Volume 5, Number 5 March 1993 $2.50

MUUG Lines

Manitoba UNIX® User Group

Newsletter of the Manitoba UNIX® User Group

Books Worth Reading

By Royce Howland

Reprinted from edm.announce (the monthly newsletter of /usr/group/edmonton, Edmonton's UNIX user group), November & December, 1992

I review two books this issue, *Computer Ethics, Cautionary Tales and Ethical Dilemmas in Computing*, by Tom Forester and Perry Morrison, and *The Whole Internet User's Guide & Catalog*, by Ed Krol.

Computer Ethics authors Forester and Morrison begin the preface of their book with this statement: "The aim of this book is two-fold: (1) to describe some of the problems created for society by computers and (2) to show how these problems present ethical dilemmas for computer professionals and computer users." In this, they succeed. If you have a professional concern with ethical issues in computing, or are just curious to read a broad treatment of the darker side of the computer revolution, you will find raised in this book many thought-provoking issues.

While the authors pose many ethical questions, they make no attempt to provide the answers. Instead, they wrap up each section with a "what if" scenario designed to make the reader consider for him- or herself the application of ethics to situations that could conceivably occur. These scenarios are also suggested as group discussion starters, as the book is designed to serve as reading material in a university or college computing ethics course.

On that latter subject, the authors state this in their introductory chapter: "Computing educators should not merely be in the business of training technicians. They should be producing articulate information technologists – technologists endowed with communication skills, 'people skills' and possessing an appreciation of the social and ethical implications of information technology. They must prepare people for the messy real world, not the tidy, imaginary one inside a VDT screen." Whether the reader agrees with this education sentiment or not, it will be difficult, after reading this book, not to arrive at the authors' conclusion that there are many consequences of the computer revolution that we are presently poorly equipped to handle.

The ethical implications of computer-related failures that caused fatal crashes of the USAF's UH-60 Blackhawk helicopter and the European A320 Airbus, for instance, may be hard for an individual to grasp, but the legal implications are probably within most of our imaginations. And these situations involve computer control systems that are less complex than other systems we rely on daily, such as digital telephone, air traffic control, medical, and nuclear power station systems. As computers continue to penetrate more areas of society, the consequences of failures become more severe. On a less dire, but highly confused, legal subject, just try to resolve the issue of software ownership and theft, combined with copyrights, patents, and so on!

There are many responses to these situations, ranging from ignoring them (hoping they will go away), to government regulation of computer professionals (of which there has been some rumblings in the US, Canada, and elsewhere), to Luddism (an outright rejection of technology).

Continued on page 5

This Month's Meeting

Meeting Location:
Our next meeting is scheduled for Tuesday, March 9, at 7:30 PM. Once again, the meeting will be held in the auditorium of the St-Boniface Hospital Research Centre, just south of the hospital itself, at 351 Taché. You don't have to sign in at the security desk – just say you're attending the meeting of the Manitoba UNIX User Group. The auditorium is on the main floor, and is easily found from the entrance.

Meeting Agenda: See inside for details.

Inside This Issue

Coping With Change

By Gilbert Detillieux

Last month, I talked about the interesting times in which we live, and the problems of dealing with rapidly changing technology. Now more than ever, it is important for us to be able to cope with change.

This month, our diverse collection of articles seem to have one unifying theme – that of coping with change. Because this means different things to different people, the articles look at it from very different perspectives.

First off, in the article entitled *Books Worth Reading*, we look at two very different books – one dealing with computer ethics, and another dealing with the Internet. Part of coping with technological change is working out the ethical issues that arise. It is important for us, as users of technology, to consider these issues and be prepared to deal with them.

As our society becomes more information oriented, we also have to find effective ways of dealing with information, and this usually means better communication. The Internet is rapidly becoming one of the most important tools to improve communication, not only for research, but for anyone who can make use of it. As well as being a tool for inter-personal communication, the Internet allows access to a wealth of information on just about any topic – you just have to know where to find it.

Coping with change also means designing systems that will survive the change. *Writing Software for Portability* addresses some of the issues programmers face in trying to design such systems.

