TOC

The Practical Manager's Guide to Linux

Can you profitably use Linux in your organisation?

An osOpinion.com column

Authored by: "Ganesh C. Prasad"

A downloadable version of this entire document is available "**here.**"

Synopsis:

This document on Linux is unique in that it speaks the language of business, from the viewpoint of a corporate user. It addresses the issues managers want to talk about, -- cost savings, ease of use, support, uptime, productivity, vendor independence, staffing and training, -- backed up by detailed references. You may or may not decide to use Linux after reading it, but you will certainly come away with a better understanding of the options that Linux now gives you.

Table of Contents

"Why should I bother about Linux?"

A Brief History of Linux

1. The GNU Project and the Free Software Foundation
2. The importance of the GNU General Public License
3. Enter Linus Torvalds

A Hard Look at Linux's Claimed Strengths

1. Zero price tag
2. Do-it-yourself flexibility
3. Freedom from licensing headaches
4. Stability
5. Performance
6. Standards-compliance
7. Diverse hardware support
8. Native Internet support
9. Interoperability with existing systems

TOC

10. Inherent Y2K compliance
11. "Virus-proof" design
12. Strong cryptography worldwide

Linux's Weaknesses Revisited

1. User-unfriendliness
2. Installation problems
3. Scarcity of applications
4. Poor documentation
5. Lack of high-end features
6. Security concerns
7. Lack of support
8. Absence of legal recourse
9. Lack of ownership
10. Unviable business model
11. Uncertain roadmap

Linux Tomorrow

1. Variations in hardware
2. Synergy with Java, XML and CORBA

Skepticism about Linux

1. "Linux will splinter just like Unix"
2. "There is no quality control in free software"
3. "We can't trust production data to a freeware program!"
4. "There's no such thing as a free lunch"
5. "Open Source products are good copycats, but poor innovators"
6. "Linux is not yet ready for the Enterprise"

"What does Linux imply for me professionally?"

"Where can I use Linux today?"

About the author

"Why should I bother about Linux?"

This document is aimed at IT managers, -- decision makers who need to make technology choices to solve business problems. From the viewpoint of this group, the arrival of Linux is A Good Thing, because, if nothing else, it represents another choice to add to the options available. In the worst case, if Linux doesn't meet your requirements, you can simply walk away from it. It is an option, not an obligation.

Linux has seemingly appeared out of nowhere and captured 17% of the server market, -- a growth rate of 212%

TOC

over the past year. And this was *before* commercial vendors got onto it in a big way. IDC now estimates that the next 4 years, Linux will grow faster than *all other operating systems combined*, including Windows. Its growth trajectory could soon push it in the direction of your organisation (if it's not there already), and you could find yourself having to make some decisions quickly.

New technologies are risky, and adopting one before it becomes completely mainstream could prove either career-enhancing or career-limiting, depending on how it plays out in the market and how you deploy it within your organisation. There is both risk and opportunity in Linux. As an IT manager, you need to analyse Linux carefully, understand where it is strong, where it is weak, where it is likely to go, and then evaluate where you could profitably use it, if at all.

This document is an attempt to collate all the relevant facts and market information about Linux, without hype or prejudice, into one easy-to-read guide, as it were, so that most common questions about this operating system are answered satisfactorily. It should easily be worth an hour of your time.

References:
Linux 1998-99 growth and server marketshare: http://www.news.com/News/Item/0,4,30027,00.html?st.ne.fd.gif.a
IDC estimate of Linux growth (1999-2003): http://www.it.fairfax.com.au/breaking/922943886.html

A Brief History of Linux

From a purely technical standpoint, Linux is just another variant of Unix. What makes it unique is something other than its technology. To really understand the reasons for its amazing popularity, it may be worth delving into a bit of history.

The GNU Project and the Free Software Foundation

The GNU project was started in 1984 by Richard Stallman, a researcher at MIT's Artificial Intelligence labs, in reaction to the (then) new practice of keeping source code secret and enforcing software licensing. Stallman saw the withdrawal of source code as a curtailment of programmers' freedom to modify and improve software. He also saw the license restrictions on copying as being at odds with his philosophy of being a good neighbour and sharing ideas. So he set out to rewrite all the software then commonly in use, single-handed if need be, and make it free for everyone to use, modify and redistribute, without any restrictions. (A task as enormous as this would have put off a lesser man, but Stallman's determination, self-confidence and technical skill are now legendary.) His goal was to recreate a complete operating environment that was free of such restrictions, with all the tools and utilities that a computer user would ever need.

The model he chose was Unix, because it was technically better than the other operating environments of the day. But because he was against the restrictive licensing of Unix by AT&T, he called his project by a recursive acronym, GNU, for "GNU's Not Unix".

(Free software programmers often display a wacky sense of humour. For example, the free equivalent of the Unix Bourne shell is called *bash*, for "**B**ourne **a**gain **sh**ell".)

Richard Stallman proved to be a formidable hacker. (He uses the word "hacker" in the positive sense of master programmer, reserving the word "cracker" for people who break into systems.) He single-handedly wrote free versions of many popular Unix utilities. Among his lasting software contributions are the GNU C compiler *gcc* and the *emacs* text editor.

Richard Stallman established the Free Software Foundation to raise funds to produce free software. For him, the

TOC

"free" in free software refers to freedom, not price. He is not against software being sold for money as long a
source code is available and other programmers have full rights to modify and redistribute the software. As h
fond of saying, "When you say Free, think free speech, not free beer."

References:
The GNU project: http://www.gnu.org

The importance of the GNU General Public License

Richard Stallman is a great hacker who wrote some really amazing software, but the contribution for which he will probably be remembered is not a piece of software but a legal document. He quickly realised that even if he wrote great software and gave it away, someone else could come along, make a few changes to the code and then copyright the whole lot by claiming it to be a differentiated product. Thus, the aim of sharing would be defeated and he would be foolishly giving away something which others could simply exploit.

He came to the conclusion that he had to design a special license to ensure that the software remained public and all modifications and improvements, no matter who made them, were made available to everyone. Ironically, as the legal system has no mechanism to protect publicly-owned intellectual work, Stallman had to rely on copyright law itself to design a license that was opposed to it in spirit! The way it works is very interesting, demonstrating that even Law can be a malleable medium to a creative mind. To protect his software for everyone, he first copyrights it, thereby preventing someone else from seizing control of it at a later date, then gives it away under controlled conditions that are essentially protected by Contract Law. The conditions are that anyone modifying the code for later redistribution has to make their source code public on the same terms. No proprietary modifications are allowed, except for private use. This license is known as the "GNU General Public License" or GPL. It's also called *copyleft*, because in a deep sense, it is the opposite of a copyright. It gives freedom instead of restricting it. (Stallman has often been accused of being a socialist or communist, an anti-commercial crusader, but the reality is probably simpler than that. He is an idealist who just believes very strongly in the "right" of programmers to share code without artificial restrictions. A naive philosophy, according to some, but one that is nonetheless shaking up the software industry.)

Many people think that free software, public domain software and shareware are the same thing, but this is not so. Shareware is commercial software. Authors of shareware programs expect to be paid, just like authors of any commercial software, but they are willing to allow free distribution of their software to popularise it. Upgrades and bug-fixes are available to those who pay for the copies they receive. The source code is typically not available. Shareware is more a marketing technique than a form of software freedom. Public domain software, while free, is not under copyright at all, which means that someone making modifications to it can claim copyright to the modified version and "take it out of circulation". GPL-ed software, on the other hand, is copyrighted by the original author and licensed to the public, albeit under very generous terms. It ensures that the software remains perpetually free. GPL could be thought of as **G**uaranteed **P**ublic for **L**ife. There are other free licenses as well, the most famous of which is the BSD license, which has sometimes been called "copy-neutral", because it enforces no restrictions at all on copying and redistribution, not even the GPL's condition that changes should be made available to the public.

The GPL and other free software licenses must seem very quaint concepts to people from the commercial world of copyrights, patents and non-disclosure agreements, but increasing numbers of high-quality software products are given away every year under such licenses and are being used by increasing numbers of computer users, forming a credible threat to established vendors of commercial software, so they cannot be pooh-poohed as mere idealistic nonsense. You need to understand how they work, even if you don't agree with their philosophy.

References:

The GNU General Public License: http://www.gnu.org/copyleft/gpl.txt
The BSD License: http://www.dislessici.org/opensource/bsd-license.html
The Artistic License: http://www.weblint.org/artistic.html
"Linux may be running on some spindly legal legs" -- an analysis:
http://www.businessweek.com/cgi-bin/bwdaily_full?right=/bwdaily/dnflash/apr1999/nf90427b.htm

Enter Linus Torvalds

Richard Stallman wrote an amazing amount of software, even suffering painful carpal tunnel syndrome for a few years from so much typing. But after all his efforts, he did not succeed in creating a complete, working system. The core, or "kernel", of the system did not yet exist; only peripheral utilities were available. These are now famous as the "GNU Utilities", and a Windows version is even distributed with Microsoft's NT Resource Kit. (Bound by the terms of the GPL, Microsoft includes the source code as well!)

In 1991, a Finnish Computer Science student named Linus Torvalds wrote the first version of a Unix-like kernel for his own use, and posted the code on the Internet with a request to other programmers to help him build it up into a working system. The response was overwhelming, and what began as a student's pet project rapidly developed into a non-trivial operating system kernel. Looking around, Torvalds was pleasantly surprised to find that virtually everything else he needed was already there in the form of the GNU Utilities and other free software. He put them all together and named the complete operating system after himself, -- Linux, for Linus' Unix. (He pronounces it LINN-ucks, not LYE-nucks.)

Richard Stallman is miffed that the contribution of GNU software to Linux is not often acknowledged. According to him, the operating system should properly be called GNU/Linux, but most people, even those sympathetic to Stallman, find it quite a mouthful, so the *de facto* name for the OS remains Linux. However, Stallman cannot complain about another decision by Torvalds -- to release the source code for the Linux kernel under the GNU General Public License.

Today, the entire Linux system, kernel and utilities, are freely available with source code for anyone to use, modify and redistribute. Judging from the frenetic activity on the developer websites, thousands of qualified programmers from around the world are accepting the GPL's invitation to modify and improve the system in all the ways they think fit.

References:
Linus Torvalds' first post (?) re. Linux: http://www.debian.org/Lists-Archives/debian-user-9804/msg02622.html
How Linus Torvalds pronounces "Linux": http://www.ssc.com/lj/images/english.au

A Hard Look at Linux's Claimed Strengths

You may have already heard the most commonly cited strengths of Linux. Let us examine each of them carefully and try to separate hype from reality.

Zero price tag

Linux is often touted as being "free", though in practice, no organisation will install software without a support agreement in place. Given that support for Linux by reputed vendors as well as the retailer next door is mushrooming, it is very likely that Linux will be used by many organisations with this kind of third-party support arrangement. So hype notwithstanding, Linux will never be a zero-cost solution.

However, when an organisation looks at *licensing* costs, especially over multiple users and multiple computers, it may find to its pleasant surprise that Linux does deliver a significant cost advantage after all.

For example, Computer Currents magazine estimated that a fully configured Windows NT server with webserver, e-mail, development tools and database would cost more than $4500 ($4636) to set up, while an equivalent Linux setup would cost only $50 for a Red Hat CD with all this software bundled (hardware costs being the same). What's more, the Windows license fees need to be multiplied by the number of such installations, whereas the Linux solution incurs only a one-time cost, -- the CD's purchase price -- since the software can be freely installed on an unlimited number of machines.

Now factor in support costs. In a recent announcement, Hewlett-Packard has offered unlimited, 24/7 worldwide phone and e-mail support for Linux for a fee of $130 per month per server, or $1560 a year per server. This seems a reasonable ballpark figure for support from other third-party vendors as well. So the cost argument in favour of Linux appears to be justified, and of the order of $2500 per server in the year of purchase, even assuming zero support costs for Windows NT. Microsoft and its partners offer a variety of support options with different pricing terms. So the Linux cost advantage should be even greater when NT support costs are factored in.

Another hidden cost advantage of Linux is its ability to run on older machines with less memory and disk capacity, which translates into savings on hardware upgrades. Each subsequent release of Windows, on the other hand, seems to require upgrades to hardware as well. Faster chips constantly appear, but are saddled with increasingly bulky software, neutralising their speed advances. Hence the saying, "Andy giveth and Bill taketh away", -- a reference to Intel's former chairman Andy Grove and Microsoft's Bill Gates. Linux provides excellent performance on new hardware while still running adequately on older machines.

Software bloat is another hidden cost on traditional Windows platforms. The GNU C/C++ optimising compiler on Linux occupies 10 MB of disk space, and the associated editing, debugging and project management tools together account for less than 2 MB. Microsoft Visual C++ 6.0 Professional Edition, on the other hand, requires 290 MB of disk space. Even the fact that Visual C++ is a visual tool with an integrated editor and debugger doesn't seem to justify such a large difference in size.

In the past, it has been pointed out that "Linux is free only if your time is worthless", a valid reference to the difficulty (for a relative novice) of finding and editing various configuration files, which was the only way to administrate Linux. However, new administration and configuration tools, such as Red Hat's *linuxconf* and Caldera's *lizard* (for Linux Wizard), provide centralised, graphical administration, largely eliminating the need to edit configuration files by hand. As such tools improve, Linux system administration effort should reduce to acceptable levels.

Last year, the Mexican government embarked on an ambitious project to equip 140,000 schools with computers. They found the license costs of Microsoft Windows so high, even with volume discounts, that they opted to use Linux instead, saving an estimated $124 million. Over a large installed base, the cost advantage of Linux becomes compelling.

When Digital Domain rendered the visual effects for the movie "Titanic", they needed a large server farm to handle the processing load. Ultimately, they settled on 105 servers based on the Compaq/Digital Alpha chip. The operating system they chose was Linux. Though estimated savings are not publicly available, Digital Domain's website says that cost was the primary reason for the choice of Linux. (Of course, the system also performed adequately, otherwise the savings would have been meaningless.)

Your mileage might vary, and it's best to do a rough calculation before making a decision.

References:

The Computer Currents article: http://www.currents.net/magazine/national/1524/inet1524.html
HP's 24/7 worldwide support package for Linux: http://www.news.com/News/Item/0,4,35392,00.html
The Mexican school computerisation project (ScholarNet): http://www.wired.com/news/news/technology/story/16107.html
Linux and the Titanic: http://www.linuxjournal.com/issue46/2494.html

Do-it-yourself flexibility

A commonly-heard Linux "advantage" is that users can easily modify the software to suit their requirements. There are two aspects to this. One is that unlike with most commercial software, which is distributed only in binary form, the Linux source code is readily available, making it *physically* possible to modify and recompile it. The other is that the GNU General Public License expressly permits anyone to modify and redistribute the software, making this *legally* possible as well.

So, should you make changes to Linux code just because you can? Unless your needs are very specialised and you know exactly what you're doing, don't. Among other things, you risk making your version incompatible with future Linux upgrades.

So is this open source feature really an advantage?

This is a subtle point to note: The availability of Linux source code is important to users because it makes it easier to modify, *but it is not necessary that they do it themselves*. It's like when you're looking to buy a car, and choosing brand A over brand B because of the easier availability of spares. It's not that you intend to replace parts yourself. It just gives you greater confidence that you can easily get it done.

There was a report last year that provided anecdotal evidence of such a benefit. Microsoft decided against developing an Icelandic version of Windows 95 because the limited size of the Icelandic market couldn't justify the cost. When approached by volunteers from Iceland who offered to do the port, Microsoft refused, on the grounds that the Windows source code was secret. There is no similar "dog in the manger" problem with Linux, because there are no cost considerations and the software requires no permission to modify. Unsurprisingly, an Icelandic version of Linux's 'K' Desktop Environment exists. With Linux, minority users with special needs are not at the mercy of any vendor.

References:
Microsoft vs. Iceland: http://kyle.seattletimes.com/news/technology/html98/alticel_063098.html
Icelandic KDE: http://www.mbl.is/frettir-ifx/?MIval=forsida&frontcatform=7&nid=400041&tp=1

Freedom from licensing headaches

Using commercially licensed software carries with it the responsibility of ensuring that you stay compliant with the license at all times. Exceeding the licensed number of installations is a crime. In many countries, the CEO of a company that is found to be in breach of software license contracts is personally accountable and, in theory, can even go to jail for it. This means that organisations must keep track of the number of purchased licenses and the actual number of installations of every piece of software that they use, -- an administrative overhead. Organisations that purchase large numbers of licenses of many different software products find that they need the help of special "license management software", a product which seems like a solution to an artificial problem.

Sometimes (and this may be familiar to many), production systems fail to scale under unexpectedly heavy loads because a piece of software enforces limits on the number of concurrent connections or transactions, based on the

number of purchased licenses. Some products, for example, BEA's Tuxedo middleware, permit utilisation sli
above the licensed limits, but it must be particularly irritating when services are affected, not for technical, b
legal/commercial reasons that can be easily regularised without the need for such drastic enforcement.

