Hot **Topic**

The weight of useful, and useless, data available on the web prompts Chris Bidmead to dust off his Topic text retrieval system. And, a web site which compiles your kernel to order.

Last year I mentioned the text retrieval system I use on my network here, a product called Topic, from Verity. It looks after a database of practically everything I've written since the early eighties, together with snippets of useful stuff scanned in or collected electronically from the outside world.

Five years ago, Topic was the centre of my computer writing activity. But the arrival of the web, with its comprehensive search engines, has shifted the balance somewhat. At one stage, I came round to thinking that an in-house text retrieval system risks becoming close to irrelevant under the sheer weight of information available in cyberspace. I neglected Topic, and began keeping my current output in a number of Digital Library files, a vastly more simple text retrieval mechanism built into NeXTStep.

I still had Topic as a method of searching the Bidmead legacy archives, but my access to it was through ageing character-based OS/2 client software that even in its day was somewhat clunky. In comparison with the NeXTStep user interface, it looked like something out of the ark.

But I was wrong about the value of the web. There's a ton of stuff out there and it's a valuable on-going education. But free text searching can waste an awful lot of time when the raw material can be just any old junk put up by anyone who wants to build a web page. Democracy is a fine thing but if you throw open the Opera House to anyone who fancies a sing-song, you can't expect La Traviata.

The web is no substitute for a carefully qualified and managed in-house text retrieval database. Clearly, I had to start cleaning the rust off my Topic implementation and getting it back to work.

The historical roots of Topic lie in UNIX and from this spring a couple of characteristics that I find valuable.

Firstly, unlike all the "personal text retrieval" products I've used in the past, it's died-in-the-wool client-server software. Secondly, it is configured and administered through an initially baffling collection of plain ASCII files riddled with Unixy black magic incantations. "He thinks this is good?", you mutter. Yup. And the reason I do is the reason I write this column.

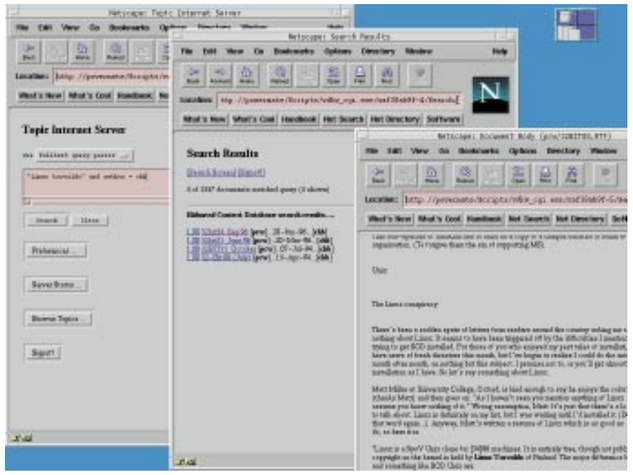

Fig 1 Topic Internet Server is the Verity search engine linked into the web server of your choice. The result is a ubiquitous text retrieval system that can be accessed across the network from a variety of different machines. This is how it looks from Caldera

Topic, as I mentioned here last year, runs under DOS, OS/2, Unix and a number of other operating systems. What I didn't mention is that thanks to its client-server design, Topic can also run under a combination of these environments. For example, you could do the indexing under Unix, run the search engine on Windows NT and have OS/2 handling the client software.

In my time with Topic, I've used combinations of DOS, OS/2, Unix and Windows NT to prepare and serve the data, each part of the work being handled by the operating system best equipped to do the job. The appeal of the arcane text files that control Topic is harder to explain. After three years of neglect, I had to delve back into configuration files which looked like Fig 2. Each text database, or "collection" as Verity calls it, is controlled by a directory tree stuffed with a variety of files like this. I won't go into details here, but the general principle is that the stream of your documents coming into the index system is filtered into plain ASCII, examined for particular patterns to pull out fixed fields (I use Title, Source, Date and Author in my standard collections) and then an inverted index is created of all the text in the body of each document.

Many simple text retrieval packages just index which words are in which document but don't bother to log exactly where each word is (the offset into the text). NeXT's own Digital Librarian works like this, which means it won't allow you to do "proximity searches" (find "relational" within ten words of "Codd"), search for whole phrases, or weigh the relevancy of a returned document on the basis of how many times a particular word or phrase occurs. Topic does all this and more — Boolean searches of course ("marsupial OR reptile") — but the name of the product derives from the way it can also search on "topics", complex clusters of words and phrases representing concepts.

The words and topics are related to one another hierarchically in family trees of topics, sub-topics, and sub-sub-topics extended as far as necessary to define the particular family of ideas on which you are trying to home in. Very useful if you regularly need to profile a sea of electronic documents into predetermined subjects in which you're interested.

My chief use for Topic has always focused on the basics, like being able to combine fixed field searches with free text searches ("Linus Torvalds and source = PCW"). Fundamental to any text retrieval system, in my humble opinion, is the ability to search on one or more date fields, a feature that's often seriously neglected.

