Volume 7, Number 5 March 1995 $2.50

MUUG Lines

Manitoba UNIX® User Group

Newsletter of the Manitoba UNIX© User Group

Rightsizing Your Network

By Andrew Trauzzi

With all the talk of 'downsizing', and 'rightsizing' CPUs and software over the past few years, one large rightsizing issue has been virtually ignored — the network. One reason for this is that there really hasn't been much of a selection of, network protocols. Sure, 16/4 token ring networks exist, but that's as close as we've seen a challenger attempt to knock off the (so far) undisputed king — 10 Mbps ethernet.

Over the past year, several new network protocols have been proposed — 100 Mbps ethernet, switched ethernet and token ring, and (most notably) ATM. Now, network designers and managers are faced with quite a choice. Should your company leave the comfortable but saturated arms of ethernet for the next new kid on the block? These new kids have much to offer — very high data transfer rates, simpler hardware and software requirements, and easier debugging and maintenance. They are also currently quite expensive, and most companies haven't created a need for digital video on everyone's desktop just yet.

Many network companies foresaw the dilemma that network managers would be faced with, and came up with alternative solutions. Instead of performing a massive and risky overhaul of a network overnight, companies can now ease into the new offerings, while still maintaining the old system.

Many managers would ask themselves if they required the full bandwidth of ATM, and many said no. IBM responded by introducing less-expensive, more usable 25 Mbps ATM, but that seems to have caused confusion among the vendors and users alike. As all of these fantastic new architectures are being dreamed up, your 10 Mbps ethernet network is becoming more and more saturated. What should you do?

Cisco Fusion Architecture

Internetworking architectures have evolved profoundly within the last ten years, from early designs based on bridging to today's popular designs based on routing. In the coming years, internetwork architectures will continue to evolve as new technologies such as Asynchronous Transfer Mode (ATM), LAN switching, and virtual LANs weave their way into the internetworking fabric. This month's speaker will identify the design characteristics, operational features, and constraints of today's most popular internetwork architectures, and introduce a new design for switched internetworks — the CiscoFusion™ architecture — that anticipates and satisfies internetworking goals into the twenty-first century.

The CiscoFusion architecture is unique because it allows sustainable growth through multilayer intelligence concurrently operating at different layers of the OSI model. Cisco's strategy and product plans for implementing this new architecture will be presented by a short video.

Thom Wolstencroft of Cisco Systems Inc. will present the video at our March meeting, and will also be answering questions after. Hope to see you there!

This Month's Meeting

Meeting Location:
Our next meeting is scheduled for Tuesday, March 14, 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

Printing Courtesy of Xerox Canada Ltd.

RAMBLINGS

Empty Pipes

By Andrew Trauzzi

Well, another blast of winter seems to have set upon us (well at the time of this writing anyway), and I scrape together whatever few free hours I have to create another newsletter issue. One problem though — I haven't had any articles sent to me for over three months! I really enjoy writing, but I just don't have the time to research a topic to my satisfaction before I write a story.

Really Rambling

You may have noticed your newsletter arrive later and later these past few months. This is due to a combination of things. First, I procrastinate. A lot. Second, I also rely on a few other regular columns written by other people, and they don't always come in on time (probably procrastination again). Third, we don't stuff the envelopes by hand anymore, so the third party factor has introduced some variability in the mailing time. Hopefully this month's newsletter arrived a week before the meeting instead of a day!

Solutions

Back in the good old days of the newsletter, people used to love to submit articles. Something about seeing their name in print giving them a thrill. (I know I get one!) Now, everyone has tight deadlines, more work, less pay, less staff, etc. God forbid a life outside the office! Sometimes, your boss may ask you to do some research on a cool new product. Write up a report, send one copy to your boss (the one with all the spelling mistakes), and send one copy to me. At this point in time, I would publish a study comparing user interfaces to religions. (That one exists — really!) Say you are surfin' the net (terms I truly despise). You suddenly come across an interesting site, article, picture, whatever. Mail me a copy — it only takes a second! Maybe you found some cool UNIX utilities or commands. Send them to me — you'd be surprised how many other people would find them useful as well.