Linux and other free software products do away with such considerations altogether. You can install the software on any number of machines without breaking the law. In effect, Linux gives you an "unlimited-user, unlimited-installation" license. That is an undisputable plus for those who are currently held accountable even for inadvertent license violations. Note, however, that commercial products that run on top of Linux may still be subject to license restrictions.

Linux's free license also means that you don't need to worry about staying within a budget, or about exposure to adverse changes in vendors' licensing terms. The elimination of Microsoft's concurrent licensing for Office and BackOffice was an unpleasant surprise to many organisations, who saw their licensing costs go up sharply. The Microsoft antitrust trial also brought to light internal company e-mails in which executives planned a move away from the current practice of one-time licenses to annual (recurring) licenses, and this was planned to start as early as 2001. Linux and other free software can be a godsend to managers who are exposed to such harsh and volatile regimes.

References:
The Business Software Alliance on penalties for licensing violations: http://www.bsa.org/uk/penalties
Fines paid by companies that breached software license agreements: http://www.elronsw.com/metering.html
Microsoft eliminates concurrent licensing:
http://www.idg.net/idg_frames/english/content.cgi?vc=docid_9-69142.html
Microsoft mulls annual Windows fee: http://www.news.com/News/Item/0,4,29088,00.html?st.ne.fd.mdh
Forbes reports on Microsoft's license fee hikes: http://www.forbes.com/forbes/98/0907/6205050a.htm

Stability

"It never crashes!"

It is sometimes remarked that the reason why Linux rarely crashes is that it isn't required to do as much as other operating systems. If Linux is loaded to comparative levels, the argument goes, it will crash as often. Observation leads us to conclude, however, that Linux shares this stability trait with other Unix strains and larger proprietary systems like VMS and IBM mainframes. The only computer systems that are known to crash are PCs and Macintoshes. The reasons may not be far to seek. Stability is largely an architectural issue, because implementation bugs can be squeezed out over time. A hardware architecture that has never grown too far from its 1981 design roots, early design compromises, and the continuing requirement for backwards-compatibility with poorly-written applications hobble even the latest versions of Windows. Also, the feature-set of Windows is a moving target, frustrating attempts at eliminating implementation bugs. The other crash-prone system, the Macintosh, is as notorious for its lack of protected memory and pre-emptive multitasking as it is famous for its user-friendly interface. With such crippling constraints, it is a wonder that these operating systems do not crash more often than they do.

Therefore, far from stability being a lucky accident in Linux, it is a commonplace feature that is taken for granted in most mainstream operating systems. It is only Windows and the Mac that are aberrations. The Unix design which Linux shares is a time-tested one. Linux, it must be noted, has the advantage of a quarter century of Unix experience to draw on, and the right lessons appear to have been learnt. Linux's design shows aspects of the most modern operating systems concepts and the most time-tested ones, cherry-picked with a luxury only a newcomer can afford. Most significantly, the open-source code model of Linux seems to ensure that bugs are detected and fixed early.

As even IBM says on its website, Linux is stable, functional and offers value.

Loading Linux with applications may make it slower, but is hardly likely to make it crash. That is a patently spurious argument.

References:
IBM's endorsement of Linux's quality and stability: http://www.software.ibm.com/data/db2/linux
Diagnostics page for Mac freezes and crashes: http://www2.northstar.k12.ak.us/help/mactips.html
Windows 95 and 98 may crash every 49.7 days -- news item: http://news.com/News/Item/0,4,33117,00.html
Windows NT's "blue screen of death" -- reasons:
http://www.webshopper.com/jhtml/templates/display_content.jhtml?id=129634

Performance

Many benchmarks have been conducted by independent organisations, pitting Linux against Windows NT and against the various flavours of Unix. Apart from one study that is discussed later in this section, it appears that Linux quite consistently beats NT not only on single-processor machines, but also on multi-processor machines on which NT is expected to scale better on account of its multi-threaded architecture. Linux seems able to deliver good performance even with "heavyweight" processes instead of the "lightweight" threads that NT uses. Process forking in Linux is particularly efficient, almost obviating the need for threads. Suprisingly, Linux also narrowly beats Solaris on its home ground, SPARC hardware, albeit only single-processor SPARC machines. Again, a little analysis reveals a major reason. Unix systems, Linux included, treat graphics as a user-level application that can be optionally run. Reasonably sophisticated graphical interfaces exist for Linux, but these are not tightly integrated with the operating system kernel. They can be "switched off" when not required. On servers, graphics capability is an unnecessary overhead most of the time. Typically, on Unix servers, Linux included, the graphical interface is invoked only when the system is being administrated, and turned off at all other times, delivering a significant performance boost to the system, because graphics is a very resource-intensive capability.

The Windows design, by contrast, suffers from an overly tight integration of the graphics subsystem with the kernel (unsurprising in view of its desktop roots), and this design model will forever handicap it in its role as a server operating system. Perhaps Linux will lose its slim performance advantage against Solaris when it acquires more high-end features and grows in size. Time will tell. However, the constant fine-tuning of the kernel by a worldwide group of expert systems programmers could continue to give Linux a performance lead even in future. It will be an interesting battle to watch.

In April this year, a benchmark conducted by a company called Mindcraft reported that Windows NT with Microsoft's IIS (Internet Information Server) performed 2.5 to 3.7 times faster than Linux as a Windows fileserver (running Samba) and as a webserver (running Apache). This is somewhat surprising, because it is contrary to what several independent testers have found in the past, including Sm@rt Reseller On-line.

Subsequent information on the benchmark found that the NT machine used was highly tuned and optimised for the tested load, while the Linux machine was not, a fact later admitted by Microsoft. It was also discovered that the benchmark was sponsored by Microsoft, and was not as "independent" as it was made out to be. (Microsoft still touts these figures, though). These facts therefore put the results under a cloud. For now, rather than debate the legitimacy of the Mindcraft benchmark, it is best that you design and conduct one yourselves, tailoring it to your situation. After all, not everyone runs servers that are as high-end as that used in the benchmark.

Re. client-side performance, even the famous leaked Microsoft memo, the second of the "Halloween documents", admitted after internal testing that on the same hardware, Netscape Navigator on Linux was 30-40% faster than Internet Explorer on Windows NT.

TOC

"Benchmarketing", of course, ranks with lies, damned lies and statistics, but from a variety of sources, the impression one gets is that Linux is one of the leanest and fastest operating systems available. With the new kernel (version 2.2), it has reportedly even drawn level with the ultrafast FreeBSD. Nevertheless, the Linux vendors need to submit formal SPEC or TPC benchmark figures, otherwise this sort of controversy could keep arising. Incidentally, the TPC benchmarks have a price/performance parameter, and Linux should do very well on that!

References:
Smart Reseller Online's webserver test in which Linux/Apache beats NT/IIS:
http://www.zdnet.com/sr/stories/news/0,4538,2196115,00.html
Smart Reseller Online's fileserver test in which Linux/Samba beats NT:
http://www.zdnet.com/sr/stories/news/0,4538,2196106,00.html
Microsoft touts the Mindcraft benchmark: http://www.microsoft.com/windows/dailynews/042199.htm
Microsoft spokesman admits that Linux machine in the Mindcraft benchmark was not well-tuned:
http://www.itweb.co.za/sections/enterprise/1999/9904221410.asp
Linux Weekly News finds flaws in the Mindcraft benchmark: http://lwn.net/1999/features/MindCraft1.0.phtml
A commentary on Mindcraft's planned second series of benchmarks: http://linuxtoday.com/stories/5424.html
The Halloween documents:
Halloween I: http://www.opensource.org/halloween1.html
Halloween II: http://www.opensource.org/halloween2.html

Standards-compliance

By definition, open source Linux cannot have proprietary features. Under the terms of the GNU General Public License, it is illegal for any entity to make modifications to Linux without making the corresponding source code publicly available. At one stroke, this takes away the incentive to "hijack" the system and produce a proprietary variant. The license therefore ensures that the only changes to the system that will last are those that are accepted by the "community". The community has no vested interest in creating proprietary standards and protocols, and so the OS naturally coalesces around industry standards. This is not mere theory. Linux today is a POSIX-compliant OS and its constituent subsystems support all relevant ANSI, ISO, IETF and W3C standards. However, certification is a different issue, and the Linux community is against having to pay standards bodies for something that doesn't really benefit them. Therefore, Linux is currently in the state of being compliant with some standards without actually being certified.

Ironically, while Linux does a good job of supporting industry standards, there is still a lack of standardisation between different Linux distributions. True, the differences are minor, since all distributions have free access to the entire Linux codebase. They just differ on what applications they bundle with Linux, the versions of those packages, the installation utilities and the locations they use for various system files. A project called the Linux Standard Base project has sprung up that aims to unify all distributions in a few respects, such as standard directory locations for system files. It is in the interests of users for this effort to succeed, since it will make for a more predictable and uniform user experience.

Other than that, organisations looking for a fully industry standards-compliant operating system do not have to look beyond Linux, provided it meets their other acceptance criteria.

References:
POSIX and Unix 98: http://lwn.net/lwn/980611/standardseditorial.html
Linux Standard Base home page: http://www.linuxbase.org

Interesting viewpoints on Linux and standards compliance: http://lwn.net/lwn/980618/ianresp1.html, http://lwn.net/lwn/980618/Editorial.html

Diverse hardware support

This is a mixed scorecard for Linux. On the one hand, it runs on virtually every known processor, whether RISC or CISC, 32-bit or 64-bit. The most common processor for Linux is of course, the Intel x86 family, but it also runs on Motorola's 68k, the IBM/Apple/Motorola PowerPC, Compaq/Digital's Alpha, MIPS chips, Sun's SPARC and UltraSparc and Intel's StrongARM. HP's PA-RISC chip is perhaps the only major one on which Linux does not yet run, but HP is assisting an independent group, the Puffin Group, to port Linux to PA-RISC. Intel is also supporting Linux, and it is Intel's stated objective to make Linux run fastest on its chips. Intel is providing technical information about its 8-processor motherboards to the Linux community, so that high-end Xeon servers running Linux can be a cost-effective alternative to customers. Intel is also sharing advance information about its forthcoming 64-bit Merced chip, which is expected to be the way the industry will go in a few years. When Merced arrives, Linux will be ready to run on it.

On the less glamorous side, computers running Intel x86-compatible AMD or Cyrix chips are among the most inexpensive hardware available, and zero-license fee Linux can make such machines very attractive to the low-end of the market as well as to bulk purchasers. Some vendors have woken up to the ease of customisation of this general-purpose OS and its royalty-free convenience, and have started putting it on such unlikely hardware as TV set-top boxes and MP3 music players.

This breadth of chip support is a tremendous achievement that no other operating system can boast of, though the free BSD variants come close. Linux has in fact fulfilled the hardware-independence promise of Unix, which was belied when Unix split into incompatible proprietary versions. Users of Linux gain an extra degree of independence from hardware vendors.

On the other hand, Linux does not support USB (Universal Serial Bus) or PnP (Plug-and-Play) devices, though there are active efforts underway on both these projects. Intel is pushing its UDI (Uniform Driver Interface) as a common Unix approach to device drivers, and is trying to get the Linux community to help write the drivers.

Linux also does not support as many peripherals and cards as Windows does. It is still necessary to consult a hardware compatibility list before choosing a new piece of hardware to add to a Linux machine. Doubtless, this issue will decrease in importance with time, as the OS's increasing popularity encourages hardware manufacturers to release drivers (or at least specifications) for it.

For the next year or so, these are likely to remain chinks in Linux's armour. Pre-installed Linux systems with peripherals tested and guaranteed by the manufacturer are the best short-term answer.

References:

HP, the Linux PA-RISC port and the Puffin Group: http://www.hp.com/pressrel/mar99/01mar99e.htm

Sub-$600 PCs from Emachines: http://www.news.com/News/Item/0,4,35322,00.html?tag=st.cn.sr1.dir.

Cyrix reaches beyond the $299 PC: http://www.news.com/News/Item/0,4,34825,00.html?tag=st.cn.sr1.dir.

Linux on a set-top box: http://www.thestandard.net/articles/display/0,1449,4246,00.html?home.bf

Linux on 8-way Intel Pentium III Xeon SMP servers: http://www.newsalert.com/bin/story?StoryId=CnWWPWbKbytaXmtC&FQ=Linux&SymHdl

Linux and Intel's Merced: http://www.crn.com/dailies/weekending030599/mar02dig09.asp

Cygnus GNUPro toolkit for Merced enables Linux to be compiled for that platform: http://linuxtoday.com/stories/5434.html

Linux-USB project mirror: http://www.nv.org/linux/USB/snapshots.html

TOC

Linux Plug-and-Play project home page: http://www-jcr.lmh.ox.ac.uk/~pnp
Intel's UDI and the Linux community: http://www.zdnet.co.uk/news/1998/37/ns-5501.html
Linux runs an MP3 music player for cars: http://www.wired.com/news/news/technology/story/18236.html
Linux hardware compatibility list: http://metalab.unc.edu/LDP/HOWTO/Hardware-HOWTO.html

Native Internet support

Linux was born of the Internet, and its Unix pedigree virtually guarantees that it will support all the standard Internet protocols. (Indeed, Linux was perhaps the first OS to support IP version 6.) Linux is a very popular server OS among ISPs (Internet Service Providers) on account of its low cost, reliability, and an abundance of Internet-related software. E-mail, file transfer and network news are available out of the box with any Linux distribution.

The world's most popular webserver, the Open Source Apache, runs naturally on Unix, and is most commonly used in combination with either Linux or FreeBSD. Lots of add-on modules for Apache make it a very powerful solution for web applications. The *mod_perl* module allows Perl CGI scripts to be interpreted and run within Apache's memory space, rather than starting up the Perl interpreter each time in a separate process. The *mod_jserv* module allows Apache to use Java servlets. The Java-Apache project builds on this and proposes some very innovative ideas. The *mod_php* module allows Apache to run HTML-embedded scripts in a Perl-like language called PHP (Hypertext Pre-Processor), a program that works exactly analogously to Microsoft's Active Server Pages. PHP has drivers for every major database and for the OpenLDAP directory server, and is a very powerful tool for web-based applications.

Most importantly, there is a module called *mod_ssl*, which, when used in combination with a cryptography package called *SSLeay*, gives Apache 128-bit strength SSL (Secure Sockets Layer) capability, which US-developed commercial webservers are not allowed to export. This can give a website strong cryptographic capabilities for e-commerce, at zero cost.

Many hardware vendors are now selling "thin servers", which are basically small-sized machines running on cheaper RISC chips rather than on Intel's Pentium IIs and IIIs. Obviously, the best OS for non-Intel chips is Linux, because Linux runs on almost anything. Apache is of course the obvious webserver choice. Configuration of such machines is very easy, and is usually done using a browser. Thin servers are very cost-effective for small Intranets.

The move now is towards Application Servers, which use webservers as a front-end but do the bulk of the actual processing themselves. IBM's Websphere application server uses Apache as its front-end.

Lutris Technologies has donated its Java/XML-based Enhydra Application Server as Open Source to the web development community. This is likely to prove a very popular product.

The latest Linux kernel (version 2.2) supports firewall functionality through a tool called *ipchains*. There are web-based configuration tools for *ipchains*, freeing a user from having to edit a configuration file by hand. The Squid caching proxy is also a very popular one among ISPs. Mail products on Linux include the venerable *sendmail* and recent competitors *qmail* and *smail*. Mailman is a good mailing list manager written in the powerful Python language. Some vendors have ported Linux onto PCs and are selling them as routers. They provide a much cheaper alternative to Cisco routers. Virtually all databases are available for Linux. Directory servers running the Open Source OpenLDAP software are also here, and support directory replication as well.

A recent announcement for FreeS/WAN, an Open Source encryption software, makes it possible to build secure VPNs (Virtual Private Networks).

TOC

On the client side, the Mozilla project is slowly wending its way towards a release of the industry's first 100 standards-compliant browser. Mozilla is the project created by Netscape to manage the development of Communicator after it went Open Source. The greatest achievement of the Mozilla project so far has been the complete rewrite of the browser's Layout Engine, or Rendering Engine. This is called Gecko and fits into a single floppy. Gecko is also available as an ActiveX control. This gives users the flexibility to choose the rendering engine they want in a browser. It is possible to embed the Gecko ActiveX control as a plug-in to Microsoft's Internet Explorer, achieving a completely standards-compliant Microsoft browser with no help from Microsoft! The first company to profit from Gecko, ironically, is not Netscape. NeoPlanet has embedded Gecko in its latest browser/portal, NeoPlanet 3.0. With Gecko being an embeddable plug-in, the current practice amongst web developers to target Netscape and Microsoft browsers whenever they develop web-based applications can finally end. They can work with the guarantee that their applications will be rendered by a single, standards-compliant rendering engine. Mozilla's networking libraries are also being rewritten and are collectively called Necko. Mozilla is an excellent example of code reuse within the Open Source community. It uses the Hungry Programmers' *Japhar* JVM (Java Virtual Machine) to run Java applets, and incorporates James Clark's *expat* XML parser. Mozilla promises to be the smallest, fastest and most standards-compliant browser. That should be a refreshing change from IE5's 100 MB disk requirement. Mozilla on Linux clients should give desktop users a fast and "correct" browser.

