# NEWSLETTER

# EUUG

European UNIX® systems User Group

Volume 9, No. 1
Spring 1989

## CONTENTS

# EUROPEAN UNIX® SYSTEMS USER GROUP NEWSLETTER


*Volume 9, Number 1*
*Spring 1989*

**Editorial Team:**

Philip Peake
Publications Executive
Axis Digital
135 Rue d'Aguesseau
92100 Bonlogne
France
philip@axis.fr

Alain Williams
Parliament Hill Computers Ltd.
7 Prospect Street
Caversham
Berkshire RG4 8JB
U.K.
addw@phcomp.co.uk

**Typesetting:**

Laura Dekker
The Instruction Set Ltd.
City House
190 City Road
London EC1V 2QH
U.K.
laura@inset.co.uk

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

**Published by:**
The EUUG
Owles Hall,
Buntingford
Hertfordshire SG9 9PL
U.K.
euug@inset.co.uk (...!mcvax!ukc!inset!euug)



ISSN 1011-4211

# Editorial

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

*Parliament Hill Computers Ltd*
*Berkshire RG4 8JB*
*UK*
*+44 734 461232*

## New Columnists

This issue we welcome two new columnists to the EUUG Newsletter.

**Donal Daly** is writing *Software | Review*. Donal looks at some of the more interesting software that has appeared over the net: software that is useful, software that is instructive, and software that is fun.

Many of you will remember in Cascais (Portugal) **William Roberts** as a co-author of the longest paper to be ever presented at an EUUG conference. His subject then, as now, is *Window Systems*. A few years ago graphic terminals was the realm of the lucky few – now they are within the budgets of most organisations and look set to soon replace ASCII terminals.

You can make the jobs of both Donal and William by sending them relevant material (reviews), comments and ideas on what they have written.

## Calendar of UNIX Events

Another new item in this issue is a calendar of UNIX events. Only a little information is given on each event as this is intended as a long term *planner*.

But most of you do not yet need to think further ahead than early April and the EUUG conference in Brussels. To help you get there you will find a complete time table (to show your boss) and a booking form (to fill in today and send to Owles Hall).

## BSD 4.3 Manuals and EUnet Directory

There are still a few copies of this excellent manual set available from Owles Hall. The price *includes* postage and makes them as cheap as if you bought them in the USA.

By the time that you read this the EUnet directory will be available. This useful publication will let you find out what you want to know about all EUnet sites.

You will find an order form for both of these, along with other EUUG Publications in this issue. (Save postage and send in your order with you Brussels booking).

## Contributions to the EUUGN

Keep supplying me with ideas and articles, contact current authors and let them know what you think of what they have written; all comments are useful – polite or not!

The copy dates for the next two issues of the Newsletter are:

24 April for 1 May
24 July for 1 September

# A Port of the MINIX Operating System to the Atari ST

*Aarron Gull & Sunil K Das*
*aarron@cs.city.ac.uk*
*sunil@cs.city.ac.uk*

*City University London*
*Computer Science Department*
*London EC1V 0HB, United Kingdom*

Aarron Gull is a (''scoff if you like, but it's true; the force surrounds us, holds us, binds everything together'') trainee UNIX guru. Currently working on his PhD (''I feel ... a grave disturbance in the force'') at City University London. He is interested in distributed processing, films and lemonade.

Sunil K. Das is a founder member of the UNIX Travel Club (at the expense of others).

**Abstract**

The STIX project consisted of two activities. The first was the creation of a MC68000 cross-development environment on a Gould PN6000 host computer. This would be used for the down-loading of C binaries to Atari ST target machines. The second was the port of the MINIX operating system, designed to run on the IBM PC, to the Atari ST range of micro-computers.

This paper describes elements of the STIX project which were particular interesting or unusual. Since the Atari ST does not have a hardware MMU, emphasis is placed on the process and memory management techniques employed in STIX. The impact of these ideas upon the system call interface, in particular *fork*( ), *exec*( ) and *exit*( ), concludes the discussion.

## The STIX Project

The **MINIX** (mini UNIX) operating system runs on the IBM PC range of micro-computers [Tanenbaum 1987a]. Externally, MINIX resembles seventh edition UNIX (V7) by having a similar system call interface and being distributed with many UNIX-like commands. Internally, however, it is very different; MINIX is a modern operating system whose modules communicate via *message rendezvous*. System call requests from user processes are effected by a message pass to the appropriate server module.

For programmers of IBM PC micro-computers MINIX is a low-cost, UNIX-like system which is a viable alternative to MS-DOS. Teachers and students will find MINIX of particular interest because it is fully documented [Tanenbaum 1987b] and the source code is distributed without licencing restrictions.

The STIX project, a MINIX port to the Atari ST, was conceived so as to allow the operating system to be studied on the 100-or-so Atari ST micro-computers at the City University. The successful outcome of the project would form the basis of future undergraduate or postgraduate courses and projects in computer science. For example:

- operating system design, directly or indirectly based upon MINIX;
- development of commands and tools which run under MINIX; and
- use of the Gould/Atari ST cross-development system.

The remainder of this paper is devoted to an exposition of the Gould/Atari cross-development environment which was constructed and the MINIX port itself. Emphasis is placed on a description of the novel techniques employed to overcome the lack of hardware memory management.

## The Host-Target Environment

A host-target environment typically offers several advantages over the traditional method of developing software on the target machine; the host tends to be faster, more stable, and has a greater disk capacity. Moreover, using a host which runs UNIX provides access to a range of tools (e.g., *lint, make*, *SCCS* and *RCS*) which are specifically intended for C program development.

It was decided that the creation of an Atari cross development system, running on a Gould PN6000, would greatly increase the speed of the port. This decision had the additional advantage in that, on completion, the system would be available to Atari programmers who require a more hospitable development environment. For this purpose, it was decided to also create a library of function calls to interface to the resident Atari operating system TOS [Gerits 1988].

Version 2.0 of the Stanford UNIX Mac C Development Kit (**SUMacC**) [Croft 1984][1] was chosen as the basis for the cross-development system. Packaged for a VAX computer running either BSD UNIX or Eunice, the SUMacC system includes compilers from the Universities of Berkeley and Stanford. These are merged with conditional compilation statements. The Berkeley code is based on Steve Johnson's portable C compiler [Johnson 1978] and was used to form the basis of the host-target system; a detailed description of which is available from the authors [Gull 1988].

The complete compiler suite comprise:

- System-wide header files;
- C compiler (*cc68*, *cpp68*, *ccom68*, *c268*, *as68* and *ld68*); and
- C library, *nm68* and *lorder68*.

The majority of the STIX-dependent modifications made to the compiler suite (as opposed to porting and bug fixes) were located in the linkage loader *ld68*. Details of these changes are covered by the discussion of the *exec*( ) system call.

The source code of the C library was not originally available to the authors. As a consequence, the decision was taken to reimplement it. This proved to be a long and rewarding exercise (13,000 lines of C and 2,000 of assembler) but does not merit a full description.

The commands *nm68* and *lorder68* were ported to the Gould. These were used to rebuild the library whenever a module was changed.

## From MINIX to STIX

For the purpose of the ensuing discussion, the term STIX will describe the port of MINIX operating system, thus enabling a clear distinction to be drawn between the IBM PC and Atari ST versions of the code.

### Booting STIX

The File System CheckK program (*fsck*) is bootstrapped on power-up by the TOS program *Gemboot*. This loads the fsck text and data from disk, clears the BSS space and starts running the code.

*Fsck* loads the modules of the STIX KERNEL into low memory according to the information held in the system configuration file *config.dat*. The BSS and headers are constructed for each KERNEL module. Execution then begins at the main KERNEL entry point.

### STIX Memory Layout

The composition of STIX is very similar to that of MINIX in that it consists of four modules which are collectively called the *KERNEL* (n.b. upper case). It differs, however, in that each KERNEL module is contained in a separate *.dat* file on the boot disk: *kernel.dat* (kernel); *mm.dat* (memory manager); *fs.dat* (file system manager) and *init.dat* (init process). The *kernel* (n.b. lower case) is

1. SUMacC is distributed in the public domain under the condition that it may be used but not sold because it is subject to a Stanford copyright.

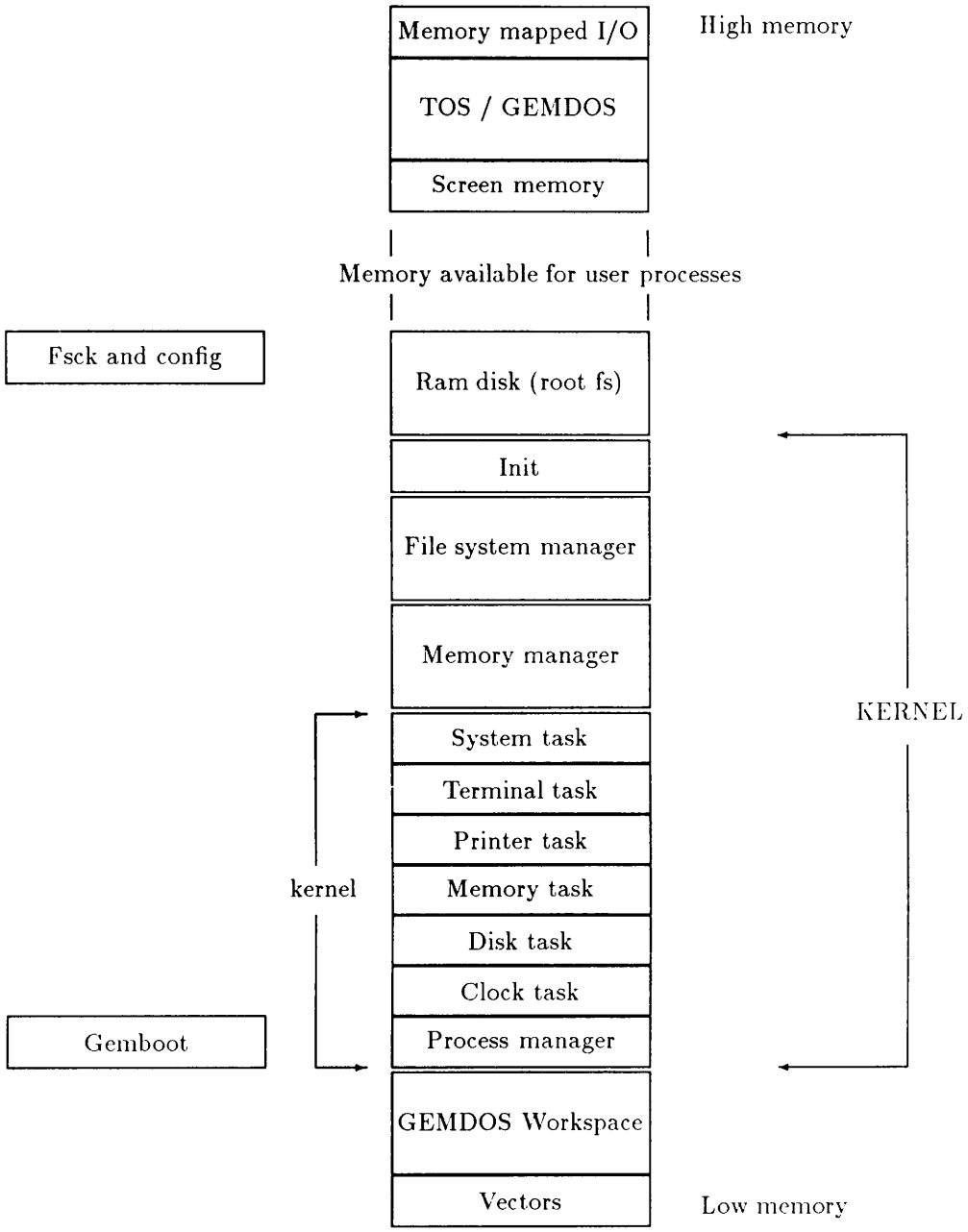



**Figure 1: The layout of the STIX memory**

further subdivided into a process manager, a system task, and device drivers for the terminal, printer, disks and clock.

Because of the nature of the ST hardware the modules of the STIX KERNEL are linked relative to each other while the modules within MINIX can be linked independently.

The modules of the STIX KERNEL can make reference to one another though use of headers: arrays of eight longs which precede their text in memory. These contain lengths from which the positions of the other modules may be calculated:

```
#define UNUSED_LENGTH   _begtext[-8]
#define KERNEL_LENGTH   _begtext[-7]
#define MM_LENGTH       _begtext[-6]
#define FS_LENGTH       _begtext[-5]
#define INIT_LENGTH     _begtext[-4]

/* Used for booting off hard disk */
#define BOOT_DEV            _begtext[-3]

/* _begtext[-2] to _betext[-1] are currently unused */
```

The overall memory layout of STIX is illustrated in figure 1.

### An Overview of STIX Processes

A process is the image of a program in execution and includes amongst its associated resources: physical data such as the text, data, BSS, stack, the invocation arguments and the environment; and the concept of state as defined by the processor registers, the open file descriptors, the current working directory and scheduling information. The memory map of a typical process is shown in figure 2.

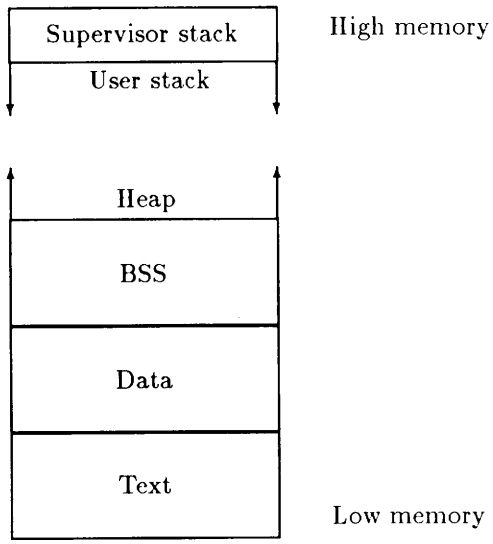



**Figure 2: A typical STIX process**

STIX processes are very similar to those of V7; each user process is created when another executes a *fork*( ).

STIX processes, like those of MINIX, are fixed in size at creation; the *brk*( ) and *sbrk*( ) system calls do not change the overall amount of memory consumed by the process: the memory requirement is simply validated.

The code images created by the *fork*( ) and *exec*( ) system calls could, in theory, be loaded into any area of free memory[2]. The Atari ST does not have memory management hardware to assist with this. Therefore, three alternative strategies for process creation in the STIX environment were considered:

- Find a compiler which produces totally relocatable code. Although C compilers exist which produce position independent code (PIC) the authors do not know of one which creates code which is relocatable while in execution.

  In theory, it would be quite simple to write a compiler which generates such code. The code produced, however, would be very inefficient and system performance would be significantly affected.

- Discard the *fork*( ) and *exec*( ) system calls and adopt a *forkexec*( ) call. This new call would create another process running a specified filename. In order to support the new call it would be necessary to place relocation information in executable files.

  This approach would have the advantage of being simple to implement, but the considerable disadvantage of losing V7 compatibility. Since compatibility with V7 was thought to be of primary importance this method was rejected outright.

- Use swapping[3] of the data, BSS and stack images of related processes to implement forking. Relocation information would be needed in executable files for *exec*( ). Substantial revisions would have to be made to the memory manager code to control such process migration.

  This approach has the advantage of retaining the V7 interface, but the disadvantage of adding substantial complexity to the kernel and memory manager.

It was decided that the final method was the most satisfactory.

*Fork( )*

In order to retain the *fork*( ) system call, STIX supports process swapping. The concept is simple: STIX processes comprise contiguous regions of TEXT (which is static) and DATA (which are dynamic), both of which are of fixed size. When a process forks, the child requires a private copy of the parent's DATA but can share its TEXT. When executing, both parent and child need to occupy the parent's original memory space.

When a process is scheduled its DATA will be moved into the memory it requires to run, swapping it with any DATA already residing there. Related processes will execute by swapping DATA in and out of the memory originally occupied by the parent.

The *fork*( ) algorithm knows little about swapping: this is handled by the context switching code. The *fork*( ) code simply creates another entry in the process table containing the new current address. The image of the child is created by the swapper; the child is flagged as *copy_on_swap*. This produces a marked reduction in the overhead of a typical *fork*( ) *exec*( ) sequence but only works because the child is scheduled to run before it's parent.

The modular nature of the STIX KERNEL means that it is necessary to introduce the concept of the *creation*, *current* and *execution* address of a process. The *execution address* is the memory a process' DATA must be located in while running while the *current address* is its present location. When a process is running its execution and current addresses will be the same. The DATA of processes which are not running, however, may be relocated so that these addresses differ.

Due to the fork mechanism, if two processes share the same execution address, they must be related and, therefore, are the same size. The *creation*

2. All MINIX processes actually start and end on 16-byte boundaries (*clicks*). This is due to address resolution on the 8088 being indexed through 16-bit segment registers which function as 20-bit registers. STIX processes start on even byte boundaries and are multiples of 4 bytes in length. This enables the *swapper*( ) to move DATA as quickly as the ST architecture allows.

3. The STIX *swapper* function should not be confused with the UNIX swapper process.

*address* space is the memory allocated to a process on execution of either the *fork*( ) or *exec*( ) system calls. It is also the memory released by *exit*( ). On termination process DATA is returned to its creation address space so that its memory can be released.

The STIX process manager's tables contain the *execution* and *current address* of each process while the memory manager stores the *execution* and *creation addresses*. The union used to store process memory maps is illustrated in figure 3.

```
union mem map  {

   struct proc_m_map {          /* Process manager only */
      unsigned mem_exe;         /* Data execution address */
      unsigned mem_curr;        /* Current DATA address */

      unsigned text_len;        /* Length of text only */
      unsigned data_len;        /* Length of data, BSS and stack */

      char copy_on_swap;        /* As of yet without DATA */
      unsigned child_creat;     /* Address of child DATA storage */
    };

   struct mem_m_map {           /* Memory manager only */
      unsigned mem_exe;         /* Process execution address */
      unsigned mem_creat;       /* Process creation address */
      unsigned whole_len;       /* Length of process */
    };
};
```

**Figure 3: The mem union.**

In STIX, the *swapper*( ) is called whenever a process needs to be restarted. It is the program which ensures that the data, BSS and stack of the process are located in the required region of memory. If necessary, the *swapper*( ) interchanges the DATA spaces of two processes which are co-resident in user memory. The algorithm of the *swapper*( ) is shown in figure 4.

Some care must be taken when copying data to and from processes. This will occur as part of message passing or due to kernel system call activities. Processes pass system call addresses which are valid when the process is swapped into its execution region. Such addresses are invalid when the process has been swapped out. Several functions have been modified in order that this is taken into account.

There is a significant flaw in the way processes are implemented on the Atari ST; the hardware provides no mechanism for protecting processes from each other; there is no way of stopping a program, accidentally or maliciously, reading or writing outside its allotted memory. This is attributable to the hardware and not to the implementation of the STIX KERNEL.

```
algorithm swapper
input:  proc table pointer
output: none
{
    if (!currently panicing &&
            process DATA not in execution region) {
        if (KERNEL process)
            panic(attempt to swap in kernel);

        found = false;
        if (!copy_on_swap)
           for (all user processes)
               if (process DATA is in execution region) {
```

```
                    found = true;
                    break;
                }

            if (found == true) {
                if (DATA are of different lengths)
                    panic(cannot swap different length processes);
                swap DATA with that in execution region;
            } else
                if (copy_on_swap)
                  copy DATA into child region;
                else
                  copy DATA into execution region;

            update the current address fields of the processes;
        }
    }
```

**Figure 4: The swapper( ) algorithm.**

Finally, although swapping DATA might seem slow, the actual cost in system performance is small. Most programs immediately follow a *fork*( ) system call with an *exec*( ) in the child. The system only needs to make one copy and perform one swap; a copy of the parent is made prior to the child running and a swap is made to release the child's memory after the *exec*( ).

The overhead of forking can be further reduced by increasing the timeslice allotted to running processes.

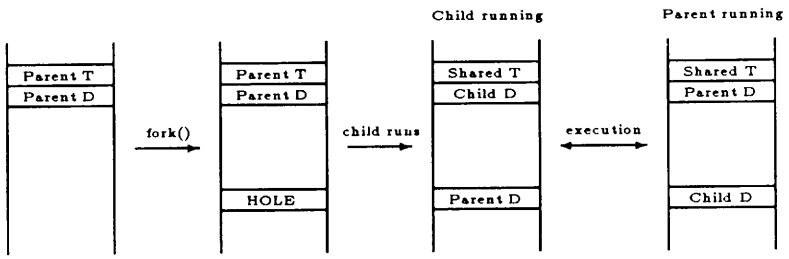
Swapping, perhaps surprisingly, reduces the complexity of the *fork*( ) system call. It does, however, make process termination and subsequent memory reallocation considerably more complex. This is accomplished during the *exit*( ) system call.

**Example 1:** This is introduced in order to clarify the usage of the address fields defined in *union mem_map{}*.

A process whose execution address and creation address are equal is scheduled to run. The process forks a child. The kernel initiates a search for a process table slot for the child. This is followed by the memory manager searching for a 'hole' in memory large enough to hold a copy of the parent's DATA image. Assuming the *fork*( ) is successful, a child process is created and scheduled to run before its parent.

When the child begins execution a copy of the parent's DATA is made into child's creation memory space. The DATA of the parent process then becomes that of the child. The parent's TEXT is shared between parent and child.

From then on whenever either process is scheduled, they must be relocated to their execution address space to run. Swapping in this way is only possible because related processes are of the same length.




**Example 1: The fork sequence**

*Exit( )*

The *exit*( ) system call has the task of clearing up after processes which have terminated. The algorithm of the exit system call is illustrated in figure 5. Points of interest are:

- The memory manager has no understanding of swapping; In order to reduce the number of message passes required only the kernel knows the current address of a process. This makes releasing the memory of a terminated process difficult: the memory manager does not know which region of memory should be released. By calling *sys_swap*( ) using the terminating process number and creation address as parameters, the process is immediately located.
- A parent process will typically have an execution address and creation address which are identical. The execution address of a process is shared with any children it created. This memory must not be released until the last child exits, even if the parent terminates. The memory of a process with no children may be released when the process exits.
- MINIX does not release the memory associated with a process until the parent process has executed a *wait*( ) system call. This could result in a *fork*( ) or *exec*( ) system call erroneously failing due to lack of free memory. STIX releases the memory of processes when they I exit ( ). This leaves entries in the process table which have no memory associated with them. This is similar to the *zombie* process state in UNIX.

```
algorithm system_call(exit)
input:  return code for parent process
output: none
{
    store exit status for next wait( ) system call;

    if (parent is waiting)
        release parent and tell everybody;
    else
        suspend process;

    tell kernel(to swap process into creation region);
    tell kernel(to stop scheduling process);

    if (no other process runs in execution region)
        free(text && execution regions);

    if (execution region != creation region)
        if (no other process runs in creation region)
            free(creation region);

    /* Process is now in zombie state */
}
```

**Figure 5: The exit( ) algorithm.**

*Exec( )*

The *exec*( ) system call has been made more complex due to the lack of hardware memory management on the ST. The main problems which had to be overcome were program relocation and memory reallocation.

The format of a STIX executable file *b.out* is shown in figure 6; there is no space between the header, text, data and relocation segments.

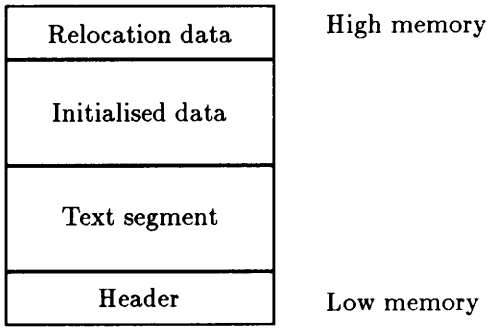



**Figure 6: The b.out file format**

STIX user processes may be any size up to a soft limit of 120K imposed by the superstructure and a hard limit imposed by available memory.

Code executed in user mode differs from that of the STIX KERNEL in the following ways:

- Whilst all executable files in STIX have headers, files intended to run in the KERNEL have a different header format to those intended to run in user mode. The KERNEL headers hold data about the size of the KERNEL modules; user headers contain details which are necessary to create processes.

  STIX executables which are intended to run in user mode differ slightly from those of MINIX in that two previously empty header fields are now used to hold the size of the text and data relocation information. The new header structure is shown in figure 7.

- User code must be able to be loaded anywhere in memory; KERNEL code is loaded at a fixed address in memory. Hence, user code must retain relocation information whilst the KERNEL has no need for it.

- User programs always begin execution at their first instruction; the KERNEL has multiple of entry points.

- User code requires a startup routine to set up the *argc*, *argv* and *envp* parameters; the KERNEL has no need for startup parameters.

To inform the linkage loader *ld68* that code is intended to run as a STIX *user process* the *-M* (MINIX) flag is passed from the compiler front-end.

```
struct {
    long fmagic;        /* Executable type */
    long hsize;         /* Header size */
    long tsize;         /* Text size */
    long dsize;         /* Data size */
    long bsize;         /* BSS size */
    long rtsize;        /* Text relocation items */
    long tmem;          /* Total memory required */
    long rdsize;        /* Data relocation items */
} stix_header;
```

**Figure 7: The header format**

The algorithm of *exec*( ) is illustrated in figure 8. A process invokes *exec*( ) by calling one of its front-end functions to build an image of the new process stack from the supplied arguments and environment. The stack image is passed as an argument to the *exec*( ) system call.

After a process has executed a successful *exec*( ) system call and prior to being restarted, its memory map will resemble that in figure 9.

The most important features of *exec*( ) are:

- new text and data segments have been loaded and relocated;

- the BSS, heap and stacks have been filled with zeros;
- the arguments and environmental information have been copied into the user stack; arrays of pointers to each entry have been set up; argc, argv and envp have been pushed into the user stack; and
- the supervisor stack has been constructed so that the process will restart execution in the C startup routine.

The C startup routine *crtso* is an assembly language routine which simply calls *main*( ) as a subroutine. No arguments are passed by *crtso*; these have already have been placed on the stack by the kernel. *Crtso* simply builds the standard parameter passing stack frame around the arguments. The routine main can then be treated as a standard C function.

```
algorithm system_call(exec)
input:  pathname and stack image
output: only if exec fails
{
    if (invalid stack image || filename is !executable)
        return(error);
    Fetch the new stack from the user;

    Read the file header and extract the header sizes;
    if (!enough free memory for new image)
        return(error);

    /* Last point a failed exec( ) will pass back to user */

    tell_kernel(to swap process into creation region);

    if (no other process runs in execution region)
        free(text && execution regions);

    if (execution region != creation region)
        if (no other process runs in creation region)
            free(creation region);

    Reclaim memory to hold new image;

    Read in the new image from the file;
    if (the header information is invalid)
        exit(process);

    Relocate the image to its new address;
    if (the relocation information is invalid)
        exit(process);

    zero the stacks and BSS of the process;
    Copy the stack image into the stack;
    Relocate the stack image;
    Take care of setuid and setgid bits;
    Reset all caught signals;

    tell_kernel(tidy the stack and reschedule the process);
}
```

**Figure 8: The exec( ) algorithm.**

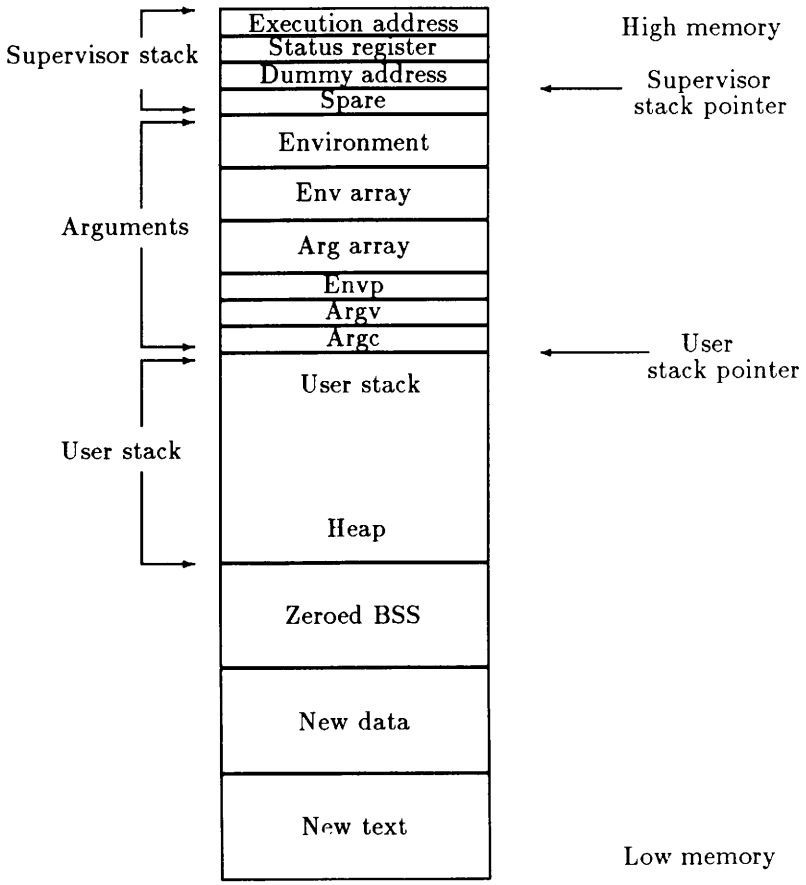
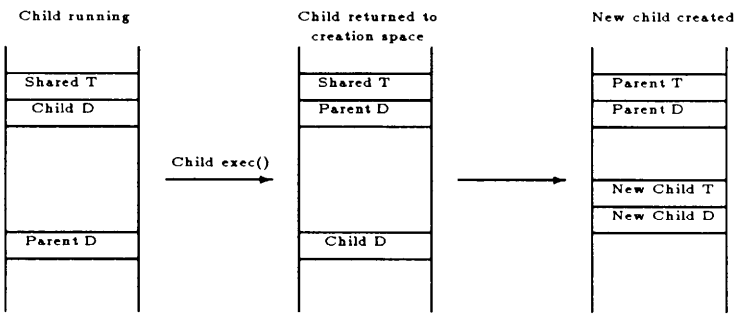


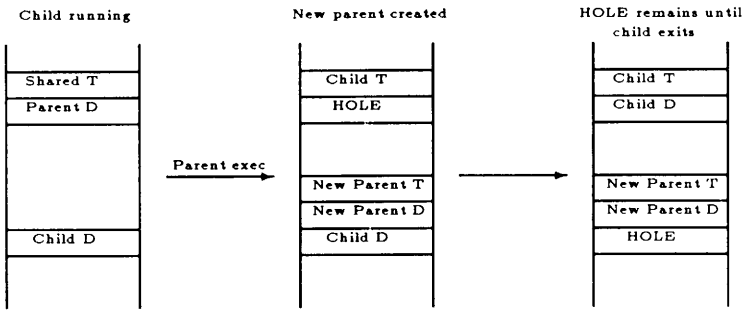

**Figure 9: The memory map of a process after exec( )**

**Example 2:** A process forks a child and the child process executes an *exec( )* system call. Once it is certain that the *exec( )* can succeed, *sys( )* is called to return the child to its creation space. The function *free_mem( )*, inside the memory manager, is called to release the creation address space. As the parent process still exists, the child's execution space is retained. Finally, memory is allocated to the new child:




**Example 2: The child fork( ) sequence**

**Example 3:** This highlights the unusual case when a process forks a child and the parent issues an *exec*( ) system call. The parent's creation and execution space is the same, but the space *has* to be retained for child execution. A 'hole' is created in memory which is only removed when the child process terminates:




**Example 3: The parental fork( ) sequence**

A complete discussion of the cycle of events during process creation, existence and termination is available from the authors [Das 1988].

## Results

In material terms the result of the project is 6 disks: the STIX *boot* disk, a 400K *root* disk and 720K */usr*, */user*, */src1*, */src2* disks. These are all that are required to run STIX as a stand-alone operating system. The KERNEL passes the exhaustive Tanenbaum Validation Suite and is V7 compatible; programs which use V7 calls run successfully in the STIX environment. Moreover, a large number of utilities have been provided.

As a side product of the port an Atari ST development system has been created on the Gould computer. This system can and is used by anyone wishing a more hospitable Atari ST program development environment than that provided by the target machine.

A summary of the changes made to the KERNEL is recorded in the following:

| Module | Total lines | Lines changed | changed |
|---|---|---|---|
| File System | 3949 | 491 | 12 |
| Memory Manager | 1445 | 456 | 31 |
| Kernel | 2450 | 1669 | 68 |
| Totals | 7844 | 2616 | 33 |

Performance tests indicate that the additional overhead involved in a typical *fork*( ) *exec*( ) sequence (in which the child immediately execs) is small. On an 8Mhz Atari 1040 this amounts to around 1/285th second per K of data, BSS and stack in the parent process (1/12th of a second for */bin/sh*).

The C compiler supplied with STIX benchmarks at 762 dhrystones with high register usage and 720 without.