Informal Survey

Finally, I would like to find out if any of you like/dislike the current regular articles in the newsletter. Am I missing any topics you would like covered? Let me know — I promise I won't make you write a new column (unless you want to). E-mail `<editor@muug.mb.ca>` with your opinion on the newsletter. I take criticism well, and kudos even better, so don't be afraid to insult the newsletter. No personal shots please — I get enough of those at work! I will publish the results (or lack thereof) in the next MUUGLines.

The 1993-1994 Executive

President:	Bary Finch	(W) 934-1690
Vice-President:	Rob Wright	(W) 488-5175
Treasurer/Secretary:	Doug Shewfelt	(W) 986-3748
Publicity Director	Rory Macleod	(W) 488-5168
Programs:	Doug McLean	(W) 985-1643
Mailing List:	Roland Schneider	1-204-785-9179
Membership Sec.:	Roland Schneider	1-204-785-9179
Newsletter editor:	Andrew Trauzzi	(W) 986-3898
Past President	Susan Zuk	(W) 989-3530
Information:	Bary Finch	(W) 934-1690 (FAX) 934-1308
(or)	Andrew Trauzzi	(W) 986-3898 (FAX) 586-8567

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.

Advertising Rates

Quarter page	$50
Half page	$75
Full page	$100
Insert (1-4 pages)	$100

Above prices are per issue. The first ad is charged at the full price; each successive month is 1/2 price.

Ad copy must be submitted by the final copy deadline for an issue (usually 3 weeks prior to the monthly meeting) in a format acceptable to the editor. (Please make arrangements with editor beforehand.)

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 at any meeting, or by mail.

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

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

PRESIDENT'S CORNER

UniForum Canada is Dead; Long Live UniForum!

By Bary Finch

A number of you have been wanting to renew your UniForum Canada memberships, or join UniForum Canada for the first time. It has been a number of months that we have been trying to re-establish our contacts with UniForum Canada, as we had not seen or heard anything from them. We were becoming suspicious that UniForum Canada was going through some difficulties, and was going to be unable to fulfill their membership requirements. I had been trying for quite a while by phone and by email to get in contact with someone (anyone) from UniForum Canada.

The board agreed that if we did not get in contact with UniForum Canada by the end of January, we would assume them to be out of business, and unable to fulfill their membership requirements. This means that any of you that intended to join UniForum Canada will not be able to do so, as we have no-one to forward your UniForum Canada membership fees to. As such, we will be refunding your UniForum Canada membership fees to each of you.

California Dreaming

On a brighter note, I have contacted the worldwide parent organization, UniForum, which is located in Santa Clara, California. They are the organization that puts on the annual UniForum conference, which is coming up soon, as well as many other events. They also provide the same benefits for membership that UniForum Canada would have (the Open Systems Product directory, the UniForum Monthly magazine, UniNews newsletter, etc.).

We will be including a membership form for UniForum along with each of your refund cheques. This way, if you are still interested in joining UniForum, you may do so directly to the parent organization in the US. You cannot join thru MUUG as we are not an affiliate at this point.

Also, UniForum confirmed that UniForum Canada had been having problems for a while, and was no longer an organization providing UniForum memberships and benefits. However, a number of the existing UNIX groups in Canada are becoming affiliates with the parent UniForum, as UniForum Canada is no longer available to be an affiliate of. MUUG will also be looking at this, as we still believe it is an organization that has many merits in promoting the Open Systems technologies and marketplace.

It will take a while for MUUG to evaluate whether we want to become and affiliate, or whether we qualify to become an affiliate, of UniForum. At this point looks like the requirements from us are very similar to that for UniForum Canada, and we will proceed towards becoming an affiliate. So hopefully we will again be offering memberships in UniForum as part of our membership package.