Linux is an excellent, standard platform for web applications. You can use it to build a complete, secure Internet site, including router, firewall, proxy, webserver, mailserver, database server and directory server.

References:

Apache webserver home page: http://www.apache.org
Enhydra application server home page: http://www.enhydra.com
Firewall configuration tool: http://rlz.ne.mediaone.net/linux/firewall
Firewall-building tool "Mason" licensed under GNU GPL: http://users.dhp.com/~whisper/mason
Squid home page: http://squid.nlanr.net/Squid
Sendmail home page: http://www.sendmail.org
FreeS/WAN news item: http://www.infoworld.com/cgi-bin/displayStory.pl?990421.icfreeswan.htm
FreeS/WAN home page: http://www.xs4all.nl/~freeswan
Mailman mailing list manager: http://www.list.org
Python home page: http://www.python.org
Linux-based routers: http://www.zdnet.com/sr/stories/column/0,4712,381687,00.html
Network Concierge, a Linux-based thin server: http://www.nc4u.com/linux.htm
Cobalt Networks, Inc. Linux-based thin servers: http://www.cobaltmicro.com
Corel Corp.'s Netwinder thin server running Linux: http://www.corelcomputer.com
IBM's "WebSphere" E-commerce software runs on the Apache webserver:
http://www.software.ibm.com/webservers/appserv/awb.html
The Mozilla home page: http://www.mozilla.org
NeoPlanet embeds Gecko layout engine in 1.3 MB browser/portal:
http://www.neoplanet.com/press_OpenSource.html
Gecko as an embeddable ActiveX plug-in for Internet Explorer: http://www.iol.ie/~locka/mozilla/mozilla.htm
The Java-Apache project: http://java.apache.org
James (Java-Apache Mail Enterprise Server) MailServlet page: http://java.apache.org/james/index.html
ServletCentral website for server-side Java: http://www.servletcentral.com
PHP (ASP-style HTML-embedded scripting language): http://www.php.net
OpenLDAP project home page: http://www.openldap.org

The Hungry Programmers' Japhar JVM: http://www.japhar.org
James Clark's free XML tools: http://www.jclark.com/xml

TOC

Interoperability with existing systems

Linux is widely claimed to be able to coexist with other operating systems and even talk some proprietary protocols. Linux can talk SPX/IPX in a Netware environment, Appletalk in a Mac crowd, and even SNA to IBM mainframes. But for most organisations, the most important and relevant aspect of Linux's claimed interoperability is its ability to coexist with Windows machines. Considering that Windows can talk the Unix-native TCP/IP protocol, the ability of Linux to communicate with Windows is no great feat. However, there are two areas where Linux aims to emulate Windows so well that it threatens to replace it altogether.

The first way is by providing Windows file and print services through "Samba", a product which, like Linux, is released under the GNU General Public License. By all reports, a Linux server running Samba emulates a Windows NT server so well that it completely fools Windows clients. Windows workstation users can use their favourite Explorer file manager to manipulate files on the Linux server, even using drag-and-drop! Microsoft had the Samba developers foxed for a while with their NT Domain Security encryption system, but the resourceful bunch soon worked around that. At the time of writing, Samba can let Linux do anything an NT server can do except emulate a Backup Domain Controller, and that capability is reportedly not far away.

Given Linux's ability to scale upto multi-processor, 64-bit machines, Samba gives users a serious way to obtain "NT-like" file servers at levels of power and stability that NT itself cannot currently reach, and to operate a complete Windows network without using any NT servers at all. (As a side-benefit, this also eliminates the need for Windows client-access licenses.) Samba is such a popular piece of software that SGI (formerly Silicon Graphics) is bundling it with its high-end Unix servers.

The second way Linux is trying to emulate Windows is proving to be a lot harder to pull off. The Windows Emulation Project (WINE for short) aims for nothing less than the ability to run Win32 binaries on Linux. This will obviate the need for porting Windows applications to Linux, since they will run "out of the box". Corel is supporting the WINE project, for this reason. They can avoid porting their Windows software to Linux, and can simply have it run with no additional effort. If WINE succeeds, Linux can one day replace Windows workstations just as it can replace Windows servers today with Samba. Some Windows applications, notably games like Solitaire, have been available on Linux for over a year. However, the really complex applications are the ones to watch. It will be considerably more difficult to run Microsoft Word, and such applications are reportedly only 90% compatible at the time of writing. These are precisely the kind of applications where anything less than 100% compatibility is not acceptable, so the task is not easy, considering also that WINE must emulate Windows bug for bug to display identical behaviour.

Linux and Windows can easily coexist on the same network because both speak TCP/IP. So compatibility fears should not stand in the way of Linux adoption. Linux with Samba can also provide tangible value in terms of license savings, better performance and improved stability. WINE is a project to be watched, though at present, nothing usable has emerged.

References:
Linux and Appletalk: http://www.idg.net/idg_frames/english/content.cgi?vc=docid_9-129189.html
Linux and IPX: http://pinkfloyd.student.okstate.edu/~arc/howto/IPX-HOWTO-1.html
Linux and SNA: http://linuxworld.com/linuxworld/expo/lw-wednesday-server.html, http://www.linux-sna.org
Samba home page: http://au1.samba.org/samba/samba.html
Samba on SGI: http://www.zdnet.com/products/stories/reviews/0,4161,394079,00.html

Samba 2.0 release announcement: http://linuxtoday.com/stories/2298.html
High-end Unix servers from SGI come bundled with Samba:
http://www.sgi.com/newsroom/press_releases/1998/december/samba.html
Samba - A license to kill Windows NT?: http://www.zdnet.com/sr/stories/issue/0,4537,396321,00.html
WINE project home page: http://www.winehq.com
Corel's support to the WINE project: http://www.zdnet.com/zdnn/stories/news/0,4586,380599,00.html

Inherent Y2K compliance

The year 2000 is an issue of great significance to IT, with many organisations having formal policies against installing software products that are not certified Y2K-compliant. Systems that are not Y2K-compliant can fail because they store dates in a format that only records the last two digits of the year. This leads to the year 2000 being confused with 1900, resulting in all kinds of errors in applications.

Linux and other Unix variants claim to be *inherently* Y2K-compliant because they record dates very differently. Dates in Unix are internally stored as the number of seconds elapsed since 1st January, 1970. The 32-bit Unix variants store such dates in 32 bits, and this storage will only overflow in the year 2038, not in 2000. 64-bit Unix versions store dates in 64 bits, and this is sufficient for *billions* of years! Most 32-bit Unix versions will probably upgrade to 64 bits well before 2038, comfortably averting their doomsday.

Thanks to the free availability of Linux source code, the operating system has been independently verified by many to be Y2K-compliant. Any organisation can have the OS audited to its satisfaction before accepting it. However, it must be remembered that this compliance does not automatically extend to the applications that run on it, or to the BIOS of the hardware on which it runs. These need to be separately certified. Open Source applications that run on Linux (e.g. GNU software) can obviously be independently audited for Y2K compliance. For other (commercial) applications, users are forced to rely on the vendors' own guarantees.

(To show how easy it is to upgrade the Unix date format from a 32-bit to a 64-bit field, we need to dip a little into techno-speak. The size of the date field is defined in only one place in any Unix system. This is a variable called *time_t*. Moving from a 32-bit chip to a 64-bit chip involves, among other unrelated things, merely changing the definition of *time_t* in this one place from 32 bits to 64 bits. When the operating system is then recompiled, it will use the new definition and store all dates in 64 bits from then on. Of course, moving from a 32-bit chip to a 64-bit chip is not trivial because the very instruction set changes, but this is all that needs to be done about the date.)

References:
Linux and Y2K: http://www.zdnet.com/enterprise/zdy2k/stories/0,6158,2218229,00.html
Y2K compliance status of GNU software: http://www.gnu.org/software/year2000-list.html

"Virus-proof" design

Boot-sector and file viruses have historically been known only in the PC world. Higher-end systems like Unix have two clearly demarcated privilege levels -- call them "user" and "system". A normal user, or a program owned by a normal user, has no privilege to delete system files or files belonging to other users, because such actions require "system" privileges. The administrator of a Unix system, or "super-user", is the only one with system privileges. Therefore, normal users of Unix have limited ability to cause damage to their systems by importing suspect files from elsewhere. That is why we never hear of Unix viruses. (We sometimes hear of Unix "worms", programs that choke systems by replicating themselves endlessly and filling up storage, even if they have no privilege to actually delete or corrupt files. Worms are not as destructive as viruses, and can also be blocked with a little diligence.)

All said, Linux, like Unix, can be considered relatively "virus-proof" compared to the "lightweight" operatin systems -- MS-DOS, Windows 3.1, Windows 95, Windows 98 and the Macintosh. The deadly Chernobyl v rus that irrevocably damaged hundreds of thousands of Windows 95/98 PCs around the world on April 26th lef Linux machines unaffected. (The Mac is affected by a different, but no less deadly, set of viruses. Examples of Mac viruses are INIT-29 and Autostart 9805.)

Windows NT, like Unix, has separate "user" and "system" privilege levels, so NT is in theory as virus-proof as any version of Unix. However, Windows *applications*, even on NT, are vulnerable to a new kind of virus, the "macro virus", that spreads through e-mail attachments and infects Word and Excel documents. "Melissa" is one such macro virus that was recently in the news.

Computer users whose experience is limited to Windows PCs and Macintoshes could be excused for thinking that viruses are an inescapable part of life with computers. It should be pleasant news to them that there exist operating systems that are inherently resistant to viruses because of a better security design, -- Unix, Linux, and to a lesser degree, Windows NT.

Though Linux at present enjoys a virus-proof reputation, it has largely escaped the attention of virus writers because of its limited market presence compared to the ubiquity of Windows computers. As Linux gets more popular, viruses targeting it will certainly appear. A Linux virus could result in careless users losing their own files, even if system files and the files of other users are unaffected. Personal computer owners should be particularly careful not to log into their systems as the super-user for anything but system administration tasks. Inadvertently downloading viruses while logged in as the super-user can result in wholesale damage just like on a Windows PC.

Moreover, the addition of macro-like programming features to the free Gnumeric spreadsheet program is cause for concern. The developers claim that Gnumeric allows only trusted code to execute in a spreadsheet, and that a Melissa-type attack will not be possible. Unless backed up by a public-key infrastructure that authenticates external code, it is not easy to see how imported spreadsheets with useful macros can be viewed without danger.

So using Linux today certainly puts you in a more comfortable position with respect to virus protection, but there is no place for complacency. Eternal vigilance is the price of freedom.

References:
Report on the Chernobyl virus: http://news.bbc.co.uk/hi/english/sci/tech/newsid_329000/329688.stm
Report on the Melissa virus: http://news.bbc.co.uk/hi/english/sci/tech/newsid_307000/307162.stm
Screenshots of gnumeric: http://www.gnome.org/gnumeric/screenshots

Strong cryptography worldwide

If your company is not based in the US or Canada, and if it's not a bank or financial institution, you cannot get a US-developed hardware or software product with strong cryptographic capabilities for love or money. Even if you are a North American company, you cannot implement a truly secure transnational network (a "Virtual Private Network") linking your various international branches and business partners together. The problem is not technical but legal/political. Most popular commercial software (such as operating systems, webservers and communication software) comes from US companies and is subject to US laws that forbid the export of encryption software that exceeds 56-bit strength. (The policy is aimed at preventing international terrorists from being able to communicate through a channel so secure that even US law enforcement agencies cannot tap it.) But when you know of the Electronic Frontier Foundation's demonstration that cracked a secret message (encrypted using the 56-bit DES algorithm) in just 56 hours using a computer costing just $210,000 to build, you can be excused for being unenthusiastic about the security of products that US companies are allowed to sell you. For

TOC

this reason, the Indian government has warned against the use of US security software, and may even ban its altogether.

For a few years now, companies like Germany's Brokat have made a comfortable living selling independently developed strong cryptographic products to customers outside the US and Canada. A large number of German and other European banks have used Brokat's proprietary solutions to legally acquire strong crypto (before the US Commerce Department made an exception for banks and financial institutions, depressing Brokat's business somewhat!). However, these products are not cheap. The artificial scarcity of what are, after all, just (well-known) mathematical algorithms has raised the price of legally available cryptographic software from other sources. But it is not as if you can get a cryptographic "patch" from a source outside North America and apply it to commonly-used US commercial software. US law forbids even cryptographic "hooks" in US-developed software that would allow such patches to be applied. You would need to buy entire solutions from non-US vendors, and this is likely to be both costly and not as elegant as the original standard software would have been with strong crypto. The Wassenaar agreement between 33 advanced countries to restrict the export of products with strong cryptography makes this task even harder.

If you, for example, want to set up a website for secure electronic commerce, you would need to encrypt the communication channel between your customers and your site, so that no prying eyes can tell what business is being transacted as the messages go up and down over the Internet. The standard technology to do this is called SSL (Secure Sockets Layer), pioneered by Netscape. If you are outside North America and are not a bank or financial institution, commercial webservers like Netscape Enterprise Server and Microsoft's Internet Information Server are only allowed to give you 40-bit SSL, which as you know, can be broken fairly easily through a brute-force attack. You cannot get a standard and simple solution that employs "strong" cryptography, at a reasonable price.

The solution to this seemingly intractable problem is delicious in its irony. You cannot *buy* any such product, but you can get one *free*. You can download the market-leading, free Apache webserver in source code form from anywhere, including the US, then legally download the popular and free "SSLeay" SSL package (independently developed by an Australian, Eric A. Young) and the Apache "mod_ssl" package from any site *outside* the US. A small utility included with the mod_ssl package overwrites parts of the Apache source code itself, in effect *placing* cryptographic hooks in the code to call the SSLeay routines. You then compile the source code yourself to build the secure webserver, a process that is simpler than it sounds. Note that the original Apache source does not contain any cryptographic code or hooks at all. So this solution is clean, legally bypasses US export laws, and gives you a fully standards-compliant webserver with 128-bit strong SSL, which you can use to build an e-commerce solution.

What's more, you can have the SSLeay source code independently audited to satisfy yourself that the cryptographic algorithms have been faithfully implemented with no hidden trapdoors or dilution in strength, something you cannot do with any commercial (binary-only) software. Interestingly, Netscape always uses 128 bits for SSL even in 40-bit strong products. They merely send the remaining 88 bits in cleartext. Without the ability to check the source code and compile it afresh, there is no way to ensure that such surreptitious dilution of strength is not taking place in the "strong" cryptographic products that are available commercially.

This entire issue has only indirect bearing on Linux. However, the Apache-Linux combination is a very common and standard one. Apache runs on all versions of Unix as well as on Windows NT, though at present the NT version is not as stable. So if you fall outside the boundaries the US government has laid down for the legal use of strong cryptography, Apache-on-Unix may be the only way to go to implement (verifiably) secure e-commerce, with Apache-Linux being a very natural solution.

Other strong cryptography products that are free include PGP (Pretty Good Privacy), -- a public-key cryptography package, its GNU equivalent GPG (GNU Privacy Guard) which uses no patented algorithms, and the Cryptix

Java classes for free use in Java applications. Also recently announced is the comprehensive FreeS/WAN str cryptography package to provide secure communications over TCP/IP networks like the Internet. Unlike SSL which can only encrypt the Hypertext Transfer Protocol (HTTP) traffic between browsers and webservers, FreeS/WAN can encrypt any TCP/IP traffic, including FTP (File Transfer Protocol), which has traditionally been insecure on account of its sending passwords over the network in cleartext form. FreeS/WAN was entirely developed outside the US for obvious reasons, and is the cheapest legal way to implement a Virtual Private Network or Extranet. All these free products work with Linux, and the source code can be audited to your satisfaction.

The best things in life *are* free, at least as far as strong cryptography is concerned...