Finally, even our Fortune File column deals with the subject of coping with change in its own tongue-in-cheek way – bemoaning the hardships endured by those who are forced by their vendor into migrating to a drastically altered operating system. While this may not be a very productive attitude toward coping with change, sometimes it helps to laugh about it.

One of the big benefits of the rapid change in the industry is the amount of computing power users can now afford. With a low cost 486-based PC, and a system like Linux (which will be featured at this month's meeting), the investment is smaller than ever. ✒

The 1992-1993 Executive

President:	Susan Zuk	(W) 788-7312
Vice-President:	Bary Finch	(W) 934-2723
Treasurer:	Rick Horocholyn	(W) 474-4533
Secretary:	Roland Schneider	1-482-5173
Membership Sec.:	Richard Kwiatkowski	589-4857
Mailing List:	Roland Schneider	1-482-5173
Meeting Coordinator:	Paul Hope	(W) 237-2361
Newsletter editor:	Gilbert Detillieux	489-7016
Publicity Director	Gilles Detillieux	489-7016
Information:	Susan Zuk	(W) 788-7312
		(FAX) 788-7450
(or)	Gilbert Detillieux	(H) 489-7016
		(FAX) 269-9178

Copyright Policy and Disclaimer

The Manitoba UNIX User Group, the editor, and contributors of this newsletter do not assume any liability for any damages that may occur as a result of information published in this newsletter.

Our Address

Manitoba UNIX User Group
P.O. Box 130
Saint-Boniface, Manitoba
R2H 3B4

Internet E-mail:
editor@muug.mb.ca

Group Information

The Manitoba UNIX User Group meets at 7:30 PM the second Tuesday of every month, except July and August. Meeting locations vary. The newsletter is mailed to all paid-up members one week prior to the meeting. Membership dues are $25 annually and are due as indicated by the renewal date on your newsletter's mailing label. Membership dues are accepted by mail, or at any meeting.

PRESIDENT’S CORNER

The Calm Before the Storm?

By Susan Zuk, President

This is really strange, the UNIX group has not had a very hectic February. We have ongoing projects but there has not been any panic situations. What a nice change.

At the present time, we are preparing for the ELSY Group’s Computer Expo ’93. This event is sponsored by the Muddy Waters Computer Society and will be a larger event than the computer show held last May. It is two days in length, May 8th and 9th, and should be just as successful as last year’s. We will be staffing a booth on these two days and will be asking for volunteers to give us a hand. Please mark these dates in your calendar and let us know if you are available to work at the MUUG booth and help promote MUUG to the Winnipeg community.

In continuing with the theme of promoting the group, we have placed an order for MUUG mugs. These mugs have our logo on one side and the caption “Wake up and smell the coffee, UNIX is here to stay” on the opposite side. We are selling these cups to members at our cost and will have some available at our next meeting. The cost for these mugs will be between 2 and 4 dollars. Did the caption have anything to do with BYTE Magazine’s article about Windows NT? Of course not? Why would we be bothered about a article titled “Is UNIX Dead?”

In last month’s newsletter, there was a request for nominations for UniForum’s Canada’s Open Systems Superuser Award. The final date for nominations was March 10. This date was incorrect and should have been May 10. Please read the article later in this issue and let me know if you have a person (or persons) that could be Canada’s Superuser. What a great way to promote our Manitoba talent.

In April, CIPS is holding their Annual Spring Seminar. This seminar is called *Networks – Now and in the Future.* The topics are varied and discussions will be on types of WANs and LANs being used today and what will be available in the future, networking software (networking, work group, work flow), the various standards appearing (both hardware and software) and how to smoothly implement a network strategy in your organization. If you are interested in this seminar please call me for details.

In the upcoming months, we have quite an interesting selection of speakers. This month, Greg Moeller will be speaking on LINUX. This is a UNIX variant for PC’s. In April, we will hear about Windows NT, and the May topic will be Sybase. We would like to bring a special speaker in to Winnipeg for the September meeting. If you have any suggestions please e-mail them or call Paul Hope.