Currently, the server end of my Topic system is running on my Windows NT box but fundamentally it's still UNIX software at heart. Windows software would use a GUI to launch and configure the indexing and retrieval engine which would certainly be nice and simple, but would inevitably restrict the possibilities. Topic launches from the command line (obviously under Windows NT you can knock up a few icons backed by batch files if you want to make it look pretty) and uses command line parameters and this nest of plain ASCII config files to define exactly what you want to happen. How the text is broken into fields, what to do with those fields, where the main document starts and stops, what kind of filters to apply, how to tune the indexing and so on are all defined by the config files.

I haven't said anything about the client end, which is the bit the user sees. Verity has traditionally offered a choice of client-end packages to cover all the main operating systems and inevitably they've all worked slightly differently and been out of phase in their versions. The solution Verity has come up with is, as far as I'm concerned, near Nirvana and The Future of Computing. Many software and hardware manufacturers are doing it now in various ways. It works, it's simple and it's delightfully cross-platform. I'm talking about web browsers, of course.

There's at least one for every operating environment these days (NeXT has a choice of four or five but that's because browsers were invented on NeXT!). Forget the browser wars as Netscape and Microsoft haggle over advanced features. Keep it simple: stick to basic HTML 2 conventions and the network is your oyster. What this latest implemention of Topic does is offer an extension at the server end that works alongside your regular web server. I'm using Microsoft's Internet Information Server but any server with a common gateway interface (CGI) can do the job.

You create an HTML query form which can be squirted across the network, collect the query through any browser on any operating system and return the result list as a second HTML page. The result list contains skeleton details about each hit, combined with an HREF pointer to the document itself. Like the server, these client pages are all capable of being tailored via ASCII files. The neat thing is that all the gubbins is kept together at the server end. No complicated client software or configs to distribute to each workstation. All each client needs to know is the web address that gets the initial query page started. I've gone on about this at some length, partly because I'm in the heady throes of getting it up and running (extraordinarily painlessly, as it happens), but also because I'm sure we're going to hear more about this "thin client" style of computing in the near future.

Fig 2 — delving into config files

```
$control: 1
descriptor:
  /collection = yes
{
  data-table:  _df
    /num-records=1
    /max-records=1
  {
    # Header information for partition management
    constant:  _DBVERSION      text "vdk11"
    fixwidth:  _DDDSTAMP       4 date
    varwidth:  _DOCIDX         _dv
    fixwidth:  _PARTDESC       32 text

    fixwidth:  _SPARE1         16 text
    fixwidth:  _SPARE2         4 signed-integer
  }
  data-table:  _df
    /offset=64
```

(… and lots more.)

Linux kernel compilation while-u-wait

Here's another twist on the browser: use it to compile your Linux kernel! Probably the most alarming thing that migrants from off-the-peg operating systems like Windows have to face when they install Linux for the first time is the suggestion they recompile the kernel. This is because a typical Linux startup will come stuffed full of drivers for all sorts of peripherals and bus connectors you don't actually need.

What you'll see on the console at boot time is a warning to the effect that memory is tight. You probably don't need to do anything about this straight away but eventually you'll want to slim down the kernel to only those features you need. You might also want to recompile so as to bring your kernel up to a later version. (You can check the kernel version by running "uname -a" from the command line. But if it is earlier than 1.2.13, then it's a little long in the tooth.)

I was impressed (read "terrified" !) when first presented with this kernel compiling challenge: building an entire operating system from source code isn't something you take lightly. But tens of thousands of Linux users have done it, many on a regular basis. It's actually easy because the whole

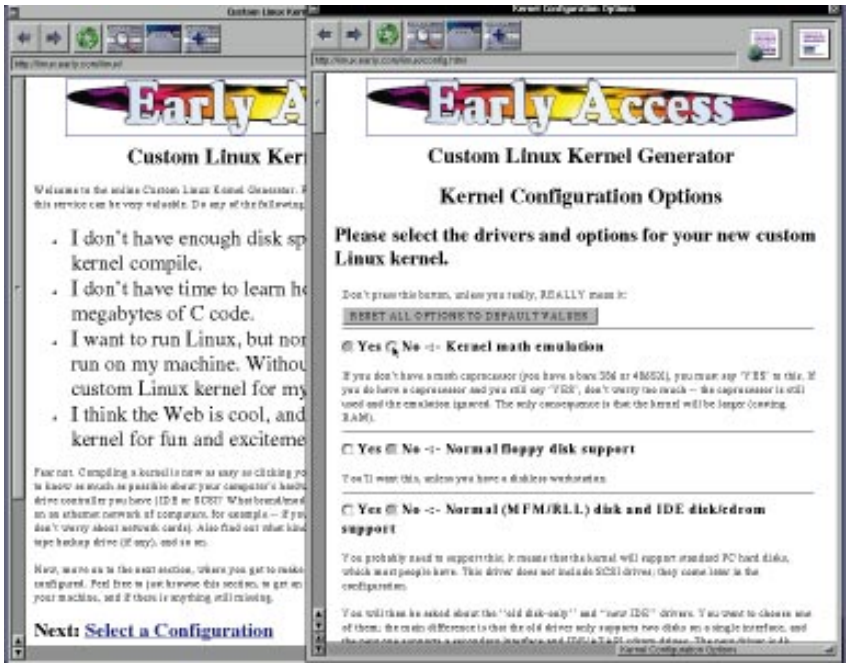

Fig 3 The Early Access Web Site at linux.early.com/linux is another cross-platform web proposition which allows you to build yourself a Linux kernel to your own specification over the internet from any machine capable of running a simple browser, in this case my NeXT machine