References:

The Electronic Frontier Foundation website: http://eff.org/

Indian government warning on US security software: http://www.economictimes.com/120199/lead2.htm

Brokat's website: http://www.brokat.de

Apache home page: http://www.apache.org

SSLeay home page: http://www.psy.uq.oz.au/~ftp/Crypto

SSLeay download page: ftp://ftp.pca.dfn.de/pub/tools/net/ssleay

mod_ssl download page: http://www.engelschall.com/sw/mod_ssl/distrib

(Naming convention: "mod_ssl-2.2.8-1.3.6.tar.gz" means this is the patch for SSLeay version 2.2.8 and Apache version 1.3.6)

Netscape's 40-bit SSL hack: http://www.cs.bris.ac.uk/~bradley/publish/SSLP/chapter4.html#4.4.3

PGP download page: http://meryl.csd.uu.se/~d95mno/PGP.html

GPG home page: http://www.d.shuttle.de/isil/gnupg/gnupg.html

Cryptix home page: http://www.cryptix.org

FreeS/WAN news item: http://www.infoworld.com/cgi-bin/displayStory.pl?990421.icfreeswan.htm

FreeS/WAN home page: http://www.xs4all.nl/~freeswan

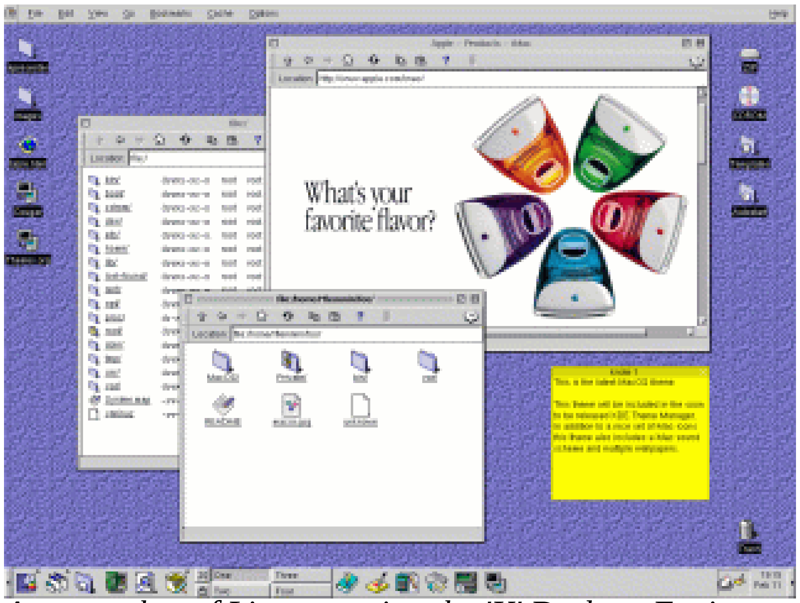
Linux's weaknesses revisited

Linux is certainly not a perfect OS, and for quite some time, we have also been hearing about its weaknesses. Let us re-examine them to see how many of them are still valid today:

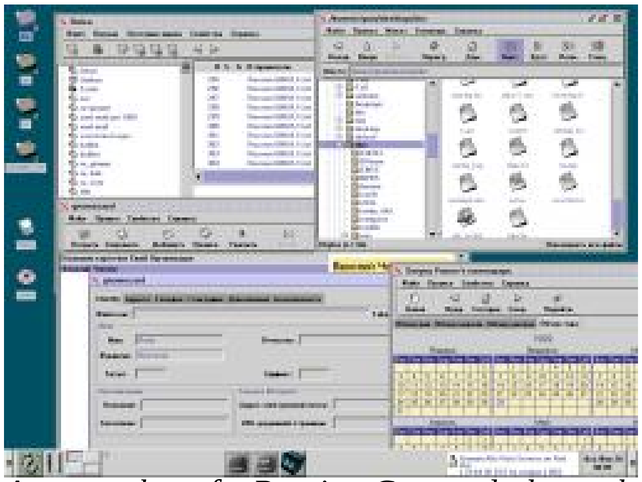
User-unfriendliness

It is impossible to read an article on Linux written for a non-Unix audience without coming across the phrases "primitive command-line", "cryptic commands" and "arcane syntax". Undoubtedly, users accustomed to the friendly graphical interfaces of Windows and the Mac would reasonably balk at having to type commands in an "arcane syntax". But things are changing fast. There are two graphical desktop environments for Linux. KDE ('K' Desktop Environment) is the slightly more mature one, and the main criticism heard about it these days is that it's too "clean" and "corporate". Gnome is a more fun interface that allows users to customise it *ad infinitum*. Neither interface is as yet as polished and complete as the Windows or Mac desktops. However, they have developed to their current state in an amazingly short time (See screenshots below). Critics of Linux must remember that the earliest versions of Windows were eminently forgettable, and it was only with version 3.1, appearing sometime in 1993, that Windows became usable. It took more than another 2 years for Windows to achieve its current levels of usability in the form of Windows 95. Given the current status of both Linux desktop projects and their tremendous momentum, it seems reasonable to expect that this argument about Linux not having a friendly graphical interface will wither away by the end of 1999.

(Already, these interfaces have pulled ahead of Windows in some respects. If you delete a program manually, a reference to it still appears on the Windows Start Menu. Gnome and KDE automatically detect the (manual) removal of a program and stop displaying references to it.)



A screenshot of Linux running the 'K' Desktop Environment with a Macintosh "theme"



A screenshot of a Russian Gnome desktop, showing its internationalisation capabilities

A screenshot of Gnome with the Enlightenment window manager, an extreme example of how users could customise their environment.

Besides, to turn the user-friendliness argument around, a command-line is an excellent alternative to a GUI in many situations. Even with an extremely friendly GUI, a user may find certain operations difficult to express with a graphical metaphor, for instance, "piping" the output of one program to the input of another one. The powerful repertoire of commands that can be chained together makes the Unix/Linux command line a very productive environment. Scripting is another very powerful Unix mainstay, an area where predominantly GUI-based systems like Windows and the Mac are notoriously weak. The DOS batch files of Windows are pitifully inadequate compared to the basic Unix shell script, while Applescript comes somewhat closer. Neither can match the breadth and power of Unix's specialised (free) scripting languages like Perl, Python, Tcl and Guile. That may explain why these languages are now being ported to Windows. It remains to be seen whether Windows 2000, with its promised Active Scripting using Visual Basic, brings Windows up to this level.

Ironically, with a graphical environment very similar to Windows or the Mac, Linux's command-line and scripting interface will perhaps soon emerge as a major selling point for *advanced* users. Windows and Macintosh treat all users alike, irrespective of skill level, and do not allow users to acquire more control over their machines with increasing experience. Unix and Unix-like systems, on the other hand, "scale" extremely well with experience, rewarding advanced users with dramatically greater productivity.

So the argument that Linux has no friendly interface is now clearly untrue. The challenge for Linux now is to develop graphical metaphors for its powerful command-line constructs as well. Piping and redirection through drag-and-drop, visual scripting, graphical representation of the powerful Unix "file" abstraction, and other innovations can push the Linux interface beyond the reach of Windows and Macintosh, because they lack the underlying architecture to support such visual representations.

References:

Gnome home page: http://www.gnome.org
KDE home page: http://www.kde.org
Simple Unix Bourne shell script tutorial: http://www.ocean.odu.edu/ug/shell_help.html
Perl home pages: http://www.perl.com/pace/pub, http://www.perl.org
Python home page: http://www.python.org
Tcl/Tk page: http://www.tcltk.com
Guile home page: http://gnu.internexus.net/software/guile/index.html
An interesting Macintosh-user perspective on Linux: http://www.applelinks.com/warpcore/apr99/wc-8.shtml

Installation problems

There have been many articles in the popular press in recent times that recount in painful detail the travails associated with installing Linux, and these are inevitably contested by Linux supporters who claim it is a breeze. All said and done, it does appear somewhat problematic for lay users to install Linux. Even though Linux does a reasonably good job of detecting all the hardware components on a PC, it does require some enlightened input from the installer. Disk partitioning and mounting of filesystems are relatively advanced concepts, especially for users who are used to the simple drive letters of Windows. With many Linux distributions, the user also has to know the details of the graphics adapter card and monitor in order to provide the information the installation program requires. Installing Linux on an existing Windows machine to gain "dual-boot" capability (the ability to run either OS) has its own traps to watch out for.

Having said this, it must be remembered that users themselves normally never install operating systems on their

TOC

computers. Operating systems always arrive pre-installed by the hardware manufacturer, whether the machir PC workstation or server running Windows, a Sparc machine running Solaris, an RS/6000 running AIX, or something else. It may well be that the toughest OS to install is (say) HP/UX, but the user doesn't know and shouldn't care.

With the availability of Linux computers off-the-shelf from manufacturers like VA Linux Systems, Compaq and Dell, OS installation is now a non-issue. However, those who want to "try out" Linux by installing it themselves on an old box might benefit by enlisting the help of an experienced person.

Linux installation is getting progressively easier, however, and Caldera OpenLinux 2.2 has been found to have one of the easiest installs. Early reports of Red Hat Linux 6.0 say that it, too, has an easy installation procedure. With such trends, Linux's installation problems could soon become a thing of the past.

References:
Fortune magazine's The Dreyfuss Report, on the difficulties of installing Linux:
http://cgi.pathfinder.com/fortune/technology/dreyfuss/1999/04/26/index.html
Caldera OpenLinux 2.2 features: http://www.comspec.com/linux/cld/open22.html
Review of Caldera OpenLinux: http://www.linuxworld.com/linuxworld/lw-1999-04/lw-04-penguin3.html
Dell to sell PCs with Linux: http://www.news.com/News/Item/0,4,34036,00.html?tag=st.cn.sr1.dir.
Compaq expects Linux to popularise 64-bit Alpha chips:
http://www.news.com/News/Item/0,4,34119,00.html?tag=st.cn.sr1.dir.

Scarcity of applications

Nobody chooses operating systems for their own sake. It's the applications that run on them that matter, and for some time now, critics of Linux have justifiably asked, "Where are the applications?"

To try and answer this, we need to look at two different environments, -- the server side and the client side.

The server side

The point about scarcity of applications was very true around this time last year. The most glaring weakness in the Linux story was the complete absence of strong database products. The only database products available were free ones like mySQL and PostgreSQL. (These have their advantages, of course. The mySQL DBMS is one of the smallest and fastest of its kind, but lacks features like transactions with rollback, which are essential for all but the simplest applications. PostgreSQL is an object-relational DBMS (ORDBMS) whose design underlay the commercial Illustra ORDBMS, which in turn was later incorporated into Informix. It allows for inheritance of tables, storing of programming logic within database columns and is, in general, a very good multimedia database. However, it is not as high-performance or as widely tested as the commercial databases such as Oracle, Sybase, Informix or DB2.)

But then, late last year, all the database vendors jumped on board the Linux bandwagon. By July 1999, all the major databases, with the unsurprising exception of Microsoft SQLServer, are expected to be available on Linux, with varying support and fee options.

After the database vendors, one by one, the large hardware vendors (Sun, HP, SGI, IBM, Compaq/Digital and Dell) revealed their plans to sell hardware pre-installed with Linux. The pace has since only increased.

Internet and web applications have never been a problem for Linux. Sendmail, a free mail transport program that is estimated to handle 70-80% of the world's e-mail, runs on Linux. Linux also has a natural fit with the free webserver Apache, which according to the Netcraft survey, has 56% of the Internet webserver market, with

Microsoft's IIS (Internet Information Server) at 23%, and Netscape's Enterprise Server a poor third at 7%. Th
Squid caching proxy is very popular with Internet ISPs. Java is available for Linux, though it is still at versio
1.1.7. The Java 2 port has been completed and is undergoing compatibility testing. Lutris technologies has donated its Enhydra Java Application Server as Open Source.

On the server side, there are very few application categories that are not available for Linux. One holdout so far was ERP, prompting many to dismiss Linux as not yet ready for the "Enterprise". However, SAP soon announced a Linux port, and that bastion fell as well. On the middleware front, BEA is planning a port of its Tuxedo TP monitor client to Linux, and is considering the server component as well. Computer Associates will port its Unicenter TNG systems management environment to Linux.

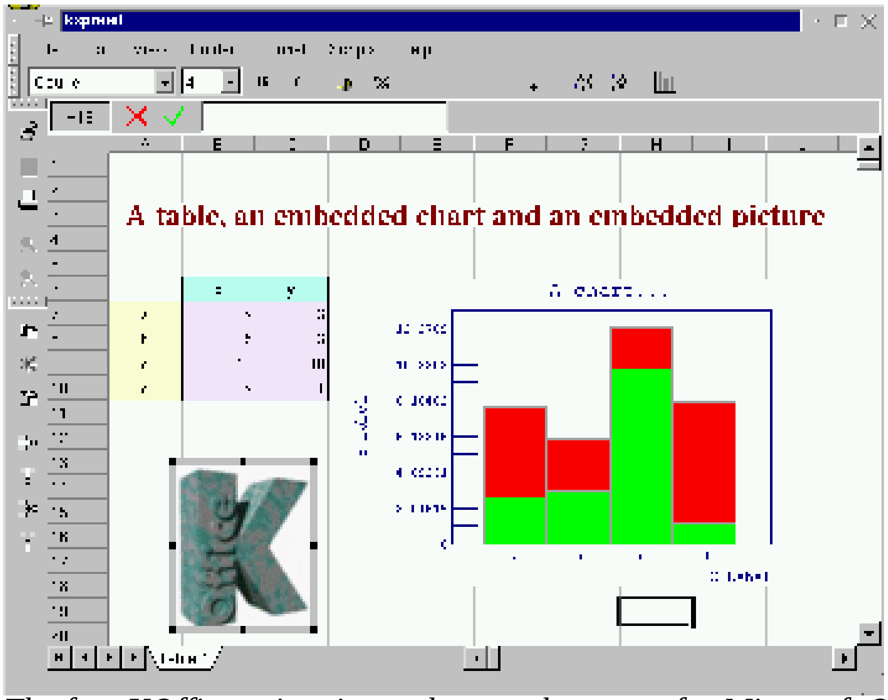
From the experience of the database vendors, it seems fairly simple to port a Unix version of a large and complex software product to Linux. Linux is a variant of Unix, after all. Therefore, as the Linux market grows, there is no reason why products available on other Unix platforms should not be available on Linux as well. From the viewpoint of applications availability, Linux servers are in a good position.

References:
mySQL website: http://www.mysql.com/
PostgreSQL website: http://www.postgresql.org
Database vendors Oracle, Sybase, Informix, IBM support Linux:
http://www.news.com/News/Item/0,4,30360,00.html
Hardware vendors HP, SGI, Compaq, Dell and IBM support Linux:
http://www.news.com/News/Item/0,4,31511,00.html
Linux on IBM's RS/6000 and Netfinity servers:
http://www.businessweek.com/bwdaily/dnflash/feb1999/nf90222a.htm
Linux on Sun's 64-bit UltraSparc servers: http://www.internetwk.com/news1298/news120898-6.htm
Big guns support Linux: http://www.zdnet.com/pcweek/stories/news/0,4153,1014026,00.html
Compaq expects Linux to popularise 64-bit Alpha chips:
http://www.news.com/News/Item/0,4,34119,00.html?tag=st.cn.sr1.dir.
Sendmail home page: http://www.sendmail.org
Apache home page: http://www.apache.org
Netcraft's webserver marketshare survey: http://www.netcraft.com/Survey
Squid home page: http://squid.nlanr.net/Squid
Java-Linux project home page: http://www.blackdown.org
Enhydra application server home page: http://www.enhydra.com
SAP jumps on Linux bandwagon: http://www.news.com/News/Item/0,4,33063,00.html
Tuxedo on Linux: http://www.fi.infn.it/DFS/news-comp.dce/msg00463.html
Computer Associates' Unicenter TNG for Linux:
http://webserv.vnunet.com/www_user/plsql/pkg_vnu_nn.homepage?p_story=77351

The client side

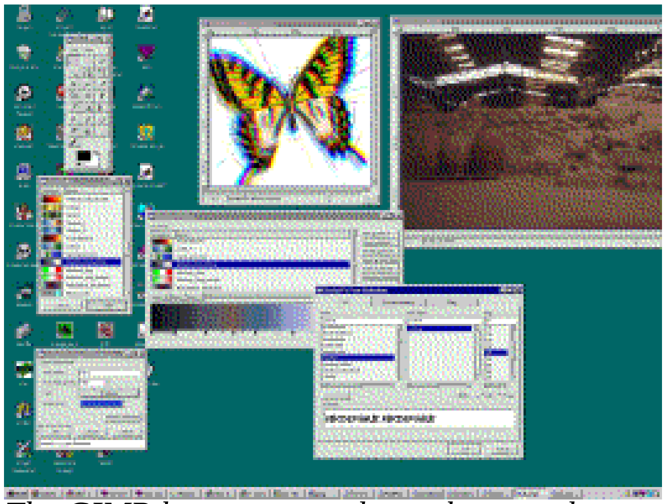
The desktop is a slightly different proposition, for a reason we will look at very soon. Desktop productivity applications like office suites are here, of course -- WordPerfect, Applixware and StarOffice. The Gnome and KDE desktops with their bundled free applications will probably make Linux very popular in the price-sensitive market.

