



# The common good

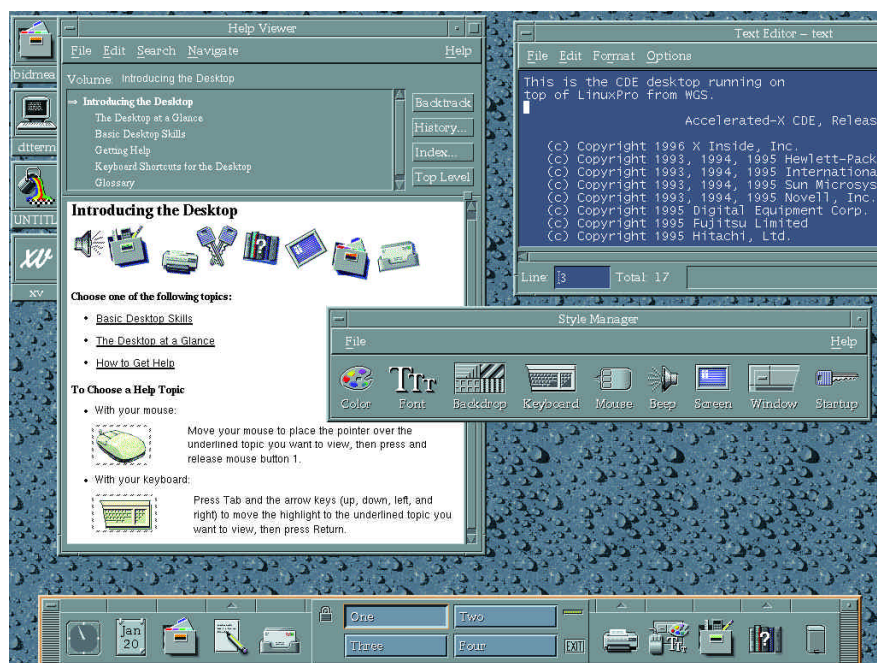
... and bad, unfortunately. But Chris Bidmead is in combative mood as he attempts to install the Common Desktop Environment on his network, and defends a remark he made earlier.

A couple of months ago I mentioned that CDE, the Common Desktop Environment, is now available for Linux, and I showed you a screenshot of it running on LinuxPro, the version of RedHat Linux supplied by WGS (Workgroup Solutions) whose web page (unsurprisingly) is [www.wgs.com](http://www.wgs.com).

I'd borrowed the screenshot from the WGS web site. But this is supposed to be the Hands On section, so I made a vow to myself that I'd follow through by getting hold of the software and installing it on my network. There's some bad news about this, some good news, some more bad news, but ultimately, I'm glad to say, some good news.

Before we get into that, I'd like to deal with some heated email from a reader who objected to my remark in the column that the CDE screenshot made LinuxPro look "uncannily like the AIX desktop — which of course is the point of CDE". When the major UNIX manufacturers got together in 1993 around an initiative called COSE (the Common Open Software Environment), of which CDE was to be the first component, the intention was to simplify things for system administrators and users by offering a "look and feel" that would be similar across all the UNIX platforms. One of the first manufacturers to implement this was IBM on AIX, and as I happen to run AIX here, it seemed worth mentioning the resemblance.

Andrew Lehane seemed to think I was suggesting IBM had invented CDE singlehandedly. If my very brief mention in the screenshot caption gave that impression, I'm sorry. Long-term readers of this column may remember that back in January 1994, speaking of IBM's



**Fig 1** This is the Common Desktop Environment. It happens to be running on top of LinuxPro, but in theory it could be any UNIX and you wouldn't know the difference. It's not just a pretty face — there are internal subtleties like drag-and-drop which I'll investigate in later columns

introduction of the LaunchPad on OS/2, I said: "And the first of the promised COSE interface features also arrives with version 2.2 — a dashboard control based on Hewlett-Packard's VUE (Visual User Environment), a point-and-click program launcher and screen manager."