## Conclusions

The STIX project has shown that it is possible to run V7 compatible operating systems on machines which have no form of hardware memory management. This particular implementation sacrifices performance when forking in order to otherwise maintain high throughput.

The authors consider 33% of changed code to be high and attribute it to the vastly different architectures of the IBM PC and Atari ST. Porting MINIX to a computer with hardware memory management would require much less work.

## References

[ [Croft 1984] ] Croft W; *SUMacC: Stanford UNIX Mac C Development Kit*; Stanford University Medical Centre; November 1984.

[ [Das 1988] ] Das, Sunil K Gull A; *The Process Life Cycle in STIX*; Proceedings of the AUUG Conference: "Networking Linking the UNIX World"; Melbourne, Australia; 13th-15th September 1988.

[ [Gerits 1988] ] Gerits K, Englisch L Bruckman R; *Atari ST Internals-The* authoritative Abacus Software; 1988.

[ [Gull 1988] ] Gull A; *STIX: A Port of the MINIX Operating System to the Atari ST*; BSc Final Year Project Report - City University London Computer Science Department; May 1988.

[ [Johnson 1978] ] Johnson S Ritchie D; *Portability of C Programs and the UNIX System*; The Bell System Technical Journal; Vol 57, No 6, Part 2, pp 2021-2048; July-August 1978.

[ [Tanenbaum 1987a] ] Tanenbaum, A; *MINIX: A UNIX Clone with Source Code*; EUUG Newsletter; Vol 7, No 3, pp 3-11; 1987.

[ [Tanenbaum 1987b] ] Tanenbaum, A; *Operating Systems: Design and Implementation*; Englewood Cliffs, NJ; Prentice-Hall; 1987.

==========

# Book Review

**The UNIX Operating System, Second Edition** Kaare Christian, John Wiley, 1988, ISBN 0 471 84781 X. Reviewed by Syngen Brown, <*syngen@king.ac.uk*>, Kingston Polytechnic

I suspect that in writing this book, the author had no clear perception of his target audience. The overall impression is of an overambitious attempt to include as much material as possible, at the expense of organisation and consistency.

The book begins with yet another minor variation of the folklore surrounding the origins of UNIX, a rudimentary introduction to timesharing systems, and the inevitable "Noddy meets the Bourne shell" dialogue.

The second phase of the book is organised in thematic groups of commands. The selection of commands is creditable, together they convey a good illustration of the tool-oriented concept that is central to UNIX. However, in this coverage; the author oscillates between simplistic explanations of fundamental concepts and tedious descriptions of more technical nuance. On occasions, such material requires the understanding of concepts which have not yet been covered. It is particularly annoying that there is no clear demarcation between basic and advanced material. This is in contrast to the style of many contemporary books, where the need to interleave different levels of material is compensated by imaginative use of typographical devices.

An unusual feature is the inclusion of sections devoted to *sccs*, *yacc* and *lex*. These are important UNIX tools, but they are seldom used by the novice. It is even more unusual to find a chapter devoted to system benchmarking. This, and the other later chapters, present an overview which misleads through oversimplicity. While such treatment of esoteric and advanced issues does add variety, it does not assist the novice and is no substitute for appropriate specialist materials.

A total omission arises in the case of the UNIX document preparation tools. The reason for this is given in the preface: Christian has another book dealing with that subject alone. Not having seen that work – it must suffice to say that, using a book on computer typesetting by Christian is rather like having Norman Tebbit as your vocational guidance counsellor.

As a reference work, the book fails for lack of coherent organisation. A prime example being the treatment of shell metacharacters. This material is distributed throughout the book, with little effort to present a unified concept. Although there is a section headed *Metacharacters*, this is restricted to those effecting filename expansion, leaving the novice with an erroneous impression of the concept.

This is a verbose work: a feature which will deter the novice by sheer information overload, and the experienced user by the effort to extract signal from noise.

To be fair, I did not find any major factual inaccuracies in the book. It is also the case that there are worse UNIX primers available. Criticism aside, it is inevitable that this book will appeal to some people, in particular those with no clear idea of what they are doing.

# AUUG Conference Report

*Jim Reid*
*jim@cs.strath.ac.uk*
*...!mcvax!ukc!strath-cs!jim*

*Dept of Computer Science*
*University of Strathclyde*
*Glasgow*

Jim Reid has been the Systems Manager in the Computer Science Department at Strathclyde University for over seven years. Away from work, he enjoys playing the trombone and listening to music – not always the same thing! Other interests are Dundee United Football Club, beer and curries.

## Getting There

Back in March last year, the Australian UNIX User's Group posted a news article advertising their forthcoming conference in Melbourne. The conference theme was to be "Networking – Linking the UNIX World". I decided to chance my arm by submitting a revised copy of my "Installing NFS on a 4.2 BSD VAX" paper that I had prepared for the Manchester EUUG conference in 1986. Nothing then happened for a few months and I duly forgot all about it, not wanting to build my hopes up.

Much to my surprise, I received a letter inviting me to the AUUG conference. Even more to my surprise, I somehow managed to get the money to go![1]

1. How the money was raised was a story of drama, intrigue, brinkmanship and eleventh hour grovelling. It doesn't bear repeating here. Special thanks are due to the UKUUG. Their support made all the difference when, at the last moment, it looked like my trip would have to be cancelled.

## Day 1 – Tuesday 13th September

After two weeks of sightseeing in Australia, I arrived in Melbourne late on the Monday evening. This meant I missed the pre-conference dinner the AUUG had arranged for the speakers. After an early night, I was up bright and early next morning (honest!) to meet the AUUG executive and get to the registration desk. It turned out that the AUUG executive were all wearing suits, a fact that was the target of some derision by many of the delegates.

I picked up my conference folder, which had been sponsored by Nixdorf. Inside was a copy of the AUUG newsletter (the conference proceedings), my name badge, a notepad, some glossy advertising material and a rather tasteful AUUG lapel badge. It was green with a gold border, shaped like Australia containing "AUUG" in gold letters.

### *Opening Remarks*
### *Peter Poole, Chairman AUUG*

The meeting was formally opened by Professor Peter Poole of Melbourne University. He said that the AUUG had come a long way since it was

founded four years before. The current conference was by far the biggest and was the first to be held in a high-quality venue, the Southern Cross Hotel. He also announced that the AUUG had just been incorporated that week, giving the group a formal status in law.

*UNIX Networks:*
*Why Bottom-Up Design Beats Top-Down*
*Michael Lesk, Bellcore*

Mike Lesk's paper was the keynote address of the conference. He gave a historical perspective on the development of *uucp*. Essentially, it started as a means for him to automatically distribute software at Bell Labs. Electronic mail was an afterthought!

He then went on to analyse current networks such as the Internet and CSnet. He said that the early lessons had not been learnt and that current trends were opposite to the early days at Bell Labs. For rapid growth, computer networks should have no central administration. (This would obviously upset the *uucp* backbone sites, the NIC and the PTTs to name but a few.) Local administration should be minimal – one reason *uucp* has proliferated because it does not require modifications to the kernel or to most system software.

Looking into the future, Mike expected that data communications will get faster – i.e., higher bandwidth – but not much cheaper, even though processors will continue to get quicker and cheaper. As the PTTs provide newer technologies to the home, networking should become as easy as the telephone or fax. If our experiences of other mass communications – netnews, television, videotex – were anything to go by, electronic mail would be used for truly awful things when it becomes commonplace.

**Coffee**

*Future Telecommunications Products*
*Steve Jenkin, Softway Pty.*

Steve's presentation discussed some of the goodies that Telecom – the Australian PTT – were promising. "Just around the corner" were ISDN – Integrated Services Digital Network - on 64-Kbit digital lines, 34-150 Metropolitan Area Networks (MANs), Broadcast ISDN on 150-Mbit optical fibres, not to mention X.400. He doubted the ability of Telecom to deliver these services at the prices and in the timescale they envisage. Customers of British Telecom would probably think "so what's new?".

*Overview of the Sydney University Network Version IV*
*Piers Dick-Lauder, University of Sydney*

If you get confused between operating system versions and CPUs when someone says Sun-N (N = 2, 3 or 4), think of the Australian UNIX community. They also have a set of network protocols with the same name! Piers gave an overview of the changes made to their protocols to make it better – quicker, more reliable, cheaper – yet retaining backwards compatibility with older versions of the protocol.

**Lunch**

At lunch, I was joined by Greg Rose and Chris Maltby, President and committee member of the AUUG. Amongst other things, we discussed the organisation of our various national (or is it continental?) user groups. It turns out that the AUUG don't have anything like the backup provided to the EUUG and UKUUG by Owles Hall. The executive have to deal with all the AUUG's administration – memberships, subscriptions, conference arrangements, etc. – themselves. Until recently, the group's affairs were chaotic as it was difficult for the committee members to spare the time to do these tasks. I suppose this shows how much work is being done almost unnoticed by the people at Owles Hall. It certainly made me more appreciative of their efforts and of the work done by the AUUG committee for the Melbourne conference.

*NeWS Programming*
*Tim Long, Sun Microsystems Australia Pty.*

This talk was more of an overview of NeWS, Sun's PostScript-based window system. It covered the usual ground of event handling, lightweight processes and multiple drawing surfaces. In a sense Tim was trying to sell NeWS as the window system that would be all things to most users.

*The X window system: An Overview*
*Tim Segall, Hewlett-Packard Australia Limited*

An overview of the X window system was given by Tim Segall. In a fairly routine presentation, he mentioned the main X concepts – the X server, a bit-mapped display, the Xlib library, the interface to the X server protocol and so on.

*HOWS: A Window System*

*Greg Rose, Softway Pty.*

This was the comedy presentation of the conference. Greg showed a few slides of window systems: "X" (a glass skyscraper), "Colour" (some pretty stained glass windows) and "Open Look" (an unfinished house with no glass in the window frames). After the jokes, Greg told the meeting about HOWS (Horrible Old Window System). This was actually a small and rather nice window system that Peter Collinson of UKC had developed for the Atari ST. It used humble and cheap hardware to make a quite useful window system.

**Coffee**

*An Implementation of FTAM*

*Andrew Worsley, CSIRO*

Andrew described the work he had been doing on implementing FTAM for UNIX systems at CSIRO. He made some comparisons between his implementation, the ISODE FTAM and the 4.3 BSD *ftp*. His version was about twice as big as the 4.3 *ftp*, mainly because the FTAM protocol is more complex than *ftp*. I was surprised that he was able to get data transfer rates for both FTAM implementations that were about 80% of those for the 4.3 BSD *ftp*. FTAM had always struck me as a big, slow beast.

He commented that it was difficult to fully test his code, though it did work OK when tested against itself. Access to the ISODE implementation had greatly helped testing and bug finding in both versions. He would have liked to be able to test his code against commercial FTAM software, but he found that vendors were very reluctant to let him have access to even beta-release versions of their products.

*IOS/OSI Under Berkeley UNIX*

*Mike Karels, University of California, Berkeley*

Mike's talk was on the work being done at Berkeley and elsewhere to integrate an ISO protocol stack into a future Berkeley UNIX release. It appears that most of the necessary bits are either ready or being worked on: X.25 sockets and ISO transport protocol software from the University of Wisconsin, X.400 from University College London and Nottingham University, the virtual terminal protocol from the Mitre Corporation and X.500 directory services from the National Bureau of Standards and perhaps UCL. Some of this software would be available in a later ISODE release, but all should appear in a subsequent Berkeley release.

Berkeley's work in this project will primarily be the kernel-level integration and updating of the socket framework to support ISO addressing. Some of this work would entail "kludging the clean design of the existing socket mechanism to cope with the warts in the ISO protocol stack". (I paraphrase Mike's words since the detail is too much to mention here.) Mike also touched on the work Berkeley had done towards giving BSD a POSIX interface.

**Cocktail Reception**

The first day closed with a cocktail reception in the exhibition area. I stuck to the Foster's Lager and Victoria Bitter since I've never been fond of drinks with fruit salad or wee umbrellas in them. I asked Mike Karels how the Berkeley internals book was coming along. Apparently it had gone to Jaap Akkerhuis for typesetting and that proof reading the first draft was completed. This meant that the book would be due to appear in late December or early 1989, which is how it eventually turned out.

## Day 2 – Wednesday 14th September

*Futures for UNIX Software*

*Larry Crume, AT&T Unix Pacific Co. Ltd.*

Larry is responsible for UNIX in the Pacific, Asia and Australia. He outlined AT&T's plans for the future of UNIX as "the platform for the 1990s". He said that Xenix, System V and Berkeley UNIX versions would be merged into one version called System V Release 4.0 by 1989. Application Binary Interfaces would be defined for the common processors – Intel 80386, Motorola 68000 and 88000, SPARC, the Clipper and the MIPS R3000.[2] It was **not** the case that AT&T were restricting UNIX to the SPARC, simply that the SPARC processor ABI was the first that AT&T had defined. He gave details of when to expect completed definitions of the ABIs for other processors.

He went on to explain what would be in System V Release 4.0 (SysVR4). The answer was just about everything: real time, POSIX conformance,

2. I assume it was an oversight that there was no mention of an ABI for the VAX or the 3B2.

internationalisation, X/Open capabilities and improved operation, maintenance and administration facilities. As part of the unification, the release would support BSD-style signalling, and it would take NFS (running on top of TLI) and Sun RPC from SunOS, as well as Sun's merged X/NeWS window system. SysVR4 would still have sockets, although they'd probably be implemented on top of streams and the TLI. NFS and RFS would both be present. A new version of SVID will be developed.

Larry spoke a little about the AT&T/Sun collaboration. SunOS releases from version 4.1 onwards would be SVID-compliant. (He didn't say if that was the existing SVID or the new one.) He also said that AT&T and Sun were already looking at System V Release 5.0. This would entail a kernel restructure and ideas such as using C++ and Mach were under consideration.

Larry said that AT&T would *participate* with OSF, but would probably not become a member. He underlined AT&T's stated commitment to open systems.

Open Look was mentioned and a videotape of a marketing presentation was available for people to see what it would look like. Open Look will be based initially on X11 with a NeWS version later. It will have a "standard look and feel" – and will be an open standard, with the source code being licensed. An application style guide was to be published by the end of 1988.

In the question and answer session, Larry was asked how much of SysVR4 would be unbundled. His answer was "you get what you pay for", so I suppose this means that things like troff will continue to be unbundled. Personally speaking, this is probably a good thing since I doubt if we'll have the disk space to take everything that would be on offer in System 5 Release 4.0. He would not be drawn on the issue of source licensing, saying the details had still to be worked out. However, I think he did say that licences would continue to be available to academics at "reasonable sums".

*Future Berkeley UNIX Developments*

*Mike Karels, University of California, Berkeley*

Mike's second presentation started with a brief description of the recent 4.3-tahoe release. He then went on to list the current projects being worked on at Berkeley. These are:

— routing protocols
— the Internet nameserver
— ISO/OSI networking
— a POSIX-compliant interface
— system interface updates
— the generic file system interface
— rearrangement of the UNIX filesystem

A new virtual memory system is under development. This will permit shared memory, file mapping, device memory mapping, copy on write and lightweight processes amongst other things. There will be one large sparse address space using multi-level paged data structures. Swap space won't be pre-allocated, but will be done through the file system. All of this will be integrated with the kernel memory allocator and the network and filesystem buffers. This might strike the reader as being similar, if not identical to Mach. This is because Mach is being used as a model for the VM system and Berkeley may even use some Mach code.

The next Berkeley release will have a POSIX-style *tty* handler and may go some way to providing a streams-like framework for the *tty* and network protocol handlers. There will also be some kernel clean-ups – the merging of the proc and user structures and a new sleep/wakeup mechanism.

On licensing, Mike said that Berkeley are considering using System V Release 2 as their base for future distributions so that they can use AT&T updated utilities such as *make*, *sccs* and *sh*. Earlier versions of BSD relied on the now defunct 32V UNIX licence. System V Release 3 won't be chosen because of its demands for SVID compliance.

*Unified UNIX*

*John Young, Sun Microsystems Australia Pty.*

This talk contained more or less the same information as Larry Crume's presentation. Sun and AT&T were working on a new merged UNIX that would have *everything* – NFS, RFS, sockets, streams, X, NeWS, POSIX compliance, SVID conformance (was that the old SVID or Larry's new one?), a nod in the direction of X/Open, all possible flavours of signals, System V this, Berkeley that and something else again from SunOS.

**Coffee**

*Open Software*

*Tom Daniel, Hewlett-Packard Australia Limited*

Tom described the Open Software Foundation that had been formed by Apollo, DEC, H-P, IBM and others. He explained (not at all convincingly to

me) how these companies would produce a common UNIX version that could somehow be proprietary to each member company. He said that OSF were using IBM's AIX as a starting point. Each company would submit its own technologies to a panel for assessment and incorporation into the OSF standard. Input from academic institutions would also be welcomed. The OSF members could then add their own features to the common standard to satisfy their customer's demands.

*ABI and OSF:*
*Technology and Future Impact on UNIX*
*Ross*

*Ross began his presentation by showing a slide he had used at a USENIX conference three or four years ago. His slide (meant as a joke then) made ridiculous and incredible predictions. Ross was very surprised to see them come uncannily true. The ones I remember were:*

— *Bill Joy leaves Berkeley to join a start-up company*

— *DEC market UNIX*

— *AT&T buy up Bill Joy's company*

— *AT&T pay Bill Joy to produce a merged UNIX*

— *IBM adopt UNIX*

*He then tried to explain the chaotic marketing and standardisation efforts currently surrounding UNIX, showing the viewpoints and concerns of users, software developers and vendors.*

*Panel Session*

The five speakers from this morning's session lined up for a panel session that turned out to be rather subdued. I had been expecting fireworks, but there were none. Robert Elz was lurking around at the back of the hall, but he was keeping silent.[3]

Most of the questions got the same answers. Who said what (I've paraphrased and perhaps slightly exaggerated their statements) is left as an exercise to the reader:

*"OSF will win in the end as their standard is not proprietary"*

*"System V Release 4 will win as it will have everything in it"*

*"We're not really interested in standards?"*

*"Just wait for a year or two until the dust settles"*

**Lunch**

I skipped lunch to take a long look round the exhibition. The biggest surprise of all was that IBM were there. They had a bunch of PS/2s and a couple of RTs (6150s) on show. There were also some glossy handouts on AIX/370. IBM now offer their UNIX – a System V/BSD hybrid – on top of VM on their mainframes.

Another surprise was the Sony stand. They were exhibiting their NEWS workstations which looked very nice. For me, the glossies had a lot of the right buzzwords in them – TCP/IP, 4.3 BSD, NFS, X-windows – to name a few. The machines had bitmapped screens and had two 68020 or 68030 processors – one for I/O and one for your application. The colour workstation had some very pretty demos and the quality of both the colour and monochrome displays was very good. Apparently, Sony went for 4.3 BSD instead of System V because it simplified the legal and licensing problems. The Sony rep. told me that System V would not be important to their marketing as they were going after the academic/scientific market. [They had some office automation software on show, so that could be another target for them.]

The Comperex stand proved popular. This was probably because they served free ice-cream. I liked the strawberry best.

Also on the Comperex stand was a video camera and frame buffer. The AUUG were taking video pictures for their face server project. By the time I got home and was back at work, an icon of my ugly face was waiting to stare at me from my mailbox.

*Distributed Trouble:*
*The University of Sydney Experience*
*with Networked Workstations*
*Rex Di Bona, University of Sydney*

This paper won a prize from the AUUG for the best student paper at the conference. Rex explained some of the security problems that he has encountered with a number of networked

3. I gathered from other AUUG delegates that Robert was usually outspoken and always ready to make blunt, direct statements. I suppose he'd be the AUUG conference equivalent of Dave Tilbrook.

workstations at the University of Sydney. Some of the attacks were mentioned were astonishingly simple.

*Installing and Operating NFS on a 4.2 BSD VAX*
*Jim Reid, Strathclyde University*

This may well have been the best paper at the conference, but then I'm biased! Actually I was thrilled and a little scared to make a UNIX presentation when Ken Thompson[4] was in the front row of the audience.

*Networks: Legal and Social Implications*
*James Watt, Griffith University*

This was a thought-provoking talk that explored the legal issues involved in using computers and networks. James posed some interesting questions like:

*"Can someone be sued over faulty software posted to comp.sources.unix?"*

*"How could a court prove that a user did something illegal on a computer?"*

*"Are printouts admissible as evidence?"*

*"Who, if anyone, can be prosecuted over obnoxious electronic mail?"*

Sadly, the answer appears to be that the questions will only be resolved when someone is taken to court and that the legal profession does not seem to understand the problem.

**Coffee**

*Human Interface Issues*
*Michael Lesk, Bellcore*

Mike closed the second day with two talks that should be familiar to those who attended the UKUUG meeting at Newcastle in July 1987. His jokes and anecdotes were the same as the ones he used then. He talked about his experiments with an on-line library index at Bell Labs and then discussed his work on computerised direction finding.

When Mike worked at Bell Labs., he experimented with two on-line index systems for the library. The first used a simple keyword lookup taking as input a string that was either part of a title or an author's name. The second used a top-down menu system based on the Dewey classification system. His results suggested that the keyword system was more popular. However, when he repeated the experiment with keyword and menu interfaces to an Associated Press news wire service, he found that there the menu-style interface was more popular.

The computerised direction finding work was interesting. It is a straightforward enough problem, but for me it was memorable for Mike's story of how a US car-hire firm wanted to exploit the idea and how it was killed by the bureaucracy in AT&T.

**The Conference Dinner**

Most of the delegates went off to the bar to kill the hour or so before the conference dinner. I got into conversation with Ken Thompson[5] who started to talk about Plan 9, his new operating system. This was named after the notoriously bad 1950's science fiction B-movie "Plan 9 From Outer Space". We spent the rest of the time in the bar discussing this movie and other films in that genre. Apparently, they have quite a cult following at Bell Labs.

As I wandered into the dining room, I met George Michaelson of Yorkbox fame. Some of you may remember the acrimonious mail I used to send him when he worked at York University on the dreaded Yorkbox. Now he works in Australia (surely not because of my mail?) and this was the first time we'd met in person. I felt it was rather strange to travel 12,000 miles to meet for the first time someone I sort of knew who used to work only 200 miles away.

The dinner was sponsored by Pyramid, who gave everyone a small plastic pyramid with their logo on it. The pyramid was magnetic and came with some paper clips, so I reckoned it had a useful purpose apart from being a welcome advertising freebie.

John Mashey tried to give his talk about comparing software development to a military mission. Sadly, hardly anyone seemed to be paying attention to him. There was too much background noise for him to be heard and most folk looked to be more interested in the food and drink.

4. If you're going to namedrop, you might as well make sure the name is one worth dropping into the conversation.....

5. There I go, namedropping again....

The dinner became quite boisterous. There were lots of paper aeroplanes flying about and some of the helium-filled balloons ended up on the ceiling. The high spot came when one bloke collected all the balloons and tied them to a heavy piece of silverware that held a card with a table number on it. This thing must have weighed close to one kilogram and it ended up on the ceiling about 5 metres above us, suspended from about 50 balloons! Clearly, AUUG dinners can be as interesting as those held at EUUG conferences.

## Day 3 – Thursday 15th September

*AUUG ACSnetSIG Meeting*

Thursday started with a Special Interest Group meeting on ACSnet – the Australian UNIX network that uses the SUN protocols. There was some concern about a plan by the Australian Universities to set up their own network. This would be a bit like JANET, though it would use high bandwidth (> 1 Mbit) connections. People were worried about how this would be funded, perhaps by stopping the money spent on ACSnet. There was also concern over the choice of protocols that may be used – TCP/IP, SUN, DECnet, SNA, ISO, even Coloured Books were being suggested! Some people felt the lines were too fast until it was pointed out that there was talk of running telephone and video circuits on the proposed network.

*Link Management and Link Protocol in SUN IV*
*Bob Kummerfeld, University of Sydney*

This talk was a followup to the one given by Piers Dick-Lauder the previous day. Bob went into much more detail on the new link level protocol used in version IV of the SUN protocol. The new protocol had been designed to be adaptable enough to work efficiently and cheaply over a variety of links including TCP/IP, telephone and X.25.

*A New C Compiler*
*Ken Thompson, Bell Labs.*

Ken had taken some sort of study leave and was working at the University of Sydney. He talked about a new C compiler he had been working on. The main design goal was that the compiler had to be *fast* – people tend to compile code more often than they execute it! The new compiler was also to be fairly portable, generate "not terrible" code and had to support ANSI C. The compiler was targeted for the NS32032, M68000 and CRISP processors, though people were working on other targets such as the VAX.

The compiler takes 8 passes to convert C code into a .o file. The last pass is an assembler. Ken described some fairly quick but effective optimisation techniques that the compiler uses: for instance, a simple peep-hole optimiser, register allocation based on loop depth, assessing the costs of saving registers for subroutine calls (arguments are not popped after a call) and so on. There was no distinction made between automatic and external variables in what Ken called the compiler's "registerisation".

I managed to note down the times he gave for comparison between the Sun C compiler (*cc*) and his new compiler (*2c*). The times were for compiling *dc* – roughly 2,100 lines of C and 2,400 lines of *#include* files – on a Sun 3/60.

| | *cc* | *cc -O* | *ld* |
|---|---|---|---|
| cc | 34.8 | 57.0 | 0.3 |
| 2c | 6.5 | 10.9 | 2.9 |

Interestingly, a version of dc that was compiled with the new compiler was about 10% quicker than one using the standard Sun-supplied compiler.

**Coffee**

*Distributed Processing*
*in the Queensland Government*
*Peter Waller, CITEC Queensland*

This paper discussed the development of computerisation of the Queensland State Government. The presentation explained how, in consultation with the University of Queensland, the state is making increasing use of UNIX throughout its departments. Peter discussed the particular problems of educating naive users and of networking and data communication, especially with sensitive installations such as the police computer.

*Time Synchronisation on a Local Area Network*
*Frank Crawford, Q.H. Tours*

Frank described the real problem of keeping time synchronised on the 4 Pyramids that Q.H. Tours maintain their database on. This was solved by using the 4.3 timed utility, though it had to be modified for his circumstances. Time could not be allowed to go backwards or stop for more than a second as this would break the database. A

mechanism was also needed to make the systems fault-tolerant. This was achieved by making all the computers keep time rather than by selecting a single master.

*On MICROLAN 2 - An Office Network*
*Richard Lai, ICL Australia Pty.*

This described a simple and cheap network developed by ICL for use in office environments.

**Lunch**

I had a lengthy chat with John Carey, the editor of the AUUG Newsletter, over lunch. The AUUGN (pronounced 'organ'), comes out every two or three months, much like the EUUG newsletter. He usually had difficulty getting contributions for the newsletter, much like the editor of the EUUG newsletter. I sympathised with him and, to salve my conscience, that's why I've prepared this article.

*Writing Portable Programs*
*Stephen Frede, Softway Pty.*

Stephen was unable to present his paper, so Greg Rose did it for him. The paper outlines the differences between the main UNIX variants – System V and BSD. The differences were discussed at the level of system call names and parameters, library routines (i.e., When do you use *index*( ) instead of *strchr*( )), and *#include* files.

The paper was incomplete, but if Stephen does complete the job (if this is even possible), a finished version would be an invaluable aid to every UNIX programmer.

*Real-time Disk I/O Scheduling on UNIX Systems*
*Jeonbae Lee, ETRI, Korea*

The Electronics and Telecommunications Research Institute in Korea have been working with Samsung to develop a multiprocessor UNIX machine using M68000s. This machine has a similar design to the Sequent and Encore boxes. Lee described how the group at ETRI had modified the UNIX kernel (called KONIX – Korean UNIX) to give precedence to disk I/O requests from real-time processes. He produced a table that showed how little the response time for a real-time process degraded as the load on the computer increased.

*The Process Life Cycle in STIX*
*Sunil Das, City University*

Sunil wasn't as lucky as I had been in getting to Australia. Yours truly gave a presentation on the work that Sunil and Aaron Gull had done to port Minix to the Atari ST. I had also been asked to tell the Australians about the UKUUG, so I did that as well.

**Coffee**

*RISC and the Motion of Complexity*
*John Mashey, MIPS Computer Systems*

Sorry, but my notes for this talk have disappeared and there wasn't a paper in the conference proceedings. As I remember, John's talk was about the evolution of the MIPS RISC processor. He went into great detail about various trade-offs that the hardware and software engineers made to make the chip run fast. It turned out that the hardware people were able to do clever tricks that made things easier for the compiler writers. They in turn were able to do clever tricks that helped the hardware design people.

I also remember that John brought a few of the latest MIPS chips along for the audience to look at. He counted them all out and counted them all back in again, which was a pity. I would have liked another souvenir to bring home along with the AUUG badge.

**Close**

At some point over the three days, the AUUG held their business meeting. It was brief, even by UKUUG and EUUG standards. The committee was quickly re-elected and details were given of the 1989 AUUG Conference. This will be held in the Hilton Hotel, Sydney between the 8th and 11th August, 1989. The conference theme will be "Nobody Ever Got Fired for Using UNIX".

## Final Impressions

I had thoroughly enjoyed the conference. The presentations had been of a high standard – at least as good as at any EUUG conference I've been to – and most of the speakers had something interesting and informative to say. I'm glad to say that UNIX is very much alive and well "down under". I hope that it won't be long before I get the chance to return there.

# Journées UNIX

*Rosalie Hurtado*

## Introduction

Rosalie Hurtado is a journalist specialising in UNIX. She has been the driving force behind several UNIX-oriented publications in France, the best known being */usr/info/sm90* which was dedicated to the SM90 workstation, and */usr/info* which is a more general UNIX publication.

Rosalie has also been active in promoting Grenoble as a centre of UNIX competence in France, and was heavily involved in the organisation of both the 'UNIX club of Grenoble' and the recent combined exhibition, conference and AFUU AGM, known as the 'journées UNIX' held at Grenoble.

This is her account of that meeting.

## UNIX: Grenoble dans la course à la standardisation

Grenoble sort renforcée des journées UNIX organisée pour la premiere fois en province. Une avalanche d'informations a peine stabilisées et interessant la capitale des Alpes ont emergées:

Le centre de recherche Européen d'OSF;

Le centre de recherche du constructeur de supercalculateurs, Alliant;

Et enfin la premiere adhesion universitaire francaise a OSF est Grenobloise (ENSIEG).

Une accumulation de petits succes, signe de l'attraction qu'exerce la ville sur le plan scientifique a un niveau mondial.

Inutile de dire que l'IMAG (Institut d'Informatique et de Mathematiques Appliquees), le Centre de recherche Bull, son groupe de competence UNIX, les developpements menes par Hewlett-Packard dans le domaine des reseaux, l'implication UNIX de Cap Gemini Innovation, Sema group et Cisi informatique contribuent a ce dynamisme et a cet engouement pour UNIX.

Selon une etude publiée par le Comité d'Expansion Economique de l'Isere (Grenoble, pole informatique Europeen), 80 societes grenobloises utilisent UNIX.

En augmentant son potentiel, Grenoble espere participer aux grands mouvements strategiques qui se jouent aujourd'hui pour accelerer la standardisation d'UNIX.

Grenoble, the capital of the Alpes, has emerged from the 'journées UNIX', re-enforced by the experience. There has been an avalanche of interesting, but only just stable information:

The establishment of the European research centre of OSF;

The centre of research for the manufacturer of the super-computer, Alliant;

And, at last, the first French university to join OSF is a Grenoble university – ENSIEG.

An accumulation of small successes, which are a sign of the attractions of the town of Grenoble for the scientific world.

It goes without saying, that IMAG (the Institute of Computing and Applied Mathematics), the Bull research centre, with its UNIX competence centre, developments brought by Hewlett-Packard to the domain of networking, the implication of Cap Gemini Innovation in the UNIX world; the Sema group and Cisi Informatique all contributed to the dynamic feel of this event.

According to a study published by the Committee for the Economic Expansion of Isere (Grenoble axis of European Computing), 80 companies in Grenoble actually use UNIX.

In augmenting its potential, Grenoble hopes to participate in the strategic developments taking place today pushing forwards the standardisation of UNIX.

Il semblerait que chacun y aille du sien pour accelerer ce processus.

Objet de standardisation: les interfaces utilisateurs qui aujourd'hui sont loin d'êtres stabilisées.

Les resultats de l'appel d'offre d'OSF a ce sujet devraient permettre une avancee, selectionnant le meilleur produit du marche.

Gilbert Eloy, Directeur regional d'.SM OSF (France, Benelux, Pays-Bas) souligne que le nombre des membres associes ne cesse d'augmenter:

"Nous sommes aujourd'hui une soixantaine, un grand utilisateur, Shell, nous a rejoint."

Jean-Claude Frobert, Directeur Marketing des systèmes ouverts d'Unisys annonce l'ouverture du premier 'Open System Center Européen' a Zurich.

"Nous allons ouvrir 16 centres de competences UNIX d'obedience X/Open en Europe. Notre objectif est de tester des logiciels et du materiel et de decerner des labels officiels X/Open"

indique-t-il.