TOC



The free KOffice suite aims to be a replacement for Microsoft Office and comes bundled with the 'K' Desktop Environment. Shown above is the "KSpread" spreadsheet package, part of the suite.

Perhaps the best showcase example of a free commercial-quality Linux desktop application is the GIMP (GNU Image Manipulation Program), a worthy competitor to Adobe's Photoshop. GIMP packs about 80% of Photoshop's features in 0% of the price, and even has a scriptable interface that Photoshop lacks. A recently-published book, "The Artists' Guide to the GIMP" is a hot-selling companion guide. Organisations designing web page graphics may find GIMP and this book a far cheaper alternative to Photoshop.



The GIMP has proven to be such a popular package on Linux that it is now being ported to Windows (above). The scripting tool Script-Fu is also shown.

In today's climate, a vendor whose roadmap does not include a Linux version looks increasingly out of touch with the market. Two speech-recognition software packages were announced very recently for Linux, -- IBM's ViaVoice and Nuance Communications' Foundation SpeechObjects, the latter as Open Source. However, the one application on everyone's mind when considering a move to Linux is Microsoft Office. More than Windows itself, this is the "killer app" that gives Microsoft its corporate desktop monopoly. If moving to Linux entails losing compatibility with MS-Office, most organisations would choose to remain with Windows. This compatibility is of two kinds. Most corporate desktop users have been trained on the interface and features of the Microsoft products, and there is some understandable reluctance about moving to new products and having to relearn those skills. The other form of compatibility is in file formats. Document formats are perhaps the last stronghold of proprietary technology. Every other category has strong industry-standard alternatives. But ind standard XML is relatively new and currently in no position to dethrone the Microsoft Word and Excel file formats. And indeed, this is a serious issue. Microsoft is known to change its Office file formats with every new release, partly to support new features and partly to shake off competitors.

The Wordperfect Office Suite and StarOffice have very similar interfaces to Microsoft Office, minimising the training effort involved in switching to them. They also claim complete compatibility with MS-Office file formats. Even if this is true, it must be remembered that the next version of MS-Office will again leave them scrambling to catch up.

Microsoft Outlook and Exchange Server are also very popular integrated e-mail products that dramatically improve productivity. They are excellent examples of value-added proprietary extensions to standard protocols, but they carry with them the reality of vendor lock-in. There is no painless way to switch.

In a larger sense, the Microsoft Office lock-in with its attendent forced-upgrade nuisance has occurred in part because of the failure of the user community to demand compliance with a set of industry standards. Indeed, there have been no strong industry standards until recently. However, with the plateauing of the office suite feature set required by an organisation, it is now possible, more than ever before, to call a freeze on features. How many more bells and whistles do you want?

Standardising on the file format of a particular version of MS-Office may be beyond the ability of any single organisation, however, because the policy runs the risk of being unable to handle documents produced by other organisations using later versions. It is an interesting issue that perhaps reflects more on the collective lack of bargaining power of the user community than on any drawback in Linux. Some organisations try to use HTML, RTF (Rich Text Format) and PDF (PostScript Distribution Format) to gain a certain amount of vendor-independence, but the Office products are clearly in a higher league.

The WINE (Windows Emulation) project discussed earlier aims to solve the problem by letting Win32 applications like MS-Office run on Linux without "porting". However, WINE has not yet reached a sufficient level of success at the time of writing to recommend it as a solution.

So, on the desktop, Linux has a long way to go. The applications are there, but the most important one is not. And it will be difficult to break that lock. Users need an industry-standard, vendor-neutral document format to break it. XML is the most credible threat to Microsoft's proprietary formats, and Linux is a good vehicle for XML. As a user organisation, you need to track developments in this space, and throw your weight behind such standardisation efforts, because only that can generate the real competition that will make this a buyer's market and get you off the "upgrade treadmill".

Unviable business model

Besides, software vendors already face an unviable business model, as pointed out in a recent article, "How Microsoft took the 'Win' out of Windows". Independent software developers need to target Microsoft's Windows, because it is the most popular desktop platform, but they soon face competition from Microsoft's own products. Microsoft often "bundles" these products for competitive advantage, and there is a common perception that Microsoft applications are better "integrated" with Windows. This perception could well be justified. Software developers rely on the published Win32 APIs (Application Programming Interfaces) to get their products to run on Windows. But there are known to be many *undocumented* APIs that utilise Windows more efficiently. Microsoft's in-house application developers are believed to use these APIs to gain competitive advantage over external developers. This is a hot topic in the antitrust trial now underway.

As a user, such a situation may not concern you immediately as long as you get the products you want from

Microsoft, but in the long term, monopolistic conditions lead to lower quality and higher prices.

For software vendors, Linux, far from being an unviable business model, offers a second chance to compete by providing a more level playing field than Windows. For example, Corel Computer (now Rebel.com) is making a comeback with WordPerfect on Linux. So Linux is not anti-business. It could just be anti-monopoly.

References:

The KOffice home page: http://koffice.kde.org
Corel Wordperfect for Linux: http://linux.corel.com/linux8/index.htm
Applixware home page: http://www.applix.com/appware/linux/index.htm?&Orig=7
StarOffice personal version: http://www.linuxit.com/pages/html/body_star_office_per_ver.htm
The GIMP home page: http://www.gimp.org
Review of the GIMP: http://webreview.com/wr/pub/1999/03/05/studio/index.html
The Artists' guide to the GIMP: http://www.ssc.com/ssc/gimp.html
IBM's ViaVoice speech recognition software for Linux: http://biz.yahoo.com/bw/990426/ny_ibm_1.html
Nuance Communications gives away FoundationSpeechObjects speech recognition software as Open Source: http://www.newsalert.com/bin/story?StoryId=CnYpKWbWbsfnnmdu2&FQ =open-source&SymHdl=1&Nav=na-search-&StoryTitle=open-source
XML information: http://www.xmlinfo.com
WINE project home page: http://www.winehq.com

Poor documentation

The greatest software in the world is pretty much useless without documentation. Documentation includes, at the very least, good installation, user and administration manuals, and on-line help for all software products. Development tools must, in addition, have reference manuals and sample code.

One of the downsides to Linux having started out as a programmer's operating system is that programmers require minimal (but highly technical) documentation to understand and improve upon the work of other programmers. Corporate users, on the other hand, are not interested in the software for its own sake, but as a tool to get their work done. The documentation required by them is of a very different kind.

Documentation as a weakness has long been recognised by the Linux community, and there are strong efforts to bring the Linux documentation in line with that enjoyed by commercial systems. The current problem with Linux is *not* scarcity of documentation, but the *overabundance* of it, which makes it difficult to search. The Linux Documentation Project is a fairly successful effort to provide a centralised access point to Linux documents. In some ways, a web-based document collection is better than a set of physical manuals, because that is the most dynamic and flexible way to document an ever-evolving piece of software. Linux is an Internet-era phenomenon, and the Internet is perhaps the most apt tool to manage its documentation.

Having said that, the need for physical documents cannot be dismissed. Publishers O'Reilly Associates have an impressive selection of books and manuals for Linux and other Open Source software, and new editions are released fairly frequently. These are mainly of use to software developers and system administrators. Books by independent authors are also available for office productivity software such as WordPerfect. Fortunately, Linux *is* Unix for most practical purposes, so most Unix documentation applies to Linux as well, and there is plenty of that around.

Users in the English-speaking world have it relatively easy. Most Linux documentation is in English, although the quality is uneven. The GNU Project is seeking to address this through a mailing list of voluntary proof-readers (proofreaders@gnu.org). They hope to maintain, on an ongoing basis, a collection of high-quality documen all free software. Speakers of other languages sometimes organise themselves on a voluntary basis to transla documentation from English. Linux documentation in German, Japanese, French and Korean, for example, is relatively good.

The new Linux desktop environments, Gnome and KDE, have good on-line documentation as one of their explicit objectives. Both environments support internationalisation, making it possible to provide help in several languages. It is significant that Windows 95/98 comes with only a slim user manual for the operating system. The system is so consistent and intuitive that some initial training and a little practice are sufficient for most users, with on-line help filling in the gaps. Gnome and KDE follow the same model.

The commercial distributors of Linux bundle installation manuals with their CDs, but they don't provide extensive user manuals. Part of the reason is the desire to keep the cost of a distribution low. Manuals will certainly appear if customers express a desire for them and are willing to pay for them. Vendors of commercial software that is ported to Linux do of course provide documentation of similar quality to what they provide on other platforms. There is likely to be documentation available for the overwhelming majority of Linux software, whether Open Source or proprietary, though it may not always be the proverbial mouse-click away.

In short, the situation on the documentation front is by no means dismal, and like Linux itself, keeps getting better. Rather than bemoan the relative scarcity of elegantly bound, printed manuals, users should exploit the ability to browse HTML versions of Linux documentation on-line using their standard desktop browsers. This, in combination with other standard Internet tools like search engines, newsgroups and mailing lists, guarantees access to the most up to date information on Linux software, in a way that printed matter never can.

References:
The Linux Documentation Project: http://metalab.unc.edu/LDP
The Linux System Administrators' Guide: http://metalab.unc.edu/LDP/LDP/sag/index.html
The Linux Network Administrators' Guide: http://metalab.unc.edu/LDP/LDP/nag/nag.html
The Linux Programmer's Guide: http://metalab.unc.edu/LDP/LDP/lpg/index.html
The Linux HOWTOs: http://metalab.unc.edu/LDP/HOWTO/HOWTO-INDEX.html
The Linux "man" (manual) pages for download: ftp://ftp.win.tue.nl/pub/linux/docs/manpages
Linux FAQs (Frequently Asked Questions): http://metalab.unc.edu/LDP/FAQ/Linux-FAQ.html
O'Reilly computer books home page: http://www.oreilly.com

Lack of high-end features

Linux is a relatively new OS developed largely by volunteer programmers. These developers have not so far had access to high-end, expensive hardware (though many vendors are now making large systems available for independent developers to work on). So Linux today has simply not had enough "flying hours" on high-end hardware. Hence, it ranks (along with Windows NT) as a low-to-mid-range operating system, a category where it nevertheless performs quite well.

Many users worry about the scalability of Linux. They fear that if their workload increases, it may not be possible to upgrade their hardware and continue to use Linux. The latest Linux kernel (version 2.2), however, enables Linux to scale almost linearly in performance with additional CPUs, upto 4 processors, though it can run on machines with more processors as well. On 32-bit chips, Linux can use the full theoretical maximum of 4GB of virtual memory (2 raised to the 32nd power), except on Intel chips. (According to a senior kernel developer, it is not possible to run over 2GB of memory efficiently on a PC, so Linux uses a maximum of 2 GB there.) Linux can use 64GB on the UltraSPARC. It supports software RAID (Redundant Array of Inexpensive Disks) for disk

TOC

fault-tolerance, and clustering through "Beowulf" technology. IBM recently demonstrated Cray supercomputer-like performance using a Beowulf cluster of 17 Intel-based Netfinity servers running Linux, astonishing 3% of the cost!

However, Linux does not yet scale well beyond 4 CPUs, has no support for high-availability clustering (Beowulf is a *high-performance*, not a *high-availability* cluster) or ccNUMA (Cache-Coherent Non-Uniform Memory Access) architectures, cannot host multiple independent OS "domains" on the same box, or let a cluster of Linux systems appear as a single "image", features that many high-end Unix versions support today. A journalling file system and a logical volume manager are other notable absences.

There are projects underway to address all the above limitations, and the results should begin trickling in over the next year or two. In spite of these current deficiencies, Linux is likely to be adequate for quite a range of application requirements, although it cannot yet match the high-end offerings from IBM, Sun, Compaq/Digital, SGI or HP.

These established Unix vendors seem to have adopted a defensive marketing strategy. They offer Linux on their low-end proprietary hardware, and their own in-house OS for their high-end hardware. Thus, temporarily at least, they project a "complete" product line. It will be interesting to see how long this strategy works. It is an oft-repeated market maxim that "90% of the market is at the low-end". That is what enabled a PC maker like Compaq to grow large enough within a decade to take over makers of larger systems like Digital and Tandem. That is also what has contributed to the marketshare gains of Windows NT against these more powerful operating systems.

These Unix vendors may soon find Linux to be a cuckoo's egg. It could help them sell so many (low-end) machines that they would soon have to make the hard decision whether to continue pushing their own OS at all. Linux could make them realise that hardware is what gives them their margins. In a bid to maintain relative marketshare, each of them may be forced against their will to donate the high-end features of their own operating systems to Linux, moving it up the scale and ultimately replacing those proprietary systems altogether.

This has already started to happen. SGI (formerly Silicon Graphics) have announced that they will migrate many features of their Irix operating system to Linux over the coming year, such as ccNUMA, SMP, advanced memory management, high-speed networking and graphics. They seem to be keeping their promise with the recent release of the hitherto proprietary OpenVault removable-storage management software as Open Source. Market pressures could, before long, tear the "crown jewels" away from proprietary operating systems and hand them to Linux. This is sad in a way, but it would merely mark another milestone in the inexorable move towards open standards and systems, which is nothing less than a power shift away from vendors and towards users. Within a couple of years, Linux could even kill off the proprietary Unix versions and become *the* standard Unix of the industry.

The first signs of that have already started to appear. SCO (Santa Cruz Operation) is a vendor of a commercial version of Unix that runs on Intel servers, the very same market space occupied by Linux. SCO makes no hardware at all, and so it has absolutely no insurance against the Linux juggernaut. There are signs that Linux is starting to hurt SCO badly. As distrust of "freeware" Linux abates, users are increasingly unwilling to pay good money for a commercial product that isn't being improved with the same rapidity. A recent interview with SCO's CEO tells its own story.

In this context, the announcement of a tie-up between SCO, Sequent and IBM is puzzling for its timing. These three companies have announced that they will develop a common version of Unix, code-named Monterey, that will run on Intel's forthcoming 64-bit chip, Merced. Sequent is a niche player, and SCO has well-documented troubles. It is interesting to speculate on IBM's motivation for tying up with (what could, perhaps uncharitably, be called) two "losers".

Intel has publicly announced its support for Linux on Merced, and GNU software-based compiler tools for

Merced have already been released. With such formidable competition waiting on the Merced platform, Proj Monterey appears to be a mating of dinosaurs, -- a grand spectacle, but doomed to ultimate extinction.

Whether that happens or not, in view of the tremendous effort currently going into addressing Linux's high-end failings, users may not have to worry about Linux scalability in a couple of years.

References:
Linux Beowulf cluster-based supercomputers:
http://www.mercurycenter.com/svtech/news/breaking/internet/docs/350387l.htm
IBM's Linux-based supercomputer: http://www.infoworld.com/cgi-bin/displayStory.pl?99039.ecsuperlinux.htm
SGI to release technologies: http://www.it.fairfax.com.au/990309/openline1.html
SGI donates OpenVault:
http://www.newsalert.com/bin/story?StoryId=CnYpKWbWbu0znmduZ&FQ=open-source
SCO grasping at Linux straw: http://www.sunworld.com/swol-03-1999/swol-03-sco.html
SCO's CEO on Linux: http://www.computerworld.com/home/print.nsf/all/990426A08E
Project Monterey: http://www.linuxworld.com/linuxworld/lw-1998-10/lw-10-monterey.html
Cygnus GNUPro toolkit for Merced enables Linux to be compiled for that platform:
http://linuxtoday.com/stories/5434.html

Security concerns

"If the source code is open, hackers will be able to break into our systems more easily"

The fact of Linux's source code being common knowledge is enough to make many companies' security advisors recommend against its use. Many organisations subscribe to the "security through secrecy" doctrine, even though that has been discredited in professional security circles. In fact, a truly secure system is one that does *not* rely on the secrecy of its internals. Encryption algorithms, for instance, are generally publicly known. The DES algorithm and its derivative, the Triple-DES algorithm, have been intensively studied over the past two decades. No fatal weaknesses have been found. Triple-DES is now regarded as one of the most secure encryption algorithms, because it has stood up to close scrutiny. There is great advantage in having a system hardened through exposure, analysis, review and attack. A closed, secret system cannot be hardened so quickly, and can have bugs and loopholes in it that go undetected for a long time. If anything, the open source nature of Linux lets security problems be found and fixed quickly. There are many stories of security fixes for Linux being made available within *hours* of an attack being known (the FTP bounce attack, the teardrop or IP fragmentation attack, and the "ping of death"). So from a security angle, Linux, FreeBSD and other Open Source systems are preferable to the closed source OSes available from commercial vendors. The bugs and security deficiencies in commercial systems will continue to remain undetected until the day a hacker exploits them.