Firewalls

On another note, we have decided to put on a seminar this fall on Firewalls. Firewalls, for those of you that are unfamiliar with the term, are a combination of hardware and software that can isolate network traffic. This is a technology that has been use d for many years by organizations that require the ability to protect networks from other networks. A typical example is to isolate an internal network from the Internet, for rather obvious reasons, unless of course you WANT to have 20 million people looking at your network.

Our first step is to form a committee that will organize the seminar. This committee will work out all the details for the seminar, from where it will be held, to exactly what the seminar content will be.

(continued on page 8)

CORPORATE SPONSORS

The Manitoba UNIX User Group

gratefully acknowledges the
generous support of the following

Corporate Sponsors

IBM

digital™

hp HEWLETT
PACKARD

Great-West Life Assurance
Company

THE DOCUMENT COMPANY
XEROX

TANDEM COMPUTERS

ONLINE
BUSINESS SYSTEMS

sun
microsystems

C++ Q&A

By Marshall P. Cline

This month's C++ Q&A completes last month's look at standards. The complete C++ FAQ is now available in a book format—Addison-Wesley Publishers 0-201-58958-3 $32.25.

Question 77: What are some good C++ coding standards?

Thank you for reading this answer rather than just trying to set your own coding standards. But please don't ask this question on Usenet. Nearly every software engineer has, at some point, felt that coding standards are or can be used as a 'power play'. Furthermore some attempts to set C++ coding standards have been made by those unfamiliar with the language and/or paradigm, so the standards end up being based on what *was* the state-of-the-art when the setters where writing code. Such impositions generate an attitude of mistrust for coding standards. Obviously anyone who asks this question on Usenet wants to be trained so they *don't* run off on their own ignorance, but nonetheless the answers tend to generate more heat than light.

Question 78: Are coding standards necessary? sufficient?

Coding standards do not make non-OO programmers into OO programmers. Only training and experience do that. If they have merit, it is that coding standards discourage the petty fragmentation that occurs when organizations coordinate the activities of diverse groups of programmers.

But you really want more than a coding standard. The structure provided by coding standards gives neophytes one less degree of freedom to worry about, however pragmatics go well beyond pretty-printing standards. We actually need a consistent *philosophy* of implementation. Ex: strong or weak typing? references or ptrs in our interface? stream I/O or stdio? should C++ code call our C? vise versa? should we use ABCs? polymorphism? inheritance? classes? encapsulation? how should we handle exceptions? etc.

Therefore what is needed is a 'pseudo standard' for detailed *design*. How can we get this? I recommend a two-pronged approach: training and libraries. Training provides 'intense instruction', and a high quality C++ class library provides 'long term instruction'. There is a thriving commercial market for both kinds of 'training'. Advice by organizations who have been through the mill is consistent: Buy, Don't Build. Buy libraries, buy training, buy tools. Companies who have attempted to become a self-taught tool-shop as well as an application/system shop have found success difficult.

Few argue that coding standards are 'ideal', or even 'good', however many feel that they're necessary in the kind of organizations/situations described above.

The following questions provide some basic guidance in conventions and styles.

Question 79: Should our organization determine coding standards from our C experience?

No matter how vast your C experience, no matter how advanced your C expertise, being a good C programmer does not make you a good C++ programmer. C programmers must learn to use the '++' part of 'C++', or the results will be lackluster. People who want the 'promise' of OOP, but who fail to put the 'OO' into OOP, are fooling themselves, and the balance sheet will show their folly.

C++ coding standards should be tempered by C++ experts. Asking `comp.lang.c++` is a start (but don't use the term 'coding standard' in the question; instead simply say, 'what are the pros and cons of this technique?'). Seek out experts who can help guide you away from pitfalls. Get training. Buy libraries and see if 'good' libraries pass your coding standards. Do *not* set standards by yourself unless you have considerable experience in C++. Having no standard is better than having a bad standard, since improper 'official' positions 'harden' bad brain traces. There is a thriving market for both C++ training and libraries from which to pool expertise.

One more thing: whenever something is in demand, the potential for charlatans increases. Look before you leap. Also ask for student-reviews from past companies, since not even expertise makes someone a good communicator. Finally, select a practitioner who can teach, not a full time teacher who has a passing knowledge of the language/paradigm.