I would just like to take this opportunity to thank Michael Doob for his presentation last month on TeX. Everyone is so busy these days and we really appreciate when members can give us some of their precious time. Michael discussed and displayed one of the tools that can be used to produce documents in the UNIX environment. Thank you Michael for a very interesting talk.

I look forward to seeing you at the nest meeting, on March 9th. ✒

FEEDBACK

Ask Monsieur Ex

A column in which our resident UNIX expert answers questions submitted by members, or discussed at round table sessions.

By Gilbert Detillieux

Dear Monsieur Ex,

After watching the TeX demo at the last MUUG meeting, I am curious how I can convert TeX files, that I have gathered over the Internet, into PostScript format so they can be printed on the Mac printers at work.

Are there any such TeX-to-PostScript apps lurking on MONA somewhere? Is there any share- or free-ware that I can use at work for this purpose?

Thanks.
Kirk J. Russell
Programmer Analyst
Iris Systems Inc.

Dear Kirk,

En fait, such software is being set up by the same TeX guru who spoke at last month’s meeting. I can’t say when it will all be set up, but it is in the works. The software is indeed free, and Michael Doob pointed out at the meeting, and is available from various sources. I believe you can get it by anonymous FTP, but I don’t know from which sites. Of course, the source will soon be available on MONA.

If you have a TeX (or LaTeX) file to print, you'll first need the “tex” command itself (or “latex”) to generate a DVI file. You’ll then need a DVI to PS converter – “dvips” is one of the better ones, and you’ll need the required font files, either as PK format bitmaps, or as MetaFont source.

I can’t get into all the details in this short space; look for an announcement on-line at some future date. ✒

***Monsieur Ex**, a mysterious Frenchman who claims to be an old editor and an expert in UNIX, will return again next time he’s asked questions, so don’t forget to write in, kids!* ***Gilbert Detillieux**, a French-Canadian of non-mysterious origins, is mysteriously still the MUUG newsletter editor.*

Writing Software for Portability, Part 1

By Gilbert Detillieux, Info West Inc.

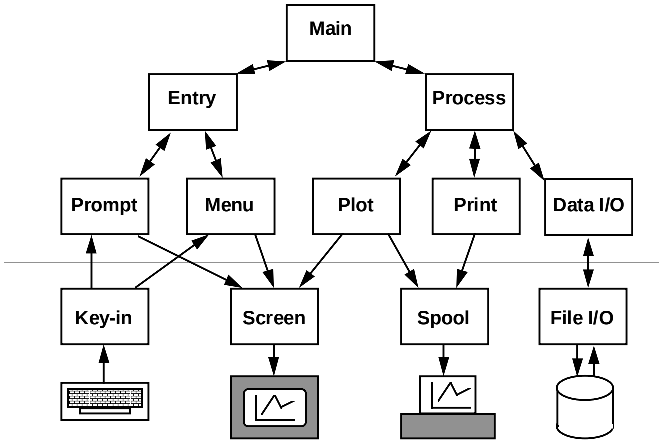
As the computer industry continues to evolve at an ever increasing pace, old systems and their software are becoming obsolete more quickly. The move to open systems technology and decreasing hardware costs make it feasible to throw out old hardware and entirely replace it every few years. But, what about our investment in software? The cost to produce and maintain it is increasing, and the complexity of the software is also increasing. To avoid having our software become obsolete too quickly, we have to design it so it will survive the transition to new systems. We must make it portable.

What does software portability mean? First, software must be machine independent, so it can easily move to different hardware architectures and systems of different scales, from micros to mainframes. It also should be operating system independent, as much as possible, since this also will change with time and with hardware changes. Software also should be designed to be independent of the environment in which it runs – user interfaces and interfaces to other programs it interacts with can change. It should also be independent of language and country specific features, if it will be used internationally. Finally, it should have version independence – it should survive changes in hardware, operating system, and programming language version changes, as well as changes in its own code.