Un budget de 3,5 millions de dollars est consacré a cette operation qui montre qu'Unisys a fait d'UNIX son cheval de bataille.

Ce systéme represente 5%, soit 500 millions de dollars du chiffre d'affaires consolide d'Unisys. Selon une enquete Infocorp Unisys couvre 15 % du marche UNIX.

Faire partie du groupe de travail X/Open devient un argument commercial:

Top Log annonce que LPI (Language Processors Inc.), dont les produits sont diffusés par la societé francaise, rejoint le groupe de travail X/Open, la norme globale, internationalement reconnué et supportée par CAE (Common Application Environment) devient un atout.

Synergie entre la recherche et l'industrie

Le salon, de petite taille, mais ou les plus grands fournisseurs de materiel UNIX etaient representes (IBM, Unisys, Bull, DEC, HP, Apollo, Alliant, Convex, Apple), a fait l'objet d'une grande premiere.

En effet, nos gourous UNIX francais:

Pierre Laforgue de l'IMAG, Eric Paire du Centre de Recherche Bull et Yves Devillers, responsable du backbone FNET a l'Inria,

It seems that everyone has to do his best to speed this process up.

One object of the standardisation process: user interfaces, which today are far from being stable.

The results of the 'Requests for Technology' made by OSF on this subject should permit an advance in this area, by selecting the best existing products on the market.

Gilbert Eloy, the regional director of OSF (France, Benelux and Holland), underlines that the number of members associated has not stopped growing:

"Today, we are sixty, a large user, Shell, has joined us."

Jean-Claude Frobert, marketing director of Unisys open systems, announces the opening of the first 'European Open Systems Centre' in Zurich.

We are going to open 16 centres of UNIX competence, following the X/Open philosophy, in Europe. Our objective is to test software and hardware to give an official X/Open label"

he said.

A budget of 3.5 million (US) dollars has been created for this operation, which shows that Unisys has made UNIX 'its battle cry'.

This system represents 5%, that is 500 million dollars, of turnover of the Unisys group. According to a survey by Infocorp, Unisys actually takes 15% of the UNIX market.

Being a part of an X/Open working group becomes a commercial necessity:

Top Log (a subsidiary of Metrologie) announces that LPI (Language Processors Inc.), whose products are distributed in France by Top Log, has joined the X/Open working group, The global standard internationally recognised and supported by the CAE (Common Applications Environment).

Synergy between research and industry

The exhibition, was small in area, but had representatives from the biggest UNIX suppliers (IBM, Unisys, Bull, DEC, HP, Apollo, Alliant, Convex, Apple), this is perceived as being a good 'premiere'.

Some French Gurus:

Pierre Laforgue of IMAG, Eric Paire of the Bull research centre and Yves Devillers, who is responsible for the EUnet backbone at INRIA,

ont decidé, non seulement de relier les ordinateurs entre eux grace a un cable Ethernet et a TCP/IP, mais egalement de creer une connexion entre ce reseau local heterogene et Internet (via Transpac). Ainsi le DPX2000 de Bull, sur le stand ANL (Agence Nationale pour le Logiciel) a servi de relai aux autres ordinateurs, qui, sur l'espace de deux jours ont appris a connaitre le monde specifique 'des News et des mails'.

Internet donnant ainsi acces aux deux plus grands reseaux heterogenes francais bases sur UNIX (l'Imag et l'Inria) et aux nœuds de communication avec les Etats Unis.

Grace à la premiere experience ont a pu voir un ecran Bull sur le stand de DEC par exemple, ou une fenêtre de Vax Station dessinait une image transmise par un DPX 2000 par X Window (X11).

De même X11 a marche entre le DPX 2000 du stand de l'ANL et le Stand Apple.

Le Mac II supportait le nouvel AUX (vu en France pour la premiere fois, mais non encore commercialise aux Etats Unis!). Cet AUX supporte non seulement X11, mais également un UNIX system V fort honorable, avec meme une caracteristique rare (car tres difficile a implementer sous SV): le 'job control' sous system V (control-Z sous csh).

Devant la camera de FR3, Eric Paire et Yves Devillers ont pu faire une demonstration en reel de l'interet de communiquer par les News:

"Nous avons recupere le fichier de patch officiel de sendmail pour protection anti-virus en 21 secondes par un 'anonymous ftp' sur un serveur americain, via la liaison satellite entre Sophia et Princeton"

expliquent-ils.

A cette occasion, l'AFUU decentralisait pour la premiere fois sa conv les resultats de leurs travaux. Pour le grand public des plaquettes ont ete editees:

Vivre avec UNIX, reussir avec UNIX.

"La bande 'benchmark' SSBA est aujourd'hui connue de tous les constructeurs"

indique-t-il.

Un sous-groupe 'Benchmark-SGBD' s'est donne pour objectif

decided to not only to interconnect their machines via Ethernet and TCP/IP, but also to make a connection this heterogeneous network and the Internet (via TRANSPAC). A DPX2000 from Bull, on the stand of ANL served as a relay to the other machines, and during the two days of operation introduced everyone present to the world of News and mail.

The Internet connection thus gave access to two large heterogeneous French networks based upon UNIX (IMAG & INRIA), and to communication nodes in the USA.

Thanks to this effort, it was possible to see Bull software via a machine on the DEC stand, or watch a window on a Vax Station drawing an image transmitted by a (Bull) DPX 2000 running X Windows.

The same X11 setup worked between the DPX 2000 on the ANL stand, and the Apple stand.

The Mac II was running AUX (seen in public for the first time in France but not yet commercially available in the USA). AUX supports not only X11, but also a complete System V, with a rare characteristic (because it is difficult to implement on SV): 'job control'.

Before the TV cameras of FR3, Eric Paire and Yves Devillers were able to give a real time demonstration of the Internet being used to read News:

"We have transferred the official patch for sendmail, to protect against 'virus infection' in 21 seconds by anonymous ftp, from an American server, via satellite, between Sophia (in the south of France) and Princeton"

they explained.

On this occasion, the AFUU decentralised, for the first time, its 'Convention AFUU', during which the results of their various working groups were made public. They had produced some literature for the public:

Living with UNIX, and succeeding with UNIX.

"The benchmark tape (SSBA) is now recognised by all of the manufacturers"

they said.

A sub-group 'Data Base Benchmarking' has set itself the following objectives:

"d'offrir le plus tot possible une version 0, d'une suite de benchmarks SGBD-independant et UNIX-portables significatives des performances essentielles des SGBD sous UNIX",

dixit J.M. Saglio, responsable de ce groupe.

Dans le groupe de travail Securite, CHAOS (pour Controle Hermetique et Automatique des Objets Sensibles sous UNIX) a été mis au point pour debusquer les pirates.

Selon les propos de Philippe Dax:

"Face a la multiplicite des attaques sur les systemes informatiques accessibles a distance, les outils d'audit, preconises dans le Trusted computer System Evaluation Criteria (TCSEC) dit 'Livre Orange', sont devenus necessaires. Sous UNIX, il n'existe pas, de façon satisfaisante, d'outil standard qui permette d'auditer un système. CHAOS est un logiciel qui essaie d'y remedier."

Les systemes informatiques se complexifiant de plus en plus, ils entrainent une fragilisation globale de l'organisation. UNIX ou pas UNIX, tous les systemes communicants sont loges a la meme enseigne!

"To offer, as soon as possible, a 'version 0', of a benchmark suite, which will be data-base independent, and UNIX portable, which will give an indication of the essential performance of a data base system under UNIX",

said J.M. Saglio, the head of this group.

In the working group on security, CHAOS, there was an explanation of steps to take to deter pirates.

According to Philippe Dax:

"Faced with the multiplicity of attacks on computing systems accessible via networks, the auditing tools recommended in the Trusted Computer System Evaluation Criteria (TCSEC), also known as 'The Orange Book', have become necessary. UNIX doesn't have, at least in a satisfactory way, any standard tools to perform a system audit. CHAOS is a system which will try to remedy that problem."

Computing systems become more and more complex, and they cause a certain fragility in the global organisation. UNIX or not, all systems which communicate are sailing under the same flag!

# EUUG Executive Report

*Helen Gibbons*
*euug@inset.co.uk*

*EUUG*
*Owles Hall*
*Buntingford, UK*

Helen Gibbons is also the business manager of the EUUG and is contactable at the EUUG secretariat.

The EUUG Executive Committee held its last meeting of 1988 on the 12th December. The meeting was hosted by AFUU at its offices in the Kremlin-Bicetre in Paris, which incidentally must be in one of the quietest locations ever, as it overlooks a vast graveyard! The AFUU Secretary, Anne Garnery, whom many will remember from the Paris Workshop, made everyone very welcome and helped greatly with transport, as the date fell in the middle of the Paris strike. Teus Hagen chaired the meeting which was attended by Vice Chairman, Michel Gien; the Treasurer, Nigel Martin, who was suffering from a very severe bout of influenza; Philip Peake, Publications Executive; Kim Biel-Nielsen, P.R. Executive; Ernst Janich, Conference Executive; Daniel Karrenberg, Network Executive; and the Business Manager. Unfortunately Neil Todd, the second Conference Executive, was unable to attend this meeting. This did not however prevent him from doing a great deal of work behind the scenes.

As always, a great deal of time was spent on examining the financial position of EUUG, and financial sheets were presented which showed that the present situation was well up to budget and that a healthy operating surplus had been achieved giving reserves almost equal to one year's operating costs.

In order to be truly European a decision had been taken to send out the 1989 subscription invoices in ECUS. These went out to all the National Groups in January but it remains to be seen whether they will find this a convenient currency to use as it is subject to fluctuation and to bank charges. The Executive Committee will be interested to hear the Groups' views on this.

At the last Governing Board Meeting in Portugal the Executive was mandated to implement a system of debtor control that would result in the fair treatment of all groups, since from experience it is known that some pay their subscription invoices promptly, while some pay only after several months. The following scheme was therefor designed and implemented at the beginning of the year.

1. Invoices are issued in accordance with current policy requesting payment within 30 days of date of invoice.
2. Groups will attract a credit of 2.5% of the invoice value if the money is received within this time, the credit being offset against future invoices.

3. For payment between 30 and 60 days the invoice remains at the amount stated.

4. After 60 days 2.5% per month interest will be charged from the date of the initial invoice on the outstanding balance.

As the aim is fairness to all Groups and not the general benefit of the EUUG, any interest received will be offset against discounts and paid back to the Groups in the form of a credit note against their October invoices.

It was noted that some Groups had still not paid their 1988 October invoices, but on the whole the Secretariat was congratulated for having done an excellent job on debt collection.

The Committee also deliberated on new financial control measures which would make each officer responsible for the expenditure of the allocated budget under his control with a purchase order system being operated from Owles Hall.

The Committee greatly welcomed the return of the German Group, GUUG, to the EUUG and agreed that, as a gesture of goodwill, the last issue of the 1988 EUUG Newsletter should be sent free of charge to all the members of the German Group along with a welcoming letter.

The Portugal Conference was reviewed and it was thought to be a great success with approximately 240 delegates at the Conference and some 270 members attending the Tutorials. Although not all the bills had yet been received it was forecast that the event will have generated a sizeable surplus.

Thereafter the Conference Officers reported on the Brussels Conference on the 3-7th April 1989, and the Vienna Conference on 18-22nd September 1989, both of which were well advanced. Fill details of the Brussels Conference appear elsewhere in this newsletter. The conference Officers have also already started work on the Munich Conference on 23-27th April 1990 and the French Conference in Nice on 15-19th October 1990. Plans are also afoot for a Conference on Norway in Spring 1991.

The executive Committee then turned its attention to publications and learned that the Publication Executive, Plilip Peake, was looking into the possibilities of increasing the Newsletter to six issues per year. Exchanges with other publications were discussed, but it was noted with regret that the sale of the 4.3 BSD Manuals, now stocked by the EUUG Secretariat was disappointingly low. These well produced glossy manuals sell at £60 per set of seven including postage and packing and are available on order from Owles Hall for immediate delivery.

The main item under Public Relations was the decision to order an EUUG portable stand which could be used at various exhibitions and events. A budget of £3000 was set aside for the provision of a suitable stand which could also be loaned to the National Groups for display at their events. A slide show describing the EUUG had already been compiled by Kim Beil-Nielsen.

Daniel Karrenberg impressed the meeting by tabling a draft of the EUnet Directory, a document which had taken a great deal of work but will surely prove very worthwhile and already a great many forward orders have been received for it. It is now going into production and will be available soon at a cost of £18.

The final session of this very busy meeting, which went on all day with only the briefest break for a light lunch, was devoted to the planning of the EUUG Workshop to be held in Dublin on 5-7 May this year. The purpose of the Workshop is to discuss objectives, strategies and policies to go forward to the Governing Board. It will look at existing services to see how they can be improved and consider what new ones might be beneficial.

The next meeting of the Executive Committee will be held on 13th February in Amsterdam. The policy continues to try to meet each time in a different country, but all possible efforts are made to keep meeting costs to a minimum.

# Dutch User Group Report

*Frances Brazier*
*frances@psy.vu.nl*

*Department of Cognitive Psychology,*
*Vrije Universiteit,*
*Amsterdam,*
*The Netherlands*

Frances has a master's degree in Mathematics and Computer Science, and has been doing research at the Department of Cognitive Psychology for the past 7 years. Human-machine interfaces and information retrieval are her major fields of interest.

## News from the Netherlands

### Conferences

Our last conference was a success! Never before have we been confronted with the problem of just not having enough room. The programme included tutorials, technical presentations and an exhibition, all open to all participants. Approximately 300 people attended the conference, including members of the BUUG, DKUUG, FUUG, UKUUG and GUUG. Our first experience with the publication of proceedings was well received.

At the moment the NLUUG is busy preparing for our next conference, May 9th, on human-machine interfaces. As the programme is still not definite, interested readers should either read news or contact the NLUUG at a later date. Our experience with 'foreign' (non-Dutch) speakers at our last conference on standards, was promising. Not because the speakers were not able to speak Dutch but because they were able to provide first-hand information. This does not imply that the Dutch speakers were any less qualified or less interesting – in contrary – but additional contributions from 'abroad' were well appreciated. Finding good speakers willing to give a presentation at a level which can be appreciated by most of the audience, is difficult. But I know we're not the only group or organisation with this problem.

### The backbone

Since the first of January *mcvax* is free of the burdon of being both the Dutch and the European backbone. The new Dutch backbone, *hp4nl* is now responsible for all Dutch mail and news. Although still stationed at the CWI, the NLUUG soon hopes to take over its maintenance and support, at least financially. We are currently in the process of looking for candidates.

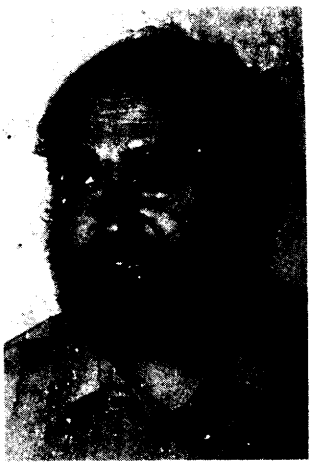
### Rumour

Rumour has it that the crypt competition will be of a more impressive calibre than originally planned. Time will tell.

# AFUU Diary

*Philip Peake*
*philip@axis.fr*

*Axis Digital, Paris*



## November 1988

Last week (18th of November) was the AFUU AGM. Since the AFUU constitution states that governing board members are elected for a period of three years, and that they cannot re-present themselves for election for at least one year, I thought that I had written my last article for the AFUU slot in the EUUG newsletter.

Alas, it seems that I was mistaken. Due to a postal strike in France, virtually no-one had received their voting papers, and so an EGM (because we didn't attain the quorum for the AGM – it gets difficult with 700 members!) decided that the existing governing board should rest in place until the postal strike is over, and we can re-start the election process.

So, once again, I have to face the prospect of Alain Williams printing that *horrible* photograph that he keeps just to annoy me ...

This election is more interesting than those of the past, because, for the first time we have more candidates than posts available. 17 candidates for 7 posts.

In some ways it is unfortunate that our constitution limits the number of members of the governing board, because we certainly have work for as many people as we can find.

Apart from the problems of voting, the AFUU meeting seems to have been successful. There were presentations from each of the working groups, explaining what the group does, and what results they have generated so far. It was unfortunate that all had to fit into one day, and so some of the presentations were in parallel, meaning that people had to decide which presentation to miss.

The AGM took place in Grenoble, taking advantage of the UNIX exhibition and conference organised by 'The UNIX club of Grenoble'. This conference took place on the 17th and 18th of November.

Now we have to start preparations for the *Convention UNIX '89*, which takes place in Paris at the end of February. As well as this, we have a certain amount of work to to to help the EUUG with preparations for the conference which is due to take place in the south of France in autumn 1990.

When the elections are over, and the new members of the governing board are installed, I wish them good luck, they have lots of work already lined up for them ...

## January 1989

On the 5th of January, we have (at last!) held our EGM, and I can safely say that I am no longer responsible for writing the AFUU page for the newsletter; but since I had already started ...

The ballot was carried out by post, and for the first time, we had a computer program to perform the counting, ordering and production of statistics – thanks to Michel Sutter for that.

Being seasoned computer users, there was also a parallel manual count ... Surprising as it may seem, both systems agreed!

The governing board of the AFUU (Conseil d'Administration or CA as it is called in French) is as follows:

| Name | Organisation | Responsibility | Duration |
|---|---|---|---|
| Jean-Luc Bellet | AT&T | | 3 years |
| Jean-Louis Bernard | Consultant | | 1 year |
| Christophe Binot | Hewlett-Packard | Vice President | 3 years |
| Nicole Blaine | Sun Microsystems | | 3 years |
| Matthew Capril | Bull | | 3 years |
| Anne Francois | Hommes & Communications | | 1 year |
| Jaques Guidon | Agence National du Logiciel | | 3 years |
| Dominique Maisonneuve | Sligos | President | 3 years |
| Veronique Mansart | Dataware | | 2 years |
| Pierre-Louis Neumann | Inria | Tresurer | 2 years |
| Jean-Christophe Petithory | University (Paris VIII) | | 3 years |
| Jean-Jaques Rousset | Inforama | | 2 years |
| Alain Saint-Patrice | Comtec | | 2 years |
| Michel Sutter | Matra Datasystems | Secretary | 2 years |
| Philippe Vaudou | Individual (user) | | 1 year |

As you can see from the *duration* column, the scheme of retiring one third of the board each year is now fully in effect (the *duration* column indicating how much time the various people have left to serve – those with 3 years are the people just elected).

The 'executive' (or Bureau) is elected by the board each year, congratulations to Dominique Maisonneuve on being elected President for a second time!

# UKUUG News

*Mick Farmer*
*<mick@cs.bbk.ac.uk>*

*Department of Computer Science*
*Birkbeck College*

Mick is a lecturer at Birkbeck College (University of London) and the Secretary of UKUUG. He has just completed a video training course on the C programming language and is starting one on UNIX. His interest is in all aspects of Distance Learning and he is the Senior Consultant (Software) for LIVE-NET, an interactive video network connecting London's colleges. He is also a member of the University's VLSI Consortium, mainly because the design tools draw such pretty pictures.

## Reorganisation

The UKUUG Committee was recently restructured into an Executive Committee and an Advisory Committee. The Executive Committee consists of the elected officers (Chair, Secretary, and Treasurer) and the Advisory Committee consists of those helping in various ventures such as Meetings, Workshops, and Networking. Collectively, we can be reached as *ukuug@ukc.ac.uk*.

## UKUUG and UKnet Winter Technical Meeting

This was held over two days on December 19/20, 1988 at the University of Kent. The annual UKnet meeting needed some quick scene shifts, as we had a lot of good things to get into one afternoon and one morning. The reason for this was the lack of a summer UKUUG meeting, because of the EUUG meeting in London. We therefore decided to start with a UKUUG afternoon featuring several talks previously given at EUUG and USENIX meetings, along with a couple of new talks.

The annual UKUUG business meeting followed. Sunil Das (Chair) and Mick Farmer (Secretary) were re-elected. Zdrav Podolski (Treasurer) reported on last year's accounts and this year's projected expenditure. His report was approved.

The evening session followed the usual *Bar, Dinner, Bar*, meeting schedule.

Tuesday started early as we had a lot of network talks to get in followed by a *UKUUCP* tutorial in the afternoon.

The general feedback suggested that most people enjoyed themselves and found the talks interesting. The tutorial was particularly popular with Lee McLoughlin even making UUCP intelligible to us, which shows the high level of presentation. All in all it was an enjoyable and, we hope, worthwhile two days.

## UNIX Security Workshop

We are holding a Security Workshop on 16 February 1989 at the Institute of Education, London WC1. It provides UNIX System Administrators with an insight into the security problems relating to UNIX systems, and will act as a forum to share knowledge and experience.

## UKUUG Summer Meeting

The UKUUG Summer meeting will be held at Strathclyde University, Gl... on June 27/28. Details are being finalised, but we hope that the meeting will reflect some of the interesting things being done with UNIX by people in the West of

Scotland. There may be a small exhibition.

Following the success of the UKUUCP tutorial at the Winter Technical meeting, the UKUUG is considering running tutorials at this meeting. Suggestions for tutorials and speakers can be made to the UKUUG(*ukuug@ukc*) or to the local organiser.

Local arrangements are being made by:

Jim Reid
Dept of Computer Science
Strathclyde University
Glasgow G1 1XH
United Kingdom
Phone: (+44) 41 552 4400 x3319
Email: jim@cs.strath.ac.uk

## London UNIX User Group (LUUG)

The LUUG lecture series is now well-established. The first two lectures by Steve Kille on *ISODE*, and Eddie Bleasdale on *CUE* were very popular.

The 1989 programme starts with a security discussion on 26 January. February's topic, *Making software installation easier*, is another discussion meeting, which has the potential to turn into a major project. Dominic Dunlop will be talking about *Technological red herrings* on 30 March. Ray Anderson will de-mystify *X-windows* on 27 April. Jim Crammond is the speaker on May 25, when he will describe the new *UK-Sendmail* configuration package. Beyond that? Well, watch **uk.events** for details!

## Glasgow UNIX User Group (GLUG)

This local user group is hoping to have its inaugural meeting sometime in February 1988. We wish them well.

That's all for now folks!

Mick Farmer
Secretary, UKUUG

# EUUG

European UNIX® systems User Group

## Spring '89

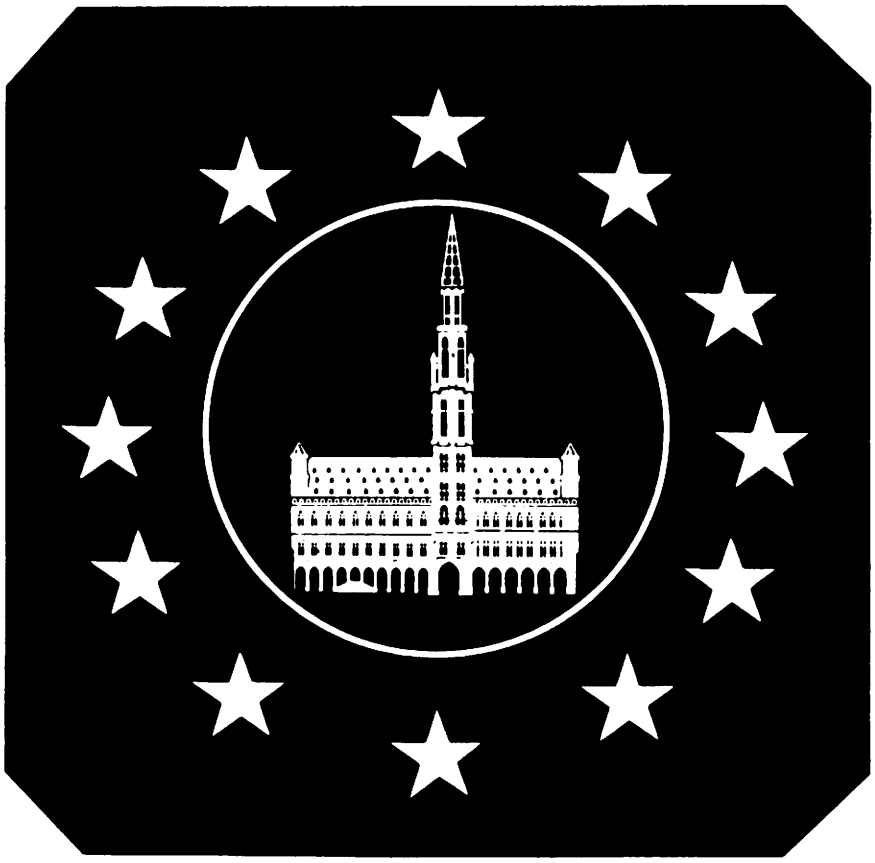
# CONFERENCE

## UNIX: European Challenges

**3-7 April 1989**

**at**

## Palais des Congrès
## Brussels



# EUUG Spring Conference

# 1989

## UNIX: European Challenges

The European UNIX systems User Group is a non-profit making organisation which exists to provide a body representing the users of UNIX and UNIX-like systems in European countries.

The EUUG membership is comprised primarily of institutional and individual members of affiliated National Groups. There are, at present, 14 National Groups with a total membership in excess of 3,500 coming from such diverse backgrounds as software engineering, information technology, computer manufacturers, end-users, software houses, universities and research centres. The EUUG runs two **major** conferences each year on a European level.

The Spring 89 Conference will be held in Belgium in conjunction with the Belgium UNIX systems User Group (BUUG) at the Palais des Congrès, Coudenberg 3, 1000 Brussels, The Technical Sessions will be held on 3, 4 and 5 April and the Tutorials will be held on 6 and 7 April. The event will be accompanied by a major Exhibition.

Conference & Tutorial Enquiries

**Mrs Helen Gibbons**
**European UNIX systems User Group**
**Owles Hall, Buntingford**
**Hertfordshire, SG9 9PL**
**England**

**tel: + 44 763 73039**
**fax: + 44 763 73255**
**email: euug@inset.uucp**

Exhibition Enquiries

**Mr Jean Destrée**
**Congress Service**
**Palais des Congrès**
**Coudenberg 3**
**1000 Brussels**
**Belgium**

**tel: + 32 2 513 4130**
**telex: 25919**

# PROGRAMME OF EVENTS

## Provisional Technical Programme

## MONDAY APRIL 3rd

### Opening Session (9:30-10:30)

Welcoming Address
*Marc Nyssen (Belgium)*
*Free University Brussels, Medical Informatics Dept.*

Object Oriented Program Development of "Making Hacking Respectable"
*Dennis Tsichritzis (Switzerland)*
*Centre Universitaire d' Informatique, Genève*

### Distributed O.S. Developments (11:00-12:30)

UNIX and Load Balancing a: Survey
*C. Jacqmot. E. Milgrom (Belgium)*
*Université Catholique de Louvain, Unité d' Informatique*

Amoeba — High-Performance Distributed Computing
*Sape J. Mullender (Netherlands)*
*Centre for Mathematics & Computer Science, Amsterdam*

Implementation of an Event Distribution Mechanism (EDM) on a Network of Workstations
*Martin Zellweger, Peter Gloor (Switzerland)*
*University of Zurich, Institut fuer Informatik*

### Networks (14:00-15:30)

EUnet: an overview
*Daniel Karrenberg (Federal Republic of Germany)*
*Centrum voor Wiskunde en Informatica, Amsterdam*

Recent changes in North American Computer Networks
*John S. Quarterman (U.S.A.)*
*Texas Internet Consulting, Austin*

The EDUNET-Project
*Manfred Wöhrl (Austria)*
*Higher Level Secondary Industrial School, Vienna*

### Graphics (16:00-17:30)

Extending User Interface Toolkits for Picture Processing
*S.T. Jones, W.L. Franklin (U.K)*
*University of Essex, Dept. of Electronic Systems Engineering*

The Design of a UNIX Workstation Environment for Medical Image Processing
*Dirk Tombeur, Frank Deconinck (Belgium)*
*Free University Brussels, Experimental Medical Imaging Lab.*

Occursus cum novo — Realistic computer Animation on a UNIX Network
*Wolfgang Leister, Achim Stösser, Burkhard Neidecker, H. Müller*
*University of Karlsruhe, Dept. of Computer Science (F.R.G.)*

Designing A Virtual Toolkit for Portability Between Window Systems
*Marc J. Rochkind, Carol J. Meier (U.S.A.)*
*Advanced Programming Institute Ltd., Boulder Colorado*

# TUESDAY APRIL 4th

## Network Software (9:00-10:30)

Transport Interfaces in Sinix Systems
*Norbert Richter (Federal Republic of Germany)*
*Siemens A.G., K D ST DF154 Munich*

Implementing Internet Protocols on a small UNIX System
*Danny Backx, R. Derkinderen, M. Snyers, P. Verbaeten (Belgium)*
*Katholieke Universiteit Leuven, Dept. Computerwetenschappen*

Striping Network Device Driver for TCP/IP
*John A. Pew (U.S.A.)*
*GE Aerospace, NASA Ames Research Center*

## Trends and Security (11:00-12:30)

More Haste, Less Speed
*David Rosenthal (U.K.)*
*Sun Microsystems, Mountain View, California*

Authentication in a UNIX Network Using Smart Cards
*S.T. Jones, M.J. Clark (U.K.)*
*University of Essex, Dept. Electronic Systems Engineering*

Into the Padded Cell
*D. Griffiths, Steve Hinde, A. McPherson, A. Standford (U.K.)*
*Information Technology plc, Hemel Hempstead*

## UNIX "Standards" (14:00-15:30)

UNIX Standardisation: an Overview
*Cornelia Boldyreff (U.K.)*
*Dept. Computer Science, Brunel University, Uxbridge*

System V release 4
*Robert L. Duncanson (U.S.A.)*
*AT&T Unix Europe Limited, London*

BSD 4.4
*Steve Bostick, M.K. McKusick (U.S.A.)*
*University of California Berkeley*

## UNIX "Standards" (16:00-17:30)

Computing Policies and the European Commission
*Walter De Backer (Belgium)*
*Commission of the European Communities, Luxemburg*

X-OPEN
*John Totman (U.K)*
*X/Open Company Limited, Reading*

O.S.F.
*Henning Oldenburg (Federal Republic of Germany)*
*Open Software Foundation, Brussels*

# WEDNESDAY APRIL 5th

## General O.S. Developments (9:30-11:00)

NFS refreshes the filesystem(s) A/UX cannot reach
*Matthew Kempthorne-Ley-Edwards, William Roberts (U.K.)*
*Queen Mary College, London*

A Contiguous High Performance File System
*Shigetoshi Yokoyama, Shin-ichi Yamada, Takehiko Nishiyama, Kazuo Ito (Japan)*
*NTT Data Communications Systems Corp., Kanagawa*

Dp: a System for Inter-Program Communications
*Robin Faichney (U.K)*
*University of Kent, Computing Laboratory*

MINIX Portability and the Atari ST
*Sunil K. Das, Aarron Gull (U.K.)*
*City University London, Computer Science Dept.*

## Tools (11:00-12:30)

UNIX Tools
*Brian Kernighan (U.S.A.)*
*AT&T Bell Labs, Murray Hill, N.J.*

A Model-Based Diagnostic System for the UNIX Operating System
*Gail Anderson (U.K.)*
*University of Edinburgh, A.I. Applications Institute*

## Languages (14:00-15:30)

SCOOP: a Software environment for C + + Object Oriented Programming
*P. Fontaine, Ch. Masson, L. Stefanelli (Belgium)*
*Intecs International s.a., Brussels*

Step-by-step Transition from dusty-deck FORTRAN to Object Oriented Programming
*Michel Bardiaux, Philippe Delhaise (Belgium)*
*Plant Genetic Systems N.V., Brussels*

Distributed Logic Programming
*Andreas Krall, eva Kühn, Ulrich Neumerkel (Austria)*
*Technische Universität Wien, Institut für Praktische Informatik*

## Tools-2 (16:00-17:00)

Performance Evaluation: The SSBA at AFUU
*Christophe Binot, Philippe Dax, Nhuan Doduc (France)*
*Université de Valenciennes*

An Interactive Spelling Checker for use with UNIX Systems
*Sunil K. Das, Philip M. Sleat (U.K.)*
*City University London, Computer Science Dept.*

Context-Reflecting Pictures of a Database
*Agnes Hernadi, Zalan Bodo, Elod Knuth (Hungary)*
*Hungarian Academy of Sciences, Computer & Automation Institute, Budapest*

***Although all details are correct at the time of going to press, the EUUG reserves the right to change any item on the technical and tutorial programme as required.***

# TUTORIALS

Only EUUG National Group or Direct Members are permitted to attend Tutorials

Each Tutorial lasts for one whole day and will start at 09.30

# THURSDAY, APRIL 6th

## Tutorial T1 Programming in ANSI C
### *Tutor: Carol J. Meier, UNIX & C Consulting Services*

This tutorial takes a practical look at a number of C topics. Students will learn how to use pointers to functions, write functions with a variable number of arguments, use setjmp and longjmp, use conditional compilation to aid in portability and debugging, parse command line arguments and access environment variables, write typedefs for complex declarations, use UNIX system calls, create subprocesses with fork and exec, and set up pipes between two processes. The tutorial will also cover ANSI C function prototypes, dynamic memory allocation, enumerations, multidimensional arrays and pointers, bit fields and unions, and new ANSI preprocessor features.