Question 80: Should I declare locals in the middle of a fn or at the top?

Different people have different opinions about coding standards. However one thing we all should agree on is this: no style guide should impose undue performance penalties. The real reason C++ allows objects to be created anywhere in the block is not style, but performance.

An object is initialized (constructed) the moment it is declared. If you don't have enough information to initialize an object until half way down the fn, you can either initialize it to an 'empty' value at the top then 'assign' it later, or initialize it correctly half way down the fn. It doesn't take much imagination to see that it's cheaper to get it right the first time than it is to build it once, tear it down, then rebuild it again. Simple examples show a factor of 350% speed hit for simple classes like String. Your mileage may vary; surely the overall system degradation will be less that 300+%, but there *will* be degradation. *Unnecessary* degradation.

A common retort to the above is: 'we'll provide "set" methods for every datum in our objects, so the cost of construction will be spread out'. This is worse than the performance overhead, since now you're introducing a maintenance nightmare. Providing 'set' methods for every datum is tantamount to public data. You've exposed your implementation technique to the world. The only thing you've hidden is the physical *names* of your subobjects, but the fact that you're using a List and a String and a float (for example) is open for all to see. Maintenance generally consumes far more resources than run-time CPU.

Question81: What source-file-name convention is best? 'foo.C'? 'foo.cc'? 'foo.cpp'?

Most Un*x compilers accept '.C' for C++ source files, g++ preferring '.cc', and cfront also accepting '.c'. Most DOS and OS/2 compilers require '.cpp' since DOS filesystems aren't case sensitive. Some also advocate '.cxx'. The impact of this decision is not great, since a trivial shell script can rename all .cc files into .C files. The only files that would have to be modified are the Makefiles, which is a very small piece of your maintenance costs. Note however that some versions of cfront accept a limited set of suffixes (ie: some can't handle '.cc'; in these cases it is easier to tell 'make' about CC's convention than vise versa).

You can use '.C' on DOS or OS/2 if the compiler provides a command-line option to tell it to always compile with C++ rules (ex: 'ztc -cpp foo.C' for Zortech, 'bcc -P foo.C' for Borland, etc).

Dr. Marshall P. Cline is the founder and President of Paradigm Shift, Inc., a firm that specializes in on-site training for C++, OOD, OOA, consulting, and reusable/extensible C++ class libraries. For more information, send e-mail to "info@parashift.com".

UNIX Q&A

Originally Compiled by Ted Timar

Submitted by Andrew Trauzzi

This month's UNIX Q&A examines some shell issues.

Question 1: Can shells be classified into categories?

In general there are two main class of shells. The first class are those shells derived from the Bourne shell which includes sh, ksh, bash, and zsh. The second class are those shells derived from C shell and include csh and tcsh. In addition there is rc which most people consider to be in a "class by itself" although some people might argue that rc belongs in the Bourne shell class.

With the classification above, using care, it is possible to write scripts that will work for all the shells from the Bourne shell category, and write other scripts that will work for all of the shells from the C shell category.

Question 2: How do I "include" one shell script from within another shell script?

All of the shells from the Bourne shell category (including rc) use the "`.`" command. All of the shells from the C shell category use "source".

Question 3: Do all shells have aliases? Is there something else that can be used?

All of the major shells other than sh have aliases, but they don't all work the same way. For example, some don't accept arguments. Although not strictly equivalent, shell functions (which exist in all shells from the Bourne shell category) have almost the same functionality of aliases. Shell functions can do things that aliases can't do. Use unalias to remove aliases and unset to remove functions.

Question 4: How are shell variables assigned?

The shells from the C shell category use "`set variable=value`" for variables local to the shell and "`setenv variable value`" for environment variables. To get rid of variables in these shells use unset and unsetenv. The shells from the Bourne shell category use "`variable=value`" and may require an "`export VARIABLE_NAME`" to place the variable into the environment. To get rid of the variables use unset.

Question5: How can I tell if I am running an interactive shell?