The selection of the right programming language is important. A high level language is essential to assure machine independence, but it's important that this language follows industry standards, such as those set by ANSI and the ISO. Also, pick a language that will fit into corporate standards and coding practise, and that will be available on all your target systems. Avoid non-portable features, such as a particular compiler's language extensions, or ill-defined or implementation dependent features in a language. The last thing you want is to rely on a side-effect of an operator or other non-standard feature that will break when compiled on another system. Also avoid standards or language features that are too new, since these aren't yet widely supported or may change in future standards. A good choice for UNIX systems is ANSI C, but there might be other choices that are reasonable for your situation.

Regardless of language, structure and modularity are important. Use a top-down design consisting of small independent code modules. If your language supports them, use header files for public declarations of module interfaces. Use these headers in modules that will need these interfaces, as well as in the module that defines them – this ensures consistency in the declarations. Header files should not result in storage allocation or code generation – i.e. they should only declare variable and functions, not define them. If the system has standard headers for declaring library routines and system calls, use them -don't make assumptions about these by declaring them yourself.

With a well structured program, only a few low-level modules will be machine dependent or system dependent – the higher level code should all be portable. These low-level modules, on the other hand, should be general enough to be reusable in any application. There should be a clear line between the high-level, application specific but portable code, and the low-level, application independent but machine specific code. System calls, and other non-portable constructs, should only occur in the low-level modules. You might have to create stubs to handle missing features on a particular system – these should either fake the missing feature or return an error to the higher level modules to say this feature isn't supported.



You also might need optional modules, that will be linked in only if certain system-specific features will be required. This might be the case for interfaces to other programs, such as mailers or spoolers, for example. You can design plug-in replacements that will be selected based on the system configuration (such as a mailer interface to either SMTP or UUCP, or a spooler interface to either LP or LPR).

If your language supports conditional compilation constructs, you can use these to further isolate machine dependencies within modules. That way, one source module can support several systems, by simply selecting the right parts to compile. (If your language doesn't support conditional compilation, but you're developing on UNIX, consider using the C preprocessor or M4 on your source files to get that capability.) As much as possible, use predefined constants and macros as switches to control compilation of system specific features. (The C preprocessor usually defines a few symbols for the target operating system and architecture, such as “unix” and “sparc”.) If required, you can define your own constants and macros, either as compilation switches, or to define system specific quantities. Make all such definitions easy to find and change – don't bury them all over the place, but rather put them in a configuration header file, so they are localized and editing for a new configuration is kept to a minimum.

HANDS-ON

(This not only saves time, but reduces the chances of introducing errors in the code.)

Finally, use the “make” command and a Makefile, if your systems support it. This makes it easy to control the compilation, and eases configuration and installation. Configuration options, constants, and macros can all be defined in the Makefile – this isolates them from the code, and allows flexibility in customizing the code without making any changes to the modules themselves. If you want to get fancy, you could even have a program or script that builds a Makefile tailored to a particular configuration, based on the system and user selected option. (Most X11 Window software now includes an Imakefile, a system independent Makefile, which can be run through a program that will produce a custom Makefile for your site.)

So far, we’ve only looked at program structure and modularity. Even more important to assure portability is the way we use data structures and data types, in memory and in data files. We’ll look at that next time.

```
% cat Imakefile
# Imakefile for xcoloredit
       INCLUDES = -I$(TOP)
        DEPLIBS = XawClientDepLibs
LOCAL_LIBRARIES = XawClientLibs
           SRCS = xcoloredit.c color.c utilities.c
           OBJS = xcoloredit.o color.o utilities.o

ComplexProgramTarget(xcoloredit)
% xmkmf
mv Makefile Makefile.bak
imake -DUseInstalled -I/usr/lib/X11/config
% make
cc -O -I. -DXAW_BC -c xcoloredit.c
cc -O -I. -DXAW_BC -c color.c
cc -O -I. -DXAW_BC -c utilities.c
rm -f xcoloredit
cc -o xcoloredit xcoloredit.o color.o utilities.o -lXaw
   -lXmu -lXt -lXext -lX11
```

A sample Imakefile for an X11 client program. The file uses abstract symbol names for the various directories and libraries, and for the actions to be performed. A site specific Makefile can then be generated, containing specific path names, commands, and options appropriate for the site.

