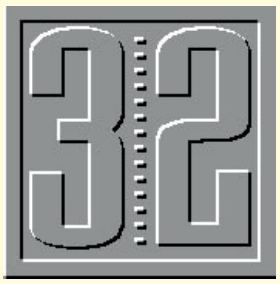


A Most Excellent pest control

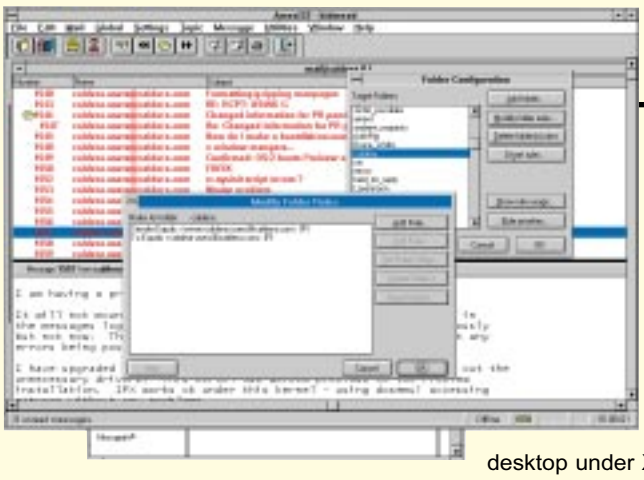
Chris Bidmead looks at ways of bug-bashing via a mailing list on Caldera Preview, stands his ground on Emacs evangelism, and defies the old adage: "Time waits for no man".

As you might expect from the name, the Caldera Preview releases have had their share of minor bugs. Pest control is being carried out through a mailing list which is a kind of electronic conference that works through your regular email box. The mail server at the far end ensures that whenever you address a message to a particular recipient, in this case caldera-users@caldera.com, your missive is automatically re-sent to all the subscribers to the mail group.

This is a simple, elegant and downright Unixy way of running a private conference, but it does have one snag. If your mail-reading app is too simple-minded, it won't offer any way of separating the mail list messages from the rest of your mail traffic. The mail reader I use, Ameol (A Most Excellent Offline reader) doesn't know how to sort mail, so on signing up to the hyperactive Caldera list, my regular daily correspondence quickly disappeared into a blizzard of chat about Caldera.

I'm currently running the 32-bit version of Ameol under Windows NT. Ameol is primarily designed as a front-end to the CIX conferencing system, but CIX also handles my mail and uploads it to me automatically whenever I go on line. Yes, I suppose I ought to be exploring purer Unix-based mail systems, and sendmail is definitely on my agenda. But Ameol and CIX have been my faithful postmen for several years now, and I'm loath to fix 'em when they ain't broke.

The mailing list problem, however, was on the point of driving me to settle down with the O'Reilly "Sendmail" book, when I came across an Ameol add-on, written by Martyn Lovell. Called Mailsort, it's an electronic filtering system that checks incoming mail and categorises it according to rules that you program into it. The particularly nice thing about Mailsort for me is that like Ameol itself, it comes in 16-bit and 32-bit flavours, so I can run it on my OS/2 system as well as under NT.



The 32-bit version of Ameol, running on Windows NT

A hitch for Portage

I've recently written in this column about the Consensys product, Portage, which bolts on to Windows NT and gives it most of the functionality of Unix System V right down to kernel level. Unfortunately, Windows NT 3.51 breaks Portage, throwing up a segmentation fault error when it loads the Portage kernel at boot time. I'd come to depend on Portage as a bridge between Windows NT and my other Unix systems. But the new version of Windows NT now fits much more comfortably onto my 16Mb DX2 machine, so it's Portage that gets the elbow, and I've promised myself to seek out interesting non-Unixy things to do with the operating system.

What finally set the seal on Portage for me was the discovery that despite the promises of the Consensys brochures, the company will not after all be coming out with an X Windows System to accompany the product. One of my hopes for Portage was that it would eventually fulfil my ambition to unify Windows NT and Linux on a single desktop under X, but I'm now going to have to come to that via a different route. Pity, because judging from the email that flooded in when I first wrote about Portage, it's clearly a product that fills a gap. And as a core Unix-on-NT product it continues to work fine if you don't mind sticking with the older version of Windows NT.

I expect I'll be coming back to Portage when the compatibility problem is fixed, but I think Consensys would probably be wise to skip this present version and wait for Windows NT to settle down first. I gather that version 3.51 was originally intended to be distributed with the Windows 95-like interface, but the final build of that didn't arrive in time for shipment. Hopefully, by the time you read this Windows NT will be in sync with Windows 95 and the updated version of Portage will be winging its way to me.

Beating the clock

In common with many laptops, my faithful Tonto has a suspend mode that shuts down the system when you close the lid, then restores everything next time you open up the machine again. This seems to

I stand by Emacs evangelism!