In the C shell category, look for the variable `$prompt`.

In the Bourne shell category, you can look for the variable `$PS1`, however, it is better to check the variable `$-`. If `$-` contains an '`i`', the shell is interactive. Test like so:

```
case $- in
  *i*)  # do things for interactive shell
        ;;
  *)    # do things for non-interactive shell
        ;;
esac
```

Question 6: What "dot" files do the various shells use?

Although this may not be a complete listing, this provides the majority of information.

csh

Some versions have system-wide `.cshrc` and `.login` files. Every version puts them in different places.

```
Start-up (in this order):
    .cshrc    - always.
    .login    - login shells.
Upon termination:
    .logout   - login shells.
Others:
    .history - saves the history.
```

tcsh

```
Start-up (in this order):
    /etc/csh.cshrc - always.
    /etc/csh.login - login shells.
    .tcshrc    - always.
    .cshrc     - if no .tcshrc was present.
    .login     - login shells
Upon termination:
    .logout    - login shells.
Others:
    .history   - saves the history.
    .cshdirs   - saves the directory stack.
```

sh

```
Start-up (in this order):
    /etc/profile    - login shells.
    .profile  - login shells.
Upon termination:
    any command (or script) specified using the
    command:  trap "command" 0
```

ksh

```
Start-up (in this order):
    /etc/profile    - login shells.
    .profile  - login shells.
    $ENV- always, if it is set.
Upon termination:
    any command (or script) specified using the
    command:  trap "command" 0
```

bash

```
Start-up (in this order):
    /etc/profile  - login shells.
    .bash_profile - login shells.
    .profile  - login if no .bash_profile exists
    .bashrc   - interactive non-login shells.
    $ENV- always, if it is set.
Upon termination:
    .bash_logout  - login shells.
Others:
    .inputrc  - Readline initialization.
```

zsh

```
Start-up (in this order):
    .zshenv   - always, unless -f is specified.
    .zprofile - login shells.
    .zshrc - interactive shells
    .zlogin   - login shells.
Upon termination:
    .zlogout  - login shells.
```

➡

INDUSTRY

The Sendmail Interview

By Tim O'Reilly

In this article, Tim O'Reilly (of O'Reilly books), interviews sendmail's creator — Eric Allman.

TIM: Can you tell us a little bit about how sendmail came about? I know it's old history for you, but it may be of interest to our readers.

ERIC: The exact chronology is a little bit hard for me to get right, so we'll just make this up as we go along. I was working on the Ingres Project at Berkeley. This was about the same time that Ernie CoVax, the 11/780, the first 32-bit machine that Berkeley had, came in. I think they had UUCP of some sort. There were a about a dozen UNIX machines around on campus at that time. Eric Schmidt had written BerkNet, which was a sort of UUCP style network, only instead of having dial-up lines, it was connected all the time. It was a store and forward net. You could do remote execs painfully, but really it was for file transfer and mail transfer. Then the Arpanet came in, and that had a whole different set of e-mail standards. The people working in the Arpanet world at that point were mostly Tenex (later known as Twenex because it became TOPS20). They used different mailbox formats, different mailers, etc. So when you sat down on the Ingres machine, you had to use msg to send Arpanet mail and Berkeley Mail to send to anywhere else, and you couldn't send one message to both places at once. That didn't last very long before hooks started to be made to send mail to other networks. Those were mostly done by Kurt Shoens and other people surrounding him. This was back in the days of Berkeley where all source code was publicaly writable by anybody at anytime, and so if you wanted a new feature, you put it in. Common courtesy demanded that you ask the author first, but sometimes common courtesy was less than common.

TIM: When was that?