REVIEW

Books Worth Reading

Continued from page 1

The reasonable starting point to the solution, whatever it (or they) may be, however, is to have the people involved inform themselves and others of the issues, and begin discussing them. *Computer Ethics* is a step in that direction. It’s also a good read for people who like disaster stories. :-)

The book has 181 pages of text in 8 chapters and one appendix, and covers an introduction to ethical dilemmas, computer crime, software theft, hacking and viruses, computer unreliability, invasion of privacy, AI and expert systems, computerizing the workplace, and autonomous systems (systems that require no human intervention; a case study is the proposed US Strategic Defence Initiative system – “Star Wars”). References follow each section; there are some 270 pointers to further reading in other books and periodicals, for the ambitious. The book is published by the MIT Press in 1990. I got my copy at Edmonton Computer Books at a cost of $16. (That’s cheap!)

The Whole Internet User’s Guide & Catalog

One of the most sought after and talked about, but perhaps least fully understood and utilized, computing resources must be the Internet, and the services available through it. I have heard a number of likely and unlikely people talking about “Internet access,” with differing degrees of knowing What It’s All About. Naturally, I thought I had a pretty good handle on the important stuff. However, after reading a copy of a recent O’Reilly & Associates book, *The Whole Internet User’s Guide & Catalog*, by Ed Krol, I began to see the error of my ways. As usual, the more one knows, the more one knows how little one knows.

The Internet itself is a computer network spanning the globe. More properly, it is a network of networks that communicate using the Internet Protocol (IP) network standard. The communication technology involved in the Internet is enough to keep network gurus talking for a long time, but what most of us are more interested in are the resources and services made available on the Internet. In *The Whole Internet*, Krol covers basic network functions like remote login and file transfer, and then launches into the more commonly used “applications” such as electronic mail, network news, and user and file directory services (e.g. *whois* and *archie*, respectively). The real meat of the subject begins to appear, however, as Krol dives into decreasingly familiar resource searching tools: *gopher*, Wide Area Information Servers (WAIS) and the World Wide Web (WWW). These application topics are not dealt with at all in Krol’s *The Hitchhikers Guide to the Internet*, a document he wrote in the late 1980’s that still floats around the Internet. Indeed, because these tools are new and still under development (some, like WWW, might even be called experimental), Krol can’t address them definitively in the book, either.

However, this latter subject matter is, to me, the really exciting part. The amount of resources and services spanning the Internet is tremendous, but making effective use of it before tools like *gopher*, WAIS and WWW, Krol likens to searching for something in a library by lining up all its resources in a row and starting to search them one at a time, from end to end. In fact, I think such a search might even be easier than dealing with the Internet, since in the library you at least would have direct physical access to all the material you’re searching, and you’d know the nature and magnitude of it. By comparison, the Internet is probably comprised of much that you don’t know about at all, some of which may need to be accessed in ways that you’re unfamiliar with or unable to use without acquiring specific access or software.

The basic tools like mail, *whois* and *archie* were only a beginning, since they limit your access to only certain types of information, and have totally different interfaces. Mail is for talking to humans; *whois* is for finding mail addresses given user names, so you can talk to humans (but only from a specific subset of address directories that use the *whois* standard; there are others, for example the X.500 directory service); and *archie* is for searching file location databases (sometimes not very up to date), given a file name or specific key word, so you can then use *ftp* or some other means to get the files to your system, if you can.

By contrast, newer applications like *gopher*, WAIS and WWW try to provide a single interface to a number of differing services that in turn offer access to different kinds of resources, without regard to where or what those resources are. They may also offer additional features of their own. For instance, the main purpose of WAIS is to allow people to search through indexed information (which could be text, graphics, whatever) for a set of words. All sources that contain those words are given a normalized score to represent how well WAIS thinks they match, and the top few sources are shown to you, regardless of where they reside! You can save the documents to your system, and the search criteria for future use. You can also use “relevance feedback” to use the results of a search to further tailor or refine the search.