Most of these features will be explained by way of examples taken from production programs and/or systems headers and libraries. The tutorial is orientated towards a UNIX environment, although portability is stressed.

This tutorial is aimed at people who have been programming in C to the point where they are comfortable with the basics and are ready to explore some of its more powerful features. It assumes a programming knowledge of C basics. Attendees should understand the basic data types, control flow statements, and use of functions. Familiarity with preprocessor constant replacement, simple arrays and structures is assumed. However, more advanced use of the preprocessor and more complex data types will be covered.

*After receiving B.S. and M.S. degrees in Computer Science from the University of Pittsburgh Carol Meier worked as a computer scientist at Bell Telephone Laboratories. From 1983 through 1988 Carol worked as an independent consultant presenting hundreds of public and on-site UNIX and C professional development seminars, and providing software development and consulting on a contract basis for a variety of companies. Ms. Meier developed and regularly teaches a series of courses for the University of Colarado, and she presents tutorials at Uniforum, Usenix and EUUG conferences.*

*In her current position as Vice President of the Advanced Programming Institute, a company specializing in advanced user interface management software, she is responsible for the implementation of a virtual user interface toolkit that allows applications to be portable across a variety of popular windowing systems.*

## Tutorial T2 — System V Device Drivers
### *Tutor: Dan Klein, Carnegie-Mellon University*

Delegates attending this tutorial need to be licensed to view the System V Source Code

This tutorial will cover the major aspects of driver design, implementation, and device integration. Both DMA and programmed I/O device drivers will be covered, as well as block and character (buffered and unbuffered) interfaces. It will outline the design and implementation of structured I/O devices (i.e. disk drives), and semi-structured devices (i.e. tape drivers and serial communications links). The tutorial will also discuss all aspects of adding a new device to the kernel (i.e. auto configuration, special files, device tables, and debugging). Although the tutorial will be geared towards System V, a comparison between the Berkeley and Bell Laboratories approaches will be offered.

This tutorial is designed for people who wish to become familiar with the fundamentals of designing UNIX device drivers. It is intended for systems programmers who will be actively engaged in the maintenance or design and implementation of UNIX device drivers.

*Dan Klein has been a teacher for the Institute of Advanced Professional Studies in Cambridge, MA, served as an independent consultant and has developed an interactive computer based training generator.*

## Tutorial T3 — Writing ONC Distributed Applications with NFS as a Case Study

### *Tutors: Sally Ahnger & Mark Stein, Sun Microsystems*

There has been much discussion recently about Network Computing and the various implementations and strategies to bring it about. Sun's Open Network Computing (ONC) environment is a public domain programming platform which speeds development of network-based applications. ONC applications use the Remote Procedure Call (RPC) protocol and External Data Representation (XDR) standard to communicate between heterogeneous machines on the network. The Network File System (NFS) is a distributed file sharing service based on the ONC platform.

This tutorial is aimed at software professionals who would like to learn more about the technical details of the ONC platform. It will discuss the underlying RPC/XDR architecture and how to write network applications using this platform and the RPC protocol compiler. In addition, it will explore the architecture and implementations of some existing ONC services (NFS, Mount service, Yellow Pages data lookup service, and file locking service).

**ONC and NFS are trademarks of Sun Microsystems, Inc.**

*The instructors have been members of the Open Network Computing Software Engineering department at Sun Microsystems for over three years. They have extensive experience with the ONC/NFS vendor community, which includes developing the ONC reference implementations and conducting classes and seminars.*

*Mark Stein is a Staff Engineer in the Portable ONC group and works on implementations of ONC for platforms other than Sun workstations. He received an M.S. in Engineering-Economic Systems from Stanford University and a B.S. in Electrical Engineering from Virginia Tech.*

*Sally Ahnger is the manager of the ONC Network Applications group, which develops new network services based on RPC. She holds M.S. degrees in Computer Engineering and Engineering Management from Stanford University, and a B.S. in Computer Science from the University of Illinois.*

## Tutorial T4 — Xlib

### *Tutors: James Watson, Adasoft with Dr Charles Clanton & Patricia Caruthers*

The focus of this tutorial is on programming with the X Window System widget sets. Three widget sets are provided on version 11 release 3 of the M.I.T. distribution of the X Window System, each based on the Xt intrinsics. These widget sets will be explained and compared. Then, simple applications using these widgets will be examined in detail. Along the way, a number of other useful facilities of the X Window System for application programmers will be explained, including context management, selections, properties, the resource database, and translation management.

The tutorial is intended for programmers who will be developing applications or utilities for the X Window System. Attendees should be experienced C programmers with knowledge of windowing systems concepts in general and at least some exposure to the X Window System as a user. General concepts of the X Window System and Xlib programming will only be presented as needed to understand the widgets.

On completing the tutorial attendees will be able to begin writing their own programs with the widget sets to continue their learning.

*James A. Watson is a senior UNIX application programmer with extensive experience consulting on UNIX and the X Window System. He has taught numerous classes on both in Europe. He started working with the X Window System Version 10, Release 3.*

*Dr. Charles Clanton has been doing system and applications development under UNIX since the mid-1970's. He specializes in the design and development of high-technology software and in user interface design for applications.*

*Patricia Caruthers is a senior UNIX system and application programmer who has been involved in UNIX for a decade and the X Window System code since version 10 release 4. Although she defers to her less shy colleagues in formal presentations, past classes have found her comments and answers to the hard questions extremely valuable.*

# FRIDAY, APRIL 7th

## Tutorial F5 — UNIX File System Administration

### *Tutor: Hendrik-Jan Thomassen, AT Computing bv*

The Fsck command has automated the old guru-type system administrators from the 7th Edition UNIX years. Despite the disappearance of this species, it may still be useful for current-day system administrators to know what is going on behind the scenes of a file system.

The tutorial focusses on the organisation of a file system on disk : the functionality of a disk device driver, kernel block buffering, block and character drivers (also for magtapes), major and minor number schemes, disk special file names, and disk partitioning.

Then the file sytem is explained from both a user point of view, and a system implementation point of view. Topics covered are: the implementation of directories, file links, symbolic links, filename parsing, i-nodes, the superblock, free diskspace book keeping, indirection levels of block access, and phantom blocks.

Building upon this, it is demonstrated why and how several maintenance programs do their work: ncheck, fsck, mkfs, clri.

Finally, backup strategies and file system tuning are discussed.

The tutorial is aimed at medium-experienced UNIX system administrators. It covers mainly the AT&T file system design, with examples also drawn from the BSD (McKusick) implementation and from modifications made by several vendors. Network file systems will not be covered.

*Hendrik-Jan Thomassen is the founder of AT Computing in the Netherlands. He is currently working on a book on UNIX System administration. He was formerly employed at the University of Nijmegen and was a member of the Governing Board of EUUG.*

## Tutorial F6 — Programming International Applications

### *Tutor: Matthew Caprile, Bull MTS*

This tutorial discusses the issues involved in developing international applications, based on existing standards. It is orientated towards programmers who wish to write internationalised applications, i.e. independent of country and language. This is NOT a tutorial on localisation (how to kludge programs to run in a particular country). It is assumed that attendees are experienced C programmers, this is not a tutorial on general programming techniques. No prior experience with internationalisation is necessary.

Topics covered:

- Historical context (where it all began)
  - — localisation vs. internationalisation
- European environment
  - — what must be provided
  - — what must be supported
  - — periphery (terminals, printers, connectivity)
- Today's standards for internationalisation
  - — ANSI C
  - — IEEE 1003.1 (POSIX)
  - — X/Open Portability Guide
- Programming interfaces (C binding)
- Programming advice (Don't use that 8th bit...)
- User interface
  - — how the user can specify his/her needs

*Matthew Caprile is head of the internationalisation development team at Bull. After graduating from the University of California in 1984, he moved to France, working as system software engineer, where he was quickly faced with the problems of getting American software to run correctly in a French environment.*
*Since joining Bull in 1986, he has been working towards solving the problems encountered in writing international applications. Mr Caprile is Bull's representative in the X/Open internationalisation working group, as well as vice chairman of the /usr/group technical subcommittee on internationalisation.*

## Tutorial F7 — UUCP & Sendmail Configuration

### *Tutor: Yves Devillers, INRIA*

This tutorial is devoted to advanced features in sendmail in the framework of EUnet, intended to make users trying to modify or customise their configuration file feel more comfortable. It will focus on the routing model and debugging methods. Topics covered are:

Quick scan through definitions:
- macro, config files, frozen conf (and hidden effects)
- mail header, unix mail, rfc822
- user mail interface (berk-mail, mh, ...)
- mail delivery agent (uux, binmail, sh, smtp, others)
- addressing, envelope

How does sendmail work:
- mail internet model, routing, local peculiarities
- address analysis and rewriting, routing consideration
- interactions between sendmail and user interface or delivery agent.

Using sendmail
- installing, config files
- what is in logs
- interpreting headers
- debugging modes, interpreting output, local and remote debuging
- test scenarios
- caveats and pitfalls

Case study of some configuration files
- standard or national files
- local customisation (cluster of workstations, gateway, ...)

Extra features
- recent versions of sendmail
- using name-servers
- IDA enhancement
- understanding other mail standards (Earn, X400)

*Yves Devillers is presently working with INRIA where he is managing the French backbone for EUnet and gateways to several other mail networks.*

## Tutorial F8 — Open Look Graphical User Interface

### *Tutor: Chris Schoettle, AT&T UNIX Europe*

The OPEN LOOK™ Graphical User Interface provides a standard user interface across many UNIX® system hardware platforms. OPEN LOOK incorporates conventions found in other user interfaces and improves upon existing component designs. This design provides a functional balance of simplicity, consistency, and efficiency to meet the needs of all users. OPEN LOOK uses and complements the simplicity, power, and openness found in UNIX System V; for example, it operates efficiently in a multitasking and networking environment.

OPEN LOOK has a specific Look (visual) and Feel (operation) that gives all its applications a consistent appearance. The OPEN LOOK Graphical User Interface is a specification and style guide, not a software program. UNIX System V Release 4.0 will provide two OPEN LOOK Toolkits with the components necessary to build applications with the OPEN LOOK "Look and Feel".

The OPEN LOOK X Toolkit is based upon the X Window System (from MIT). The OPEN LOOK NDE (NeWS Development Environment) Toolkit is based upon the NeWS graphical platform (from Sun Microsystems). These toolkits will be "object orientated", providing pre-programmed components to facilitate the building of a user interface. UNIX System V Release 4.0 provides a merged window system based on X11 (X Version 11) and NeWS. This is a client-server based window system that is compatible with applications developed for X11 or NeWS.

Although the emphasis of this tutorial is the OPEN LOOK Graphical User Interface, the entire UNIX System V Graphical User Interface Architecture will be discussed.

*Chris Schoettle is a senior consultant with AT&T UNIX Europe, based in London, England. Mr. Schoettle has Bachelor and Master of Science degrees in computer science and has worked in the international UNIX© systems market for the last two years. Mr. Schoettle is currently part of a multinational AT&T development team investigating the internationalisation issues involved in the OPEN LOOK™ Graphical User Interface, the OPEN LOOK X Toolkit, and the X Window System.*

# CONFERENCE DINNER

## TUESDAY, APRIL 4th

The cost of the Conference dinner is included in the price of the conference and will be held on 4th April at 20.00 hours at the:

**AUTOWORLD MUSEUM (Palais Mondial de l'Automobile) Parc du Cinquantenaire 11, B1040, Brussels.**
**Phone + 32 2 736 41 65**

AUTOWORLD is the largest veteran car exhibition in Europe, if not in the world. The history of the motor car from 1896 up to the 1960s is represented in the exceptional setting of the Palais Mondial, which is part of the Cinquantenaire architectural building complex. The latter was the scene for a number of international exhibitions between 1880 and 1934.

The bulk of the vintage cars on display belong to the Mahy and the De Pauw collections, with among them a unique Ford model T, a Rachet Schneider, and several other rare American and European vehicles.

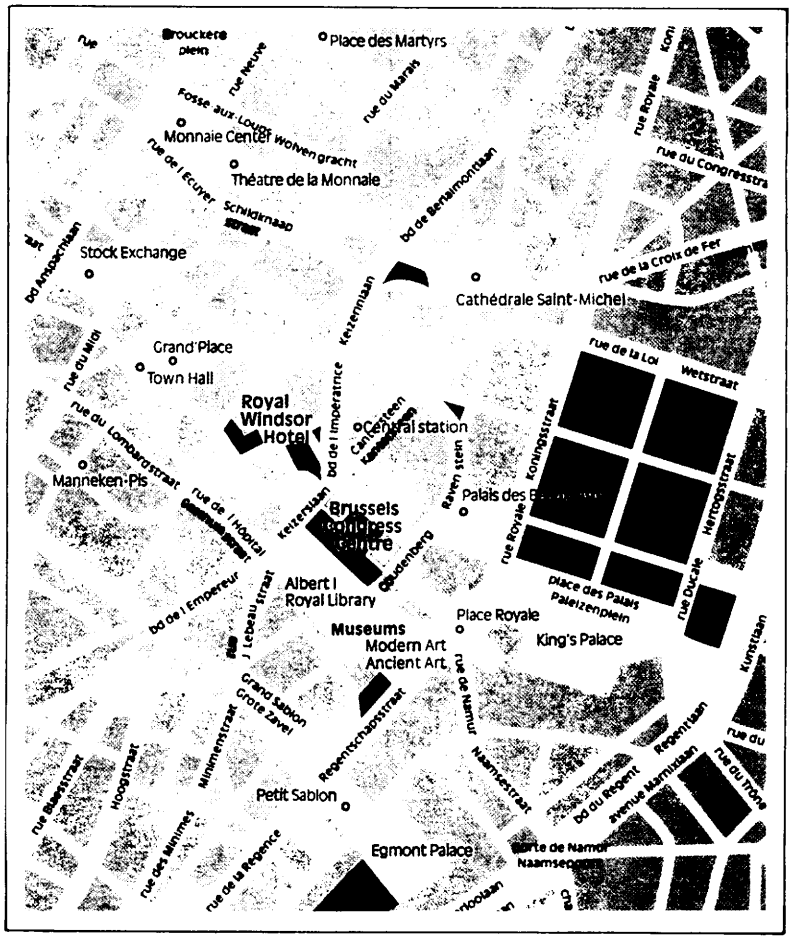
Delegates who wish to take their partners to the conference dinner may purchase an extra invitation at a cost of £25 sterling from the Registration Desk on Monday, 3rd April.

# VENUE AND ACCOMMODATION

The venue for the Conference, Tutorials and Exhibition will be:

**The Palais des Congrès**
**Coudenberg 3**
**B-1000 Brussels, Belgium**
**Phone: + 32 2 513 4130**
**Telex: 25919**
**Fax: + 32 2 514 2112**

The Palais des Congrès is situated in the heart of Brussels on the "Mont des Arts" and lies between the Upper Town with its shopping district, and the Lower Town, close to the Grand Place with its numerous restaurants.



The Congress Centre is approximately 12kms from Brussels Airport. Please note that fares and travelling times are approximate.

**Means of transport:**

taxi: BFr 1,000/journey time 30 minutes.
train: BFr 70 2nd class — 100 1st class/ journey time 20 minutes

The train leaves from the airport building and goes to the Central station. From there it is a 3 minutes walk to the Centre.

The conference hotel will be the:
**New Hotel Siru, Place Rogier Plein, 1210 Brussels, Belgium**
**Phone: + 32 2 217 75 80**
**Telex: 21722**

Special rates have been arranged for EUUG conference delegates of BFr 2,800 for a single room with breakfast and BFr 3,600 for a double room with breakfast.

The distance between the New Hotel Siru and the Congress Centre is approximately 1.5kms.

**Means of transport:**

taxi: BFr 400/journey time 10 minutes
metro: BFr 40/journey time 20 minutes

There is a metro station in front of the hotel (Place Rogier), with a direct connection to Place Brouckère; change and then direct line to Central Station.

Bookings may be made direct with the Hotel quoting EUUG. For those delegates who whish to book accommodation elsewhere, this may be done through our agent BBL Travel by using the enclosed reservation form. This may also be used to book for excursions.

The address for BBL Travel is:

**Cours Saint-Michel 60**
**1040 Brussels, Belgium**
**Phone: + 32 2 738 4536**
**Telex: 26255**

## THE EXHIBITION

The Exhibition will be held in the Leopold II Exhibition Hall at the Palais des Congrès on 3, 4, and 5th April 1989 and will be open to delegates throughout the conference.

Those companies wishing to book stands should contact the exhibition organiser:

**Mr Jean Destrée**
**Palais des Congrès**
**Coudenberg 3**
**1000 Brussels, Belgium**
**Phone: + 32 2 513 4130**
**Telex: 25919**
**Fax: + 32 2 514 21 12**

Stands will measure 3x4m and will cost £1100 sterling if booked before 1st March and £1300 if booked after that date.

## TAPE DISTRIBUTION

The EUUG will provide a tape distribution service at the Brussels conference. The tape will contain an assortment of public domain software extracted from the network and software donated by EUUG members. The tape will cost £35 for collection at the conference and may be ordered by ticking the box on the booking form.
**Only EUUG National Group or direct members are permitted to order a tape.**

## CANCELLATIONS

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

## LANGUAGE

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

## LIABILITY

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

# STUDENT GRANTS

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

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

Priority will be given to:

**1. Students giving a talk at the Conference**

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

**3. Students**

**4. Other deserving cases like research students.**

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

# HOW TO BOOK

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

The Secretariat
European UNIX systems User Group
Owles Hall, Buntingford
Hertfordshire SG9 9PL, UK

phone: +44 763 73039
fax: +44 763 73255
email: euug@inset.uucp
or ...!mcvax!ukc!inset!euug

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

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

Payments may be made in one of 3 ways.

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

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

   The Bank of Scotland
   61 Grassmarket
   Edinburgh
   Scotland EH1 2JF

   Account Number: 00613997
   Bank Sorting Code: 80-31-50

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

3. By VISA, ACCESS, EUROCARD or MASTERCARD, with card details appearing on the Booking Form.

NOTE: Closing date for all bookings is 28 March.

# COSTS

| ***EUUG national group or direct members*** | | Cost if booked and paid for | | |
|---|---|---|---|---|
| | | before 1 March | after 28 Feb | on door |
| 3 day conference | 1 person | £240 | £270 | £300 |
| 1 day tutorial | 1 person | £180 | £205 | £220 |
| ***Non-members*** | | | | |
| 3 day conference | 1 person | £290 | £320 | £350 |

# Booking Form for Conference and Tutorials

Please complete this form and send it, with cheque or evidence of payment, to **EUUG Secretariat, Owles Hall, Buntingford, Herts SG9 9PL, UK** (Block capitals please). Please note that forms sent without cheque or evidence of payment will be returned to you unregistered.

**Surname** _____ **Usual first name** _____

**Company/Organisation** _____

**Address** _____

**Country** _____ **Post/Zip Code** _____

**Telephone/Fax/Telex/Email** _____

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

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

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

**TUTORIALS** (members only)
Please reserve me a place for Tutorial No_____on Thursday 6 April ... ... ... £ _____
Please reserve me a place for Tutorial No_____on Friday 7 April ... ... ... £ _____

Do you require Vegetarian meals? Yes ☐ No ☐

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

**TAPE** Please reserve me a copy of the Conference Tape £ _____

Please read the section **"HOW TO BOOK"** Total £ _____

**PAYMENT METHOD**

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

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

☐ by **VISA**

☐ by **ACCESS/EUROCARD/MASTER CARD**

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

**Address of cardholder** _____

_____

_____

**Card Account No.** _____ **Date of Expiry** _____

**Signed** _____ **Date** _____

# STUDENT GRANT APPLICATION FORM

Return to **EUUG Secretariat, Owles Hall, Buntingford, Herts SG9 9PL, UK**
(Block Capitals Please)

**Name** .....

**Address** .....

.....

.....

.....

**Telephone** .....

**EUnet address** .....

**Position** .....

**University/Organisation** .....

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

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

Expenses requested/reclaimed (Give best estimate where necessary)

**Travel** .....

**Accommodation** .....

**Conference fee** .....

**Total amount** .....

**Date** .....

**Signature** .....

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

# Calendar of UNIX Events

This is a combined calendar of planned conferences, workshops, or standards meetings related to the UNIX operating system. Most of this information came from the various conference organizers, although some was taken from ;login: (USENIX), 13, 1, Jan/Feb 1988, CommUNIXations (/usr/group), VII, 6, Nov/Dec 1987, and the /usr/group UNIX Resources Guide.

If you have a UNIX related event that you wish to publicise then contact either John Quarterman at *jsq@longway.tic.com* or Alain Williams at *addw@phcomp.co.uk* giving brief details as you see below.

Abbreviations:

| | |
|---|---|
| C | Conference |
| G, MD | Gaithersburg, Maryland |
| S | Symposium |
| T | Tradeshow |
| U | UNIX |
| W | Workshop |

| year mon days | conference | (sponsor,) (hotel,) location |
|---|---|---|
| 1989 Feb 28-Mar 2 | UniForum | Moscone Center, San Francisco, CA |
| 1989 Feb 28-Mar 3 | U Convention | AFUU, Paris, France |
| 1989 Mar 1-2 | GOSIP Users' W | NIST, G, MD |
| 1989 Mar 2-3 | GUUG W | Saarbrueken, Germany |
| 1989 Mar 30 | LUUG (UKUUG) | 18 Bedford Square, London |
| 1989 Apr 3-7 | EUUG | Palais des Congres, Brussels, Belgium |
| 1989 Apr 10-11 | ANSI X3J11 | Phoenix, AZ |
| 1989 Apr 18-20 | IETF | IAB, (NASA, Kennedy Space Center, FL) |
| 1989 Apr 18-20 | IETF | IAB, (PSC, Pittsburgh, PA) |
| 1989 Apr 24-28 | IEEE 1003 | Minneapolis-St. Paul, MN |
| 1989 Apr 25-27 | ISDN in Europe | IFIP-TC6, ICCC, The Hague, Netherlands |
| 1989 Apr 26-29 | Networking Forum '89 | Sendai/Tokyo, Japan |
| 1989 Apr 27 | LUUG (UKUUG) | 18 Bedford Square, London |
| 1989 Apr | Soft. Management W | USENIX, New Orleans, LA |
| 1989 May | U 8x/etc C&T | /usr/group/cdn, Toronto, ON |
| 1989 May 8-12 | DECUS S | Atlanta, Georgia |
| 1989 May 9 | NLUUG | The Netherlands |
| 1989 May 14-16 | AMIX | Kfar Hamacabia, Israel |
| 1989 May 16 | POSIX Appl. W | NBS, G, MD |
| 1989 May 25-28 | ENA conference | Allentown, PA |
| 1989 May | U 8x/etc C&T | /usr/group/cdn, Toronto, ON |
| 1989 Jun | NZSUGI | New Zealand |
| 1989 Jun 7-9 | Italian U C | i2u, Milan, Italy |
| 1989 Jun 7-9 | COMUNIX | /usr/group/UK, |
| 1989 Jun 7-9 | Eur. U User Show | EMAP Int. Exh., Alexandra Palace, London |
| 1989 Jun 12-16 | USENIX | Hyatt Regency, Baltimore, MD |
| 1989 Jun 19-23 | ICX89 | USM, IEEE, Valparaiso, Chile |
| 1989 Jun 27-28 | UKUUG | Glasgow |
| 1989 Jul 10-12 | 13th JUS UNIX S | Tokyo |
| 1989 Jul 10-14 | IEEE 1003 | San Francisco, CA |

| | | |
|---|---|---|
| 1989 Jul 26-28 | IETF | IAB, Stanford, Stanford CA |
| 1989 Aug 8-11 | AUUG | Hilton Hotel, Sydney |
| 1989 Sep 18-22 | EUUG | WU Vien, Vienna, Austria |
| 1989 Sep 19-22 | ACM SIGCOMM | Austin, TX |
| 1989 Oct 16-20 | IEEE 1003 | Brussels, Belgium |
| 1989 Oct | UNIX Expo | New York, NY |
| 1989 Oct 31-Nov | 2 IETF | IAB, U. Hawaii, Honolulu, HI |
| 1989 Nov 1-3 | UNIX EXPO | Javits Conv. C, New York, NY |
| 1989 Nov 6-10 | DECUS S | Anaheim, California |
| 1989 Nov 9 | NLUUG C | The Netherlands |
| 1989 Nov 9-10 | JUS 14th S | Osaka |
| 1989 Nov 24 | AFUU C | Paris, France |
| 1989 Dec | UNIX Fair | JUS, Tokyo |
| | | |
| 1990 Jan | U in Gov. C&T | Ottawa, ON |
| 1990 Jan 22-26 | USENIX | Washington, DC |
| 1990 Jan 23-26 | UniForum | Washington Hilton, Washington, DC |
| 1990 Jan 29 | IEEE 1003 | New Orleans, LA |
| 1990 Feb 6-8 | IETF | IAB, (FSU, Talahassee, FL) |
| 1990 Mar 27-30 | AFUU | Paris, France |
| 1990 Apr | IEEE 1003 | Montreal, Quebec |
| 1990 Apr 23-27 | EUUG | Munich, Germany (tentative) |
| 1990 May 2-4 | IETF | IAB, (U. Washington, Seattle, WA) |
| 1990 May 7-11 | DECUS S | New Orleans, Louisiana |
| 1990 May | U 8x/etc C&T | /usr/group/cdn, Toronto, ON |
| 1990 Jun 11-15 | USENIX | Marriott, Anaheim, CA |
| 1990 Jul 31-Aug | 2 IETF | IAB, ?, not in North America |
| 1990 Autumn | EUUG | south of France |
| | | |
| 1991 Jan 21-25 | USENIX | Dallas, TX |
| 1991 Jan 22-25 | UniForum | Infomart, Dallas, TX |
| 1991 Feb | U in Gov. C&T | Ottawa, ON |
| 1991 Spring | EUUG | Tromso?, Norway (tentative) |
| 1991 May | U 8x/etc C&T | Toronto, ON |
| 1991 Jun 10-14 | USENIX | Opryland, Nashville, TN |
| | | |
| 1992 Jan 20-24 | USENIX | Hilton Square, San Francisco, CA |
| 1992 Jan 21-24 | UniForum | Moscone Center, San Francisco, CA |
| 1992 Jun 8-12 | USENIX | Marriott, San Antonio, TX |
| | | |
| 1993 Jan | USENIX | northeast North America |
| 1993 Mar 2-4 | UniForum | Washington, D.C. |

## Organising Bodies

NIST/NBS/POSIX
Roger Martin
National Institute of Standards
and Technology
Technology Building, Room B266
Gaithersburg, MD 20899
+1-301-975-3295
+1-301-975-3295
rmartin@swe.icst.nbs.gov

IEEE Computer Society
P.O. Box 80452
Worldway Postal Center
Los Angeles, Ca. 90080

/usr/group
4655 Old Ironsides Drive, Suite 200
Santa Clara, California 95054
U.S.A.
+1-408-986-8840
+1-408-986-1645 fax

/usr/group/cdn
241 Gamma St.
Etobicoke, Ontario M8W 4G7
Canada
+1-416-259-8122

Tracy MacIntyre
Exhibition Manager
EMAP International Exhibitions Ltd.
Abbot's Court
34 Farringdon Lane
London EC1R 3AU
United Kingdom
+44-1-404-4844

AUUG
P.O. Box 366
Kensington
N.S.W. 2033
Australia
uunet!munnari!auug
auug@munnari.oz.au
+61 3 344 5225

AMIX, c/o IPA
P.O. Box 919
Ramat-Gan
Israel, 52109
+972-3-715770
+972-3-715772
amix@bimacs.bitnet
amix@bimacs.biu.ac.il

Japan UNIX Society (JUS)
#505 Towa-Hanzomon Corp. Bldg.
2-12 Hayabusa-cho
Chiyoda-ku, Tokyo 102
Japan
bod%jus.junet@uunet.uu.net
+81-3-234-5058

UNIX Fair '88 Association
1-1-1 Hirakawa-chu,
Chiyoda-ku, Tokyo 102
Japan

DECUS U.S. Chapter
219 Boston Post Road, BP02
Marlboro, Massachusetts 01752-1850
U.S.A.
+1-617-480-3418

USENIX Association Office
P.O. Box 2299
Berkeley, CA 94710
USA
+1 415 528 8649
office@usenix.uucp

EUUG National group addresses can be found on the back cover of this newsletter.

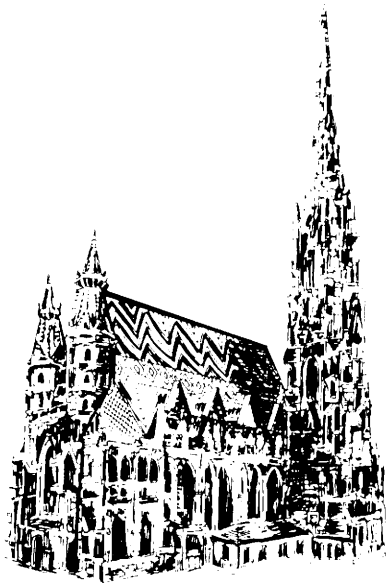
# EUUG

European UNIX® systems User Group

## PRELIMINARY ANNOUNCEMENT and CALL FOR PAPERS

# EUUG Autumn '89 Conference and Exhibition

## Wirtschaftsuniversität Vienna, 18-22 September 1989



## Preliminary Announcement

The UUGA will host the Autumn '89 European UNIX® systems User Group Technical Conference in Vienna, Austria, Europe.

**Technical Tutorials** on UNIX and closely related subjects will be held on Monday 18th and Tuesday 19th September, followed by the three day **Conference** with commercial **Exhibition** finishing on Friday, 22nd September.

A pre-conference registration pack containing detailed information will be issued in June 1989.

## Call for Papers

The EUUG invites papers from those wishing to present their work. Full papers or extended abstracts must be submitted. All submitted papers will be referred to be judged with respect to their quality, originality and relevance.

Suggested subject areas include:

***Security***
***Fault Tolerance***
***Transaction Processing***
***RISC architectures***
***Scheduling in multi-processor systems***
***User Interfaces***
***Network management***
***Interoperability with other systems***
***Artificial intelligence and 5th GL***
***Standardisation***

## Important Dates

| | |
|---|---|
| Abstract deadline | 25th April, 1989 |
| Acceptance notification | 12th June, 1989 |
| Final paper | 24th July, 1989 |
| Closing date for Student Grant Application | 1st August, 1989 |

## Method of Submission

Full papers or extended abstracts must be submitted by post to the EUUG Secretariat. All submissions will be acknowledged by return of post.

Papers may also be submitted electronically to eva Kühn, but this is not the formal method of submission.

## Tutorial Solicitation

Tutorials are an important part of the EUUG's biannual events providing detailed coverage of a number of topics. Past tutorials have been taught be leading experts.

Those interested in offering a tutorial should contact the EUUG Tutorial Executive as soon as possible.

## Additional Information

We will be pleased to provide advice to potential speakers. We can be contacted at the addresses below.

If you wish to receive a personal copy of any further information about this, and future EUUG events, please write, or send electronic mail, to the Secretariat.

## Useful Addresses

**Secretariat**
EUUG
Owles Hall
Owles Lane
Buntingford
Herts
SG9 9PL
UK

Phone: (+44) 763 73039
Fax: (+44) 763 73255
Telex:
Email: euug@inset.uucp

**Conference Executive**
Dr. Ernst Janich
Nixdorf Computer AG
Söflinger Str. 70
D-7900 Ulm
W. Germany

(+49) 731 1526 464
(+49) 731 1526 105

janich@nixulm.uucp

**Tutorial Executive**
Neil Todd
1ST
60 Albert Court
Prince Consort Road
London
SW7 2BH
UK

(+44) 1 581 8155
(+44) 1 581 5147
928476 ISTECH G
neil@ist.co.uk

**Programme Chair**
eva Kühn
TU Wien
Argentinierstrasse 8/180
A-1040 Wien
Austria

Phone: (+43) 222 58801 4417
Fax: (+43) 222 505 4800
Email: eva@vip!.uucp

**Programme Committee**
Peter Collinson
University of Kent
Computer Laboratory
Canterbury
Kent CT2 7NF
UK

(+44) 227 76400 ext. 7619
(+44) 227 762811
pc@ukc.ac.uk

Hans W. Strack-Zimmermann
iXOS Software GmbH
Rosenkavalierplatz 14
D-8000 München 81
W. Germany

(+49) 89 46020 92
(+49) 89 918114
unido!ixos!hsz

**Local Organisations**
UUGA
Unix User Group Austria
Schottenring 33 Hof
A-1010 Wien
Austria

Phone: (+43) 222 346 184
Fax: (+43) 222 743 575 ext 600
Email: uuga@tuvie.uucp

OECG
Österreichische —
Computergesellschaft
Wollzeile 1-3
A-1040 Wien
Austria

(+43) 222 512 02 35
(+43) 222 513 37 735

**For further details, contact:**
**The Secretariat**

# European Unix® systems User Group

**Owles Hall, Buntingford, Herts SG9 9PL, UK**
**Tel: +44 763 73039 Fax: +44 763 73255 Network Address: euug@inset.uucp**

# Introduction to Window Systems

*William Roberts*
*liam@qmc-cs.UUCP*

*Department of Computer Science*
*Queen Mary College*
*London, UK*

William Roberts has been programming computers since age 11. After graduating from Oxford with an Honours degree in Mathematics, he worked for 2 years with microcomputers of various sorts, before returning to higher education. In 1985 he was awarded an MSc with Distinction by the Department of Computer Science at Queen Mary College, London and has remained there ever since. He is currently a systems programmer supporting the Departmental network of over 120 UNIX machines, and actively working on X11 and NeWS.