Yes, the panel at the bottom of **Fig 1** derives heavily from VUE, but Andrew's complaint ("Why make these statements when, as I am sure you realise, CDE is in fact a decedent of HP's VUE Windowing System, an enhancement to OSF/Motif developed in post-Apollo days and released prior to 1991 for HP-UX 8.0?") seems to suggest that this is the whole story. In fact, all the COSE members

pooled their technologies to create CDE, as you can see from the copyright notice in the picture. As COSE was an initiative rather than an organisation in its own right — or as Sun's CEO, Scott McNealy, famously put it, "COSE is a verb and not a noun" — the product needed to be nurtured through the existing alliance of UNIX manufacturers, the Open Software Foundation (OSF) which has since been rolled into The Open Group. Andrew's loyalty to Hewlett-Packard (it turns out that he's working in its Telecom Systems Division) is commendable, and I'm delighted that he reads and likes the column, but I hope I've managed to deflect his accusation of "inaccuracy".

## Good and bad

Now for that good news/bad news stuff. As soon as I discovered there was a Linux version of CDE, I dropped an email to Mark Bolzern, the Linux guru who runs WGS. The first lot of bad news was that he was very nervous about sending me review software because, as it turns out, it costs him an arm and a leg in royalty payments each time he ships the product. I find it ironic that someone whose core business is distributing royalty-free software should be held to ransom by the so-called Open Software Foundation, which charges him \$100 for the CDE and Motif components in each CDE shipment.

It's a tribute to Mark that he managed to get a package out to me, comprising the six-CD set that makes up LinuxPro, along with the CD and manuals for CDE. This was the good news. The implementation of CDE he uses comes from an outfit called Accelerated X, mainly known for its souped-up commercial version of the X server for UNIX on PCs. Its CDE requires the Accelerated X server, which is what Mark ships with his LinuxPro.

I installed LinuxPro and was all ready to go on to the CDE when the shattering news came through about Apple's acquisition of NeXT. I dropped everything and spent the next week or so on the internet, news gathering and chatting with gurus by email for some articles I was writing. After the dust had settled and it became clear that Gilbert Amelio may turn out to be the best thing that ever happened to Apple and NeXT (although it's a rocky road ahead) I finally caught my breath and went back to the relative tranquillity of LinuxPro and CDE. Or rather, just LinuxPro. Because — the next bit of bad news — I simply couldn't get my system to read the CDE CD.

I'd installed LinuxPro on the old Apricot Xen LS-II. You may remember the trouble I had with SCO OpenServer not being able to recognise the Xen's Sony CDU31a CD-ROM drive because it uses a proprietary non-SCSI, non-IDE interface. There's no problem with Linux though — a driver for the CDU31a is part of the standard distribution. One of the Linux system developers, Mark Evans, has even written a driver for the rather obscure on-board busmastering ethernet port on the Xen.

The Apricot Xen LS-II had read the LinuxPro installation CDs perfectly, and as far as I can remember had never shown any sign of trouble with other CDs before. But

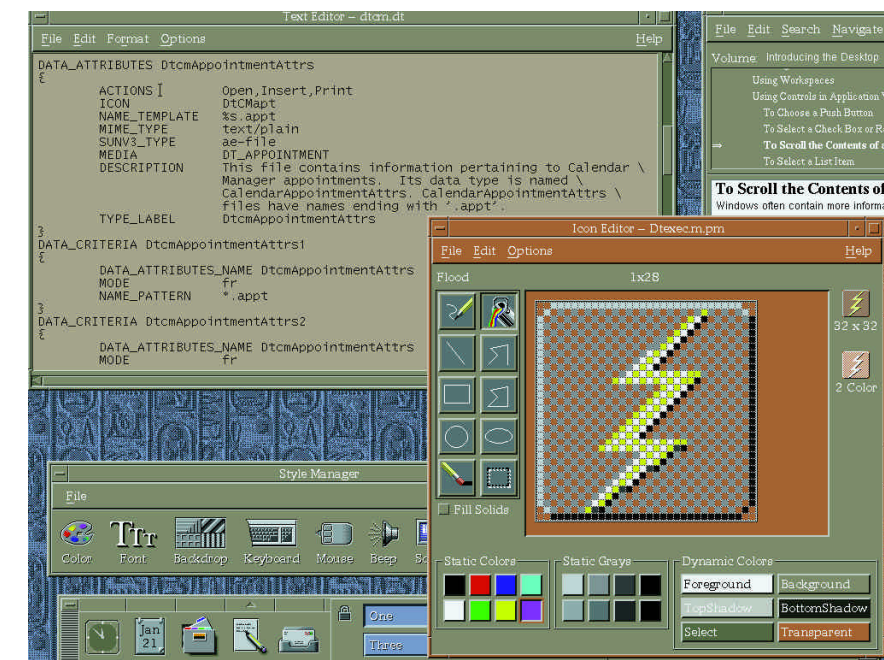
with this particular CD it was spinning its wheels, just at a time when I was desperate to install CDE and get a screenshot to you. Attempting to mount the CDE disk produced the error message (after a long hang) "can't read superblock". Even before I tried mounting the drive, its flickering LED indicated that it was labouring during the spin-up following inserting the CD. Oddly, the NeXT machine and the AIX PowerPC had no difficulty reading the CDE CD. It was just this combination of the Apricot drive and the CDE CD that was coming unstuck. After some experimentation — and I'm not going to tell you how many hours I spent messing around with this — I was forced to the conclusion that some drives, particularly older drives, just don't like some CDs.