ERIC: About 1977. This was the Bill Joy period, when everything seemed to be happening all at once. It was really very exciting. The hooks for UUCP mail were done in /bin/mail. The hooks for BerkNet were done in Berkeley Mail and the hooks for Arpanet were done somewhere else altogether. Networks didn't talk to other networks. This was a clearly unstable situation. People were starting to hack up different programs, so that for example, if there was an "@" sign, the mailer would send it off to the Arpanet. This quickly became unstable because you couldn't say, "Oh, my address is this." You had to say to somebody, "What mail program do you use? Oh! With that program my mail address is such and such." It was far worse than you can possibly imagine. Add to that the different mailbox formats. Fundamentally, if you were sitting on Ernie, you couldn't send mail to the Arpanet, but you could receive mail from the Arpanet. So we had a lot of people who wanted accounts on our machine, and that started to become unstable because it was only an 11/70. I'm not really telling this very well, but I'm trying to present this image of chaos.

Somewhere along in here, it became clear that this wasn't going to work. I started to work on something that would fix the situation. I spent a tremendous amount of time trying to figure out what to do. Couldn't figure it out. One day I just said, "Fine, it's become critical, I'm going to write the ad hoc code, and I'll worry about how to do it right later." I started writing the ad hoc code. I can remember where I was sitting, and the way the light came into the room. I got out the pad of paper, and I started writing. I wrote about two pages and I said to myself "Oh, that's the way to do it!" That turned into delivermail, which was the precursor of sendmail. It had compiled-in configurations, which was okay when you only had a dozen hosts. It was all dependent on characters. If there's an exclamation point in the name, do this. If there's an "@" sign in the name do that. That sort of thing.

TIM: Sounds like sendmail was an outgrowth of just trying to create order out of the chaos at that time.

ERIC: I think the really interesting programs are not written by people who sit around and say, "Oh, let's think of the next new widget we can build," but they have a real problem that they're trying to solve. UNIX was in its best state when people were solving real problems that they had then and there, as opposed to anticipating what somebody's problem might be somewhere...sometime.

Anyway, the world continued to become more complicated. For example, at one point, all UUCP links were connected to Ernie CoVax, so if you saw an address with an exclamation point, you'd just send it off to that machine. It didn't take very long before there were multiple machines that had UUCP connections.

Back in the NCP [precursor to TCP/IP] days of the Arpanet, there were a grand total of 254 possible sites on the network. You addressed messages to, for example, "user@MIT-XX." It was small enough so that at one point there was actually an RFC [Request for Comment document] where they were talking about standardizing the reply codes for e-mail. One weekend some guy connected to every single host on the network — the entire network — and checked to see what happened when he presented them with certain inputs. A couple of days later he published this RFC where he said 70% of the hosts do this, so let's standardize it. Then TCP came along with 32-bit addresses instead of 8-bit addresses. We came up with domains. We converted formats from RFC 733 to 822 message formats. We changed from mail being sent through the FTP protocol to having its own protocol, SMTP.

Through all of this period, I was developing sendmail, and running as fast as I could to keep it up-to-date with this week's version of the protocols. There were literally new drafts coming out every week. The configuration file turned out to be a very valuable tool because I usually had to make minimal code changes, plus a few config file changes. There were several times when a new draft of RFC 821 (only it wasn't numbered then) came out, and the next day I had it implemented and could provide feedback on how thus and such worked. So in some sense, sendmail really helped the development of those protocols. There were a bunch of things in the protocols that were simply unimplementable, and they were going to standardize it. In a lot of the protocols you find, typically with ANSI and ISO, a lot of things are designed but never built. That's why POSIX.1 really does work because it's standardizing something we have been working with and playing with for a long time. I mean, there are things in "DOT 1" I don't ☛

like, but at least we know what's in there, whereas some of the "DOT n's" where "n" is greater than 200, are just pie-in-the-sky magic things. I don't think standards of that sort can really survive.

TIM: It's often been said that when you originally implemented the format of the configuration file, you decided to make it easy for the computer to deal with rather than easy for people to deal with because you figured the computer had to read it often, whereas people had to deal with it only once or twice. Is that a fair statement?