## Introduction

Hello! This is the first article in a regular column on window systems: I hope to cover more or less everything that combines UNIX with on-screen graphics. If there are any questions you have, any products or systems you'd like more information about, let me know and I'll try to get an article from someone who knows. If there is anything you'd like to write an article about, mail me and I'll be happy to oblige.

Lined up for the near future are articles on; X11 and the Atari ST, Presentation Manager and OS/2, and Open Look (the new interface specification from Sun/AT&T). To start the column off, I'll summarise the current major systems and where the UNIX+graphics world seems to be going at the moment.

## ProprietarySystems

Most manufacturers of 'UNIX workstations' supply a fairly powerful machine intended for use by a single person sat at a graphics screen; such systems always include a *window system* which allows ways of sharing the screen between several applications and has a programming interface for writing graphical programs. Familiar examples include the Sun Microsystems workstations with the SunWindows and SunView systems, Apollo workstations with the Domain Graphics Manager, and Hewlett-Packard with Windows 9000. The main feature of such systems is that they are proprietary: the programming interface is only available on one manufacturer's machines, and a supplier writing graphics programs for several machines has to have a different version of the source code for each machine. This can be extremely annoying, especially when the graphical techniques are actually the same in two systems and all that changes is the *typedefs* and the library routine names.

## Standards

The usual weapon against proprietary systems is non-proprietary standards, and the graphics world is no exception. Early graphics standards which are available on some machines include the ACM SIGGRAPH standards CORE and CGI, the ISO GKS (Graphics Kernel System) standard and the most recent, PHIGS (Programmers Hierarchical Interface to Graphics Systems). The distinguishing feature of these systems is that they are all derived from the world of Computer Aided Design, and the use of *vector graphics* which

describe the display in terms of lines and filled areas. These are usually aimed at producing realistic pictures of three dimensional objects or two dimensional graphs with labelled axes, etc. Neither of these is appropriate to the typical use of a UNIX workstation, where the one graphics display is used to provide multiple terminal windows, each of which behaves like a separate VDU screen. To tackle these *raster graphics* displays where the final image is composed of a rectangular array of dots (called *pixels*), a new collection of standards and/or proprietary systems has been formed.

## Raster Graphics Standards

There are a number of systems which have some claim to be called 'standards', though none of them have been accepted by any of the national or international standards-making organisations. The main ones are listed below, and summarised in the following sections.

*Presentation Manager*
: This is important because it is the standard window system with the IBM/Microsoft OS/2 operating system. It is proprietary and runs only on OS/2, which implies an Intel 80286 or Intel 80386 processor and a machine that is very like one of the flavours of IBM PC or PS/2. However, anything that IBM does will affect most of us whether we like it or not, so this one counts as being ''important''.

*Macintosh*
: The Macintosh window system is another proprietary one, but is of some importance in that it is very successful and has inspired almost all of the other systems: so much so that Apple are currently suing Microsoft and Hewlett Packard for stealing the ''look and feel'' of the system by copying it too closely. Don't expect this to appear on any other machines.

*X*
: This is the window system developed originally by MIT, but now actively promoted by the X Consortium, a large group of computer companies including all of the graphics workstation manufacturers you can think of (e.g., Sun, DEC, Apollo, Hewlett Packard, IBM) and some surprises (Kodak-Eastman, Cray Research). It is the window system chosen by the Open Software Foundation, and also features in the plans of the AT&T/Sun/Unisys grouping now known as 'UNIX International' (a ghastly name!), so it is likely to feature in all new UNIX systems for the next few years.

*NeWS*
: This is a development by Sun Microsystems, but is available under licence and has been ported to other systems. Its main claim to fame is that a combination of NeWS and X will be the window system distributed as standard with AT&T System V release 4, as part of the Sun/AT&T commercial collaboration.

Having listed the important systems (if your product isn't in this list, I'm afraid that you aren't currently important...), what are the strengths and weakness of each, and which ones are in direct competition with each other?

### Macintosh

This system has no direct competition because it only available for the Macintosh, but its pride of place in the Macintosh world is being slightly eroded by Apple's decision to supply the X system with their A/UX version of UNIX System V.2.2 in addition to supporting parts of the Macintosh system. The problems stem from the architecture assumptions used in the original Macintosh, which didn't have a multiprocess system or memory management hardware. Its only advantage is the ability to run 'well-behaved' Macintosh applications.

The Macintosh QuickDraw library is fairly typical of current graphics libraries. The facilities are more complex that traditional pen plotters and fall into three groups.

The first group are operations on *rasters* (also known as *bitmaps*), which are rectangular arrays of dots. Each dot is called a *pixel* and is a number indicating the colour to be displayed on the screen; these are normally packed into memory as tightly as possible, so a *monochrome* bitmap with just the colours white or black will be stored as one bit for each pixel, 8 pixels to a byte and so on. Operations on bitmaps combine each pixel in one *source* bitmap with the corresponding pixels in another *target* bitmap (sometimes two others), replacing the pixel in the target bitmap with the result. The combination of pixels is usually done as a bitwise logical operation (AND, OR, XOR, etc.) on the numeric representation of each pixel, and the whole process is called *RasterOp* or *BitBlt*. If the source bitmap is smaller that the target, it will be *tiled* by placing (imaginary) copies of the source raster next to the original

repeatedly until the target is completely covered; this is used to do cover large areas with a regular pattern. If three bitmaps are used, the third one is called a *mask* and controls which pixels in the target will be modified and which will be left alone. To put a complex shape onto the screen, a mask is prepared with all of the pixels in the shape set to one, and all the other pixels set to zero: using this as a mask when copying the source into the target ensures that only the pixels which form part of the shape are changed. Combining the use of a mask with tiling the source raster gives the ability to *fill* complex areas with a colour or pattern.

The second group of operations involve drawing lines and curves: they describe areas to be filled using the raster operations from the first group. Lines are drawn using a *pen* which will have a shape (given by a mask raster) and a colour (given by a source raster). Lines are drawn by working out the pixels 'brushed' by the pen as it traces the line, and filling the result with the pen colour. Line-drawing operations include straight lines, arcs of circles and a general form of wavy line called a *Bezier curve*. There are also assorted convenience routines including arrow heads, rectangles and rectangles with rounded corners.

The final group deals with text. Text is placed on the screen by specifying the character string to be printed, the pixel position of the first character and the *font* which specifies the character shapes for each code. Every character code has a corresponding bitmap (sometimes called *glyphs*), and the text primitives copy the character bitmaps into the target raster, computing the start fo each character based on the width to the preceding characters. This allows for proportionally spaced text in various sizes, but the problems of rotating bitmaps mean that such fonts have a fixed orientation (usually going across the screen).

There is some discussion of QuickDraw on the *comp.sys.mac* newsgroup.

## Presentation Manager

Presentation Manager is part of the OS/2 scheme of things and again has little direct competition in the OS/2 market. Microsoft offer a look-alike for MS-DOS systems, called Windows 2.03, and the current litigation from Apple over the "look and feel" of Windows 2.03 is clearly intended to affect Presentation Manager as well (Windows 2.03 is sold as looking identical to Presentation Manager as far as the user is concerned): suing Microsoft is perhaps a way of getting a useful precedent without taking on IBM immediately. Regardless of these legal troubles (which really can't succeed in the long run, not against the IBM money mountain), Presentation Manager is a fairly typical example of how a window system works on a multiprocess operating system.

Each application is written using a large library of graphics routines for handling line drawing, multifont text and so on, and also using the *window system toolkit*, another large library of routines which provide the now-familiar buttons, menus, scroll bars, etc used by graphical user interfaces. The toolkit in particular uses object-oriented programming techniques: all buttons are very similar and share the same code but applied to a separate data structure for each button. The *instance data* for a given button will include things like the label written on the button (e.g., 'Save File') and a pointer to the *callback routine* to be executed whenever the button is pressed. The typical structure of a program has an initialisation section in which all the necessary buttons etc are created and placed onto a single window belonging to the application, followed by an *event loop* which involves waiting for something to happen (e.g., a key press, a mouse motion, perhaps an input from some external sensor in a process control environment) and then responding to that *event* by calling the appropriate routines. The toolkit routines will normally be handled directly within the library, though Presentation Manager itself doesn't work like this. This sort of program is difficult to analyse because it has a beginning, a middle and lots of other bits, and the flow of control cannot be determined without knowing exactly how the toolkit works. Tools such as lint, adb and dbx won't be much help because so much of the object-oriented machinery involves pointers to functions and the state of the program resides much more in the values stored in all those small data structures than in the current stack backtrace. The closest equivalent in UNIX experience is trying to debug the UNIX kernel, though window systems are usually slightly easier.

The graphics and toolkit libraries interact with a central part of the window system which is normally embedded in the operating system kernel: this deals with the handling of mouse and keyboard interrupts, passing them as events to the application programs, and with managing the screen. Some systems run outside the kernel, in which case the kernel must supply fairly direct

access to mouse and keyboard events. The purpose of a window system is to share the physical screen between several applications, without those applications having to know in advance about each other; this is similar to the task of the kernel in sharing the available CPU time and memory between applications. The usual technique is to let each application create one or more *windows* which are treated like the whole screen, but to combine these 'virtual screens' together on the real screen. This involves distancing the application from its window and requiring that all window operations are carried out via the window system: in particular, the part of the screen used for one window may be partly hidden by another window which has been placed 'in front', so applications cannot write directly into screen memory. For this reason, applications have to deal with the problem of redrawing the content of their window whenever necessary, because the window system may not guarantee to remember everything that was drawn before.

Turning back to the specifics of Presentation Manager, its main advantages are the notion of presentation and device spaces, the lightweight process mechanism in OS/2, and its strategic role in IBM's announced plans. A *presentation space* is a data structure which filters application graphical commands before passing them to the main part of the window system. This allows an application to change coordinate systems and other such graphical transformations, but also includes two possible ways of avoiding the 'redraw your window' problem; first, an off screen copy of the screen window, so missing portions can automatically be transferred to the screen from the copy, or second, a *display list* which stores a list of all the graphics commands and can replay then when necessary. A *device space* is another filter which deals with device dependent issues and copes with the multitude of different graphics hardware available for IBM machines; a closer UNIX analogy might be that of a module in the System V.3 STREAMS system. The OS/2 lightweight process mechanism is used in Presentation Manager to allow multiple event loops, so that large parts of the application can operate in parallel. This of course increases the kernel-like complexity of applications, but it is useful and does make some things much simpler. Finally, Presentation Manager is part of a grand IBM vision called SAA, so someone in IBM believes in it.

Presentation Manager is occasionally mentioned on the *comp.windows.misc* newsgroup.

## X

As its supporters keep having to point out, this is "a window system called X, not a system called X-Windows". X is really the main window system to watch in the UNIX world because all of the graphics system manufacturers are supporting it, because it offers device-independent graphics that can operate over network connections, and because versions of it are available free (including source code).

X is a *distributed window system*. Recall that an application has to manipulate its windows indirectly by asking the window system to carry out graphics operations; X takes this notion a step further and allows the window system to be on a separate computer with the application requests passed through a network connection (or even a simple serial line). The part of the system which actually carries out the work is called a *graphics server* and has to run on a machine with a graphics screen (and usually a mouse and keyboard as well); the application end is called a *client* of the server. It is entirely possible for several clients to use the same server, and for those clients to be on different machines. Pushing this idea further, X defines an *abstract graphics device* with a fixed set of operations, and any particular graphics server will implement those operations on one particular type of hardware, emulating the abstract device. X is carefully designed (as the result of experience with using earlier versions, we are now at X version 11 release 3) so that it can be implemented on almost any raster graphics display currently in existence, and so that almost any existing window system could be built on top of X: for this reason X is known as a *platform* and is said to "implement mechanism, not policy". In principle, proprietary systems such as Presentation Manager and the Macintosh system could be built as libraries on top of X, allowing Macintosh applications to run using an IBM graphics server or Presentation Manager applications to appear on your Macintosh screen.

The most important things about X are that it covers the things which people know how to do well (overlapping rectangles), it leaves open issues which aren't settled yet (how to handle colour) so that existing systems fit into the model, and it does nothing else. It is a rationalisation exercise intended to remove once and for all the

nuisance of different source code for different vendors hardware: the vendor now supplies an X server and the existing program still works.

The distributed nature of X offers further advantages, especially to multi-user systems such as UNIX. Manipulating large graphics displays is a processor-intensive activity and makes big performance demands on memory (a Sun 3/50 runs benchmarks 20% faster if the screen is disabled!); this makes it impractical to have lots of graphics displays on even the fastest of minicomputers. However, with a distributed window system, you can add graphics displays simply by buying small computers to run the graphics server, and running the applications on the multi-user computer as before. We are beginning to see companies producing 'X terminals', simple diskless graphics workstations whose 'operating system' is X and which connect to Ethernet rather than RS232 ports. The cost of these is currently around $2000 US, which is cheaper than most 'add-on graphics displays', and cheaper still are X servers based around small micro computers such as the Atari-ST (of which more in a future issue). At the other end of the scale, Cray Research is a member of the X Consortium (which now controls changes to the X system) because systems like X offer the only rational way to add interactive graphics to supercomputers.

The advantages of X are many; it provides manufacturer independence, it simplifies the production of graphics software over a wide range of machines, and it raises the capabilities of many existing systems. The drawbacks are mostly to do with its flexibility. You are expected to make your own policy decisions and so you are faced with a choice of toolkits, a choice of window managers etc, all of which make life harder: "I wanted an answer, but you have given me just a new set of questions".

X is already widely available: it is available on Hewlett-Packard machines under HP-UX, it will be available on all DEC machines shortly (under MS-DOS, VMS and Ultrix), it will be available on Sun systems shortly (and SunView2 will be built on top of X) and is available from Apollo and Apple (for A/UX). The free source code mentioned earlier is available from MIT for the cost of the magnetic tapes, but is also available from various sites in Europe.

One of these is from Jamie Watson. He offers a distribution on QIC-24 (Sun, IBM RT/PC, Arix, NCR others) or TK-50/TK-70 (DEC MicroVax and Vaxstation) cartridges. He makes the distributions free; all you have to do is agree to send back his tape, or an equivalent one. By the way, someone sent back a tape *FREIGHT COLLECT*. Pretty tight. Please don't do that.

The tapes that he sends contain the following:

— The complete MIT X.V11R3 distribution, including core, and user contributed 1 & 2.

— All 'official' patches issued by MIT up to the day that he makes the tape.

— The Purdue Speedups.

— Everything that has been posted to comp.sources.x

Contact him at:

Jamie Watson
Adasoft AG
Nesslerenweg 104
CH-3084 Wabern
Switzerland

EUNET: *jw@pan.uucp*
Tel: +41 31 54 35 70
Fax: +41 65 42 10 71

The difference between the sample server on the tape and the one you buy from Hewlett-Packard (for example), is that Hewlett-Packard have put effort into tuning their version to run as fast as possible on their hardware; the sample server is intended to be easy to port so that you get something working, and doesn't claim to be particularly fast. There are some very active USENET newsgroups discussing X, in particular comp.windows.x and comp.sources.x, plus there are other mailing lists on specialised sub-topics.

Finally, X has no serious competition. Both the Open Software Foundation and UNIX International are supporting X, it is the window system recommended by X/Open, and best of all, none of these groups has control of X so it will remain an open standard.

## NeWS

NeWS (the Network extensible Window System) was developed by Sun Microsystems and was viewed as direct competition for X. It is now more clearly understood to be different from X and represents one possible direction for future window systems.

NeWS is a device-independent distributed window system, but it carries device independence much further than X by using the PostScript language developed for use with laser printers. In systems like X, everything is described in terms of the pixels on the screen, so applications are not independent of things like the shape of a pixel (not all dots are square?!). NeWS however takes its descriptions in absolute terms using real number coordinates and real number proportions of red, green and blue for its colours; you can draw anything in any colour at any size. The NeWS graphics server then has the task of reproducing the ideal description as best it can with the hardware available, and it can use any and every piece of trick hardware available. This makes the NeWS server 'future-proof' because it can incorporate new developments in graphics hardware without changes to the application programs, and also make it a much better environment for applications which produce on-screen an approximation to the output you will produce on a printer. All of this is much harder work computationally, so a NeWS server is likely to be more expensive than an X server and might run slower.

NeWS has facility which gives a definite advantage over X: the language used by clients to control the server is an interpreted programming language rather than a sophisticated but fixed set of commands, so subroutines can be downloaded to the server and avoid some of the overheads of network communication between the server and client. NeWS extends the PostScript language to add lightweight processes, so it is possible to create processes which run entirely within the NeWS server (for instance, the notorious 'eyeball' demonstration where a cartoon-style eye watches the mouse as it moves around the screen). This sort of technique allows parts of the toolkit to live in the graphics server, which makes the response time of pop-up menus much better than in a system such as X, where everything is delayed by the round trip times of the network communication.

Despite its advanced features, NeWS is not currently strong competition for X because it is about two years younger and is not so widely available. There is also resistance to NeWS because it doesn't allow the traditional hands-on access to specialised hardware and so cannot be made to perform some of the tricks of high-speed animated graphics. Faced with this problem, Sun have produced a solution which combines all of both worlds (not just the best!): a combined X and NeWS server, called X11/NeWS. The underlying 'pixel shuffling' is common to both servers, so it is relatively easy to build a server which works as both an X server and a NeWS server on the same screen with the same mouse and keyboard: problems only arise in areas of conflict such as control of the screen background, and in dealing with how the two systems should be visible to one another. Sun claim to have tackled these problems and will shortly be Beta-testing the combined server, which will be the X server available for Sun machines, and which will be part of the standard distribution of AT&T System V release 4. This will enable NeWS to filter out into the marketplace inside the X server, where it will be waiting as applications come along which require its special capabilities.

At present NeWS is available without the X server on Suns (naturally) and a number of other machines including the Macintosh II under A/UX and some specialised graphics hardware from Parallax and Silicon Graphics. Sun's source code is reasonably easy to port, and Sun themselves did a 'reference port' to the DEC VaxStation II under Ultrix, and to an unnamed 80386-based machine running System V.3; both of these ports come as part of the NeWS source code. They also did a 'joke port' to the Atari ST; the ST uses the same 68020 processor as the Sun 3, and apparently one Atari owner at Sun noticed that the Sun load file ('dot-o') format could be mangled into something that the Atari loader could read.... The Atari ST port indicates a possible further development in distributed window systems, the use of a graphics-only server which is connected simply (perhaps even via a modem) to another machine which deals with the local area network connections and anything which doesn't directly affect the pixels on the screen.

Newsgroups relevant to NeWS include *comp.windows.news* and *comp.lang.postscript*.

## Glossary

The above discussion has tried to introduce the jargon of window systems at the same time as discussing the important systems currently around. You'll have to refer back to find them, but the important words are:

> pixel, raster, bitmap, RasterOp, source, target, tiling, mask, font, glyph, event, event loop, toolkit, callback routine, graphics server, abstract graphics device.

There are no good general books on this subject (though I'm helping to write one!), but there are reasonable books on the Macintosh system and probably Presentation Manager (there was a good book by Petzold on Microsoft Windows, the ancestor of Presentation Manager). The computing press have details of courses on specific window systems and/or overviews, though for proprietary systems such as NeWS it is easiest to approach the manufacturer for information on courses.

## Conclusions

The above mixture of survey and introduction should have given you some idea about what is going on in window systems at present. This is currently an active area, and one in which UNIX is very much involved with window system standards being pulled along by the UNIX standards activity. Fortunately there is an open window system (X) which has been gathering momentum in recent years and is well placed and well suited to being the standard window system for UNIX; an open window system for an open operating system.

==========

# Puzzle Corner

*Mick Farmer*
*mick@cs.bbk.ac.uk*

*Birkbeck College*
*University of London*

Hello,

One day last year I casually submitted something amusing (to me :-) to Alain for inclusion in the Newsletter and now look! A regular feature.

Anyway, I hope you find these puzzles interesting and stimulating. First, the solution to Puzzle Number 1 in the last issue:

1. Thompson's book is dedicated to Mike, because the other four are accounted for.
2. Thompson's christian name cannot be Steve, Mike, or Brian.
3. Thompson's christian name cannot be Ken, because Mike Ritchie dedicated his book to Ken.
4. Thompson's christian name is therefore Dennis.
5. Bourne's christian name is therefore Brian.
6. Ken's surname is Kernighan.

Second, Puzzle Number 2 which is dead easy:

Last night I dreamt that the EUUG had overcharged some of the national groups and that they were going to refund 1,000,000 ECUs. With no guideline available they decided that the amount received by each group was going to be some power of seven, e.g., 1, 7, 49, 343, ... They also decided that no more than six groups would receive the same amount. How did they distribute the 1,000,000 ECUs?

Third, Puzzle Number 3 which is slightly more difficult:

It was a windy day and Brian was flying his kite from the Hill using a new wire developed at the Labs. This wire had a diameter of only one-hundreth of an inch yet it was extremely light and strong. The wire had been provided in the form of perfect sphere just two feet in diameter. What is the length of the wire? You can assume that when the wire is wound up the ball is solid. Answers to the nearest mile!

For those not used to British measurement of distance: 1 foot == 12 inches, 1 mile == 5280 feet.

That's it then. Mail me if you want some clues otherwise read the solutions in the next issue.

Mick

EUUG

European UNIX® systems User Group

# 4.3BSD MANUALS

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

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

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

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

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

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

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

*4.2BSD Manuals are No Longer Available*

# 4.3BSD Manual Reproduction Authorisation

Date _____

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

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

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

Signed _____

Institution (if Institutional Member) _____

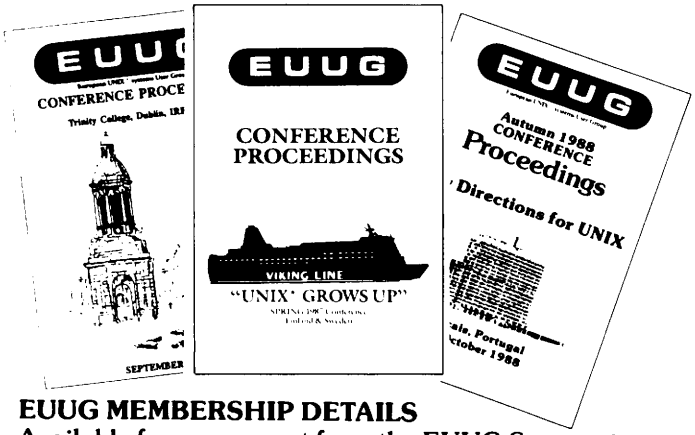
† Members of EUUG National groups are automatically members of the EUUG.

# EUUG

European UNIX® systems User Group

## ADDITIONAL PUBLICATIONS AVAILABLE FROM EUUG

| Title | | | Cost |
|---|---|---|---|
| EUUG – CONFERENCE – PROCEEDINGS | | | |
| Spring 1983 | – | Bonn | £2.00 |
| Autumn 1983 | – | Dublin | £2.00 |
| Spring 1984 | – | Nijmegen | £5.00 |
| Autumn 1984 | – | Cambridge | £5.00 |
| Spring 1985 | – | Paris | £5.00 |
| Autumn 1985 | – | Copenhagen | £10.00 |
| Spring 1986 | – | Florence | £20.00 |
| Spring 1987 | – | Finland/Sweden | £20.00 |
| Spring 1988 | – | London | £20.00 |
| Autumn 1988 | – | Portugal | £20.00 |
| European E-Mail Directory | | | £18.00 |
| 4.3BSD MANUAL (Members only) | – | User's Manual Set | £25.00 |
| | | Programmer's Set | £25.00 |
| | | System Manager's Manual | £10.00 |
| Language C – Standard Proposal | | | £8.00 |
| USENIX Conference Proceedings C++ – Santa Fe, 1987 | | | £30.00 |




### EUUG MEMBERSHIP DETAILS

Available free on request from the EUUG Secretariat.

### UKUUG PROCEEDINGS

Winter 1988 –UKNet Meeting, Canterbury **£10.00**

If you wish to order any of the above publications, or receive details of EUUG Membership, please use the order form below and return it to the EUUG Secretariat, Owles Hall, Owles Lane, Buntingford, Herts. SG9 9PL, U.K. If you are ordering publications then your remittance should be enclosed by cheque, bank draft etc. made payable to EUUG.

# ORDER FORM

Name _____

Address _____

_____

_____

_____

Phone _____

Network Address _____

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

Please send me _____ no. copy/ies of the following:

_____

_____

I would like to receive membership details of the EUUG YES/NO

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

We accept
Visa, Access, Euro Card, Master Card, UK cheques, and Euro cheques.

Please complete name and address of credit card holder if different from name and address on order form.

NAME _____

ADDRESS _____

_____

_____

Expiry Date: _____

Credit Card No:

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | |

I enclose my cheque, bank draft etc., in the sum of £ _____ and understand a receipt will be sent.

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

**To: EUUG Secretariat, Owles Hall, Owles Lane, Buntingford, Herts. SG9 9PL, United Kingdom.**
**Tel: Royston (0763) 73039 +44 763 73039**
**Fax: Royston (0763) 73255 +44 763 73255**

For BSD manuals please sign Manual Reproduction Authorisation Notice overleaf.

This page may be photocopied for use.

# Software | Review

*Donal Daly*
*daly@cs.tcd.ie*

*Distributed Systems Group*
*Department of Computer Science*
*Trinity College Dublin.*

Donal Daly works as a researcher for the Distributed Systems Group in Trinity. His current work is involved in developing UNIX on top of an object-oriented distributed operating system. Previously he had system management responsibilities for System V and Berkeley UNIX systems within Trinity. Donal is a committe member of the Irish UUG.

## Introduction

Welcome to *Software | Review*. This new column will review software that appears on the net. This is software that is posted to the source news groups and is in the Public Domain. In the future is may also deal with commercial software as well. For an example of what sort of software get posted to news groups, I have included a listing, at the end of this column, of all the software that has been posted to the *comp.sources.unix* group. I hope you will also find information here about soon to be released software, and what's the latest and greatest new utility is.

I welcome contributions from you. In fact I actively encourage it!! This is the reason for the pipe in the title. I will filter ('pipe') your contributions and gather them together into topics for publication. You can send contributions to me at the above email address.

Introduction over – let's get down to it! This first column includes information on a new upcoming news software releases. A short note on a new window manager package just released. A report on the newish source group *comp.sources.x* and reviews on *wanderer* (a game), *xshar* (a handy utility). We round off this month column with a brief report on archive sites. I have 'bullied' a few friends into helping me write this month's column. You will see there names at the start of the section they wrote, everything else was by me and I accept responsibility for the article as a whole! (If you knew me this would bring a smile to your face :-) ). I must thank also, our newsletter editor, Alain Williams for his gentle prodding ensuring that this column saw the light of day. Jamie Watson read a earlier version of this column and provided some usefull comments.

## MGR – A Window Management System

During the last 2 weeks of January and the first week of February a new window management package was posted to *comp.sources.unix* (Volume 17). The package was split into 61 parts and was the largest package ever for *comp.sources.unix*.

MGR was written by Steve Uhler at the Computer Science Research Division at Bellcore. The code itself is quite small according to an introductory news message: "For the basic window manager the MGR code is one tenth the size of the

comparable X code''. MGR is supposedly not tied to a given operating system or hardware. Has anybody used this 'beast', and would like to share their experiences?? Below is an abstract from the documentation which serves to best describe MGR.

MGR - C Language Application Interface
Stephen A. Uhler
Bell Communications Research

MGR (ManaGeR) is a window system for UNIX that currently runs on Sun Workstations. MGR manages asynchronous updates of overlapping windows and provides application support for a heterogeneous network environment, i.e., many different types of computers connected by various communications media. The application interface enables applications (called client programs) to be written in a variety of programming languages, and run on different operating systems. The client program can take full advantage of the windowing capabilities regardless of the type of connection to the workstation running MGR. This document describes the C interface library for MGR which provides a set of macros and functions that implement the stream protocol. This library provides client programs written in C with a function call interface to MGR. The library provides the lowest level access to MGR functions and represents a one to one mapping to the underlying stream protocol.

## ShareWare Software

The concept of Sharware software, is where you normally receive the package for free and then if decide you like it and want to use it you pay a fee. This places the trust on the user to register his package. Jamie Watson informed me about a specific Shareware package which he now uses, and is quite impressed with, and I thought you might like to hear about it as well. The package in question is *Jetroff* which was posted to *comp.sources.misc* a few months ago. Complete sources were posted as an incentive. This text processing package had a number of good features, including complete *ditroff* functionally, and the ability to handle bit-image files created by various PC drawing utilities.

To register your copy would cost you $50 as an individual, a $100 for a company. This seems a reasonable price when you compare it to a package like XROFFwhichcosts

Like Jamie, I feel that that something of this quality and usefullness deserves a little encouragement. If you have had any good/bad experiences with Shareware software let me know.

## WANDERER

**(Hugh Grant <hugh@maths.tcd.ie>)**

This game, billed as a ''mini rogue-like adventure game'' first appeared back in issues 2 and 3 of volume 5 of *comp.sources.games*. I didn't look at the game then, as I was under pressure to do some work of an academic nature, and to stop ''playing stupid computer games''. However, when version 2 of the game appeared recently, curiosity won, and the 's' key was pressed and the source lodged in a secure place. It was only when Donal Daly started badgering me to review some software from the net that I actually got around to compiling and playing the game and I must say I was quite surprised at what it turned out to be. The title does not accurately describe the game, it being neither an adventure, or for that matter like rogue.

The game is by Stephen Shipway from Warwick University. It comes in two shell archives, which unpacked without any difficulty. There are *makefiles* for UNIX and MS-DOS, though I have only tried it on UNIX. I just typed '`make`' and the game compiled perfectly first time. It's nice to see this, as it gave me hope for the next stage: actually running the thing. '`make install`' copied everything to the right place, and so I decided to sit back and have a go at playing the thing. Here one problem cropped up: the high score table was incorrectly created as 'hiscores' by the *makefile*, and the game looks for a file called 'hiscore', but this was the only problem I ran into. I'd imagine the game will run on almost anything, as I tried it on fairly minimal hardware (a PDP-11/73 running 2.10BSD).

The game is quite fun to play, although it was a bit confusing at first, until I figured out exactly what I was supposed to do! (Yes, I really should have read the *README* files first, I know, I know). The basic idea is that the player has to move around the screen collecting treasure without being killed by falling rocks, arrows, bombs, or on later screens, monsters. There are some other goodies such as teleport traps, and slides for the rocks to bounce down. This is definitely not a hack 'n slay game, as each screen must be completed carefully, or one may make

some treasure inaccessible, hidden under a pile of fallen rocks. The game uses *curses* to show a window onto a larger screen that scrolls automatically as the player moves, although a facility to zoom out and see the whole arena is available. The game also allows the editing of new screens, and though I tried this, I always seemed to make the screens far too easy or totally impossible.

The game is good fun, and makes a change from the usual nethack/rogue/moria sort of game. The scrolling window is a nice touch and is probably as good as it gets on an ASCII terminal (though I'm not sure it would be much fun at 1200 baud). A manual page would have been nice, for although nearly everything is in the *README* files, I got the impression that one was assumed to have played version 1 of the game. All in all a worthwhile addition to any UNIX games collection, and full marks to Stephen Shipway.

## XSHAR

**(Hugh Grant <hugh@maths.tcd.ie>)**

This posting, a replacement for the usual *shar* and *unshar* programs, appeared in *comp.sources.misc* in early May. It was labelled as 'shar2' in the headers, but is in fact called 'xshar' in the documentation and *makefile*, so I'll refer to it as that from now on. It really only caught my notice recently when I had to transfer a large number of files to a new VAX which had not yet got any file transfer protocol up and running, so I decided to mail the files to my account. Our existing *shar* program was fine as far as it went, but it had no option for setting file sizes, or for automatically dealing with binary files. Hazily remembering this article, I scurried back to my 'News' directory and was overjoyed to see it had all the features I needed! Whew! I was afraid I'd have to send around five megabytes of data doing the encoding and splitting by hand!