In this context, it's worth mentioning the recent TCP wrapper Trojan Horse attack. The TCP wrapper software is an Open Source program for secure network connections on Unix/Linux, and is distributed in both source and compiled forms. Given its sensitive nature, it was no wonder someone recently downloaded the source code, added a trapdoor to it, re-compiled it and loaded the compromised executable back to the website.

A security lapse? Undoubtedly. But what made the software vulnerable to attack also made it easy to detect and fix. Open Source software may be open, but the systems that its authors use are anything but naive. The checksum of every piece of software (called a message digest) is digitally signed by the author of the piece and placed on the same website as the software. Since the author's public key is publicly accessible from many independent sources, the message digest as it should be can be deciphered. When compared with a freshly-calculated message digest of the downloaded software, the user must see no difference. If there is a difference, it means the software has been tampered with. And so the Trojan Horse was detected after no mo than 52 downloads had taken place (Those 52 people had not bothered to cross-check the message digest). T was no way the attacker could have digitally signed the doctored executable without the author's private key, so it was only a matter of time before the deception was uncovered. The security of the website in question was also immediately tightened through better configuration to prevent such uploading of compromised software in future.

The lesson is that Open Source software has perfectly sound security systems, but users must use them properly to obtain their benefits. Open Source firewalls, like the "ipchains" program that comes with the new Linux kernel, are highly secure, but must be configured properly by the user to be effective. Security-related software that is readily available for Linux includes the aforementioned TCP wrapper and the *ssh* secure shell program, in addition to free, strong cryptography tools like PGP (Pretty Good Privacy), SSLeay, Cryptix and FreeS/WAN. Linux is thus as secure as you configure it to be, and inherently no less secure than most commercial software.

References:
Open Source and security:
http://www.linuxworld.com/linuxworld/lw-1998-11/lw-11-ramparts.html
TCP Wrapper Trojan Horse attack: http://www.linuxworld.com/linuxworld/lw-1999-03/lw-03-ramparts.html
Firewall configuration tool: http://rlz.ne.mediaone.net/linux/firewall
Download page for TCP Wrapper and other tools: ftp://ftp.porcupine.org/pub/security/index.html
The Secure Shell administration program ssh: http://www.ssh.fi
Firewall-building tool "Mason" licensed under GNU GPL: http://users.dhp.com/~whisper/mason
PGP download page: http://meryl.csd.uu.se/~d95mno/PGP.html
SSLeay home page: http://www.psy.uq.oz.au/~ftp/Crypto
Cryptix home page: http://www.cryptix.org
FreeS/WAN news item: http://www.infoworld.com/cgi-bin/displayStory.pl?990421.icfreeswan.htm
FreeS/WAN home page: http://www.xs4all.nl/~freeswan

Lack of support

By commercial standards of support, this argument is still partially true today, though it is fast fading away. This time last year, the only support one could get for Linux was in-house (if a company had a resident Unix guru) or through mailing lists. Mailing lists must seem a particularly pathetic way to obtain support, if it were not for the surprising observation that people have actually received quality suggestions and solutions fast, through this medium! No wonder Infoworld awarded the badge of "Best Technical Support" to the Linux community as a whole.

But today, more formal structures familiar to IT management have also arrived. Hardware vendors are prepared to stand behind the versions of Linux that they supply with their boxes. As earlier mentioned, HP now offers unlimited, 24/7 worldwide electronic support, with a guaranteed response time of under 2 hours, for a monthly fee of $130 per server. IBM is expected to announce a more comprehensive, if more expensive, support service. An organisation called LinuxCare has been formed purely for Linux support. Other third-party support can also be hired.

In short, given that a user is not dependent upon a single supplier for support and fixes, it may soon be possible to arrive at a nicely tailored support arrangement addressing the specific requirements of a site, which could include quality of service, price and any other relevant factors. Linux has made the differences in concept between software license fees and support fees explicit in the minds of users. This has already seen the rapid development of the support services market. The availability and the demand for third-party support will feed off each other, propelling Linux's acceptance. The beginnings of that trend can already be discerned today, ensuring that support will very soon cease to be an issue. This is one trend that promises to be very favourable to users.

References:
Linux community gets best technical support award 1997:
http://www.infoworld.com/cgi-bin/displayArchive.pl?/98/05/poy6a.dat.htm

IBM's ServerProven Program: http://www.techweb.com/wire/story/TWB19990323S0016

HP's 24/7 worldwide support package for Linux: http://www.news.com/News/Item/0,4,35392,00.html

Linuxcare home page: http://www.linuxcare.com

Absence of legal recourse

When organisations choose a software package, they need assurance that there is someone who can be held legally accountable if things go wrong, and who, in the worst case, can be sued for any loss incurred by using that software.

Linux, like all other free software, comes with absolutely no warranty. You cannot sue Linus Torvalds or Red Hat or Compaq if your production Linux server somehow goes berserk and trashes your precious data (No such case has been reported, of course, but we are talking of a hypothetical possibility). How can an organisation be expected to voluntarily expose itself to risk without legal recourse?

First of all, it must be clarified that when you pay for commercial software, you are not *purchasing* the product. You are paying for a "license" to *use* the product. Ownership of the software remains with the vendor. The legal framework governing software use is not Consumer Protection Law but Contract Law. Contract Law is much less consumer-friendly than Consumer Protection Law. Once you agree to the terms of the software license agreement, you are bound by it. How many people read the license terms before installing a piece of software?

The MS-Access bug is a case in point. Last year, a bug was discovered in Microsoft's Access database product. Under a certain (admittedly rare) set of circumstances, details entered by a user can end up being attached to the wrong record. You can imagine the consequences if a hospital's medical records or police records are so affected!

The interesting upshot of the bug was the light it shed on Microsoft's legal liability. It turned out that there was a clause in the MS-Access license agreement that said, in effect, that Microsoft was not responsible for any data corruption caused by the software. Sounds ridiculous, doesn't it? A database product not responsible for data? But Microsoft is not an exception. No software vendor takes responsibility for the integrity of your data. "No liability" clauses exist in all commercial software license agreements, and you cheerfully click the "I agree" button when you install the software.

Here is what Microsoft's End-User License Agreement for Office 97 (which includes Access 97) says. Note that it explicitly indemnifies Microsoft against "loss of business information".

[...]
LIMITATION OF LIABILITY. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL MICROSOFT OR ITS SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR ANY OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE PRODUCT OR THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES, EVEN IF MICROSOFT HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN ANY CASE, MICROSOFT'S ENTIRE LIABILITY UNDER ANY PROVISION OF THIS EULA SHALL BE LIMITED TO THE GREATER OF THE AMOUNT ACTUALLY PAID BY YOU FOR THE SOFTWARE PRODUCT OR U.S.$5.00; PROVIDED, HOWEVER, IF YOU HAVE ENTERED INTO A MICROSOFT

SUPPORT SERVICES AGREEMENT, MICROSOFT'S ENTIRE LIABILITY REGARDING SUPPORT SERVICES SHALL BE GOVERNED BY THE TERMS OF THAT AGREEMENT. BECAUSE SOME STATES AND JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF LIABILITY, THE ABOVE LIMITATION MAY NOT APPLY TO YOU.
[...]

If the situation you're complaining about is explicitly covered by the contract, and you have signed it, you may have no case, because *consent* is a powerful principle in law. Organisations need to consult expert legal opinion to assess the extent of legal protection they really have when the chips are down, as opposed to the protection they *think* they have. In any case, this very approach may be wrong-headed. Instead of looking for someone to sue when the chips are down, it may be better to look for someone to help keep the chips up, i.e., a support organisation.

And therefore, if commercial software doesn't give you any more legal protection than free software, using Linux instead of a commercial alternative may not increase your risk after all.

References:
The MS-Access bug: http://www.zdnet.com/zdnn/stories/zdnn_smgraph_display/0,4436,2131609,00.html

Lack of ownership

Corporate users are understandably reluctant to use software that emerged from the woodwork without a respected commercial organisation behind it. The lack of reputed ownership is one of Linux's major perceived drawbacks. However, nowadays, more and more users are beginning to think of it as a blessing. No vendor, no matter how dominant, can ever control Linux. Users and packaged software vendors alike have seen for themselves the control that Microsoft has over Windows, and the often unfair leverage it gives the company against its partners and customers. Hence, many in the industry today are secretly relieved that Linux can never be similarly controlled to suit any vendor's agenda. The GNU General Public License removes all incentive for proprietary code versions, because it forces all software changes to be released to the public. Thus, only the features most useful to the community find their way into the product, not features designed to "lock in" customers to a particular vendor.

Lack of ownership also means that Linux is viable on its own. It will continue to be used and improved as long as there are people interested in it, without being affected by the economic and business pressures that commercial products are vulnerable to. Netware's future, for example, is crucially tied to Novell's profitability, but Linux will continue even if Red Hat goes under. As users' awareness of Free Software's nature and longevity becomes more sophisticated, their confidence in Linux will increase. And so Linux's lack of ownership is a mere issue of perception, not an actual weakness.

The issue of standards, however, is a very important one. A single owner can enforce much-needed standards. The various distributions of Linux, though drawn from the same codebase, are different enough to cause confusion and some real incompatibilities, eerily reminiscent of the Unix wars. Such a bleak scenario might not come to pass this time around, but in the meantime, the Linux Standard Base project is worth watching and deserves all the encouragement it can get. Users and potential users of Linux alike should make known their support for a common Linux standard, because such a standard will benefit users most of all.

References:
POSIX and Unix 98: http://lwn.net/lwn/980611/standardseditorial.html
Linux Standard Base home page: http://www.linuxbase.org
Interesting viewpoints on Linux and standards compliance: http://lwn.net/lwn/980618/ianresp1.html,

http://lwn.net/lwn/980618/Editorial.html

Unviable business model

You must have heard this argument before: If Linux is free, how can anyone make money from it? The whole thing just doesn't make economic sense. The free software/open source idea is just a fad. It can't be sustained because there's no economic incentive anywhere to keep it going.

Well, just observe who it is who's telling you this. Certainly, if Linux and its accompanying free applications are widely adopted, that could knock the bottom out of the software market as we know it, and many vendors could lose money and even go out of business. That's a bad business model for them, perhaps, but the drop in prices all around is excellent for you. Don't confuse your interests with the interests of software vendors.

However, falling prices could mean that computers and computerisation then become far more affordable. The market could surge to new levels, making up for smaller margins with increased volumes, and IT business could actually flourish after a brief period of adjustment.

The best attitude to adopt here is not to complain about the noise when opportunity knocks. As a user, you are not looking at Linux as a possible way to *make* money. You are looking at it as a possible means to *save* money, and it can definitely help you do that. If it hits vendors in the short term, well, the market is perhaps due for a "correction".

Linux is the jewel in the crown of the Open Source movement, a tidal wave that is speeding towards the proprietary sand castles on the beach. And like any tidal wave, it is barely perceptible while still at sea. What is happening now is nothing less than a structural adjustment of the IT industry. Older vendors are already feeling the pressure, and hitherto unknown companies like Red Hat (Linux distributor), VA Linux Systems (formerly VA Research) and Penguin Computing (Linux hardware vendors), and LinuxCare (Linux support) are doing extremely well. Many companies have also sprung up to provide support for free software. Some of them even employ the original author of the software. So the argument that Linux will harm business in general is wrong. Linux is like the aeroplane in the early 20th century that put US railroad companies out of business and created a new industrial landscape with many more opportunities. The complaints you hear are perhaps from the latter-day versions of the railroad companies who will not adapt to a new era.

Besides, software vendors already face an unviable business model, as pointed out in a recent article, "How Microsoft took the 'Win' out of Windows". Independent software developers need to target Microsoft's Windows, because it is the most popular desktop platform, but they soon face competition from Microsoft's own products. Microsoft often "bundles" these products for competitive advantage, and there is a common perception that Microsoft applications are better "integrated" with Windows. This perception could well be justified. Software developers rely on the published Win32 APIs (Application Programming Interfaces) to get their products to run on Windows. But there are known to be many *undocumented* APIs that utilise Windows more efficiently. Microsoft's in-house application developers are believed to use these APIs to gain competitive advantage over external developers. This is a hot topic in the antitrust trial now underway.

As a user, such a situation may not concern you immediately as long as you get the products you want from Microsoft, but in the long term, monopolistic conditions lead to lower quality and higher prices.

For software vendors, Linux, far from being an unviable business model, offers a second chance to compete by providing a more level playing field than Windows. For example, Corel Computer (now Rebel.com) is making a comeback with WordPerfect on Linux. So Linux is not anti-business. It could just be anti-monopoly.

References:

"Linux's importance transcends itself": http://www.builder.com/Servers/Shafer/030199/?tag=st.cn.sr1.dir.
SCO grasping at Linux straw: http://www.sunworld.com/swol-03-1999/swol-03-sco.html
SCO's CEO on Linux: http://www.computerworld.com/home/print.nsf/all/990426A08E
Red Hat Software: http://www.redhat.com
VA Linux Systems: http://www.varesearch.com
Penguin Computing: http://penguincomputing.com
SuSE home page: http://www.suse.com
Linuxcare home page: http://www.linuxcare.com
Network Associates provide support and a commercial version of PGP:
http://www.nai.com/products/security/security.asp
Cygnus provides support for the GNU utilities: http://www.cygnus.com
Eric Allman's Sendmail, Inc. supports his freeware Sendmail program: http://www.sendmail.com
John Ousterhout's Scriptics Corp. supports his freeware Tcl/Tk scripting language: http://www.scriptics.com
Penguin64: http://www.penguin64.com
How Microsoft took the 'Win' out of Windows": http://www.zdnet.com/anchordesk/story/story_2890.html
Antitrust trial reveals Microsoft's competitive use of undocumented APIs:
http://www.news.com/SpecialFeatures/0,5,29271,00.html
Dr. Dobb's Journal book review of "Undocumented Windows": http://www.ercb.com/ddj/1992/ddj.9211.html
Corel ports WordPerfect to Linux: http://www.news.com/News/Item/0,4,21986,00.html

Uncertain roadmap

Where is Linux going? What features will it sport next year? The year after? Alas, there is no roadmap for the system. Linux developed in the anarchic medium of the Internet, and there is no strong visionary company that can provide reassuring plans of the features that Linux will support in the years to come. How is a user organisation to commit to an operating system that has no planned development path?

Again, a close examination of Linux's development process reveals that "this is not a bug but a feature". Linux acquires those features that its users find useful. Anyone is free to write code and extend Linux to support the features that they need. If those requirements are common enough, they form part of a mainstream distribution. It is akin to a market economy which responds sensitively to demand, unlike a planned economy where a few people make decisions based on their limited projections and often end up guessing wrong. So it would seem that the very concept of a roadmap is as outdated as a socialistic planned economy. Linux development is in ferment, and new ideas appear all the time. Contributions to Linux increase with its user base, in turn driving its acceptance. The "best" of these contributions are determined by the market, improved by proportionately as many users as are interested in it, and are gradually absorbed into the mainstream. The tongue-in-cheek Linux slogan, "Where do you want to go tomorrow?" is absolutely on target. Users decide where they want Linux to go. No company decides where they should go.

(Microsoft has begun pressuring Linux websites to remove the slogan from their pages, because of the potential confusion it could cause to customers with the Microsoft-trademarked slogan, "Where do you want to go today?")

Linux Tomorrow

TOC

Almost by definition, it is impossible to forecast the direction of what is a grassroots movement. Ideas can arrive from any quarter, and any attempt to predict the future based only on the current situation would be futile. However, in the short term, two trends are likely.

Variations in hardware

The world has barely begun to understand the freedom inherent in Linux. That is why the uses to which Linux has been put so far have been timid and traditional. But change is in the air. Many industry players have now understood the alien notion that a valuable piece of software can exist for the taking with no strings attached. They don't need to sign an expensive contract to use someone's software. They don't need anyone's permission to change the code. And if they sell a product based on Linux, they don't have to pay royalties to anyone. Linux makes it far easier for startups with a good idea to bring a product to market.

As people understand the freedom that they now have, they will start to take Linux apart and put it back together in so many different ways, getting it to run on systems that will bear little resemblance to the neat categories of "server" and "desktop" that we restrict ourselves to today. The catch-all term is "appliance", and many exciting developments are headed our way. Linux-based routers are a very cost-effective alternative to expensive Cisco routers. Other innovative products that have already arrived are the TV set-top box and the MP3 music player for cars, hardly anyone's mental model of a computer. Nevertheless, it has started to happen and will only continue.

Less dramatic, but equally useful, is the "multihead" computer. Windows 98 already supports multiple monitors on a single PC, but Linux will go further. It will support multiple *users*. The GGI (General Graphics Interface) project has contributed to the 2.2 kernel in some fundamental ways. Its EvStack and KGI offshoots now provide the basic infrastructure to support multiple monitors, keyboards and mice on a single computer. Once the other shoe drops in the form of full USB (Universal Serial Bus) support, it will be feasible to build multi-user computers, similar to the old text-mode minicomputers, except that the new ones will be graphical. This is a capability that Windows cannot currently match, because surprisingly, Windows NT is not a multiuser operating system, only a multitasking one. Only one local login is possible. It will take an overhaul of the NT code to support true multihead, and Linux will be far ahead by then.