This was distinctly bad news, and I found myself wondering how long it would be before I could free up another machine, reinstall LinuxPro on it, and then proceed with CDE. I was about to pack away the Apricot and sit down to write my excuses to you when I suddenly realised that this was the old, narrow, standalone-PC mentality that I'd supposedly escaped from three years ago when I started this column. The AIX machine, which had no trouble reading the CDE CD, is networked to the Apricot. Thanks to NFS, the Network File System that comes with every UNIX under the sun (or Sun, I should perhaps say), it's a trivial matter to hook up the AIX CD-ROM

drive straight into the Apricot and treat it as though it were a local drive. NFS is a client-server connection. Here, the AIX box is the server and needs to expressly "export" the drive in question. How you do this varies somewhat depending on the version of NFS. Under AIX, I used Smit, IBM's System Management Interface Tool, to mount the CDE CD in the drive and then export the drive — or rather, the directory I'd mounted it on — over the network. The directory doesn't actually go anywhere at this stage; it just gets added to a list of directories that the NFS daemon advertises as being available for network connection.

On the Apricot I then needed to mount the exported directory onto a local, empty directory. I created a /mnt/NFS directory and mounted the AIX CD-ROM there. If you read the manual pages for mount (you need to do this with the command "man 8 mount", 8 being the manual chapter that deals with system admin) you may come away reeling from the surfeit of options available. But the Linux mount command is pretty smart these days: if you just ask it to mount aixbox:/mnt/cdrom /mnt/NFS (mount this alien filesystem on this local directory), it should be able to figure out for itself that you're talking about an NFS mount and that it needs to be read-only because you're dealing with a CD-ROM.

Having done this, I had the full contents of my CDE CD available from the Apricot's



The CDE desktop reconfigured with a different background and colour scheme. Here you can see the icon editor together with a sample of the code CDE used to tie icons to executables and ascribe particular behaviours to them

/mnt/NFS drive. I switched to this directory, ran the dtinstall script I found there, and CDE loaded itself on top of my LinuxPro, with the handsome results you see in Fig 1.

### Networking, modules and more

LinuxPro is based on the RedHat distribution, with some refinements added by WGS. Traditionally, Linux installation kernels come loaded with drivers for a huge number of devices and rely on autoprobing software to find out which ones to activate. This can leave you with an overlarge kernel, which is sometimes a tight fit where memory is limited. Once the installation is finished, you're normally recommended to recompile a slimmed-down version of the kernel tailored to your specific machine. For beginners, I should point out that this turns out to be a lot simpler than it sounds.

My Apricot Xen LS-II has 16Mb of RAM, which is plenty of room for Linux. Nevertheless, I prefer to start with the smallest possible kernel and add features as and when I need them. In the early days of Linux you couldn't do this, but newer Linuxes let you load "modules" which can be drivers for physical devices, file systems, or translation code like the IBCS module that lets you run applications written for other versions of UNIX. Mark Evans has modularised the driver for the Apricot Xen's on-board i82596 ethernet controller, so I installed a kernel that had no networking enabled, made sure this was working, and added the networking later. If you haven't yet installed Linux I should point out that most Linux distribution CDs come with a large variety of pre-compiled kernels in the form of disk images. You choose the appropriate image, transfer it to a floppy (where it appears as a bootable kernel and a bunch of vital support files) and use that as the initial boot disk. During the installation process you get the opportunity to install that same boot kernel onto your hard drive.