*Xshar* was written by Bill Davidsen in the States, and was posted on the 10th of May, 1988. The programs come bundled as one *shar* file which contains the source and documentation for *xshar* and *unshar* as well as a *makefile*. The documentation is fairly complete, consisting of two manual pages and a *README* file. Everything compiled without a hitch and seems to work very well.

*Xshar* seems to have all the bells and whistles that anybody could ever want. It has an option for either treating everything as a plain text file, or as a binary file, *uuencoding* it before writing it to the output file. It also offers a mode where *xshar* decides for itself whether a file is text or binary and acts accordingly. This option uses 'file' to inspect each input file, so it should work fairly reliably, at the cost of forking an extra shell each time, but then that's a minimal cost compared to *uuencode* anyway. Another option allows *xshar* to split its output into several files, each with a maximum size in kilobytes. This is ideal for mailing through systems which have a 64k limit on mail messages. *Xshar* is quite intelligent about this, splitting input files over more than one output file, unlike a similar program I used which just left files larger than the limit as they were. The output files recombine the input automatically when run.

The *unshar* program is designed for stripping off the headers in mail or news messages and passing the remainder straight through to the shell for decoding. Funnily enough, the *unshar* program that comes with *xshar* seems crude compared with the one residing normally on this system, which offers a few extra options such as specifing the output directory.

In summary, *xshar* seems to be a useful tool, especially for those involved in posting files over the net. It certainly wins over the ordinary *shar* in its detection of binary files, and its ability to split input files in an intelligent way.

## comp.sources.x

**(Jamie Watson <jw@pan.uucp>)**

As most EUUG members are aware, interest in the X Window System has been increasing tremendously in the past year or so. Of course, as the net often reflects the interests of its readers, activity related to X has been increasing as well. Likewise in the tradition of the net, sources for X programs began to appear in various places on the net. This produced something of a problem, since there was no single place to watch for X sources. Recently, a new group was created specifically for posting of X sources. The name, obviously, is *comp.sources.x*.

This group is moderated, so the postings are strictly limited to source code for X programs, and occasional informational postings by the moderator. No discussions, flame wars, or other non-source postings are tolerated; we have *comp.windows.x* for that. The general quality of the programs posted has been very high; in fact,

the only shortcoming so far is that there have not been nearly enough games posted. Time should remedy this problem, though...

The following is a brief summary of what has been posted so far. More detailed reviews of some of these programs will be given in a future column.

- *xfig* – A MacDraw style line editor
- *awm* – The Ardent Window Manager
- *twm* – Tom's Window Manager
- *xwatch* – A replacement for the (awful) xbiff mail notifier
- *xipr* – A utility to dump a window to an Imagen printer
- *xmille* – The Mille Bourne game
- *xtools* – An easy way to start X programs
- *menupane* – Popup menus for the X toolkit
- *qix* – An arcade game
- *dclock* – A *real* digital clock
- *xplaces* – toolplaces for X
- *xphoon* – Show phase of the moon on the root window

Obviously, these form a very nice core of utilities and games for X. The window managers are particularly nice, since both of them do a fair number of the new window manager functions introduced with X11, such as title bars, which *uwm* does not. Also, anyone who is still suffering with *xbiff* really should get the *xwatch* program as soon as possible. The prize winner for pure aesthetic beauty in this group, though, is *xphoon*.

## Archive Sites.

There are a number of archive sites around Europe. It would be useful to build up a complete picture of what archive sites there are, what they keep and how one may access them. So if you are an archive site or if you know of archive sites mail me with the details and I will publish a list in a forthcoming newsletter. For now here are details of 3.

### Denmark

The Danish User Group (DKUUG) runs a mail-based service at diku. It is only available to EUnet users in Denmark because of accounting. It features access to the latest EUUG tape distribution which includes sources from *comp.sources.unix* and *comp.sources.games*. Also some specially collected items like GNU EMACS is available. To get in contact with this archive service, do:

```
mail diku!archive
Subject: help.
```

Archive mail enjoys a 100% surcharge compared to ordinary mail.

### England

In England there is an FTP- and mail-based server at the Imperial College London. It is for UK sites only. It is run by Lee McLoghlin and Stuart Mc Roberts. All volumes of *comp.sources.unix* are online, though some maybe in compressed form. For information about hte mail based server mail to `mail info-server@doc.ic.ac.uk`, with a message body of:

```
request catalogue
topic comp.sources.unix
request end.
```

They also store most of the GNU software, X Windows, MINIX updates, *uupc* and most other software deemed useful by the management.

### Switzerland

Jamie Watson (*jw@pan.uucp*) Service Anonymous UUCP, SnailMail of tapes and diskettes. *Comp.sources.x* games, misc, sources.

## NewsFlash!!

For those running the USENET Software (B News 2.11 patchlevel 14) there are two new News sofware packages on the way.

One Called C News by Henry Spencer, Geoff Collyer ( see USENIX Winter 1987 "News Need Not be Slow", pages 181-190 ) and News 3.0 by Eric B. Raymond. In the next issue I will include reviews of these packages.

For those who tend to resist change, or hold News 2.11 dear in their hearts ( :-) ), three patches have been posted recently to *news.software.b*. These patches are mainly small bug fixes, and speedups.

## Next Issue

Well that's the first column over. I am interesting in comments (by e-mail) if you have found anything interesting or useful. All 'flames' passed through my special mail filter to */dev/null*, this helps keep me optimistic about life, the

universe..... Remember any contributions are gratefully received.

If you use any of the GNU software, I am interested in your experiences using them.

I will gather together responses and publish them in the next issue. Also in the next column there should be a more complete list of archive sites.

## Comp.sources.unix Archive Listing

**Subject: Volume 16 (Ends January 17, 1989)**

| | |
|---|---|
| index16.1 | Introduction to comp.sources.unix |
| index16.2 | List of sources in the archives |
| conf2 | (4 parts) Multi-user conference system |
| index16.3 | Corrections to first two INFO postings |
| colm | A columnation program |
| deliver | (3 parts) Mail delivery program |
| dist2 | (7 parts, 2 patches) Larry Wall's Configure generator, etc. |
| fep | (5 parts) Front end editor program |
| fido | Watchdog for users and machines |
| hgrep | Highlighting grep filter |
| ida2 | (8 parts) IDA Sendmail kit |
| identlist | List identifiers and declarations for C sources |
| less5 | (4 parts) Less, a pager that's more than more |
| lint-proto.pch | Turn 4.2BSD lint into a prototype generator |
| lumberjack | Logfile monitor tool for Suns |
| month8.7 | (6 parts) A visual calendar and appointment system |
| narc | Archive net sources groups |
| obvious-pw | Tell if a password is "obvious" |
| pcomm2/part01 | (8 parts) Modem communications package |
| plp | (16 parts) Public lineprinter spooler package |
| psterm | (4 parts) Terminal emulator for NeWS window system |
| sao | (49 parts, 2 patches) Smithsonian Astronomical Observatory |
| spiff | (4 parts) Spiff, find approximate differences in files |
| texi2roff | Convert GNU Texinfo files to nroff/troff files |
| uniqbib | Remove duplicates from a "bib" database |
| xfmt | A simple formatter |

**Subject: Volume 15 (Ends August 12, 1988)**

| | |
|---|---|
| index15.1 | Introduction to comp.sources.unix |
| index15.2 | List of sources in the archives |
| abcd | Automatic Backup Copy Daemon |
| arc5.21 | (5 parts) ARC (PC compression program), v5.21 |
| ck | Check mailboxes for new mail |
| cshar | (3 parts, 3 patches) Tools to create and unpack shell archives |
| cu-shell | A "shell" for CU, Kermit, etc. |
| cu-shell.note | Ignore the copyright on the cu-shell posting |
| ddd | Fast, multi-process dd(1) clone |
| delta-times | Delta time routines for alarm(2) manipulation |
| dis6502 | 6502 disassembler |
| dis88 | (2 parts) Symbolic disassembler for PC/IX |
| dvipage | (4 parts) Sun previewer for TeX DVI files |
| emitc | Routine to process ctime(3) output |
| gbench | (2 parts) Graphics benchmark toolkit for X |

| | |
|---|---|
| hash8 | Hash long identifiers into unique short ones |
| ien116 | Updated IEN-116 namesever |
| inc-elim | Filter to eliminate file inclusion commands |
| index15.3 | Update on "killer" and error in "stevie" subject line |
| ioccc/part07 | (7 parts) International Obfuscated C Code Contest |
| lp-onionskin | Wrapper for System V lp, bug work-around |
| lwf | ASCII to PostScript filter |
| mcat.new | A cat(1) for mmap'able devices |
| monthtool | (2 parts) Monthly appointment calendar, for Suns |
| moontool | The moon on a Sun |
| mp | Mail pretty printer v1.4 (aka mail->postscript) |
| mush6.2.pch | Upgrade kit for Mush release 6.2 |
| mush6.3kit | (4 parts) Mush (mail user's shell) upgrade kit, version 6.3 |
| net-notify | Network message system, sort of like wall |
| newgetty | Alternate getty front-end, with speed detection |
| nroffgraphics | (2 parts) Tools for nroff graphics on dot-matrix printers |
| pages | Page accounting aide for SysVrel3.1 |
| perl2 | (15 parts) Perl, version 2 |
| ps.sun.pch | Module to make postscript interpreter work under Suntools |
| rmtlib | Remote magtape library for BSD |
| rot | Rotate text |
| ru | A users(1)-style rwho |
| siod | Scheme in one defun |
| stevie | (2 parts, 1 patch)i Stevie, an "aspiring" VI clone for Unix... |
| surun | Run commands as another (or super) user |
| touchup | (6 parts, 2 patches) A bitmap editor for Suns |
| tpscript | (5 parts) Ditroff to PostScript translator |
| twm | (4 parts) A window manager for X |
| ultrix-modem | UUCP/CU access on one modem |
| unfsd | (2 parts) User-level NFS server |
| uumailclean | Clean-up backlogged UUCP mail |
| vn.april.pch | (2 parts) VN, April, 1988, upgrade kit |
| vtree | Visual display of directory tree |
| whichtape | Tools to help find files on backup tapes |
| window-srch | Windowing search (not unlike context grep) |
| xmodem3.6 | (5 parts) Xmodem release 3.6 |
| yp-quote | Building custom Yellow Page Maps |

**Subject: Volume 14 (Ends May 20, 1988)**

| | |
|---|---|
| index14.1 | Introduction to comp.sources.unix |
| index14.2 | List of sources in the archives |
| ioccc | (5 parts) International Obfuscated C Code Contest |
| vplot | Device-independant graphics system, with drivers |
| mush6.0 | (14 parts) Mail User's Shell |
| nntp1.5 | (9 parts) Network News Transfer Protocol, version 1.5 |
| vn.nntp.pch | VN NNTP conversion kit |
| mush6.0/patch1 | Mush updates for SystemV, etc., Patch1 |
| jove4.9 | (21 parts) Jove, an emacs variant, version 4.9 |
| flex | (5 parts) Flex, a lex replacement |
| flex/patch1 | Flex, a lex replacement, Patch1 |
| 3bconnect | 3B2 Ethernet Connection and File Transfer Utility |
| rast | Sun rasterfile I/O library |

| | |
|---|---|
| splay-tree | Splay tree library |
| bsd-dyna-link | Dynamic linking package for BSD |
| cdecl2 | (2 parts) New version of Cdecl, parse C declarations |
| shellforms | (2 parts) Forms interface for shell scripts |
| sharedmem | (4 parts) Shared memory emulation for 4.2BSD |
| calc | A trig/multi-base calculator |
| pcomm | (6 parts) Dial out and terminal emulator |
| pcomm/patch1 | Dial out and terminal emulator, Patch1 |

**Subject: Volume 13 (Ends early March, 1988)**

| | |
|---|---|
| 4.3autobaud | Baud rate detection for 4.3BSD |
| atl | List jobs in at queue for 4.3BSD |
| attpc.renice | Change process priority on ATT PC |
| autoadd | Program to add users to system |
| backups | Tools to help automate backups |
| bool-eval | Boolean expression array evaluator |
| bpatch.2 | Binary file editor |
| bpatch2 | Binary patch program, ported to 80286 etc. |
| budpak | Utilities to monitor usage on system |
| casette-lbl | Cassette label formatting program |
| cfc | New version of .cf compiler |
| check | Check for mistakes in C programs |
| derez | (2 parts) Derez, remove stale files from system |
| e | Friendly front-end to vi |
| ease.pch | Patches to EASE sendmail.cf language |
| file | (2 parts) Replacement for the file(1) command |
| funcproglang | (2 parts) Functional programming language |
| iface | (2 parts) Generic user interface kit |
| korner | Convert (some) csh scripts to ksh scripts |
| labels | Program to make mailing labels |
| lit | Lit, a "better" echo |
| little-st2 | (5 parts) New release of little smalltalk |
| m4 | (2 parts) Public domaind M4 macro processor |
| mcc | Merge C code with compiler error messages |
| measures | Brute force measurement selection |
| modemcap | Hardware-independant modem routines |
| nroff-driver | Nroff driver table utility |
| pas2c.pch | Patches for Pascal-to-C translator |
| perl | (10 parts+sample) Perl, a "replacement" for awk and sed |
| perl | (2 parts) Perl patches 1 through 10 |
| printacct | Print BSD accounting file |
| process-uucp | Tools for pathalias with MMDF |
| pwget | Programs to retrieve /etc/passwd info |
| ratfor | Public domain RATFOR in C |
| rf | Rolodex-like filing system |
| rolodex | (4 parts) Screen-oriented rolodex program |
| rpc3.9 | (15 parts) Sun RPC, release 3.9 |
| rstat | Remote statistics server |
| sbbs | (2 parts) A BBS written in /bin/sh |
| sc5.1 | (3 parts) SC spreadsheet program, version 5.1 |
| sets | Perform "set" operations on command line arguments |
| slice | Split file based on patterns or line numbers |

| | |
|---|---|
| starchart | (2 parts) Starchart package patches |
| ups | BSD File delivery programs |
| vn.jan.88 | (5 parts) VN newsreader, 1/88 version |
| vt220fontedit | Font edit program for VT220 terminals |
| with | Resource allocation program |
| xmodem | (3 parts) Full featured xmodem program, v3.4 |

Subject: Volume 12 (Ends February, 1988)

| | |
|---|---|
| afio | (2 parts) Manipulate CPIO-format archive and files |
| cake | (9 parts) Cake, a make replacement |
| cnews | (14 parts) C News alpha release |
| crc.pch | CRC Graphics Package Patch#1 |
| fuser | Who's using that file? (For Unix-PC) |
| hershtools | (5 parts) Hershey font manipulation tools and data |
| index12.1 | Introduction to comp.sources.unix |
| index12.2 | List of sources in the archives |
| ln03-plot | New version of LN03 plot(3) package |
| ops5 | (5 parts) OPS5 in Common Lisp |
| musbus5.2 | (3 parts) Monash benchmark update |
| pathalias9 | (2 parts) Pathalias, version 9 |
| pdtar | (3 parts) Public domain TAR |
| postscript | (18 parts) A PostScript interpreter |
| qterm.alt | Query terminal for its type |
| starcharts | (7 parts) StarChart program and Yale star data |
| strings.coff | Find printable strings in COFF files |
| vmail | (3 parts) vmail - screen-based mail handler |
| zmodem | (3 parts) Zmodem file transfer programs |

Subject: Volume 11 (August 11, 1987 to October 6, 1987)

| | |
|---|---|
| 3bnet | (2 parts) 3Bnet utilities and printer spooler |
| avl-subs | AVL Tree subroutines |
| bsd.2.10.note | BSD2.10 available from Usenix |
| bsmtp | Batch SMTP program |
| bundle | Buffered copy to/from physical devices |
| comobj.pch | Patch for Common objects sources |
| cpmod | Copy modes/ownerships/times |
| getty-enable | Getty on/off programs for 4.[23] BSD |
| graphedit | (2 parts) Graphcs editor for Suns |
| hum.pch | Hum concordance package update kit |
| id | (3 parts) C cross-reference database system |
| inline | (4 parts) Inline code expander for C |
| inline/patch1 | Inline code expander for C, Patch1 |
| jove.pch | (4 parts) Jove upgrade kit |
| jove.pch/patch1 | Missing file from Jove update, Patch1 |
| lemming | (2 parts) Update kit for lemming editor |
| less3 | (3 parts) The 'less' pager |
| little-st | (3 parts) Little Smalltalk interpreter |
| musbus | (4 parts) MUSBUS 5.0 -- Monash University Benchmark |
| monthtool | Sunview visual calendar |
| mtools | (2 parts) MS-DOS disk tools for Unix |
| mush5.7 | (12 parts) Mail user's shell |

| | |
|---|---|
| netdata | Transfer data (and mail) between SYSV and CMS |
| number | Arabic numerals to multi-lingual natural language |
| psfig | (5 parts) Including PostScript figures in ditroff |
| qsubst | A query-replace program |
| reader.poll | Poll on copyrights |
| saver | Small SUN screen-saver |
| sc4.1 | (3 parts) Spread sheet program, sc 4.1 |
| se.pch.2 | Second update for 'SE' editor |
| smail3 | (3 parts) Smail, UUCP domain mailer |
| syslog | Development version of syslog(3), for ATT, too |
| syslog.sysv | SystemV version of syslog |
| tcsh.4.3 | (2 parts) Tcsh for 4.3 CSH |
| tcsh | (6 parts) New version of T-shell |
| tek2ps | Tektronix4014 to PostScript filter |
| templates | (6 parts) Template-mode for GNU Emacs |
| test.el | (3 parts) Test system for GNU Emacs |
| vitals | Word counts, checksums, etc. |
| watcher | (2 parts) Watcher system monitor program |
| zoo | (7 parts) File archiver programe |

Subject: Volume 10 (June 17, 1987 to August 10, 1987)

| | |
|---|---|
| agef | Show disk usage by file age |
| cbar | Another changebar program |
| cbw | (5 parts) Crypt Breaker's Workbench |
| cfc | "Compile" sendmail.cf files into EASE language |
| comobj.lisp | (13 parts) Common Objects, Common Loops, Common Lisp |
| complex-lib | Complex arithmetic library |
| copytape | Copytape, a magtape xerox (tm) machine |
| copytape2 | NEW version of magtape copy program |
| crc_plot | (6 parts) CRC Plotting Package |
| derez | Find and remove stale files from a disk |
| des | DES encryption routines and a login front-end |
| dev.fd | A /dev/fd device driver for 4.3 and NFS systems |
| ease | (4 parts) Ease translator repost |
| fastgrep | (3 parts) Reposting of world's fastest grep |
| hum | (3 parts) Bull Tuthill's "hum" text concordance package |
| ida | (7 parts) the IDA sendmail kit |
| ien116-server | IEN116 Nameserver |
| ifp | (7 parts) Interpreted Functional Programming lanuage |
| lc | An "ls" program |
| lemming | (4 parts) A graphics editor |
| logo | (6 parts) Logo interpreter for Unix |
| magtapetools | (2 parts) Magtape handling package |
| mx-macros | REPOST of Troff macros for "ACM Transactions" |
| notes-mod.pch | Patches for NOTESFILES for moderated groups |
| nrchbar | A "changebar" interface for *roff |
| ptoc | (12 parts) Pascal to C translator |
| qterm | Query terminal for its type |
| regexp.pch | Bug-fix for regexp() library |
| screen | (2 parts) BSD multi-screen manager |
| sps | (3 parts) SPS for BSD, Ultrix1.2, Sun3.x, NFS |
| sxt-sh-jobs | (2 parts) Diffs for SystemV /bin/sh job control with sxt's |

| | |
|---|---|
| top_s375 | (2 parts) Top users display, 2.1 with Symmetric changes |
| tr2latex | Translate troff to LaTex |
| x10r4.sunpch | (3 parts) X10R4 patches for Sun3/110C |

Subject: Volume 9 (March 3, 1987 to June 16, 1987 <great renaming>)

| | |
|---|---|
| assem2 | (2 parts) Generic assembler for micro's |
| bitstring | "Bitstring" package |
| elm2 | (19 parts) ELM Mail System |
| fastgrep | (2 parts) Fastest grep around |
| gwyn-dir-lib | New directory-access library |
| index9.1 | Introduction to mod.sources |
| index9.2 | Index of mod.sources archives |
| index9.4 | Change in archive sites, recent errors |
| localtime | Public Domain (Table Driven) ''localtime'' patch |
| month | (2 parts) REPOST of Visual calendar program |
| mx-macros | Troff macros for "ACM Transactions" |
| old.bad.code | Previous "obfuscated C" winners |
| printf | Printf(1), for shell scripts |
| teco | (4 parts) A TECO text editor |
| uemacs3.8b | (14 parts) MicroEMACS, version 3.8b |
| uumail.pch | UUmail 4.X patch |
| xscreen | (2 parts) Screensaver for X window system |
| xterm | (7 parts) Terminal emulator for X window system |
| zmac | (2 parts) Z80 macro cross-assembler |

Subject: Volume 8 (January 26, 1987 to March 3, 1987)

| | |
|---|---|
| ansitape | (2 parts) ANSI tape program |
| cut+paste | Public-domain implementations of cut(1) and paste(1) |
| dca2troff | Convert IBM DCA documents to troff input |
| display | Execute command repeatedly, display output |
| ease | (4 parts) Ease, a language for writing sendmail.cf files |
| fixcpio | Repair damaged "cpio -c" archives |
| foogol | A (vax) compiler for a tiny ALGOL-like language |
| getpw | Public-domain getpw*(3) routines |
| graph+ | (3 parts) A Graph Plotting Program |
| her2vfont | Hershey fonts to 'vfont' rasterizer |
| hier | Directory hiearchy scanner |
| index.1 | Accessing the archives |
| index.2 | Index of volumes one to seven |
| jove | (13 parts) The JOVE text editor |
| kurses | A program to call curses(3) functions |
| mcp | (8 parts) Account creation/manipulation program |
| microgmu | (11 parts) A Micro-Emacs variant that resembles GNU Emacs |
| multi_feed.c++ | Simultaneous multi-site news feeder in C++ |
| multivol.pch | Multivol, Patch #1 (see Volume 7) |
| pd-localtime | (3 parts) Public Domain (Table Driven) ''localtime'' |
| phoon | Phase of the moon, date routines |
| prep | (2 parts) A pre-processor for FORTRAN source |
| psfig-tex | (3 parts) Including PostScript/Mac figures in TeX documents |
| qterm | Query terminal for its type |
| se | (7 parts + 1 Patch) Georgia Tech 'se' screen editor |

| | |
|---|---|
| shrink_names | Shrink VeryLong+File.names to shorter names |
| smail2 | (5 parts) Smail (UUCP domain mailer), release 2.3 |
| soelim | A .so/.nx/.PS filter for *roff files |
| sp | (2 parts) Soundex spelling-checker |
| tabs | A tab/space conversion program |
| textool2 | (2 parts) A collection of tools for TeX users |
| trmatch | Syntax-checker for *roff |
| uk-1.4.pch | Patch for UK-1.4 mail configuration |
| unaxcess2 | (4 parts) UNaXcess Conferencing, version 1.00.02 |
| uucp.x25pad | UUCP X.25 'f' protocol and PAD dialer |
| uumail4 | (4 parts) Uumail release 4.2 |
| uutty | Bidirectional getty/login for SystemV |
| vn | (3 parts) The VN news reader |
| vtrm | (2 parts) A Unix/PC virtual terminal package |

Subject: Volume 7 (Ends January 20, 1987)

| | |
|---|---|
| 2.11news | (20 parts) 2.11 News Release |
| 4.3cpp.patch | #elif patch to 4.3BSD cpp |
| aaakeys | Ann Arbor XL key uploader |
| append | Allow additions to 'protected' directories |
| basic | (6 parts) A BASIC Interpreter |
| bpatch | Binary (file) patcher/viewer |
| cmstape | Read and write IBM VM/SP CMS dump tapes |
| csh.patch | Two CSH patches |
| des | Purported DES program in C |
| determcap | Decomposing termcaps |
| dirstack.csh | CSH tools for directory stacks |
| elm_update | (3 parts) ELM Update Kit |
| forktest | Find security holes in shell-escapes |
| getmetrics | PostScript program to generate .afm files |
| getoptprog | Getopt program for scripts |
| hostup | An alternative to the BSD ruptime command |
| idle.users | A simple BSD idle-users daemon |
| image | (5 parts) Image manipulation routines in C++ |
| index.1 | Index and Archives |
| index.2 | Complete Listing of Mod.Sources Archive |
| index.3 | Archive access and listing |
| index.4 | Index for Volume 7 and other info |
| less3 | (3 parts) New release of LESS |
| make | Public-domain MAKE |
| micro.asm | (2 parts) Generic assembler for micros |
| msdos_mk.patch | Patch to msdos_mk for Microsoft C |
| multivol | (2 parts) Multivol V1.00 - multivolume backup utility |
| nag | (2 parts) Nag reminder service |
| new_archives | Additional UUCP Access to Mod.Sources |
| patch2 | (3 parts) Release 2.0 of patch |
| paths.mk | Makefile to build UUCP paths |
| pdtar | Public-domain TAR program |
| read-vms-backs | Read VMS backup tapes |
| regex | Ed(1)/regex(3)-compatible reg. exp. package |
| remtape | Remote magtape library for 4.3BSD |
| rvi | (4 parts) Vi front-end for remote editing |

| | |
|---|---|
| safe | Limit a program's execution time |
| smail | (2 parts) Domain mailer and rmail replacement |
| sop | A .so filter for n/t/*roff files |
| sunmailwatch | A mail watcher for SUNwindows |
| tar_aids | Tools to read damaged tar tapes (tar_aids) |
| texdvi2tty | TeX DVI driver for TTY's, etc. |
| textools | (2 parts) A collection of tools for TeX users |
| tinytcp | A tiny set of TCP routines (tinytcp) |
| top2 | (2 parts) Top users display for 4.2BSD, Version 2.0 |
| tput | Public-domain tput(1) program |
| tput2 | Public-domain TPUT (corrected implementation) |
| untamo2 | Log out idle users |
| untamo3 | Log out idle users (untamo revised) |
| uucp+nuz.tulz | Erik Fair's UUCP & Usenet toolbox |
| uuencode | Uuencode and uudecode |
| vms_tools | (2 parts) Unix-like tools for VMS systems |
| vttest | (2 parts) Test VT100 Features |
| xlisp.patch | Patch to Xlisp1.6 for Pyramid machines |
| xmodem | (2 parts) Full-featured XMODEM |
| yacc.notes: | Tools to restart YACC parses |
| yacchacks | Tools to restart YACC parses |
| yearlength | Compute length of any year |

Subject: Volume 6 (Ends mid-July, 1986)

| | |
|---|---|
| intro | Introduction to mod.sources |
| untamo | Untamo, another idle daemon |
| calls.new | New calls; shows function call flow |
| vol | Create volume headers for tar |
| makekits2 | Makekits revisited |
| maildigest | Mail digest utilities |
| gr_scripts | Shell Scripts for game regulator |
| pacman.p | Apollo Pacman-like game |
| datediffs | patches for date to use elsie!ado's localtime |
| getpaths | Tools for analyzing netnews paths |
| sysVtalkA | A talk for system V.2 |
| sysVtalkB | A talk for System V |
| texdvi2lj | (3 Parts) TeX DVI driver for LaserJet+ |
| halign | Halign - line up columns |
| context | Context - generalized context printer |
| pacman.p.h | Missing files from Apollo pacman |
| less.patch | Patches for more/less interoperability |
| qterm | Query Terminal for terminal type |
| printfck2 | New printfck and manpage |
| context.1 | Manual page for context program |
| compress.xenix | Xenix patches to compress4.0 |
| fmtr.patch | Patches to fmtr |
| unrm.rm | Rm and unrm programs |
| elm | (14 Parts) Elm mail system |
| cvs | (2 Parts) CVS, an RCS fonrt-end |
| ditrev | Page reverser for ditroff |
| stringlib | X3J11/SVID/4BSD/etc string library |
| cpp.patch | Patches to 4.2BSD cpp for #elif, // comments |

| | |
|---|---|
| help | (2 Parts) Help programs |
| glob | 'Globbing' library routine |
| cdecl | English<->C translator for C declarations |
| sh.ulimit | Add ksh-style 'ulimit' to 4.2BSD /bin/sh |
| bsearchstr | Binary search for strings in a file |
| yyref | Cross-reference for Yacc |
| newbatchA | Usenet news batcher control program |
| newbatchB | Usenet news batcher control program |
| malloc | A "smarter" malloc |
| S3uuque | Uuque for System III/V in C |
| lbl | Lbl preprocessor for [nt*]roff |
| malloc.mk | Missing makefile for "malloc" posting |
| elm/Patches1 | Elm fixes for BSD, et. al. |
| Misc.Patches1 | Changes to calls, compress, ditrev, getpaths, nbatcher |
| vt100tool | (10 Parts) VT100TOOL for Sun's |
| settz.patch | Updates to "settz" data files |
| uEmacs3.7 | (12 Parts) MicroEmacs, Version 3.7 |
| bsd.ps.patch | Speed, etc., patches for BSD ps |
| watch | A multiple "tail -f" program |
| reminders | A Personal Reminder system |
| sysVdial | (3 Parts) System V generic dial routines |
| rpc2 | (11 Parts) Sun RPC Source |
| malloc.patch | Bug fix for "smarter malloc" |
| newscnt | Count unread news articles |
| less2 | (2 Parts) New version of less |
| msdos_mk | A Make for MS-DOS and VAX/VMS |
| att_which | A "which" for non-BSD systems |
| lj_filter | Filter for HP Laserjet |
| xlisp1.6 | (6 Parts) Xlisp version 1.6 |

Subject: Volume 5 (Ends late May, 1986)

| | |
|---|---|
| uEmacs30fix | MicroEMACS version 30 updates. |
| uumap | Automated UUCP maps |
| dither | Color Dither (ver 1.1) |
| retouch | Retouch(1): force changed date |
| backup | Front end for BSD dump |
| junkmail | Delete outdated mail automatically |
| smallc | (3 parts) Small C compiler version C3.0R1.1 |
| moon_sun | Sun and Moon rise/set program |
| par | More patches to par/unpar |
| smtp_send | SMTP SEND command for Sendmail |
| bmgsubs | Boyer-Moore-Gosper fast search subroutines |
| untic | Decompile terminfo description file. |
| bmfix | Fix to B/M/G for odd address optimization |
| rcsit | Prepare files for RCS (new version) |

Subject: Volume 4 (Ends early May, 1986)

| | |
|---|---|
| bm1.2 | Bm version 1.2 (blindingly fast "fgrep") |
| simplex | Simplex Curve Fitting Algorithm in C |
| chuni | Change a user's default universe (Pyramid Specific) |
| Msg | (8 parts) Screen-oriented "User Agent" mail program |

| | |
|---|---|
| sim2 | Update to "sim" (volume 3) similarity tester |
| shortc | C program to map flexnames into short identifiers |
| settz | Time conversion / time zone system |
| TVX | (10 parts) Portable editor, with "emacs" and "vi" modes |
| hershey.f77 | (2 parts) Hershey Fonts in Fortran 77 |
| rolodex | (3 parts) Rolodex database program |
| 68kdissasem | (2 parts) 68000 disassembler |
| bm1.2speedup | Speedup for bm on some machines |
| regexp3 | 2nd bug fix for regexp (volume 3) |
| tm_to_time | Convert broken-down time into time_t. |
| 68kdiss.fix | Patches to make MC68000 disassembler work on SUN UNIX |
| amiga | Amgia file browser |
| rcsit | New version of rcsit(1) - prepare files for RCS |
| hershey | (5 parts) Hershey fonts |
| egrep | More Pep for Boyer-Moore Grep |
| tc | Compile/decompile nroff driver tables (USG only) |
| regexpfix | Regexp(3) improvement |
| shortc | Shortc: sed output, and standard input |
| match1.2 | Fast grep for Vaxen |
| rlogin | 4.2bsd rlogin enhancements |
| list | List-of-numbers generator |
| client | Generic client and server commands for 4.2BSD |
| client_man | Client/server context diffs to 4.2BSD man.c |
| UK-1.4 | (5 parts) Sendmail UK-1.4 |
| ISO_Pascal | Yacc and Lex for ISO Level 0 Pascal |
| TVX | 1st batch of TVX Bug fixes |
| rpt | A program called 'rpt' |
| subnetARP | 4.3BSD IP subnet ARP hack |
| UNaXcess | (3 parts) UNaXcess (unix bulletin board) |
| uEmacs | (6 parts) MicroEmacs, v. 30 |
| travel | Travel-itinerary macros for nroff |
| aaa | The amazing awk assembler |
| sources | Two tools for organizing sources from USENET |
| load | Routines to check the load average |
| uEmacs_tc | Termcap support for MicroEmacs v. 30 sources |
| archx | Archx: suggested replacement for shar |
| ar | Portable ar: suggested replacement for shar |
| se | (8 parts) Georgia Tech 'se' screen editor |
| telnetd | Telnetd in the kernel |
| uumail3 | (2 parts) Uumail 3.0 |
| lplot | (2 parts) Lplot and quickplot |
| mail | Patches to BSD4.2 mail (SysV mailx?) |
| sticky | PostScript sticky label program |
| uEmacs3.6 | (8 parts) MicroEMACS 3.6 |
| texindex | Make an index from a LaTeX .idx file |
| chown | Improved and expanded chown/chgrp |
| calendar | (2 parts) Calendar generation program |
| strings | (3 parts) String routines |
| gr | A Game Regulator |
| printfck | Have lint check (most) printf calls |
| unparfix | Unpar compatability with Sys V (patch) |
| texindex2 | AAAAARRRRGGGGGHHHH!!!! Bugs in texindex!!! |

| | |
|---|---|
| UnaXcessfix | UNaXcess update #1 |
| xmodem | 4.2BSD XMODEM programs |
| icon | Tools for editing Sun icons |
| fmtr | Simple text formatter |