References:

Linux-based routers: http://www.zdnet.com/sr/stories/column/0,4712,381687,00.html
Linux on a set-top box: http://www.thestandard.net/articles/display/0,1449,4246,00.html?home.bf
Linux runs an MP3 music player for cars: http://www.wired.com/news/news/technology/story/18236.html
NT is not a multiuser operating system, from the horse's mouth:
http://msdn.microsoft.com/LIBRARY/BACKGRND/HTML/NTSFROMUNIXPOV.HTM
Network Concierge, a Linux-based thin server: http://www.nc4u.com/linux.htm
Cobalt Networks, Inc. Linux-based thin servers: http://www.cobaltmicro.com
Corel Corp.'s Netwinder thin server running Linux: http://www.corelcomputer.com
The matchbox PC, running Linux: http://Boole.Stanford.EDU/cebit
Articles on low-cost server appliances: http://www.pcworld.com/current_issue/article/0,1212,10227,00.html, http://www.infoworld.com/cgi-bin/displayNew.pl?/odonnell/990419od.htm
ISPs get routers based on Linux: http://www.techweb.com/wire/story/TWB19990428S0001
The GGI project: http://www.ggi-project.org

Synergy with Java, XML and CORBA

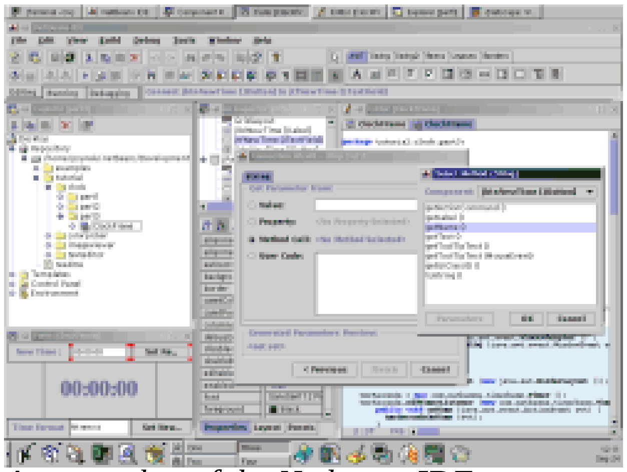
An inexorable trend over the past few decades has been towards industry standards and away from proprietary ones. Linux alone is not going to make the future. Other industry-standard technologies will complement it in the years to come, forming a compelling picture of completely open, industry-standard solutions to business requirements. Three likely companion technologies for Linux are Java, XML and CORBA.

Java is a technology that has been written off more than once, but there are signs that it is now poised for widespread adoption. Sun's court victory over Microsoft bodes well for a single, "unpolluted" Java version, and this can truly help users achieve cross-platform portability of applications. Java appears to have moved past the hype stage into a period of quiet adoption.

As a language, Java is clean, powerful and relatively easy to learn. Advanced features like multithreading are very simple to implement in Java, and will enable applications to scale well on the multiprocessor servers that are becoming increasingly commonplace. Java's automatic garbage collection feature results in applications that don't leak memory. This is very important for applications that need to run constantly without periodic system restarts. Empirical evidence also suggests that programmers exposed to Java are subsequently reluctant to return to C or C++, auguring well for a large future pool of skilled Java developers.

As a platform, Java is very close to its promise of "write once, run anywhere". When all commonly-used operating systems sport fully-compliant Java 2 virtual machines, the platform will be ready for serious "enterprise-class" applications.

Linux has never been one of the first platforms to support Java, mainly because Sun Microsystems has never before considered Linux a strategic platform. That is changing. For the first time, Sun is actively assisting an independent group of programmers, the Blackdown Group, in their efforts to port the Java 2 platform to Linux. At the time of writing, a fully certified port is likely to be announced "any day now". The Netbeans IDE (Integrated Development Environment) is an example of how slick and sophisticated the Java-Linux environment has already become.


A screenshot of the Netbeans IDE.

Java has been "saved" from Microsoft, but the battle now on is to free it from Sun itself. Clearly, Java is too important to the industry to be controlled by a single vendor, no matter how benign. Over the coming year, Sun may be forced by the industry (including its staunchest Java ally, IBM) to go beyond its "Community Source License" and yield control of Java to an industry standards body, perhaps under a BSD-style license or even the GPL. Excellent free JVMs (Java Virtual Machines) exist today, such as Transvirtual's Kaffe and The Hungry Programmers' Japhar. But Sun's certification process that mandates the use of the for-fee JCK (Java Compatibility Kit) means that it is difficult for free JVMs to obtain official Java branding, an example of Sun's excessively tight control of the Java specification. Project Mauve is an Open Source alternative to Sun's proprietary JCK, to which even HP has contributed its Chai test suite.

The increasing popularity of Java and availability of Java-based applications could spur the adoption of Linux. The Java focus would render the OS a commodity judged solely on its speed and stability, and these are areas where Linux does well.

Java has the potential to render the entire OS argument irrelevant, and it will be interesting to see the effect it has on Linux. Java is certainly not an enemy of Linux, and some analysts expect the Java-Linux combination to be wildly successful. The combination would certainly deliver good value.

If Java represents portable programs, XML is portable *data*. XML permits application-specific markup of data that can handle the most complex data relationships, and yet store it all in pure ASCII text form that is portable across all machines and architectures. Java and XML get along like a house on fire. Most XML parsers are written in Java. XML files can be easily parsed into Java objects. Both support Unicode, the coding scheme that handles virtually all the world's language scripts. Java/XML applications can therefore readily suppport I18N (Internationalisation). XML is still a new technology, and some of the specifications for its associated parts are yet to be standardised. Nevertheless, this is a technology that already seems a winner.

Linux is a natural platform for XML. Text files in Linux are also stored in ASCII format. Linux will therefore not create any impediment to implementing XML-based applications.

Finally, the component architecture for the rest of the industry, CORBA, is also coming of age. Any number of distributed applications now use CORBA. Again, like with XML, it is Java that has given CORBA new energy. Java RMI (Remote Method Invocation) over CORBA IIOP (Internet Inter-ORB Protocol) is one of the examples of Java-CORBA synergy. CORBA, as a distributed object architecture that runs on all platforms, can do almost anything with distributed objects *except* toss an object across from one host to another. And Java can do that with object serialisation and RMI. The CORBA 3 architecture has standardised on EJB (Enterprise JavaBeans), throwing aside all pretence at language-neutrality. Whatever the merits of that decision, the combination of Java, XML and CORBA provides developers with unprecedented power to implement complex applications in an open, non-proprietary way, and Linux can provide a no-fuss, solid platform for such applications.

In contrast to this happy family photograph, the Visual Basic/COM+ combination seems isolationist, a "Microsoft-only" view of the world. True, it is a well-integrated family of technologies, but with anything less than a commanding marketshare, the whole shebang will be seen for what it is, -- a vendor-proprietary technology that is best avoided. With Linux set to soon overtake Windows in the server market, a decision in favour of proprietary technology over industry-standard technology will be extremely difficult to justify.

References:

Java to run on Linux: http://www.news.com/News/Item/0,4,33086,00.html
Java not to be an ISO standard: http://www.zdnet.com/zdnn/stories/news/0,4586,1014537,00.html
Java-Linux port test status: http://www.blackdown.org/java-linux/jck-status.html
The Netbeans IDE (free for non-commercial use): http://www.netbeans.com
Transvirtual's Kaffe JVM: http://www.transvirtual.com
The Hungry Programmers' Japhar JVM: http://www.japhar.org/index.html
Project Mauve: http://www4.zdnet.com/intweek/stories/news/0,4164,2229207,00.html
XML and Java tools downloadable at IBM's Alphaworks site: http://www.alphaworks.ibm.com
James Clark's free XML tools: http://www.jclark.com/xml
ORBit, a free, GPL-ed CORBA ORB which is part of Gnome:
http://www.debian.org/Packages/stable/devel/orbit.html
OmniORB, a free, GPL-ed CORBA ORB from Olivetti/Oracle labs (now AT&T):

http://www.uk.research.att.com/omniORB

Skepticism about Linux

Many arguments have been made against Linux that cannot be neatly classified, and the most common of them are addressed in this section. These are serious questions that need to be answered equally seriously.

"Linux will splinter just like Unix"

Every year for almost a decade, the IT industry heard the mantra that this was the year Unix would finally rule. All other operating systems would die out and Unix alone would remain. Whatever happened to that fond vision? Unix ended up fragmenting into many slightly incompatible versions, each of which remained in its own niche. Meanwhile, an exciting new operating system called Windows NT made its appearance. This OS boasted some unique features. It had most of the technical features of Unix (true 32-bit memory addressing, pre-emptive multi-tasking, etc.) an attractive interface, easy-to-use administration tools, a low price tag and a single version controlled by a single vendor. In no time, Windows NT gobbled up the low end of the server market, and began its inexorable climb up the server food chain. Unix never looked so vulnerable.

But since then, Microsoft itself has been unpleasantly surprised by a challenger from the low end. Linux is cheap, standards-compliant, stable, fast and isn't controlled by anyone. The only problem is, it's Unix, so it could go the same way Unix went, right?

The possibility cannot be ruled out. However, the GNU General Public License comes to Linux's rescue. Since Linux is licensed under this contract, it is not legally possible for any proprietary version to exist. The license forces all changes to be given back to the community in source form for free use and modification. So if two different parties make different and independent changes to Linux, both have to make their source code publicly available. It is then possible, at least theoretically, to incorporate both sets of changes into a single new version. This will happen fairly quickly if both the changes are found useful. The GPL has never been tested in court, however, so its validity is not assured, though legal experts have pronounced that the license appears enforceable.

In a process that Linus Torvalds calls "healing the splinter", code forks are periodically reintegrated into a single version. In fact, current Linux development is nothing but a constant sequence of splintering followed by re-absorption, in which independent developers submit modified versions of Linux modules to a core development group to evaluate and incorporate. In a Darwinian system of ruthless meritocracy, the most useful and best-written code (regardless of the contributors' prior reputations) is chosen and becomes part of future standard distributions. Star Trek fans will recognise that this is a Borg-like assimilation strategy that is not easily beaten. Code-forking has no incentive to occur, and there will probably be just one Linux for the foreseeable future.

(There will continue to be many Linux *distributions*, which are collections of Linux software on a single CD. These could be different because the distributors differ in their views as to which programs, and which versions of those programs, are worth bundling. Note that the existence of multiple distributions is not an example of Linux splintering, because all distributions are based on a single, unsplintered codebase.)

References:
"Linux may be running on some spindly legal legs", -- an analysis:
http://www.businessweek.com/cgi-bin/bwdaily_full?right=/bwdaily/dnflash/apr1999/nf90427b.htm

"There is no quality control in free software"

TOC

The very term "Freeware" conjures up images of amateur teenage programmers, inexperienced university students and pony-tailed, rebel "hackers". These people are expected to be nerdy, and not very much bothered about polished interfaces, ease of use, documentation and other niceties that are found in commercial software. Come to think of it, free software isn't even expected to be of good quality. There's no accountability, after all. These guys are just writing this stuff for themselves. You really can't expect very much out of it. On the other hand, a commercial organisation that makes a living out of a software product is accountable, and is bound to be quality-conscious because it is a matter of business survival.

So it never ceases to amaze when free software products like Perl, Apache and Linux work so well, in many cases *better* than any commercial equivalent. Surely it's got to be a fluke.

Well, the one big change in recent times has been the Internet. Before the Internet became a phenomenon that touched everyone's life, the only decent free software was from the GNU project. This was a small group of highly talented programmers, in some ways an exception to the rule. Most PC-based freeware and even shareware was buggy and unpolished. These programs established a reputation that even today gives free software a bad name.

The Internet changed everything because it was now possible for programmers around the world to work on the same project and share files almost as if they were in the same room. All of a sudden, project teams sprang up that just happened to be geographically scattered. Programmers with similar interests could seek out and find one another. In very short order, the Internet enabled a new, distributed method of software development. Free software isn't written just by college students and amateurs. Rather, project teams have a mix of programmer types. And surprisingly, many of them turn out to be regular employees of regular companies, in short, "professional programmers". Many free software products that are born of the Internet have been written by professional programmers in their spare time, which puts quite a different slant on it. These guys know what they're doing. They've been trained in formal processes. They're not cowboys or amateurs.

Even more interesting than the demographics are the motivations behind these programmers' willingness to spend hours of their personal time working on something valuable, only to give it away. Why would perfectly normal people do that? Isn't money the best motivator? Surely a product that is paid for would be of better quality than one that isn't! All these products of "spare-time activity" should show indifferent quality, right? So why don't Linux or Apache ever crash? How does Perl do so much, so efficiently? And when you look at the source code, why is it so well-written?

It's very tempting to leave these questions unanswered in order to provoke thought and facilitate a new awareness, but let's try and answer them, anyway. Many thinking programmers have some pet project that they would like to work on, but aren't able to because the immediate business goals of their employer lie in a different direction. Such programmers are likely to be highly motivated individuals who like technology and their work. Programming to them is "fun", not a chore. When such programmers find others who share their dreams for a certain piece of software, they find it very exhilarating and energising, even if these others are halfway across the world.

The world inhabited by motivated computer programmers is one that few others can even understand. Most of humanity is condemned to forever stand outside looking in on this world. For it is these people who really live life and derive maximum satisfaction from what they do. In intense groups that share a common passion, these professional programmers put their very being into building the software that they have dreamt about. Some of these projects are too large for a single programmer, no matter how talented. But in what is virtually an infinite human resource pool, it is always possible to find the right people to work on the right modules of such a large project. And so, not surprisingly, very high-quality works are turned out by these teams. Quality is high on the

TOC

agenda. Remember, these are professional programmers who are highly motivated and take great pride in the work. Ego and the desire for peer recognition are very strong factors that ensure that code quality is always h If one programmer writes slightly shoddy code, another one will simply rewrite it because it irritates the eyes to look at it. Nothing stops them from doing this because this is FREE software, remember? If you don't believe this, take a look at a snippet of Linux kernel source code sometime. Then look at a piece of code from one of your organisation's own prestigious projects. Do you see a quality difference? Would you still blithely maintain that freeware is shoddy? This is the "independent peer review" that Open Source advocates point to, which is similar to its counterpart in science that guarantees quality research and prevents fraud.

The motivation angle stands the entire economic argument on its head. If you really *love* doing something, wouldn't you do it a lot better than if you were merely *paid* to do it? Why was the American army in the Gulf war a more motivated lot than the American army in Vietnam? Because they were volunteers, not conscripts. To take an extreme position, the freeware army is a volunteer army, while most programming teams in commercial organisations are conscript armies. The quality difference should really surprise no one.

The rigours of working in a far-flung team have yielded many good software management tools that even commercial organisations can use to advantage. CVS (Concurrent Versions System) is the source code version control software that is used by most free software projects. Bugzilla is a product that came out of Netscape as Open Source, and has been steadily improved since. It is a powerful tool to gather bug reports from a large team of testers, record the assignment of bugs to developers, and track the progress of bug fixes. Bugzilla even allows contributors to upload fixes along with bug reports. Bonsai is a tool to view specified subsets of source trees. Mailing list managers like Majordomo and more modern ones like Mailman allow a team to work in close coordination irrespective of geography.

Even more important than products are the standards that have sprung up. Most free software for Linux/Unix comes in the form of a "tarball", a entire directory tree of source files that is compressed into a single file. You uncompress this file just after you download it, exploding it once more into a directory tree.. Then you typically do just three things. You first run the simple command *configure*, which checks out the versions and locations of your compiler and libraries, and creates a machine-specific blueprint to build the application, -- a *make* file. Next, you type the single command *make*, and the entire source code is compiled into an executable version. Finally, you type *make install*, and all the executables and libraries that were created are copied into appropriate directories in your "path". That's it. You're ready to run the application without even a reboot. Even though this is a command-line procedure, it is as simple as the Windows "setup" mechanism because it involves just these four steps. Almost any piece of free software that you download will involve an identical procedure. All of them will have an *INSTALL* file in the top-level directory with detailed instructions, if you need them. Red Hat package manager files (rpm files) are even simpler to install and involve only one step.

The free software development process is quite different from what you expected, isn't it? There's far more structure and far less anarchy than you would imagine. If you have the time, get onto a developers' mailing list for some free software project and just "lurk", reading the mail that flies back and forth, and getting a feel for the energy, focus and dedication of the team. The experience is bound to make project managers in many commercial organisations long for a similar level of motivation and commitment in their in-house teams.