process is driven by a configuration file you submit to the UNIX "make" utility. Your system will arrive with the configuration file already written for you, and it will carefully trot you through a Q&A session to find out what kind of kernel you need. Fill in the answers and "make" will proceed to create your new kernel on the spot.

A couple of bright Linux hackers have taken this all a step further with a system that allows you to compile a hand-tailored kernel even though you're not running Linux. Ed Mackey and John Early have devised the "Early Custom Linux Kernel Generator", a web page that supplies you with a collection of radio buttons and tick boxes to collect details of the configuration you want. The web site then compiles your kernel to order and delivers to your system. Look for the Early Access page at linux.early.com/linux.

Readers write…

Sevan Janiyan emailed me from Hove last month with several questions that come up often enough to air them here: *"I'm 16 and very interested in Linux. I installed Linux from your cover CD a while back but I'm having problems running my Pioneer quad-speed CD-ROM drive. Is there any ftp or www site from where I could download the drivers for it? The second problem is using the mcopy and mdir commands. I can view directory listings of floppy disks in MSDOS format but how do I switch to the floppy drive? Is there any way of upgrading Linux by downloading the kernels or something like that?"*

Sevan doesn't say which model of Pioneer drive he has. As far as I know, the SCSI versions of the Pioneer drives are Sony-compatible and should be catered for in the standard kernels. Anyway, the best place to look for details is in the Linux CD-ROM How To which you can pick up from www.caldera.com/LDP/HOWTO/. The LDP, or Linux Documentation Project should be the first port of call for this (or practically any) kind of advice about Linux.

The DOS drives question comes up all the time. DOS and Windows users expect to get to the floppy drive straightaway, but in Linux, as in Unix generally, you need to mount a device before you can access it, although this can be set up as an automatic mount once you know what you're doing. The mounting process can be quite complicated, depending on the device, but mounting is one of the keys to the immense flexibility of the Unix family of OSs.

Beginners will need to learn about devices and, of course, about the mount command. The floppy disk device is generally called fd0 (or fd1 etc., depending on how many fd drives you have). The place to start learning about that is the "man fd" command. Similarly, "man mount" will give you the basics of the mount command, although this is tricky stuff and you'll probably need to delve into those How To's.

Generally speaking, you won't expect to get the very latest versions of Linux on a cover-mounted CD, not because the magazine production people are trying to short-change you but because the process of making CDs and preparing them for distribution takes time. The place to look for the latest kernels on the internet is www.crynwr.com/kchanges/, which is where kernel evolution has traditionally been tracked from. But a less academic approach for beginners is www.gulf.net/~spatula/linux/kernel.html, which will lead you to full information on where the latest kernels are and what you need to do to build them.

There's a pointer to the Easy Access site from there, too. One of the reassurances that makes compiling a new kernel less than totally terrifying is the fact that you can have several different kernels lying about on your system, with a choice of which one you boot into at any one time (via a boot loader like Lilo). Provided you can always get back to a standard kernel, this can make experimentation with new versions fairly painless. My personal tip for Sevan and others is: don't get involved with the very latest experimental kernels unless you want to experiment with them. I settle for older, known kernels that work and support the hardware I use. Then I can get on with the stuff I want to do and don't lose any sleep about the code that's holding it all up.

SCO Open Server opens up

Damn, I'm out of space and I did want to say something about SCO making its Open Server version of UNIX freely available for educational purposes. This isn't quite the Free Software Foundation flavour of freeness that you get with distributions like Linux because Open Server comes with restrictions (you can't use the free version commercially) and it isn't supplied complete with source code.

Even so, it's a really big deal that this pioneering company, the first to put UNIX on Intel chips, has seen the light, or at least glimpsed the dawn. You can find all the details on the SCO web page at www.sco.com and download the software from there, or get it on CD-ROM for a (small) handful of dollars. More about this next month, by which time I hope I'll have got hold of it and installed it.

Windows wisdom

I seem to get an inordinate number of emails asking about Windows problems. Dear people, this isn't fair. I come here to get away from Windows. It's all summed up by a sig I came across recently in the comp.sys.be Usenet conference:
Customer: "I'm running Windows 95."
Tech Support: "*Yes…*"
Customer: "My computer isn't working now."
Tech Support: "*Yes, you said that.*"

•PCW Contacts

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