Subject: Volume 3 (Ends Feburary, 1986)

| | |
|---|---|
| G-format | (4 parts) VAX BSD4.2 compiler modifications to use G-format fp. |
| GaTech | (3 parts) Sendmail patches/configuration files from Georgia Tech |
| GaTech.upd | Updates to GaTech sendmail package |
| Hey | Hey(1) [from Unix/World, Oct. 85] |
| LA50 | Convert Nroff underlines to LA50 and VTxxx sequences |
| LaserJet | (2 parts) Ditroff HP LaserJet driver |
| MSdir | MSDOS directory access routine |
| RFS | (7 parts) Public domain, kernel-resident distributed file system |
| SPS | (3 parts) Show process status - BSD only - replacement for "ps" |
| TCtoTI | Termcap to terminfo conversion program |
| TRC | (8 parts) Expert system building tool |
| agelog | Trim log files while retaining recent entries |
| att_getopt | AT&T's public domain distribution of getopt(3) |
| badm | BSD4.2 MASSBUS disk formatter utility |
| bm | Ken Yap's changes to bm (in volume 2) |
| calendar | A calendar generator program - replaces UNIX "cal" |
| calls | C program function call cross referencer |
| calls_4.2 | Patches to calls for BSD4.2 |
| chsh | Chsh,chfn for SV (password file programs) |
| chsh2 | Chsh,chfn - Original contained security bugs. |
| clr.queue | Script to clean-up the sendmail queue |
| command | Replacement for system(3). |
| ctags | Ctags source code from Ken Arnold |
| date | Formatted date program |
| decus_grep | Public domain version of grep. |
| dial | State transition controlled communications program |
| dial.sample | Example dial script. |
| dialout | BSD4.3 Kernel changes for dial in/out on modem lines |
| dtree | Directory heirarchy display program for 4.2 |
| ff | (2 parts) Simple text formatter for flexible uniform formatting |
| give | Give away ownership of files (System III/V specific) |
| hdiff | Source file compare program |
| head | Public implementations of head(1) and ctags(1) |
| help | VMS-style help facility |
| hyphen | Program to enhance troff's hyphenation capability |
| idledaemon | Yet another idle login checker (BSD 4.2 only) |
| ieee | (6 parts) IEEE Floating Point Calculator (in Pascal) |
| infer | Inference engine + demo |
| laserjet | BSD 4.2+ lpd printcap/spooler for LaserJet printer |
| lcat | Troff->laserjet filter package (uses vfont files) |
| lcat2 | Troff->HP Laserjet filter - newfonts.c |
| less | Similar to more(1) but better |
| lib_term | Datum entry using termcap |
| libc_term | Datum entry using curses |
| llib-dbm | Lint library for the DBM routines (BSD systems) |
| man | Compiled version of the 'man' program for System V |

| | |
|---|---|
| match | Faster than bm (VAX only!) |
| mdump2 | Revised mdump, the multiple dump per tape utility |
| modgen | Extract usenet moderator list from postings |
| modnotes | Notes (1.7 or later) updates for moderated groups |
| modula_pp | Pretty printer for Modula-2 written in Modula-2 |
| newspace | Determine newsgroup disk usage |
| nwho | Enchanced "who" program (uses termcap) |
| okstate | Kermit archive on OKSTATE; uucp access information |
| pathalias2 | (2 parts) Pathalias, the mod.map database path optimizer |
| pretty | Pretty printer in lisp + columnator in CLU |
| prune | Prune tops of line-oriented log files |
| rcsit | A program to prepare files for RCS. |
| regexp | Regular expression routines (like System V regexp(3)) |
| regexp2 | Bug in regexp, and fix |
| rename | A companion to restor (automated inode mapping) |
| rmsecure | Source for a safe "rm" (csh, BSD only) |
| rsend | BSD network communications program (like write & talk) |
| scpp | (2 parts) A selective C preprocessor - clean up your C files. |
| sim | Software similarity tester for C programs |
| sndml.mods | Mods to sendmail to provide translation tables |
| suntools | Improved version of Sun's window manager (suntools) |
| swho | Screen based who (uses curses - continuous update) |
| tc | Control your terminal via termcap in shell scripts |
| telno | Permute telephone numbers into letter equivalents |
| texchk | (2 parts) Syntax checker for the LaTeX TeX macro package. |
| times.awk | Uucp info from LOGFILE (awk script) |
| ttype | Typing tutor - BSD specific |
| ttyuse | Creates a Summary of daily Terminal usage |
| turbo_patch | Fix to turbo_tools, SHELL.PAS transmitted with error |
| turbo_tools | (2 parts) Turbo Pascal version of "Software Tools in Pascal" |
| uuhosts4 | Grab mod.map data for later use version1.69 |
| uumail2 | Pathalias-based uucp mailer, release 2 |
| uumail2.fix | Small fix to uumail release 2 |
| vtem | A VT100 emulator based on termcap |
| wm.new | Window manager built on top of curses |
| xargs | Execute a command with many arguments |

Subject: Volume 2 (End roughly August, 1985)

| | |
|---|---|
| Smail1 | Update to smail (in volume 1) |
| access | Kernal Hacks for access control lists |
| basic | (4 parts) A BASIC interpreter in C (needs work) |
| bgrep | Boyer-Moore based fgrep like program |
| bm | Much faster Boyer-Moore |
| bm2 | Various bm updates |
| choose | A program to select lines at random |
| compress | (2 parts) Compression 4.0 program better than pack or compact |
| cshar3 | Update to C shar (volume 1) |
| cpg+mdep3 | Cpg revisited (C formatter - original in volume1) |
| makekits | Software "kit" generation script |
| mdump | Multiple dump per tape utility (see update in volume 3) |
| remote | Remote mag tape routines |
| remote2 | Small patch to remote tape library |

| | |
|---|---|
| rtar | Diffs to tar to use a remote system's tape drive |
| runtime | Runtime memory allocation for multi-dimensional arrays |
| tools | (6 parts) Software Tools in Pascal |
| uroff | Nroff underlining |
| window | (4 parts) BSD 4.2 window manager + Patches to Curses |
| wire | (2 parts) Wirewrap program. |

Subject: Volume 1 (Ends June 1985)

| | |
|---|---|
| ANSI.C | Yacc and Lex for 11/12/84 draft of ANSI C |
| Smail | A smart net mailer - a front end using pathalias data |
| UK-1.1 | (3 parts) UK-1.1 Sendmail Configuration Package |
| Xlisp1.4 | (4 parts) Lisp written in C with object oriented extensions |
| bed | Editor for binary files. Front end for ascii editors |
| bourne | (9 parts) Bourne shell enhancements (history,tilde,job control) |
| cforth | (3 parts) Forth Interpreter written in C |
| checkin | Editor interface for RCS logs |
| cpg+mdep | Cpg - C formatter, mdep - make dependency generator |
| cpp | (3 parts) C preprocessor suitable for use with Decus C |
| cshar | Shell archive builder (shar) written in C |
| cxref | C cross referencer |
| diffc | Contextual diff (diff -c) for Bell systems |
| dynamic | Dynamic loading code for 4.2bsd |
| getopt | Public domain getopt(3) |
| lbgm | Newsgroup archiving (Little Bird Gave Me) |
| newshar | The Connoisseur's Shar, version 2 |
| newsweed | A program to delete unwanted news articles |
| patch | A program to apply diff format output to update files (1.3) |
| pcurses | (11 parts) Public domain Terminfo/Curses (needs a little work) |
| rfc_882 | RFC 882 - Domain Names - Concepts and Facilities |
| rn | (9 parts) Rn news reading program, version 4.3 |
| rpc | (10 parts) Sun "Remote Procedure Call" source code |
| sendmail.cf | GaTech Sendmail configuration |
| uucpanz.V7 | A uucp status program (V7, BSD version) |
| uucpanz.S5 | Uucpanz for System V |
| uuque | A uuwizard's utility for uucp queue snooping |
| vnews | (7 parts) New reading program for 2.10.2 news |
| vstr | Dynamic string package |
| xfernews | Uucp traffic batching system |
| xref | A general purpose cross reference utility |
| vnews.1 | Manual page for 2.10.2 vnews(1) |
| readnews.1 | Manual page for 2.10.2 readnews(1) |
| expire.8 | Manual page for 2.10.2 expire(8) |

# USENIX Association News for EUUG Members

*Donnalyn Frey*
*donnalyn@uunet.UU.NET*

*Frey Communications*

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

## 1989 Winter USENIX Association Conference

The USENIX Association's 1989 Winter Conference Conference, held January 30 – February 3 in San Diego, California, was a success. Over 2,000 people attended the conference. The first two days were devoted to tutorials, with the next three days for technical sessions. William T. O'Shea, Vice-President of Product Development at AT&T, presented the Keynote Address.

Tutorials were presented on many subjects, including new tutorials on 4BSD TCP/IP Performance Improvements, Open Systems Interconnection (OSI), Security Issues in a Distributed UNIX Environment, Using PostScript as Yet Another UNIX Tool, Network Computing System and Architecture, and Object-Oriented Design on UNIX.

Technical papers were presented on distributed systems, file systems, operating systems, window systems, internetworking, objects and memory, processes, security, and two special interest sessions. A Work in Progress session was held on Thursday afternoon to preview new work not yet ready for publication.

### Viruses, Worms, and Other UNIX Pests

A special session entitled "Worms, Viruses, and Other UNIX Pests" was held at the conference. The session focused on recent UNIX worm and virus papers and included discussion of the recent problems. The session contained a paper by Donn Seely entitled "A Tour of the Worm" and a paper by Eugene Spafford on "Some Musings on Ethics and Computer Break-ins".

Earlier in the conference, Tom Duff of AT&T had presented a paper entitled "Viral Attacks on UNIX System Security", accepted as a conference paper long before the recent worm attacks.

### Open Software Foundation & UNIX International

The USENIX Association hosted information sessions by both the Open Software Foundation and UNIX International, Inc., formerly the Archer Group, at the conference. The OSF session was held Tuesday evening, January 31 and the UNIX International session was held on Thursday evening, February 2.

This was the second time the OSF held an information session with USENIX Association conference attendees. The OSF panel included Alex Morrow, director of strategic planning, Ira Goldstein, director of the research institute, Ellis

Cohen, member of the User Environment Component team, and Eric Shienbrood, operating systems project leader. They provided conference attendees with an OSF technology update, including the initial User Environment Component offering, an overview of the selection process, a review of the delivery schedule and development period evaluation, and a summary of the licensing options.

This was the first time UNIX International held an open meeting with UNIX users. The UNIX International panel included senior technical representatives from UNIX International and member companies. They discussed the role, objectives, and purpose of the new organisation in guiding UNIX development at AT&T, how members can participate in the process, and answered questions from conference attendees.

To request copies of the Winter 1989 USENIX Association Conference Proceedings, send email to *{uunet,ucbvax}!usenix!office*.

## 1989 Summer USENIX Conference and Exhibition

The 1989 Summer USENIX Conference and Exhibition will be held in Baltimore, Maryland on June 12 – 16, 1989. The first two days will be devoted to tutorials, with the next three days for technical sessions. For further information on the conference, contact the USENIX conference office.

## Workshop on Software Management

The USENIX Association will be holding a Workshop on Software Management on April 3 – 4, 1989 at the New Orleans Hilton and Towers in New Orleans, Louisiana.

The workshop will be concerned with the management and processing of source; the discipline of managing, maintaining, and distributing software. The workshop is planning presentations and discussions of software management including release engineering, configuration management, installation tools and techniques, construction tools and techniques, source code control systems, and testing.

For further information on attending this workshop, contact the USENIX Conference Office.

## Workshop on UNIX Transaction Processing

The USENIX Association will be holding it first Workshop on UNIX Transaction Processing on May 1 – 2, 1989 at the Pittsburgh Hilton Hotel in Pittsburgh, Pennsylvania.

The workshop plans to include papers and discussions on:

— Transaction Integrity
— Two-Phase Commit
— Distributed Transactions
— Client-Server Transaction Models
— Transaction Queuing and Scheduling
— Data Entry Systems
— Transaction Benchmarking
— Transaction System Performance Modeling
— Operating System Support for Transaction Systems

For further information on attending this conference, contact the USENIX Conference Office.

## Distributed Processing Workshop and Graphics Workshop

The USENIX Association will be holding a Distributed Processing Workshop in Fort Lauderdale, Florida October 5 – 6, 1989.

The fifth Graphics Workshop will be held in November or December of 1989. For information on the call for papers for these workshops, contact the USENIX Association Office at P.O. Box 2299, Berkeley, CA 94710, at +1 415 528 8649, or by email at *{ucbvax,uunet}!usenix!office*.

## Further Information on Conferences and Workshops

If you need further information on upcoming annual USENIX Association conferences or workshops, contact the USENIX conference office at P.O. Box 385, Sunset Beach, CA 90742 USA. The conference office can provide you with information on the annual Computer Graphics, Large Installation Systems Administration, UNIX Security, and UNIX and Supercomputers workshops, as well as the new workshops on Software Management and Transaction Processing. The office can also provide information on the annual C conference and the semi-annual technical conferences.

# UNIX Clinic

*Colston Sanger*
*colston@olibc1.oliv.co.uk*

*Olivetti International Education Centre*

Colston Sanger is a lecturer at the Olivetti International Education Centre, Haslemere, UK and a visiting lecturer in the Faculty of Engineering, Science and Mathematics at Middlesex Polytechnic. In his spare time he is an art historian and is the organiser of the J.L.Agasse exhibition which has just opened at the Tate Gallery, London.

## No space on integral hard disk drive 0, partition 0

You're out of space. What's more, the message on the console says you're out of space on the root partition. What now?

First of all, don't panic: nothing is going to crash. But you are going to have to find some space before you'll be able to do anything sensible with your system. If you can, check for and delete any junk files (`core`, for example) in the root directory. Otherwise, try zeroing out `/etc/wtmp` and `/etc/utmp`. If all else fails, on an **Olivetti-AT&T 3B2**, you could try removing `/unix` (it will be rebuilt automatically when you reboot the system).

Once you've got yourself some breathing space, it's time to analyse what happened and to ensure that it doesn't happen again. The best way, of course, is to prevent it altogether. As a well-seasoned system manager, you are supposed to know pretty much from hour to hour how much free space you have (the *df* command, remember?) and, really, the root partition shouldn't change very much at all. Apart from `/etc/utmp` and `/etc/wtmp` (which log user and accounting information for commands such as *login* and *who*) nothing much else changes – assuming that you have a separate `/tmp` partition. (*vi*, *cc* and a few other commands write their temporary files in `/tmp`,† so it's certainly a good idea to have a separate partition mounted as `/tmp`: 10 – 20 MB should be ample.)

By the same token, it's probably best to keep your user directories out of `/usr` – otherwise you'll be running out of space there as well. Instead, have separate partitions for each group. On this system, for example, users' login directories are in `/staff` for lecturing staff, `/centre` for the office ladies and `/course` for course delegates (spread over three machines connected with *RFS*, but the principle holds).

The complete picture is then:

```
/
/tmp
/usr
/staff
/centre
/course
```

† Properly speaking, they reference the environmental variable *TMPDIR* – by default `/tmp` in System V Release 2, `/usr/tmp` in Release 3.

It's also tidier, in that there is a clear separation between the standard UNIX System distribution and user files. Backup is easier too, because we can backup a whole partition and know that we have captured everything – without the sort of 'pick and mix' that would be involved if user directories were scattered all over `/usr`.

What else can you do in the way of preventive system management to keep track of free space? Well, on this system, I do a lot of silent tidying-up with *cron*. Here are the relevant entries from the root `crontab`:

```
# root crontab
#
# cleanups
# clear /etc/wtmp every morning at 7 am
0 7 * * * cp /dev/null /etc/wtmp ; \
chown adm /etc/wtmp ; chgrp adm /etc/wtmp
#
# cut down /usr/lib/cron/log at 4 am, Mon, Wed, Sat
0 4 * * 1,3,6 tail -50 /usr/lib/cron/log > /tmp/cronlog ; \
cp /tmp/cronlog /usr/lib/cron/log; rm -f /tmp/cronlog
#
# cut down /usr/adm/sulog at 4 am, Wed, Sat
0 4 * * 3,6 tail -50 /usr/adm/sulog > /tmp/sulog ; \
cp /tmp/sulog /usr/adm/sulog ; rm -f /tmp/sulog
#
# get rid of any core-dumps, dead.letters at 6 am, Mon-Fri
0 6 * * 1-5 /bin/find /  -name core -o -name dead.letter ) \
-mtime +3 -print 2> /dev/null | xargs /bin/rm -f
#
# remove any leftovers in /tmp at 7 am, Mon-Fri
0 7 * * 1-5 rm -f /tmp/[Qq]* /tmp/*.dlf \
/tmp/calendar.* 1>&2 > /dev/null
#
# clear out everything once a week, at 6 am, Sat
0 6 * * 6 rm -f /tmp/* /usr/tmp/* 1>&2 > /dev/null
# get rid of preserved (but unused and forgotten)
# vi edit buffers
0 6 * * 6 rm -rf /usr/preserve/* 1>&2 > /dev/null
#
# Look through for old stuff and mail polite messages
# to users at 7 am, Sat
# 0 7 * * 6 /staff/colston/admin/old_files 1>&2 > /dev/null
```

It's probably worth going through this in some detail. The first thing I do is make sure that various log files are cut back periodically, otherwise they just grow and grow. `/etc/wtmp` is a data file, so it is zeroed out after the system accounting has finished processing. On the other hand, `/usr/lib/cron/log` (the log of all commands run by *cron*) and `/usr/adm/sulog` (the log of all successful or unsuccessful attempts to run the *su* command) are text files: they are simply *tail*-ed – but not until some other reporting scripts have checked them for anything suspicious.

Everyday, I search the whole filesystem for `core` and `dead.letter` files older than three days and quietly remove them. The rationale here is that any coredumps or undeliverable mail that are useful are likely to be current. Programmers, for example, may be debugging with *sdb* and a current coredump, but anything else is junk and has been forgotten. I also remove any leftovers in `/tmp`. (We run some third-party software and locally-developed programs that are not quite as punctilious about cleaning up as perhaps they should be.)

Then, once a week, I clear out everything in `/tmp` and `/usr/tmp`, and also get rid of unused and forgotten *vi* edit buffers saved under

/usr/preserve.

One other thing: when space is very short (or when I want to be especially objectionable) I run the script old_files. As you can see, it's commented out at the moment, but what it does is look through the user directories for old files with names like temp or mailmerge.out, then mails polite reminder messages to their owners asking that the files be removed. Anyway, here's the script:

```
# old_files
#
# Script to find old core, etc, files and send
# mail to their owners. The file old_ls will be
# the full list of old and bogus files.
#
# Adapted from a script posted to USENET,
# but I forget who originally wrote it.

# setup
PATH=/bin:/usr/bin
HOST=`uname`
FILESYSTEMS='/centre /staff'
ADMIN=colston
ADMIN_NAME='Colston Sanger'
trap "rm -f ${TMPDIR}/old_ls ${TMPDIR}/old_unknown ; \
exit" 0 2 3 15

cd ${TMPDIR}
rm -f ${TMPDIR}/old_ls ${TMPDIR}/old_unknown

# How old files should be before we complain.
AGE=10

# You might want to tweak the predicates in here.
# Typically, whatever names people use for junk files
# on your system.
find ${FILESYSTEMS} -type f \
    \( -name a.out \
    -o -name '*,*' -o -name '.*,*' \
    -o -name '*~*' -o -name '.*~*' \
    -o -name '*.out*' \
    -o -name '*.old*' -o -name '*.bak*' \
    -o -name 'temp*' -o -name 'junk*' \
    -o -name fred -o -name jim \) \
    -mtime +${AGE} -print | \
    xargs /bin/ls -l >old_ls

# Assuming a standard 'ls', third field is the users.
# Loop over list of all users. The grep commands ignore
# filenames; be careful if your 'ls -l' uses tabs.
#
for USER in `awk '{print $3}' < old_ls | sort -u`
do
```

```
    case ${USER} in
    root|bin)       continue ;; # system logins - do nothing
    helmut|recept) continue ;; # Don't bother these people
    [0-9]*)      grep " ${USER} " old_ls >> old_unknown ;;
    *)              (
                    echo "You have the following mouldy \
old files on ${HOST}:\n"
                    grep " ${USER} " old_ls
                    echo "\nIf you have finished with them, \
it would be nice\nto remove them, freeing up the disk space.\n
Many thanks,\n
${ADMIN_NAME}
Your friendly system manager\n"
                  ) | \
                  mailx -s "Old files on ${HOST}" ${USER} ;;
    esac
done

if [ -s old_unknown ]
then
    (
    echo "Found unknown old files on ${HOST}:"
    cat old_unknown
    ) | \
    mailx -s "${HOST}: No owner for these ancient files." ${ADMIN}
fi
```

I also have something here called *diskhog*, written by Dave Settle and, I believe, posted to the net sometime last year. I haven't actually installed it – it seems a little drastic. It's based (rather loosely) on the disk quota mechanism of Berkeley UNIX, but uses *du* to count the number of blocks used by each person. Here is an extract from the README:

> I've found this very useful on my system, where it encourages people to keep their disk usage down, without all the hassle of having to complain to them personally. The package runs a check each night on each user, and sends mail to those anti-social people exceeding their allowance.
>
> If, after a suitable number of warnings, these people do not remove their extra files, their next login receives a restricted shell, where the only available commands are those for removing and backing up files.
>
> You can decide on the range of commands available, and place these in the directories /diskhog and /usr/diskhog, so the restricted shell can be as nasty as you like. (You could be really vicious and just have *rm*, but it's probably advisable to have some commands to save to tape and/or floppies.)

Anybody who wants it is welcome to contact me.

### A system manager's toolkit

Once you have disk usage firmly under control, you're ready for the next step: keeping track of what your users are up to, and how well your system is coping with the load being put upon it. After all, your job is to provide your users with a computing service. If that service isn't available when 'they' need it – because one of them is making unreasonable demands on it – they will complain.

There are, to my mind, five commands that form a system manager's basic toolkit on a stand-alone UNIX system: *who*, *who -u*, */etc/whodo*, *ps -ef* and *sar*. If you get into the habit of running them periodically, throughout the day, you will be able to anticipate and, maybe, head off many problems before they actually occur. What sort of problems? The sort that arise as questions:

**Question**: *Why is fred logged in when he is supposed to be on holiday?*
**Answer**: Possible security breach – someone else knows fred's password?

**Question**: *Why is alice logged in at six terminals?*

**Answer**: Good question – why indeed?

**Question**: *Why is mutt still logged in – he left early today?*

**Answer**: The silly person probably forgot to log off.

**Question**: *Why has jeff got six vi sessions running?*

**Answer**: Because he's been doing shell escapes from within *vi* and has probably forgotten where he is – may even be editing the same file over again!

**Question**: *Why is the system so slow today?*

**Answer**: Because carol is doing a 'super-*make*' of some new statistical software, richard is compiling a *Fortran-77* image-processing program, william is *troff*-ing, sam is running a huge *lisp* object, henry is playing *rogue*, there is a *C* programming course going on downstairs, the office ladies are running the **Informix** database – and, frankly, how much more do you think you can squeeze out of a 4MB system? (Definitely losing cool here!)

Ahem... sorry about that.

## Finally

Finally, and as an addendum to last issue's column on common (but tricky to solve) problems, one other: the case of the file with the impossible name.

It happens occasionally – particularly with software that puts the terminal into 'raw' mode, such as the infamous **Whizzo-Office+** – that users create files with 'impossible' names, i.e., containing backspaces or other control characters. What can be done about it?

It depends. If all you want to do is remove the file (because it's junk anyway), then the easiest way is:

```
# cd to the directory containing the file
cd /staff/mutt
# If you have the -b or -q options
# to ls that display non-graphic characters
ls -b
# Be careful!
rm -i *
```

Alternatively, you could try:

```
cd /staff/mutt
# Find out the i-number of the file
ls -li
find . -inum NNN -exec rm {} \;
```

But what if the file contains valuable information that the user wants to keep? Nothing special: just a variation on the *find* example above:

```
cd /staff/mutt
# Find out the i-number of the file
ls -li
find . -inum NNN -exec mv {} new_filename \;
```

Or, if you really want, here's a *C* program to do the same thing:

```
/*
*     fix.c - fix filenames containing funny characters
*     From: UNIX/WORLD, February 1987
*
*     Usage:  Do an ls -li to find out the inode,
*     then fix [ inode ]
*/

#include <stdio.h>
#include <sys/types.h>
#include <sys/dir.h>
#include <ctype.h>

#define FIXOUT "fix.out"     /* output filename */

main(argc, argv)
int argc;
char *argv[];
{
    struct direct dbuf;
```

```
    int fd, fix_ino;

    if (argc != 2 || !isdigit(argv[1][0])) {
        fprintf(stderr, "Usage: %s inode-number\n", argv[0]);
        exit(1);
    }

    fix_ino = atoi(argv[1]);
    if ((fd = open(".", 0)) < 0) {
        fprintf(stderr, "Can't read current directory.\n");
        exit(2);
    }
    while (read(fd, (char *)&dbuf, sizeof(dbuf)) > 0) {
        if (dbuf.d_ino != fix_ino)
            continue;
        unlink(FIXOUT);         /* delete former fix */
        link(dbuf.d_name, FIXOUT);     /* new link */
        unlink(dbuf.d_name);     /* delete old link */
        printf("Fixed it: inode-number = %d\n", fix_ino);
        printf("old name was: %s\n", dbuf.d_name);
        printf("new name is: %s\n", FIXOUT);
        exit(0);
    }
    close(fd);
    fprintf(stderr, "Can't find it.\n");
    exit(3);
}
```

### Next issue

That's it. In the next issue I'm planning a complete change of subject: *regular expressions*, hopefully with lots of practical examples.

If anybody has any *awk*, *grep* or *sed* scripts they're particularly proud of, I'd be interested to see them. You know the address.

# EUnet

*Peter Houlder*
*uknet@ukc.ac.uk*

*Computing Laboratory, University of Kent*

Peter Houlder has been in the Computing Laboratory at the University of Kent for the last 4 years and looked after day to day UKnet admin work in the last 3 years.

He graduated in Geography from Kings College, London in 1970 and then spent 9 years in business – dropping out in 1979. He then spent a year touring North, Central, South, and Carribean America, became interested in archæology and spent three years exclavating in Britain and Europe.

Two Masters degrees, the first in Archæological Sciences and the second in Computer Science, followed in successive years. Maggie in the meantime reduced archæological funding, so he arrived in 1984 kicking and screaming into the world of Computing. He has since got to quite enjoy it.

He is married with two labradors.

## Introduction

As I write it is three days since I received the dreaded "... Your contribution for the EUnet column is required by ..." The problem is nobody had sent me anything to include. Since I started this column (7 issues ago) the EUUG newsletter has had articles on the network from the following countries:

| | |
|---|---|
| aut | |
| bel | Vol8, No 4 1988 |
| che | |
| deu | Vol 7, No 3 1987<br>Vol 8, No 4 1988 |
| dmk | |
| esp | |
| fin | Vol 8, No 2 1988 |

| | |
|---|---|
| fra | Vol 7, No 4 1987 |
| gbr | Vol 7, No 2 1987 |
| grc | |
| irl | |
| isl | |
| ita | |
| lux | |
| nld | Vol 7, No 3 1987 |

| | |
|---|---|
| nor | |
| prt | |
| swe | Vol 7, No 3 1987 |
| yug | |
| eunet | Vol 8, No 1 1988<br>Vol 8, No 3 1988 |

I realise that other articles have been placed prior to this column, but some up to date information, as a paragraph or whole article on any networking issue would be nice. Austria, Denmark, Ireland, Italy and Norway all have established networks with many sites, so information would be appreciated (to email address as above). It would also be nice to hear from the other countries about problems of setting up networks, future plans requests for help or any other topics. (End of Advert.)

The information below is strictly applicable to UK only users as our information servers are not authorised to send international mail, but I thought if I explained our on-line services in the UK, that it will hopefully inspire other countries to write about similar services in their countries. (Second End of Advert.)

## On-Line Information Within UKnet

We now have three methods by which you can get automatic systems to send you information in the mail from UKC's machine.

### Mailing information@ukc

The first method of using the information system is really designed to send simple files It is mostly used to store fundamental information about the network. To obtain an index of all the available information in the system UK users type:

```
mail information@ukc
Subject: index
```

The index is simply a file and all the other entries are obtained in a similar fashion by supplying a particular Subject line. We also put certain special information in these files like the ARPA INTERNET worm reports and on-line forms to join both the UKUUG and the UKnet network.

### Mailing info-server@ukc

The second method is more complicated but is more flexible. This is the information server. The info-server provides a way for users to obtain public domain software and other information from the UKC gateway. To access it:

```
mail info-server@ukc
```

This information server will mail the files in response to a request contained in the mail message that you have sent it. Requests are of the form:

```
request: subject
topic:   topic within that subject
request: end
```

As an example suppose you want to be mailed information about 'mailing-lists' in the subject 'uknet'. You would send a message of the form:

```
request: uknet
topic: mailing-lists
request: end
```

and the mailing-lists information would be mailed back to you. The 'uknet' request points at the same information used to run the information system. A list of the 'top-level' requests can be obtained by sending the following request to the info-server:

```
request: index
topic: index
request: end
```

Within a request subject, an index and some help information are both available. These would be (using catalogue as the subject example):

```
request: catalogue
topic: index     (or help)
request: end
```

All blank lines are ignored, and the 'request end' is optional, however if it is omitted and there are other lines in the message an automatic error message will be sent to you.

The UKC info-server contains all the files which can be returned by the information system, several public domain systems including the news code and *comp.sources.unix*, the news-group used to transmit useful UNIX sources.

Please note if the 'Subject:' line begins with 'Request', it is expected to hold all the information, using ';' to separate logical lines. Thus the following will have the same effect as the previous example:

```
mail info-server@ukc
Subject: request: catalogue;
topic: index; request: end
```

The current list of 'Requests' that can be requested from the UKC info-server are:

| | |
|---|---|
| index | This index |
| catalogue | Information on who has what and where in fixed format. |
| sources | Public domain shared software system (the actual files) |
| uknet | UKnet information files |
| comp.sources.unix | Copies of the UNIX source archive (from volume 8) |
| siteinfo | Access the UUCP maps on a site basis. |
| uucpmap | Access to the World UUCP map files |

Finally it should be emphasised that UKC is only one information service. It is linked to the information servers at The Universities of Cambridge, Nottingham, Glasgow and Imperial College London. If a user requests the catalogue index (as in one of the examples above) the user can see what information is held by other sites. The user can then mail that information service directly or remail UKC, who will automaticall forward the request. All the info-servers can perform this automatic rerouting.

## Mailing netdir@ukc

The final information service is a simplistic method of providing a USENET directory. It is used to obtain information about a particular USENET site. To obtain a map of a particular site:

```
mail netdir@ukc
Subject: ariadne
```

Back will come:

```
UUCP map data for site ariadne:

System name          :   ariadne
System type          :
Organization         :   Cretan Research Center
Contact person       :   Kostas Vassilakis
Electronic Address   :   ariadne!kostas
Telephone            :   +30 81 221171
Postal Address       :   P.O. Box 527, Iraklion, Crete, Greece
Longitude/Latitude   :   25 09 E / 35 20 N
Remarks              :
Usenet (news) links  :
Last editor & date   :   mcvax!piet 841215,870609

Connect list etc.    :

  ariadne ermhs(HOURLY*4), theseas(DAILY)
```

# UNIX is Chauvinistic

*Dominic Dunlop*
*domo@sphinx.co.uk*

Dominic Dunlop, surprisingly described by the London Times as a "software wizard", was co-founder of Sphinx Ltd., a company which distributes application software for the UNIX operating system throughout Europe.