WWW is an even newer Internet service designed to use hypertext concepts to link potentially any type of information across the entire Internet to any other type of information. Like WAIS, which incorporates previous services by allowing indexes to be created from whois and archie directories, and gopher, which provides access to WAIS index services, WWW can also access WAIS databases and sources created from other services.

Unlike the other services, Web “documents” can contain many links to other “documents” across the Internet, and you can easily move across the links to see what’s there, place markers to return to certain points, save documents to your system, etc. The Web doesn’t currently have many true hypertext documents in place, created just for it. Most of the sources it can access were WAIS databases at the time the book was published. The reason for this is that the tools to create WWW hypertext documents are not widespread. At the time of publication, Krol states that only the NeXT possessed a hypertext editor for use with the Web, with one for an X Window package to follow soon. Despite the relative shortage of tools and custom resources available to these new Internet services, it nevertheless seems that this is the right direction for development to be taking.

In addition to what I’ve described so far, Krol also goes into some of the history of the Internet, and other topics like getting connected through a service provider, dealing with problems, and acceptable uses (remember, this is a network of networks, each of which may have its own mandate and requirements of its users). There is also the *Catalog* portion of the book, a listing of many diverse resources compiled by Krol using the tools and techniques he describes.

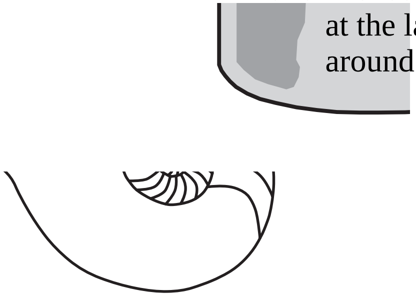
Rather than read my comments any further, if you are at all interested in the Internet, either because you have or wish to have access to it (perhaps in a more complete and effective manner), get a copy of *The Whole Internet User’s Guide & Catalog*. It contains around 354 pages of text, and is definitely targeted at an audience that wants to use the resources of the Internet, not talk about network bits and bytes. It is fairly light reading, and contains a few sprinkled references to Buckaroo Banzai, which clinched my vote. :-) The book should be available anywhere that carries the O’Reilly series. I got my copy at Edmonton Computer Books for about $35. ✒

Announcement

MUUG Mugs Are Here

The Manitoba UNIX User Group has ordered two hundred custom designed coffee mugs, which bear the group’s name and logo on one side. These mugs will be used as thank-you gifts for group members and other local speakers who give presentations at our meetings. They will also be used as promotional items at computer shows such as the upcoming MWCS Computer Show in April.

In addition to that, we will be offering them to MUUG members and friends at cost, as was mentioned at the last meeting. Cost will be around $3 per mug. Many people at the last meeting indicated an interest in buying these mugs. If you missed the last meeting, don’t worry – it’s not too late to get a mug for yourself. We will have a quantity of them on hand at the March meeting.



So there’s another good reason to come on out to the meeting!

THE FORTUNE FILE

The Day SunOS Died

Lyrics by N. R. “Norm” Lunde. Apologies to Don McLean.
Submitted by Andrew Chan

Originally posted to rec.humor.funny

Remember when those guys out West
With their longish hair and paisley vests
Were starting up, straight out of UCB?
They used those Motorola chips
Which at the time were really hip
And looked upon the world through VME.
Their first attempt ran like a pig
But it was the start of something big;
They called the next one the Sun-2
And though they only sold a few
It soon gave birth unto the new
Sun-3 which was their pride
And now they’re singing

“Bye, bye, SunOS 4.1.3!
ATT System V has replaced BSD.
You can cling to the standards of the industry
But only if you pay the right fee —
Only if you pay the right fee . . .”

The hardware wasn’t all they sold.
Their Berkeley port was solid gold
And interfaced with System V, no less!
They implemented all the stuff
That Berkeley thought would be enough
Then added RPC and NFS.
It was a lot of code to cram
Into just four megs of RAM.
The later revs were really cool
With added values like SunTools
But then they took us all for fools
By peddling Solaris . . .
And they were singing,

“Bye, bye, SunOS 4.1.3!
ATT System V has replaced BSD.
You can cling to the standards of the industry
But only if you pay the right fee —
Only if you pay the right fee . . .”