A few people have wagged fingers at me over the evangelism for the good ol' char-based interface I was parading last month. I retract nothing, but perhaps I should put this into context. Last month I was rediscovering Emacs. It was a joy to be able to get on with the job of writing without the burden of function and screen furniture that a modern word processor like Microsoft Word for Windows heaps on you. More about Emacs in a moment. My hearty endorsement of the character-based screen was, in retrospect, perhaps one of those "necessity is the mother of invention" things. As I think I said, I'd taken Linux (and Emacs) away on holiday and was stuck with a portable on which I couldn't get X to run.

Since then, a good deal of twiddling and the indispensable help of a new version of Caldera (Preview II, which includes the latest XFree86 version 3.1.2) has changed the picture. XFree86 3.1.2 now properly supports the Western Digital WD90C24 video chip used in a lot of portables, and at last I've been able to equip Tonto with a very handsome 640 x 480 x 256 X-based screen. In Caldera, the Motif-like GNU fvwm (feeble virtual window manager) goes on top of X, and on top of that the distribution runs a proprietory desktop manager called Looking Glass (note: unlike the rest of Linux, *not* for free distribution). It's in this environment that I'm currently running Emacs, writing this column in a salmon-coloured, blue-bordered window; one of several I can have up at the same time.

For and against

Okay, arguments against using X and Emacs; really only one. Emacs is already pretty huge, if you include all the macros, extensive documentation and tutorials that come with it (well, I suppose 9Mb is pretty modest by current word processor standards). X adds a whole clump more code to your hard disk and puts paid to the idea of running anything serious in less than 8Mb of RAM.

Arguments for; lots. Most importantly, you don't dispense with any of the simple goodness I was raving about last month. Entering and navigating text remains as fast as you could wish. Admittedly, the screen starts to look a little more complicated — when Emacs detects X, it puts on its party clothes in the shape of a menu bar at the top of the window. You can pull down sub-menus with the mouse in the usual way, and pop up menus directly from within the windows with commands like the Control key and Left mouse button combo (which gives you a choice of different screen fonts). I prefer the versatile keystroke combinations for the basic stuff. I find the X Windows presentation easier on the eye than a raw char screen, considering that you can set your choice of background colour and font.

Finger-flickin' good

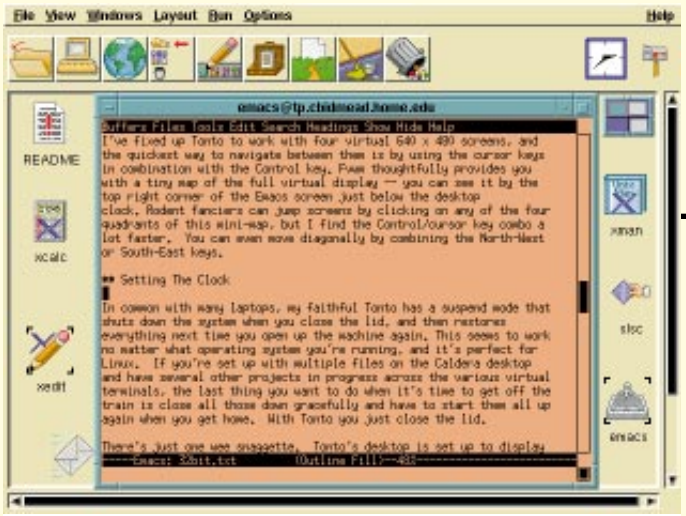
Flicking between Emacs windows ("frames") is handy too in the X version. The char-based Emacs supports this as well, although you don't get to slip and slide the frames with a mouse, just switch over between virtual screens. I've found it best not to use the mouse for this, anyway. The fvwm display can be arranged to have a virtual size that's bigger than the physical screen can show — the actual ratio depends on how much video RAM you have. I've fixed up Tonto to work with four virtual 640 x 480 screens, and the quickest way to navigate between them is by using the cursor keys with the Control key. Fvwm provides you with a tiny map of the full virtual display — you can see it by the top right corner of the Emacs screen just below the desktop clock. Rodent fanciers can jump screens by clicking on any of the four quadrants of this mini-map, but I find the Control/cursor key combo a lot faster. You can even move diagonally by combining the North-West or South-East keys.

work no matter which operating system you're running, and it's perfect for Linux. If you're set up with multiple files on the Caldera desktop and have several other projects in progress across the various virtual terminals, the last thing you want to do when it's time to get off the train is close all those down gracefully and have to start them up again when you get home. With Tonto you just close the lid. There's just one wee snaggette. Tonto's desktop is set up to display a clock, so I can keep an eye on the time as I write. This is updated by Linux's own system clock, but when you go into suspend mode, the clock does too. So when you next open the lid, on Friday morning, Linux still thinks it's Thursday evening. I decided to live with this, and apart from the clock falling drastically behind, everything worked fine and I let a week go by without rebooting the system.