Have you got your CD-ROM copy of the *Oxford English Dictionary* yet? Well, neither have I, so I still have to reach for the shelf, biceps tense, to pull down the first volume of the *Shorter Oxford*. What does it tell me?

> ‖ **Chauvin** [Fr.; from Nicholas *Chauvin* of Rochefort, a veteran soldier of the first Republic and Empire, whose demonstrative patriotism was ultimately ridiculed by his comrades.] Popularized[1] as name of a character in Cognard's vaudeville, *La Cocarde Tricolore*. 1831.
>
> **Chauvinism** 1870. [-Fr. *chauvinisme* (1843), f. prec.; see -ISM.] Exaggerated and bellicose patriotism. So **Chauvinist. Chauvinistic** *a*.

What does this tell us (apart from the fact that I'm too lazy to work out how to persuade *troff* to print the symbols which indicate the phonetic pronunciation)? That, back in the last century, we English appropriated the name of a Frenchman whom his countrymen regarded as a joke, using it to name a concept which was just too good for the French to be allowed to keep it to themselves.

What does this have to do with the UNIX operating system? Well, it turns out that most application software is riddled with cultural chauvinism of one sort or another. For example, a program which can communicate only in English ignores the needs of those who speak other languages; a program which prefixes currency amounts with a dollar sign, and which insists that they have two decimal places, is of limited use outside the USA. If that same program is concerned with taxation, and understands sales tax, but not value added tax, the whole thing becomes about as useful to a European customer as a chocolate teapot.

How has the software industry been able to get away with such chauvinism? Firstly because many individual markets in European countries are large enough – and rich enough – to support local developers willing to produce software tailored to local needs. For example, if you are a Dane wanting an accounting package which speaks Danish, you can certainly find one. It is almost bound to cost more than a package targeted at the far larger British market, but it is available.

---

1. This† must be one of the words we English really are supposed to spell with a *z* – even if *Spell -b* ("OED notwithstanding") thinks otherwise.

† Although even the most *un*progressive of dictionaries would seem to allow both 'popularize' and 'popularise' as legitimate British spellings, it is EUUGN's policy to use the 's' alternative wherever this is the case. Dominic has been allowed the liberty of a 'z' just to make his point.

Similarly, you can get Danish databases, Danish word-processors, and more. You just have to put your hand rather more deeply into your pocket than you do if you are prepared put up with software which speaks English. These packages tend not to be locally-developed, but instead have been developed for another market and subsequently "localized". The localization costs money – money which must be recovered from sales in the Danish-speaking market. You may be prepared to save money by putting up with software which is not localized.

Localization also takes time, resulting in delays in the availability of improved versions of the software. Unlocalized software is often preferred because it offers facilities that no local product can match. This is particularly true in the markets for MS-DOS and UNIX software, which are dominated by US-based software authors – authors who were able to build sizable markets in Europe by selling products originally written for their own home market. Only when when winning new customers begins to require speaking their language, does work on localization begin – work funded out of the revenue from continuing sales of the English language version.

Of course, UNIX itself is a case in point. It is only in the last couple of years that AT&T has come up with an "internationalised" UNIX – a version which allows localizations for particular markets to be added.

It is also only in the last couple of years that widely-accepted tools for the production of internationalised applications have appeared. Most notable among these is X3.159-198, the ANSI standard for the C language, although the IEEE's 1003.1 operating interface standard also has a short chapter devoted to the subject. Before these developments, any application which supported localization had to be written without the benefit of standards, and consequently each developer solved the problem in a different way.

Even now, while the standards cover the basic issues, they do not address important areas such as the means of access to sets of error messages in different languages. Thus, if you want to run a localized database and a localized word-processor on top of your preferred localization of UNIX today, you may have to go through three set-up procedures. Only when there is widespread agreement on – and use of – mechanisms for internationalisation will the need for this unnecessary duplication be a thing of the past.

The coming of 1992, with its removal of internal barriers to trade in the European Community, is exciting the interest of politicians and pundits. But if there is to be a truly open market in applications software, it has to involve programmers as well. It is programmers who will have to work out how to write software which does not suffer from chauvinism. I see late nights ahead for all of us!

# Overview of UNIX System V Release 4.0

*Janet Davis*
*janet@uel.uucp*

*UNIX Europe Limited (UEL)*
*International House*
*Ealing Broadway*
*London, UK*

## UNIX System Evolution

The UNIX System became commercially available for the first time in 1978, with the licensing of Version 7 of the operating system.

Since then standards efforts and technology advances have resulted in many versions of the UNIX System, and brought maturity to the UNIX System industry.

Three major variants have evolved:

- Berkeley System Distribution (BSD)
- The XENIX System
- UNIX System V

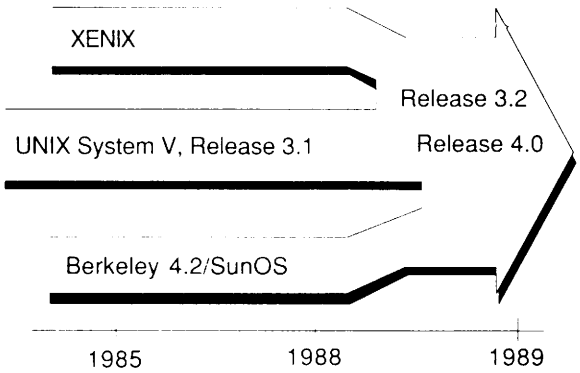



**Figure 1: UNIX System V Unification**

UNIX System V Release 4.0 will provide the first consolidation and implementation of these variants.

This evolution has seen a significant growth in units, revenue and software availability. For example:

- There are over 180 hardware companies and 700 software companies that sell UNIX System-based products.
- the 1987 /usr/group UNIX System products directory lists over 3164 products from 826 vendors, 170 of which are outside the United States, making the UNIX System marketplace international in scope.

Each of the major variants of the UNIX operating system has captured a significant segment of the marketplace.

BSD remained most popular among scientific and engineering users, capturing the high-end technical workstation segment of the market. XENIX systems dominate the so-called 'low-end' segment of the marketplace and desk-top systems. UNIX System V represents the business and commercial market that consists of multi-user medium-sized computers and large mainframes. These market segments are converging, forcing the UNIX System variants to converge as well.

Technical workstations have found their way into business and financial departments; the power of microprocessors has increased to such a degree that personal computers now rival the technical workstations; and multi-user systems have become servers in networks for personal computers and workstations.

Technology has forged a single market out of traditionally distinct market segments, and the market demands a single, unified UNIX System to serve all its needs.

### SVR4.0 General Operating System Services

UNIX System V Release 4.0 developments for General Operating System Services (see figure 2) concentrate on the following:

- file-system enhancements to support SunOS, BSD and new types of file-system:
  - — memory-mapped files
  - — fast file-system
  - — symbolic links
  - — virtual file-system
- signal-handling mechanisms that POSIX adopted from BSD
- remote procedure call capabilities required to support NFS
- XENIX system-call compatibility required for application portability
- STREAMS I/O service from UNIX System V Release 3.0 to facilitate the implementation of protocol handling and networking services
- additional shell command interfaces in the form of the Korn Shell end the C Shell.
- continued work to further internationalisation:
  - — collating sequences
  - — conventions for date and time
  - — numeric representations
  - — clean-up of remaining libraries and utilities.

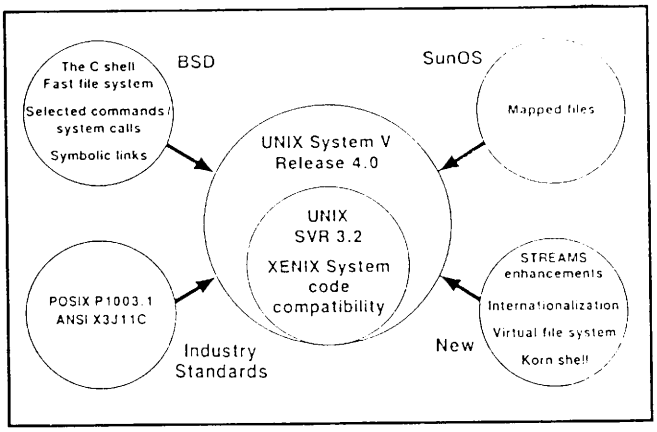



**Figure 2: SVR4.0 Basic Operating System Services**

### SVR4.0 Application Development Environment

The UNIX System is a system created by programmers for programming, with the expectation that by creating the best possible environment in which to develop software, the best possible software will be developed.

The UNIX System is the premiere software development environment, providing a wealth of tools and utilities to make writing and testing programs easier.

UNIX System V Release 4.0 continues the UNIX System tradition of providing the foremost computing environment for the creation and use of software tools.

UNIX System V Release 4.0 expands upon the already superb UNIX System programming environment, principally in the areas of standards conformance but also in file-system enhancements (see figure 3).

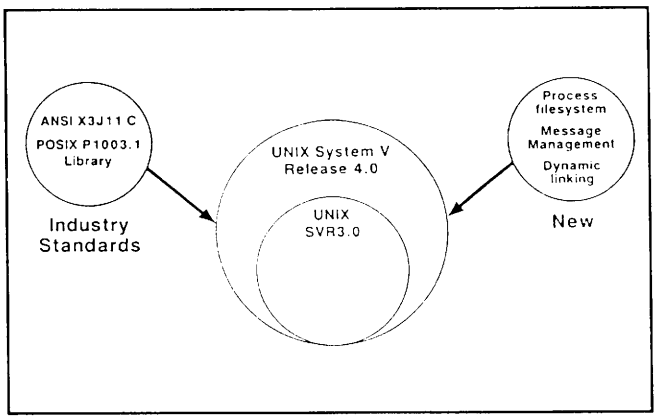



**Figure 3: SVR4.0 Application Development Environment**

## SVR4.0 Open Systems Networking

With the introduction of STREAMS I/O in Release 3.0, UNIX System V defined the fundamental capabilities needed to support Open Systems Networking. Release 4.0 builds on the foundation laid by Release 3.0 extending the networking capabilities of the UNIX System V to include the data communication services defined by the Defence Advanced Research Projects Agency (DARPA) and supported by the 'sockets' interface of BSD.

For the first time, Release 4.0 brings together Remote File Sharing (RFS) from System V and the Network File System (NFS) from SunOS, including the capabilities for Remote Procedure Call (RPC) and eXternal Data Representation (XDR) used by NFS (see figure 4).

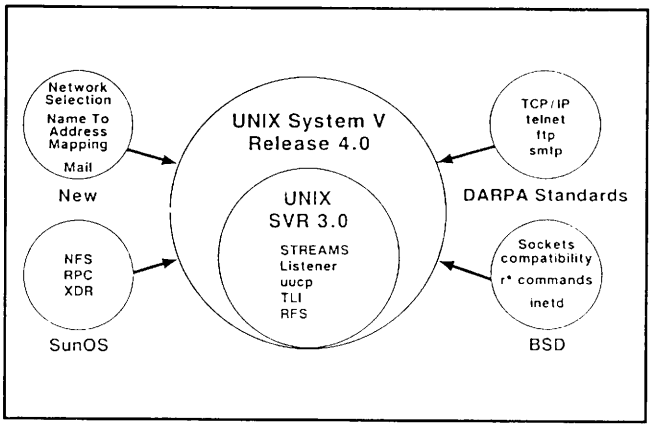



**Figure 4: SVR4.0 Networking**

## SVR4.0 User Interface

User Interface Extensions to the UNIX System V Base provide services that simplify the development of applications that interact with users. Human-computer interface components support device-independent display management in both character-oriented and bit-mapped graphic display devices. Style guidelines and development toolkits help programmers to maintain consistency in the 'look' and 'feel' of the user-interface to promote 'user portability' between different applications that follow the same guidelines. This further reduces software costs by insulating applications from display device dependencies and reduces user training costs and loss of productivity as users move between different applications.

Software developers benefit from a rich and robust set of tools for developing applications that share the same 'look' and 'feel'. End-users benefit because they can transfer existing skills and knowledge of how to use one application to the use of another and thereby avoid the need to learn a new and different user-interface.

UNIX System V Release 4.0 includes extensive feature/functionality in the area of user-interface (see figure 5):

- Character-oriented user interface
  - — curses/terminfo
  - — Extended Terminal Interface (ETI)
  - — Form and Menu Language Interface (FMLI)
  - — Framed Access Command Environment (FACE)
- Graphic-oriented user interface (not bundled in SVR4.0)
  - — X Window System
  - — OPEN LOOK

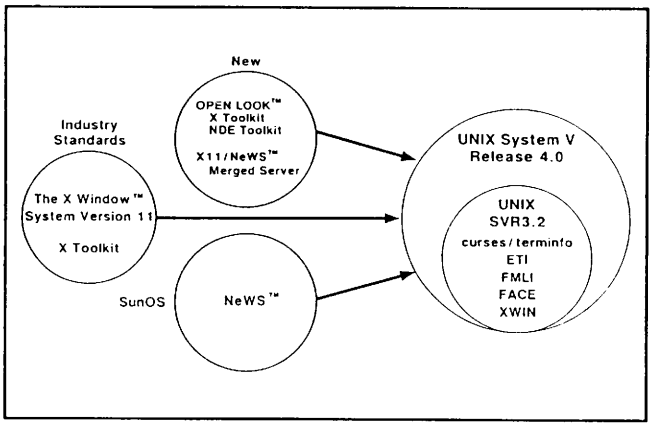



**Figure 5: SVR4.0 User Interface**

## SVR4.0 Enhanced Administration

UNIX System V Release 4.0 further enhances UNIX System administration in the areas of:

- software distribution and installation
- configuration management
- backup and restore facilities
- message management and monitoring
- administrative user-interface based on FACE

Special attention has been given to merging the administrative capabilities of the NFS and RFS (see figure 6).

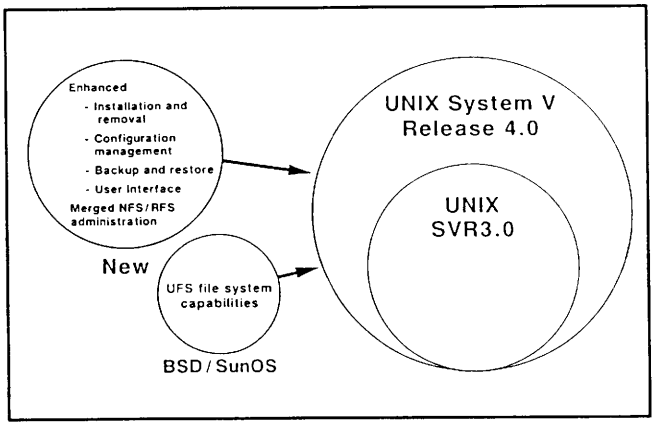



**Figure 6: SVR4.0 Enchanced Administration**

## SVR4.0 Application Binary Interfaces

UNIX System V Release 4.0 takes the next step to establish a single unified UNIX software market by defining Applications Binary Interface (ABI) for computers running UNIX System V.

ABI brings to UNIX software developers the advantages enjoyed by developers of PC software, allowing applications to run interchangeably across the platforms that support the ABI as easily as software can run in the PC world today. An ABI for the UNIX System opens up the opportunity for 'shrink wrap' software with all the benefits that with bring with it.

Software dealers and computer stores do not stock source-code, they stock packaged software. The mass market demands programs capable of running 'out of the box', there must be a 'binary standard' for UNIX software.

The System V Application Binary Interface, or ABI, defines a system interface for compiled application programs on computer systems based on a specific microprocessor architecture.

At this time, there are plans to have an ABI specification for the following architectures:

- Intel 80X86
- Motorola 680XX and 88000
- SPARC

Other architectures will be added as we move forward.

The remainder of this column is devoted to a more technical look at ABI.

## Foundations and Structure of the ABI

The System V ABI specifies two types of interfaces:

- Architecture Specific (Low-level)
- Generic (High-level)

Low-level interfaces were drawn from the specific processor reference materials and include definitions such as the processor's instruction set and register usage conventions. It also includes other definitions that are strongly affected by the characteristics of the processor's architecture. This includes information on the formats of executable and linkable files, and other implementation specific data important for application programs.

The higher-level interfaces were drawn from existing standards for operating systems, user interfaces, programming languages, and networking, including:

- The System V Interface Definition, or SVID.
- The IEEE POSIX P1003 standard operating system specification.
- The MIT X11 X Window System graphical user interface specification.
- The ANSI X3J11 C Language specification.

## How to use the ABI Interface

The ABI Interface Definition is a reference document that should be used in conjunction with the publically-available standards documents it references. The ABI enumerates the system components it includes, but descriptions of those components may be included entirely in other reference documents.

Application programmers who wish to produce binary packages that will install and run on any ABI conforming computer (for a specific architecture) should follow this procedure:

1. Write programs that use only the system commands, library functions, system calls, and other facilities explicitly included in this specification.
2. Compile the programs so that the resulting executable programs use the specified interface to all library functions and system calls, and so that the format of the executable programs conforms to the details in the specification.

3. Package the application in the format and on the media described in the specification, and install or create files only in the specified locations provided for this purpose.

The manufacturers of ABI-based computer systems who wish to provide the system interface described in the specification must satisfy a complementary set of requirements:

1. Their system must implement the specific processor architecture.
2. The system must be capable of executing compiled programs having the format described in the specification.
3. The system must provide libraries that are linked to application programs using the methods described in the specification, and contain all the library functions included in the ABI.
4. The system's map of virtual memory must conform to the requirement of the specification.
5. The system's low-level behaviour with respect to system call linkage, system traps, signals, and other such activities must conform to the formats documented in the specification.
6. The system must provide all commands, files, and utilities specified as part of the ABI, in the format defined in the ABI specification and other referenced documents. It must also provide all other components of an application's run-time environment that are included in the ABI specification.
7. The system must install packages using the formats and procedures described in the ABI, and must be capable of accepting installable software packages, either through physical media or through a network interface.

## Base and Optional Components of the ABI

The ABI provides two levels of interface specification: Base and Optional. Base components of the ABI are required to be present in all ABI-compliant systems. Optional components may not be present on an ABI-based system, but, when they are present, they must conform to the specification given in the ABI.

This distinction is necessary because some ABI capabilities depend on the presence of hardware or other facilities that may not be present, such as integral graphics displays or network connections and hardware. The absence of such facilities does not prevent an ABI-based system from complying with the ABI specification, through it may prevent applications that need these facilities from running on those systems.

## Evolution of the ABI Specification

The ABI will evolve over time, and will be reissued at intervals of approximately three years. Each new edition of the specification is likely to contain extensions and additions that will increase the potential capabilities of applications that are written to conform to the ABI.

As with the System V Interface Definition, the ABI will also include the provision for Level-1 and Level-2 support for its constituent parts. Level-1 support implies that a portion of the specification will continue to be supported indefinitely, while Level-2 support means that a portion of the specification may be withdrawn or altered after the next edition of the ABI is made available.

# Book Review

*Lindsay F. Marshall*
*Lindsay.Marshall@newcastle.ac.uk*

*The Computing Laboratory*
*University of Newcastle-Upon-Tyne*

**Numerical Recipes in C** William Press, Brian P. Flannery, Saul Teuklovsky and William T. Vetterling Cambridge University Press 1988 ISBN 0-521-35465-X (UK) Price $27.50, Hard Back, 818 pp,
**Numerical Recipes in C – diskette** Cambridge University Press 1988 ISBN 0-521-35466-8 (UK) Price $15.00 IBM PC format
**Numerical Recipes Example Book (C)** William T. Vetterling, Saul Teuklovsky, William Press and Brian P. Flannery Cambridge University Press 1988 ISBN 0-521-35746-2 (UK) Price $20.00, Paper Back, 239 pp,
**Numerical Recipes Examples (C) – diskette** Cambridge University Press 1988 ISBN 0-521-35467-2 (UK) Price $15.00 IBM PC format

This package from Cambridge University Press comes fairly close to meeting the requirements for the perfect recipe book – it is well written, entertaining to read and the dishes it describes are tasty. The main text covers a large amount of ground in an accessible style allowing readers to derive as much or as little information about particular topics as they require. Every section provides a list of references and further reading that can be followed by those who wish to learn more about a particular method or algorithm. Topics covered include simple julian date manipulation, random number generation (including information on DES), sorting, and a vast range of numerical and statistical techniques. This reviewer is, to say the least, not one of nature's applied mathematicians but found the text easy to follow and has been assured by his numerical analyst colleagues that the theoretical content is of a high standard.

However, there are some drawbacks to the book. The most important of these is that it shows its origins all too clearly – this volume is a reworking of the original text which was written for Fortran users. The majority of the functions have been translated literally from one language to the other and this means lots of variables with meaningless names like *izzy*. Only the DES functions really make use of C's more sophisticated data handling facilities though a reasonable package for complex number handling is provided. The authors also have some rather odd ideas about program structures – they dislike *continue* for some reason and suggest that one ought to use *if...else* instead of *switch* statements! They justify this later idea by invoking some supposed uncertainty about the types allowed in the control expression and by the even more dubious assertion that *if...else* is more 'recognisable' – whatever that may mean. Nevertheless these are very minor quibbles and the fact that the authors do not use the ghastly 'kernel' style for formatting their code means that they cannot be too far from the true path (at least in this reviewer's eyes). Ideally one should purchase the floppy disk (available for the IBM PC or the Mac) and so could avoid having to look at the code at all! Having the disk also eliminates the unreliable process of copying functions from the text, a task not eased by the FORTRAN-style names. The disk also includes header files with function prototypes that can be tailored for both K&R and ANSI versions of C which is very useful in this period of transition between standards. The only slightly odd thing about the IBM PC version of the disk is that the sources are held in a hidden directory which is most confusing as the disk appears at first glance to be missing all the software that one has paid for!

The companion volume contains listings and brief descriptions of some simple programs that make use of the functions provided as well as the data needed to drive them. Particularly nice to have is the listing of the NPS DES validation data which can be used to check out the encryption procedure given in the main text. However, neither this book nor the floppy that goes with it are really necessary for the programmer who just wants to get hold of a particular numerical function. The examples would be of much more use to someone teaching a course on numerical work in C – if

such a person exists – or, more likely, for someone teaching C to numerical analysts. Here again the floppy is pretty well essential as some of the datasets need to drive the sample programs are quite large (particularly the NPS data).

All in all this is an excellent package for both the programmer who only occasionally needs some numerical functions and the specialist who needs to work in C. At $27.50 the main text is quite expensive but much better value than many other Computer Science texts in this price range.

However when you add the cost of the floppy disk into this it does begin to look rather too dear, particularly for students. This is probably unavoidable but will tend to put off quite a few potential purchasers. If the floppies were sold at media price to those who had bought the text, or were simply included with the books themselves then sales would be much higher.

---

**C Programming in a UNIX Environment**
Judy Kay and Bob Kummerfield
Addison-Wesley International Computer Science Series ISBN 0-201-12912-4
(UK) Price £15.95, Paper Back, 340 pp.

The Australian UNIX tradition goes back a long way so it is not surprising that several publishers have recently set off down the Yellow Brick Road and signed up some of the Wizards of Oz. The authors of this book hail from the University of Sydney and if it is a representative sample of what we can expect from down-under then we have some treats in store. The text is easy to read and well paced for experienced programmers wishing to learn or improve their C. The examples are sensibly chosen, clearly presented using the format favoured by this reviewer and lead up to the implementation of a small but real application. This allows most aspects of developing a C program under the UNIX operating system to be covered. The authors place particular emphasis on the development of good programming style as a basis for creating reliable programs and encourage the use of C idioms to overcome the 'Pascal in C' syndrome that is often seen amongst newcomers to the language. They also stress the need to check return codes and to make use of lint to help find problems.

One of the weakest points in most books about C and UNIX is their treatment of system provided functions and libraries. The temptation to parrot the manual page is hard to avoid and most authors simply make no attempt to provide more readable information about this important area. This book is an exception, the description of the standard functions fits naturally into the style of the rest of the book and various common pitfalls are pointed out (e.g., calling *time* with a value rather than a pointer!!). The chapter on libraries takes up nearly a quarter of the book and serves also to reinforce the lessons of the earlier chapters about programming style and technique.

What of the shortcomings of this book? There are a few. Firstly the book describes K&R C and only devotes four pages in an appendix to the ANSI Standardisation effort. This is written very tentatively and would not help anyone who had to get to grips with what is actually happening in the practice. The second major problem is that the book is written for someone using V7 UNIX. Whilst we all know that V7 was a much better system than most of its successors, the world has moved on and there have been a few developments that would be worth mentioning. Perhaps the authors should cash in on this and, like so many others, have a BSD version and a System V version? This would be much more useful for someone trying to take full advantage of their environment. Even if they do not do this it would certainly be worth their while to produce a version of the book for ANSI C (when it eventually settles down). It would also be worth tidying up the chapter on arrays and pointers as there are some areas of difficult that they do not touch upon. However, when all is said and done this is a good book and is reasonably priced – only three times what one would expect to pay for an equivalent sized, non-technical paperback.....

# UKUUG Winter Abstracts

Here are the abstracts of the papers delivered at the UKUUG Winter meeting held at the University of Kent, England on the 19th-20th December 1988.

Copies of the proceedings are available from Owles Hall at £10 each including post and packing.

Thanks are due to Peter Houlder <*uknet@ukc.ac.uk*> who organised the conference and the typesetting.

## A Low-Cost Bitmapped Terminal on the Atari ST

*Peter Collinson*

The Computing Laboratory
The University
Canterbury, Kent CT2 7NF
England
*pc@ukc.ac.uk*

The development of workstations with large bitmapped screens and a pointing device has meant a revolution in computing practice. This paper describes a program written for the Atari ST micro which is an attempt to bring some of this revolution to the a wider user base by the use of low cost technology. The program, called *Term*, allows the user to control and maintain a windowing terminal. The environment may be used on BSD UNIX systems to run several shells supported by a host multiplexer. The terminal uses many features of the Atari ST, including the ability to display differently sized fonts. The paper discusses the various design issues and comments on how well the various system components perform on the low cost hardware.

## Mail Interface Enhancements to MH

*Donal Daly*

Computing Science Dept
Trinity College
Dublin 2
Eire
*daly@cs.tcd.ie*

An introduction and some history and comments on the MH system, followed by a brief breakdown of its various features and some of the author's modifications and enhancements.

## Developing and Adapting UNIX Tools for Workstations

*David Barnes*
*Mark Russell*
*Mark Wheadon*

The Computing Laboratory
The University
Canterbury, Kent CT2 7NF
England
*djb@ukc.ac.uk*

This paper describes our experiences in developing tools for high performance graphical workstations. In particular we concentrate on the ways in which some tools and concepts familiar to users of glass-teletype UNIX systems have been adapted and exploited within a new environment. The tools described are a file differencer, an execution profiler and a file browser.

## Direct Manipulation Tools for UNIX Workstations

*J D Bovey, M T Russell and O Folkestad†*

The University of Kent at Canterbury
*jdb@ukc.ac.uk*

Direct manipulation is one approach to the creation of software which can make use of the high resolution graphics and pointing device available on a workstation like a Sun 3. A direct manipulation tool is typically used to manipulate a complex system like, for example, a file system, and works by presenting a graphical image of the system which the user can manipulate in order to manipulate the system itself.

The paper starts off by discussing direct manipulation in general terms and then goes on to describe three examples of direct manipulation tools which were written at the University of Kent. The tools described are a file system editor, a graphical debugger and a front end to SCCS.

The remaining sections of the paper discuss the implementation of direct manipulation tools, outline some of the user interface techniques that are applicable, and suggest a few systems which may be amenable to the direct manipulation approach.

## Installing and Operating NFS on a 4.2 BSD VAX

*Jim Reid*

Dept of Computer Science
University of Strathclyde
Richmond Street
Glasgow
G1 1XH
Scotland
*jim@cs.strath.ac.uk*

† Now at *Ingenioerene Bond & Co, Treschowsgate 2B, 0477 Oslo 4, Norway*

The author's experience in the installation and operating of Sun's Network File System (NFS) on a VAX 11/750 running *4.2BSD UNIX*. Problems encountered during installation and operational difficulties while running the system are discussed, along with the adjustments needed to solve these problems.

## A MINIX Port to the Atari ST

*Aarron Gull and Sunil Das*

City University
Dept of Computer Science
City University
Northampton Square
London EC1 OHB
England
*aarron@cs.city.ac.uk*

The presentation covers the port of the MINIX operating system to the Atari ST. The discussion will center on the novel process and memory management techniques which were employed. The impact of these upon the STIX system call interface, in particular *fork*(), *exec*() and *exit*(), concludes the discussion.

## NeWS and X, Beauty and the Beast?

*W. T. Roberts*
*A. Davison*
*K. Drake*
*C. E. Hyde*
*M. Slater*
*P. Papageorgiou*

Department of Computer Science
Queen Mary College
190 Mile End Road
London E1 4NS
United Kingdom
*liam@cs.qmc.ac.uk*

NeWS and X11 are the two best known distributed window systems. This paper presents experience of both NeWS and X11 at Queen Mary College, highlighting strong and weak points of both systems and looking at future developments, including the much-heralded X/NeWS combined server.

## UKnet Overview

*Peter Houlder*

The Computing Laboratory
The University
Canterbury, Kent CT2 7NF
England
*uknet@ukc.ac.uk*

This paper is in two parts. The first part deals with the internal aspects of UKnet as a network; what it does, how it is administered, services offered costs, etc. The second part tries to fit UKnet into the scheme of international Electronic Mail networks. The aim is to give an overview of the main world networks along with the mail systems and transport protocols used. The relationship of these networks to the OSI 7-layer model is also discussed.

## UK Coloured Books – Current Facilities and Transition Plans

*Prof P.F.Linington*

The Computing Laboratory
The University
Canterbury, Kent CT2 7NF
England
*pfl@ukc.ac.uk*

Most EUnet users think of communication with the UK academic community in terms of mail and news. These, however, form only a part of the facilities in regular use. Other protocols support file transfer, job transfer and manipulation and screen oriented interactive terminal access, and these facilities will be described. The intention is to move towards the use of international standards, and a strategy for doing so has been drawn up. This involves both technical and organisational choices which are aimed at providing continuity of service to a steadily expanding community. The future directions for this transition will be outlined.

## EARN and BITNET

*Paul Bryant*

Rutherford Appleton Laboratory
Didcot
Oxfordshire, OX11 0QX
England
*peb@ib.rl.ac.uk*

The transition of EARN to use ISO protocols. EARN is currently putting in an X.25 international network as the first part of its transition to use ISO protocols. This paper describes the initial network and shows how the current services will be preserved. The outline plans for the further development of the network are described.

## News – Present and Future Developments

*Chris Downey*

The Computing Laboratory
The University
Canterbury, Kent CT2 7NF
England
*cmd@ukc.ac.uk*

This paper covers the history of the USENET news system, and how it works; followed by a description of the current state of the news software, and some observations about future developments, both in the medium and the long term.

## EUnet and OSI Transition Plans

*Daniel Karrenberg*

Centrum voor Wiskunde en Informatica
P.O. Box 4079
NL-1009 AP Amsterdam
Netherlands
*dfk@cwi.nl*

EUnet, a pan-European cooperative R&D network, is described in terms of applications, protocols and topology. A strategy for the introduction of OSI applications and protocols into this network is then presented. The actual talk will provide additional up to date information about other developments in EUnet.

AUSTRIA - UUGA
Friedrich Kofler
Schottenring 33/Hof
A-1010 Vienna
AUSTRIA
+43 222 34 61 84
uuga@tuvie.at

FINALND - FUUG
Johan Helsingius
OY Penetron Ab
Box 90
02201 ESPOO
FINLAND
+358-0-427 632
julf@fuug.fi

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

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

DENMARK - DKUUG
Mogens Buhelt
Kabbelejevej 27B
DK-2700
Bronshoj
DENMARK
+45 1 60 66 80
mogens@dkuug.dk

HUNGARY - HUUG
Dr Elôd Knuth
Computer and Automation Institute
Hungarian Academy of Sciences
H-1502 Budapest 112, P.O. Box 63
HUNGARY
+36 1 665 435

ICELAND - ICEUUG
Marius Olafsson
University Computer Center
Hjardarhaga 4
Reykjavik
ICELAND
+354-694747
iceuug@rhi.hi.is

IRELAND - IUUG
John Carolan
Glockenspiel Ltd
19 Belvedere Place
Dublin 1
EIRE
+353 1 364515
john@puschi.uucp

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

NETHERLANDS - NLUUG
Patricia Otter
Xirion bv
World Trade Centre
Strawinskylaan 1135
1077 XX Amsterdam
THE NETHERLANDS
+31 20 6649411
patricia@xirion.uucp

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

EUUG-S
N
I
X
SWEDEN - EUUG-S
Bjorn Eriksen
SUNET/KTH
S-100 44 Stockholm
SWEDEN
+46 8 790 65 13
ber@kth.se

UNITED KINGDOM - UKUUG
Bill Barrett
Owles Hall
Buntingford
Hertfordshire SG9 9PL
UNITED KINGDOM
+44 763 73039

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