ERIC: True story here. My first config file was about fifteen lines long. I actually came across it a couple of years ago and threw it out. What an ass! At fifteen lines, who cares what the syntax is? My philosophy at the time was that when I had to change something, I put it in as an option. That way, I wouldn't have to go in and muck up the code again, based on the assumption that something that changed once was probably going to change twice. That turned out to be relatively true. One day I printed it out again and it was 4 pages. How did this happen? I should have gotten wise and said, "Wait! Something's wrong." I also think people write config files that are unnecessarily obscure.

TIM: I think when people poorly understand something, its easier to add to it than go back and take things out because you might break something.

ERIC: Well, there is an awful lot of stuff that crept in over the years. I looked at what Berkeley was shipping as I was working on this latest release and looked it over and said, "I really do not understand what this is." So I threw it all out and started over. I came up with something that was about half the size.

TIM: It sounds like a lot of the problems in sendmail syntax are related to the fact that it grew historically in response to rapidly changing demands from a lot of corners. It really wasn't something that was designed from an overall integrated perspective.

ERIC: I am not the sort of person that goes to bed at night thinking, "Gee, I wonder what I can do to make life difficult for systems administrators."

TIM: I think a lot of people will be relieved to hear that you have suffered from your own creation.

ERIC: I've suffered probably more than anyone because I get the weirdest problems. They show up in my mailbox. It's interesting to note that sendmail — because it tends to be so adaptive and so powerful — has probably perpetuated more bad mail software than anything else around. For example, at one point I spent some time looking at messages as they came in off the wire, before sendmail got hold of them. I would say almost half of the messages going over the Net are in an incorrect format now, and sendmail fixes them before you ever see them. For example, if you are using / usr/ucb/mail or mailx or whatever, and you say

```
        To:     tim eric
```

with no comma between the names, it just sends it out. Sendmail puts the comma in. Because sendmail does that, nobody has found it essential to go in and fix that damn user interface, which is just wrong. By any measure it's wrong. (Well, that's not quite right. A long time ago — when we didn't have any interoperability to consider — we just used spaces for the separators, just like we do on command lines. But today we have standards.)

TIM: How different is the new version 8 of sendmail?

ERIC: What I really wanted to write was a whole new program tentatively dubbed Son of Sendmail. I was going to completely restructure the code. There are some things that need to be done — for example, inverting the way the queue is done so instead of processing messages, you process hosts. You open a connection to a host and send everything youve got for that host, and so forth. I haven't done any of those things. But there is a huge list of changes and some new features. A lot of them are performance enhancements, things like connection caching. Let's say you have ten messages queued up all to be sent to the same host, which is actually pretty common if it's a major host that has been down for a few minutes. The old sendmail opened a connection, sent the first message, closed the connection, opened the connection to the same host, sent the second message, then closed it. Connection caching says open it, send it, send second, send third, and so forth, and close it when you're all done. That is not the same as doing it right because all it really does is keep the cache of a small number of open connections and uses it if it's there. But, in fact, you can have connections open to multiple hosts at once.

I didn't want to make the level of changes in the code that would be necessary to do it completely right. Frankly, I wanted something that would look familiar enough to people so that even if they hated it, they would still say, yes, I recognize this, and they would be less afraid to run it.

There are enhancements for new standards. The old sendmail is not RFC 1123 compliant, but the new one is. I believe that those upgrades are important. Once again, I have some disagreements with some things in RFC 1123 but it is still a step forward, and I would like to see vendors pick up a version of sendmail that will support these things. Until the major vendors — Sun, DEC, and HP — pick up 1123 compliant mailers, there is no chance that the Net will be 1123 compliant.

TIM: What made you decide to revise sendmail after all these years?

ERIC: There are several reasons. One was simply that we were trying to put hosts into a CS subdomain at Berkeley. Berkeley has just gotten too big. It doesn't work any more to have it all as one domain. That prompted me to look at the code again to put in what is now the "user database."