When I finally rebooted, I happened to notice a system startup message which claimed to be "setting the clock", and once up and running I noticed that I was indeed back on time.

Get a grep on it

The machine's hardware clock, which stores its data in CMOS, was obviously



Tonto and Emacs, sitting on the Caldera desktop with Xclock at last showing the right time

unaffected by suspend mode. Only the Linux system clock was losing time, and on bootup something was synchronising them back again. The lesson I've learnt from Unix in general and Linux in particular is that you can usually find out how things work by snooping around, and the place to start when you're looking at initialisations is the etc/rc.d directory. This contains the shell scripts that are run every time the system powers up, and you can track down which does what by grepping through them, looking for the relevant string you saw on screen during the initialisation process.

In one of the main files, rc.sysint, I discovered the string I was looking for and found it was associated with a command called "clock". Running "man clock" returned a definition of the programming function clock() which wasn't what I wanted. If you're looking for a user command that happens to have the same name as a programming function, you have to explicitly mention the manual volume you're looking for, or use man with the -a parameter, which will show you entries in all the sections.

I happened to know that user commands are in the eighth volume of the manual, so I re-entered the request as "man 8 clock" and got the following description:

```
Clock manipulates the CMOS clock in
variaous (sic) ways, allowing it to be
read or written, and allowing
synchronization between the CMOS
clock and the kernel's version of the
system time.
```

It turns out that the -s option updates the Linux system clock from the CMOS clock, so all I had to do was just run this manually every time I returned from Suspend mode. But in fact there's a better way. Why not just run "clock -s" regularly to make sure the two stay in sync?

You can do this by taking advantage of a long-standing Unix institution, the cron daemon, one of several "hidden helpers" which chug away in the background getting things done for you. Cron keeps an eye on the clock, and runs tasks at a particular time of day, or on a particular day of the month, or whatever. You set these tasks up in a somewhat cryptic text file that lives in the directory /var/spool/cron/crontabs/ and is named after your particular account.

The system also keeps another crontab file in the directory/etc which is the responsibility of root. This system crontab takes care of things like cleaning out temporary files regularly and running scheduled updates on various files. For example, the root cron is typically set up to update the "what is" database on a regular basis (see later).

I decided that the clock update dodge I was about to install wasn't really a core system responsibility, so didn't belong in the main /etc/crontab file. (A decision somewhat influenced by the fact that this file has a slightly different format from the user crontabs, and I didn't want to risk screwing up anything critical.) But on the other hand, the clock update shouldn't just be associated with the Emacs user on my system, called "elbid". Tonto only gets used by me, but in various capacities.

Who should I be today?

After working with a Unix system for a while you get used to dividing yourself up into a gang of different users, depending on what you happen to want to do with the machine. Root takes care of system admin, el bid writes this column, bidmead issues invoices, and so on. This kind of applied schizophrenia turns out to be very useful. I solved the dilemma by logging in as root and editing root's personal crontab file, stored with the others in the /var/spool/cron/crontabs/ directory.

You are not encouraged to edit the cron file directly; instead there's a combined editor/viewer utility called crontab that evokes vi and makes sure the revised file is presented back to cron for processing. The line I added to my crontab was:

```
*/3 * * * *     /sbin/clock -s
```

The man pages for crontab will fill you in on the fine detail; essentially the line is divided into two fields. The first, comprised here mostly of stars, defines when and how often the action has to take place, and the second field defines the action.

So now when I open the lid, the clock on the screen is still wildly out, just for the first minute or so. Then the cron command kicks in, momentarily blanks the screen (for some reason I'm calling this "a feature") and updates the clock.

Hello, handsome

The AIX box has finally arrived, just as I was putting this column together. It's a very handsome PowerPC 604 machine, and I'll tell you all about it next month.

Unixes are getting more and more similar, but AIX has its little peculiarities. The standard way of getting up to speed on a new system is to use the man pages, and more particularly the apropos command. Type something like "apropos disk" on the shell command line and you'll get a short summary of every command in the man pages relevant to the word "disk".

Apropos works through an index database file called "whatis", which on Linux systems is assembled from the man pages by running "makewhatis". (Other Unixes work similarly, but may use different commands.) On the new AIX box, apropos kept returning "Cannot find matching entry", a firm indication that the whatis database hadn't been compiled.

Only when I got stuck into finding out how to do that, did I discover there weren't any man pages on the system to index. I went to install CD and fished about in there for about half an hour and turned up nothing. Eventually I contacted IBM, and we had the following exchange:

Bidmead: "So where are the man pages, then?"

IBM: "They're part of our general system information package, InfoExplorer."

Bidmead: "Ah. So where's InfoExplorer?"

IBM: "InfoExplorer is a cost option, available for £310."

Bidmead: Collapses open-mouthed in astonishment.

PCW Contacts

Chris Bidmead is a consultant and commentator on advanced technology. He can be contacted on **bidmead@cix.compulink.co.uk**