They took a RISC and kindled SPARC.
The difference was like light and dark.
The Sun-4s were the fastest and the best.
The user base was having fun
Installing SunOS 4.1
But what was coming no one could have guessed.
The installed base was sound
And software did abound.
While all the hackers laughed and played
Already plans were being made
To make the dubious “upgrade”
To Sun’s new Solaris . . .
And Sun was singing,

“Bye, bye, SunOS 4.1.3!
ATT System V has replaced BSD.
You can cling to the standards of the industry
But only if you pay the right fee —
Only if you pay the right fee . . .”

The cartridge tapes were first to go —
The CD-ROM’s a must, you know
And floppy drives will soon go out the door.
I tried to call and ask them why
But they took away my TTY
And left my modem lying on the floor.
While they were on a roll
They moved the damned Control.
The Ethernet’s now twisted pair
Which no one uses anywhere.
ISDN is still more rare —
The bandwidth’s even less!
But still they’re singing

“Bye, bye, SunOS 4.1.3!
ATT System V has replaced BSD.
You can cling to the standards of the industry
But only if you pay the right fee —
Only if you pay the right fee . . .”

But worst of all is what they’ve done
To software that we used to run
Like dbx and even /bin/cc.
Compilers now have license locks
Wrapped up in OpenWindows crocks —
We even have to pay for GCC!
The applications broke;
/usr/local went up in smoke.
The features we’ve depended on
Before too long will all be gone
But Sun, I’m sure, will carry on
By peddling Solaris,
Forever singing,

“Bye, bye, SunOS 4.1.3!
ATT System V has replaced BSD.
You can cling to the standards of the industry
But only if you pay the right fee —
Only if you pay the right fee . . .”

Canadian Open Systems Superuser

Sponsored by UniForum Canada

UniForum Canada is currently calling for nominations for its annual Canadian Open Systems Superuser award. This award is given each year to individuals who have made extraordinary contributions, in any field, to the Canadian UNIX and Open Systems Community. The award need not recognize achievements for a single calendar year, but can be more broad in scope. An award recipient need not be a member of UniForum Canada or an affiliate body. Organizations or incorporated bodies are not eligible.

Nominations for this award must be made, and seconded by members of UniForum Canada or an affiliate group. Nominations must be received by the Award Coordinator, Susan Zuk, by May 10, 1993. Anyone who would like to submit a nomination (or more) or would like more information, please call Susan at 788-7312.

The award will be presented to the recipient(s) at a future date, most likely at one of UniForum Canada's annual UNIX shows.

Agenda

for
Tuesday, March 9, 1992, 7:30 PM
St-Boniface Hospital Research Centre
Theatre, Main Floor, 351 Taché

1. President's Welcome 7:30

2. Round Table 7:40

3. Business Meeting 8:00
   a) Old Business
   b) New Business

4. Break 8:15

5. Presented Topic 8:30
   Linux: A Free UNIX!
   By Greg Moeller
   A talk about the youngest UNIX on the block. What it is, what it isn't, and do you want it? Complete with a question and answer session, where Greg can show his in depth knowledge of Linux... or his ability to bluff.

6. Adjourn 9:30

Note: Please try to arrive at the meeting between 7:15 and 7:30 pm. Thank You.

Coming Up

Meeting:
Next month's meeting is scheduled for Tuesday, April 13, at 7:30 PM. Meeting location will be given in the April newsletter. The meeting topic will be a lively presentation on Windows NT, by Microsoft.

Got any ideas for meeting topics? Any particular speaker or company you'd like to see at one of our meetings? Just let our meeting coordinator, Paul Hope, know. You can e-mail him at <phope@muug.mb.ca>.

Newsletter:
There are a few articles that should be on the way, including a book review on another book about the Internet. Of course, I can always use some more material, especially shorter articles – half a page to one page (400 to 1000 words) would be fine. Monsieur Ex has also let me know that his mail-box is empty once again – please submit your questions to the old guy via e-mail to <m-ex@muug.mb.ca> or by FAX to the MUUG Lines editor.