Typically, the installation process will mollycoddle you through procedures like setting up the network. The upside of this is that you get a working system with the minimum of effort. The downside is that you don't get to understand the fundamentals. Because I was starting without a network driver, the rest of the networking naturally refused to configure itself. Doing this manually was instructive and, luckily, not too arduous.

I found myself telling reader Michael Butler <m.butler@ic.ac.uk> about this at some length, and what follows below is a

**Fig 2 Results of ifconfig**

```
eth0      Link encap: 10Mbps Ethernet  HWaddr 00: 00: 49: 20: 26: 2F
          inet addr: 192.168.1.25  Bcast: 192.168.1.255  Mask: 255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU: 1500  Metric: 1
          RX packets: 1176 errors: 0 dropped: 0 overruns: 0
          TX packets: 555 errors: 0 dropped: 0 overruns: 0
```

**Fig 3 Kernel routing table**

Destination Interface	Gateway	Genmask	Flags	MSS	Window	Use
192.168.1.0 eth0	*	255.255.255.0	U	1436	0	569
loopback	*	255.0.0.0	U	1936	0	4810

**Fig 4 Resorting to ping**

```
PING 192.168.1.3 (192.168.1.3): 56 data bytes
64 bytes from 192.168.1.3: icmp_seq=0 ttl=255 time=1.7 ms
64 bytes from 192.168.1.3: icmp_seq=1 ttl=255 time=1.4 ms
64 bytes from 192.168.1.3: icmp_seq=2 ttl=255 time=1.4 ms
64 bytes from 192.168.1.3: icmp_seq=3 ttl=255 time=1.3 ms
64 bytes from 192.168.1.3: icmp_seq=4 ttl=255 time=1.5 ms
64 bytes from 192.168.1.3: icmp_seq=5 ttl=255 time=1.5 ms
```

condensed version of that correspondence. Michael had written to me about the trouble he was having with his network card, and he'd opened his mailing with: "I have been converted to Linux through reading your column in PCW. It's a great column and worth the cost of the magazine alone!". This is a great way of getting my attention.

Once the install of the minimal kernel is complete, I login as root and then check that there's a set of modules, including the apricot.o module, somewhere among the library files. In my present version of Linux this turns up under /lib/modules/1.2.13/net.

Linux comes with a number of tools for handling modules, and you can get the full list with the command apropos modules. Among these is the insmod (insert module) command. This knows where to find the modules, and knows that modules are .o files, so the command line to add my network driver is just

```
insmod apricot
```

We've now installed the ethernet driver for the device known as eth0, but it's not yet doing anything very useful. You connect it to the network in two stages: first we need to give our eth0 a TCP/IP address, known as the "dotted quad" address. We use the ifconfig networking utility to set this up:

```
ifconfig eth0 192.168.1.25
```

With my own small network setup this

means I've given this particular machine the identifying number 25 on a network whose name is 192.168.1.0. (The full story is much more complicated, but let's not worry about that now.) The next step is to tell eth0 where to find the network. For this we need the route utility. Again there are all sorts of complexities you can get into with route, as you'll see if you consult man route, but for now let's just go with

```
route add 192.168.1.0 eth0
```

And that's basically it. If you now run ifconfig and route without command line parameters you'll get a report on the state of play. ifconfig gives something like Fig 2 (there'll also be an entry for lo, the loopback interface). The output from route should look like Fig 3. To check that the network really is connected I can use the old standby, ping, to send test packets to another machine and have them echoed back at me. For this I need the dotted quad address of the second machine

```
ping 192.168.1.3
```

which should give something like Fig 4, confirming the network is now working fine.

### PCW Contact

Chris Bidmead is a consultant and commentator on advanced technology. Email him at [unix@pcw.vnu.co.uk](mailto:unix@pcw.vnu.co.uk)