At the same time, Bryan Costales was writing the O'Reilly book on sendmail, and he asked me if I would mind if he wrote it because he figured that, after all, it was my book. I said something along the lines of, "Please, be my guest." Bryan started writing the book, but I agreed that I would review chapters because it is not to anyone's advantage, least of all mine, to have incorrect information out there. Believe me, there is a huge amount of misinformation about sendmail. So, Bryan started passing chapters by me. He's the sort of person who was really trying everything, just everything. He was trying things no sane person would ever try. As he asked me to review more chapters of the book, the changes started to get pretty serious. He'd find something that didn't work and I'd say, "Yeah, you're right, that's pretty stupid." So I'd fix it. A huge number of corrections resulted from Bryan writing the book. It's really his fault. ➧

This interview was excerpted from the fall 93 issue of ora.com, O'Reilly's print magazine and catalog.
O'Reilly & Associates also publish the definitive guide to sendmail.

MEETINGS

SIG Sideline

By Andrew Trauzzi

The SIG group needs a new co-ordinator! If you are interested in taking an active role in MUUG activities, please contact the board at `<board@muug.mb.ca>`**.**

The next SIG meeting will be March 21, 1995, at 7:30 PM. As usual, the meeting will be at ISM, 400 Ellice Avenue (behind Portage Place). Thanks again to Wolfgang von Thuelen for hosting these meetings.

The Net

Internet Corner

By Andrew Trauzzi

A document that is on its way to Internet RFC status is something you might be interested in taking a look at. The document contains all US and international area codes, telex codes, country codes, etc. The name of the document is /internet-draft/draft-robinson-newtelex-01.txt and it can be obtained by anonymous ftp from the following sites:

Africa:	ftp.is.co.za	(196.4.160.2)
Europe:	nic.nordu.net	(192.36.148.17)
Pacific Rim:	munnari.oz.au	(128.250.1.21)
US East Coast:	ds.internic.net	(198.49.45.10)
US West Coast:	ftp.isi.edu	(128.9.0.32)

Agenda

for
Tuesday, March 14, 1995, 7:30 PM
Samuel N. Cohen Auditorium
St-Boniface Hospital Research Centre
Main Floor, 351 Taché

1. President's Welcome 7:30

3. Business Meeting 7:35
   a) Old Business
   b) New Business

5. Presented Topic 7:45
   Thom Wolstenkroft of Cisco Systems will discuss the CiscoFusion™ architecture. A short video on CiscoFusion will also be presented.

4. Coffee Break and Informal Discussion 9:00

Note: Please try to arrive at the meeting between 7:15 and 7:30, to avoid disrupting the meeting in progress.

Coming Up

Meeting:
Next month's meeting is scheduled for Tuesday, April 11, at 7:30 PM. Meeting location will be the St-Boniface Research Centre, as usual. The March meeting topic is the database shootout.

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

Newsletter:
If you are interested in a particular topic, let me know. I'm sure I could coerce you into writing an article! I could use a few articles — especially shorter ones — half a page to one page (400 to 1000 words) would be fine.

Monsieur Ex has also let me know that his mail-box has room for more of your wonderful queries again – please submit your questions to the old guy via e-mail to `<m-ex@muug.mb.ca>`. He may be old, but he's not ready for retirement yet!

President's Corner

(Continued from page 1)

So, if any of you are interested in being a member of this committee, please let me know. There will be several positions available, such as coordinating the arrangements for the speakers, covering all the details for the location of the seminar, and doing all the publicity details of marketing the seminar.

While I'm on the topic of getting assistance, we are also in need of a new SIG (Special Interest Group) coordinator. This is for our Linux / System Administration SIG, which is held the third Tuesday of each month. If you have an interest in Linux or System Administration, or are just interested in coordinating our SIG, we would love to hear from you. This would be an especially easy task for someone who already attends the SIG.

The main tasks of this role are to chair each of the SIG meetings, write the SIG Sideline monthly article (which is a recap of the meetings events, a highlight of the next meeting, and any other items of interest), and help find speakers for each meeting.

Of course, any new ideas for the role would be most welcome. We need someone to take on this role for the next SIG. I'd like to thank our former SIG Coordinator Gilbert Detillieux for all his efforts towards the SIG.

That's it for this month. I hope you can all make it out to the next meeting on March 14. We will be back to normal on the second Tuesday of the month.