References:

The Cathedral and the Bazaar (the article that influenced Netscape's decision to release the source code for Communicator): http://www.tuxedo.org/~esr/writings/cathedral-bazaar/cathedral-bazaar.html
Homesteading the Noosphere (an article that tries to explain the hacker "gift culture"): http://www.tuxedo.org/~esr/writings/homesteading/homesteading.html
How to be a hacker (another article that provides insights into the workings of Open Source): http://www.tuxedo.org/~esr/faqs/hacker-howto.html
Bugzilla page for the Mozilla project: http://bugzilla.mozilla.org

CVS (Concurrent Versions System): http://www.cyclic.com/cyclic-pages/CVS-sheet.html
Bonsai source tree filter: http://www.mozilla.org/bonsai.html
Mailman mailing list manager: http://www.list.org

"We can't trust production data to a freeware program!"

Well, if yours is one of the hundreds of thousands of organisations worldwide that process production data using Perl scripts, you already do! Perl is free software, just like Linux. If there is a bug in Perl that subtly alters your production data, it could cost you money and even expose you to litigation! And no, you can't take the main author, Larry Wall, to court for damages. The free "Artistic License" under which Perl is distributed, expressly disclaims responsibility for any damage caused by the program, and you "agreed" when you installed the software. Does this send a chill down your spine?

But wait! Before you storm in there and forbid the use of those trusty Perl scripts, ask around. As your or any IT department could tell you after years of experience with the software, Perl is extremely powerful, thoroughly reliable and of very high quality. It is more than value for money, because it doesn't even cost money!

Just think about this. If you're comfortable letting "freeware" Perl scripts loose on your production data, what's wrong with compiling your production systems' C and C++ programs using the freeware GNU C/C++ compiler? That's a tried and tested piece of software, too. And if that's OK, what's wrong with hosting your corporate website on a freeware webserver like Apache that's used by over half the Internet sites in the world (including, amazingly, Microsoft's Hotmail with its 30 million subscribers)? Well, if you can stomach all that, what makes you hesitate to use a freeware operating system that's growing faster than any other OS, and has even been used in experiments on the Space Shuttle?

The "F" word still has some strange stigma attached to it, even though there are plenty of examples of extremely high-quality free software that deliver excellent value. The pragmatic manager evaluates software based on its merits and the experiences of other users. A prejudiced attitude, on the other hand, can keep you and your organisation from enjoying the significant benefits that many good free software programs can give you. Just make reasonable arrangements for support, either in-house or third-party.

References:
Perl home pages: http://www.perl.com/pace/pub, http://www.perl.org
The GNU C/C++ compiler gcc: http://www.gnu.org/software/gcc/gcc.html
Apache webserver home page: http://www.apache.org
The webserver market and Apache's share: http://www.netcraft.com/survey
Netcraft's webserver monitor (Enter "www.hotmail.com" and click "Examine"): http://www.netcraft.com/cgi-bin/Survey/whats
Linux on the Space Shuttle: http://www.linuxjournal.com/issue39/2186.html

"There's no such thing as a free lunch"

It's commonly accepted wisdom that there's always a price to pay, even if it isn't obvious. This implies that "freeware" Linux cannot possibly be without some compensating costs, though those costs may not be immediately apparent...

This argument, though superficially reasonable, ignores the discovery of new efficiencies. By definition, improved efficiency means you get an extra return for the same investment. If your car used to give you *X* kilometres to the litre, and then one afternoon, you tuned the engine a bit, and it now gives you *X* plus something more, you've got that extra something for nothing, -- in effect, a "free lunch".

Distributed.net is an organisation that uses the *idle CPU cycles* of computers on the Internet to crack secret messages. Anyone can join the effort. The organisation has won many challenges to crack encrypted messages in a fairly short time. What is distributed.net's cost of solving these problems? Essentially nothing, because the resources used to solve it are going waste anyway.

The SETI (Search for Extra-Terrestrial Intelligence) project is distributing a new kind of screensaver that not only displays some fancy graphics when your machine is idle, but actually downloads some radio telescope data and processes it at such idle times. The SETI project coordinators hope to tap into the power of the millions of personal computers that connect to the Internet, and use their idle CPU cycles to do some useful work. Again, something for nothing.

There's no doubt that the Internet has made the marshalling of distributed resources much easier, and it could be argued that this extends to the software development process itself. Much of the free software we see today was born of Internet-based cooperative groups. In a sense, such software is developed using the "idle CPU cycles" of professional programmers worldwide, which might otherwise have been wasted in watching TV! In a few years, economists might point to this perod as the time when society as a whole discovered new efficiencies.

So yes, Linux may have its drawbacks, but they are not necessarily the inevitable "price to pay" for being free.

References:
Distributed.net website: http://www.distributed.net/
The SETI screensaver: http://setiathome.ssl.berkeley.edu/download.html

"Open Source products are good copycats, but poor innovators"

Microsoft's Jim Allchin put it quite graphically when he said Open Source projects were good at "following tail lights", implying that users need commercial software vendors to provide truly innovative products. An understandable conclusion, considering that much of the recent media attention on Linux has been on its new Windows-like interface, and the rapid development of features (from other operating environments) that were hitherto missing.

One must not forget, though, that much of Unix itself was developed as Open Source, long before the term was invented, and Unix has since served as a model for other operating systems, Windows included. One needs to look no further than BSD (the Berkeley Standard Distribution), with its revolutionary socket-based network programming model. Windows today uses sockets without a murmur of acknowledgement of the technology's Open Source roots. Virtually all Internet-related innovation, in the form of both protocols and implementations, took place in an open, consensual manner. Internet standards are typically not from a standards body, but from RFCs (Requests for Comments), a very informal mechanism for seeking feedback on new technology proposals. This is virtually identical to the Open Source methodology. Standard Internet services, such as domain naming and mail transport, are implemented as Open Source products that still power the bulk of the Internet. The products are the well-known *bind* (Berkeley Internet Name Daemon) and *sendmail*. The first webserver (NCSA httpd) was Open Source. Unix shared libraries clearly inspired Windows' DLLs (Dynamic Link Libraries). So the "closed source" world does not have a monopoly on innovation.

It is certainly true that Linux has so far been playing catch-up, because the focus so far was on proving compliance and interoperability, but now that Open Source products have reached a certain level of maturity, innovation is starting to appear in unexpected areas. The Java-Apache project is a prime example. Many good

TOC

ideas can be found there, including the revolutionary *MailServlet*, which in effect, allows a webserver to be a mailserver. The idea is so radical that Sun turned it down when it was proposed as a standard servlet to be included in the official *javax.servlet* package. The idea's proponents, however, are continuing with its development, even though it will not be part of the "official" servlet package. Sample code at the Java-Apache website shows how trivial it would be to build (say) a mailing list manager if a MailServlet entity existed.

The GGI (General Graphics Interface) project is equally innovative, and will give Linux multiuser capability on PC hardware, through EvStack and KGI. This is different from the old model of character-mode terminals on minicomputers. It will allow multiple users (each with a graphics monitor, keyboard and mouse) to work on the same machine, a capability called *multihead*. When implemented, it will be a first for the industry. True, Tektronix terminals on old minicomputers provided similar capability, but that was on very specialised hardware. GGI will let it happen on commodity PC hardware.

In any case, innovation does not mean just inventing new things. It includes standing on the shoulders of those who came before, and building on their work. By this token, the Gnome desktop's use of CORBA to coordinate desktop objects is certainly innovative.

So although one can understand Jim Allchin's motives in trying to scare you, stagnation is an unlikely outcome of adopting Open Source products. Many exciting "goodies" are even now flowing in, and users don't have to pay a hefty premium to use them.

References:
Jim Allchin talks of Open Source chasing tail lights: http://nerv-un.net/doc/halloween.html
The Java-Apache project: http://java.apache.org
James (Java-Apache Mail Enterprise Server) MailServlet page: http://java.apache.org/james/index.html
The GGI project: http://www.ggi-project.org
Gnome home page: http://www.gnome.org
ORBit, a free, GPL-ed CORBA ORB which is part of Gnome:
http://www.debian.org/Packages/stable/devel/orbit.html

"Linux is not yet ready for the Enterprise"

Following the amazing amount of positive press about Linux has come the inevitable (and welcome) hard-headed analysis of its features and potential value. D.H. Brown Associates have perhaps conducted the most painstaking and comprehensive analysis of all. The report in its entirety is available for a fee, but a summary is freely downloadable from the company's website. In the summary's conclusion, they state that they "[...] hesitate to recommend Linux distributions as enterprise computing solutions due to their lack of high-end capabilities [...]".

This conclusion has been seized upon by many to argue against investigating Linux more seriously. Some of the media headlines accompanying the report imply that Linux is currently unsuitable for use by an enterprise (in the sense of an organisation). But this is not the point the report is trying to make. If you read carefully, it does not say that Linux cannot be used by an enterprise. It merely says that Linux is currently unsuitable for certain tasks that are considered "enterprise-class". There is considerable debate over what constitute "enterprise-class" applications, and the references at the end of this section should prove interesting reading.

But coming back to the report, what the summary actually says is,

"In key application areas that employ open protocols or well-characterized closed protocols, such as low-end or midrange Web serving, e-mail routing, network printing, and file serving, Linux can provide a solution that, once properly configured, is both stable and performs adequately for at least modest workloads. Further, Linux's "good

TOC

enough" capabilities come at minimal cost and do not incur significant vendor lock-in."

So, the report is actually saying that Linux can be used *today* in roles such as the above, -- in short, all the ro which Windows NT is currently being used, as an IDC report found.

Two well-known US companies, Burlington Coat Factory and Jay Jacobs, rolled out new production systems based on Linux. Burlington installed 1250 Linux/Dell machines in 264 stores over 42 US states. Clothing retailer Jay Jacobs installed Linux/Compaq machines in 115 stores across the US. These can hardly be dismissed as something other than "enterprise" applications. From the positive experience of such early adopters, it appears that the D.H. Brown report may be unnecessarily guarded in its recommendations.

In any case, it would be a pity if an organisation walked away from the potential benefits of using Linux in such roles because of a misinterpretation of a report's conclusions.

References:

D.H. Brown Associates' report summary for download: http://www.dhbrown.com/dhbrown/linux.html
NT's market strength is on paper only (IDC report): http://cnn.com/TECH/computing/9904/21/ntpaper.ent.idg
Aventail Corporation's Extranet Strategist magazine on Linux for the Enterprise: http://www.extranet-strategist.com/trends/linux.html
Windows NT Magazine's editor on Enterprise-class computing (also read LinuxToday reader comments below it): http://linuxtoday.com/stories/5499_flat.html
Burlington coat factory installs Linux: http://www.infoworld.com/cgi-bin/displayStory.pl?99046.eccoat.htm
Clothing retailer Jay Jacobs installs Linux: http://www.techweb.com/se/directlink.cgi?INW19981109S0019

"What does Linux imply for me professionally?"

For IT managers, the advent of Linux is a crisis, in the Chinese sense of both danger and opportunity.

As the use of Linux is increasingly seen to be viable in a range of situations, hard questions will start to be asked of IT managers who continue to purchase more expensive, proprietary alternatives. Managers have a responsibility to spend their organisations' money wisely and avoid vendor lock-in. Provided adequate arrangements are made for support and maintenance, Linux could prove a very cost-effective and standard platform for many organisations, a fact that will not long escape the notice of senior management with an eye on the bottom line. As an IT manager, it is best to prepare early for the day when you will be asked to formally evaluate and report on the feasibility of using Linux. You must have sufficient knowledge of Linux to argue convincingly either way.

Linux also provides a unique career opportunity for dynamic managers who are willing to take considered risks to pull ahead of the pack in the eyes of senior management. If you manage to realise Linux's promise of high-performance computing with high uptime, radically lower cost of ownership and increased bargaining power *vis-à-vis* vendors, you can deliver impressive value to your organisation. Careers can be built by being the first to recognise such value and exploit it for the organisation's benefit. Yes, no one ever got fired for buying Microsoft. But no one got promoted for it, either.

As a manager, you are also probably responsible for staffing. The computing landscape is even now being dramatically altered, with Linux being legitimised and used by IT in very large numbers, and it follows that a severe skills shortage is imminent. Even with a GUI, Linux is quite different from Windows, and the people in the trenches who need to work on this platform will need to be trained on a different set of skills. No matter what

Microsoft might say, Unix has re-emerged from its grave thanks to Linux and the Internet, and Unix skills ar once again at a premium. Start equipping your staff with Unix skills now. Budget for Unix-related training, a look for Unix skills when hiring. It's not a one-horse race anymore.

Linux is here to stay. It will not go away if you ignore it. The longer you remain in a state of denial, the worse it will be for your career. You need to bite the bullet now and start familiarising yourself with Linux, even if you end up never using it for serious work. Call it insurance.

This document is an attempt to put enough information in your hands to help you get started, but if you have had no experience at all with any brand of Unix, you will need to gain practical experience with Linux for about 3 to 6 months before you can tame the beast. Start small, but start now.

"Where can I use Linux today?"

If you have never used Linux before, and you have limited Unix skills available in-house or on call, you would be wise to consider a period of experimentation before committing Linux to actual production applications. You need to have skilled people on board as you roll out such applications, so use the experimentation phase to train people as well.

Obviously, you're not aiming to replace your mainframe or high-end Unix server with a Linux box. (At least, not yet.) You probably also don't want to migrate the corporate desktop environment from Windows to Linux just yet. As of now, that's still risky and likely to be disruptive.

You may also not be able to fund this familiarisation exercise from your budget, so you may have to fall back on older, retired machines and only free equivalents of all software products. As you prove the concept both to yourself and to your superiors, you could replace some of the less polished free software with commercial equivalents.

Choose an application to play with

A good starting point would be any of the areas listed in the D.H. Brown report. In other words, deploy Linux experimentally in any of the following roles:

1. A webserver running the industry-standard Apache software, perhaps for a simple Intranet application
2. A mailhost for e-mail (supporting SMTP, POP3 and IMAP4)
3. A Windows file and print server using Samba

If you believe that the D.H. Brown report is overly conservative, or once you gain confidence in Linux's reliable performance, you can also explore using Linux in the following roles:

1. A secure webserver that supports 128-bit SSL using SSLeay and Apache mod_ssl
2. A more fully functional webserver using Java applets, servlets (using Apache mod_jserv), Active Server Pages-style embedded HTML scripting using PHP, and any relational database
3. A database server for development environments, using any of the popular database packages that are available for the platform
4. A development client workstation for C, C++ and Java developers, to be offered as an *option* to those of your developers who are interested
5. A graphics designer's workstation using the GIMP image manipulation software
6. A DNS (Domain Name Server)

7. A firewall using the ipchains software that comes bundled with kernel 2.2
8. A proxy server, including a caching proxy using Squid
9. A router
10. A web application server using Enhydra
11. A directory server using OpenLDAP
12. An internal mailing-list manager using MailMan
13. A multimedia application using the object-relational PostgreSQL database
14. A fax server using Hylafax

As you can see, there are plenty of potential application areas. Take your pick.

Choose hardware

Yes, they say Linux will run on a 386 with 4 MB of RAM. But you are not trying to set records. You are trying to evaluate Linux in a reasonably comfortable configuration, so that you get a true picture of its abilities. At the minimum, dedicate a Pentium 100 machine with 32 MB of RAM, 1 GB of disk, and a CD-ROM drive of any speed. Ensure that the PC has a network card and is plugged into your network, otherwise you won't truly appreciate what is, after all, a *network* operating system. If your corporate policy won't allow a Linux machine on the network, set it up along with one or two Windows clients in a separate network altogether.

Choose a distribution

The best known distributions are Red Hat, Caldera, S.u.S.E. (especially in Europe), Pacific Hitech (especially in Japan), Debian, Mandrake and Slackware. Choose a distribution that uses the latest kernel (version 2.2) and one of the two good Linux desktops (Gnome or KDE).

Choose the right people

Good managers don't bet on technology. They bet on people. The results of your experiments could turn out either way depending on the people who work on them. Since it is in your interests to evaluate Linux at its best, ensure that the right people work on these pilot projects. Don't force anybody into the role. You're best off with volunteers, because they're the ones who try to make things work. Conscripts usually give up after a couple of token attempts.

Make use of the Linux community when you need help, through newsgroups and mailing lists. The community has earned a well-deserved reputation for helpfulness. At the same time, don't strain that goodwill by asking questions to which the answers already exist in the documentation. Get your people to study the documentation before shooting off queries to other busy professionals. Getting your people onto discussion groups, even in "lurk mode", can teach them a lot, and you can leverage off the collective knowledge base at little cost.

And finally...

If you follow these steps, and get yourself and your people used to Linux, you will be ready in a few short months to reply to the inevitable memo from on high, something along the lines of "Could Linux be used by our company? Please discuss."

You can walk into the subsequent meeting, present the strengths and weaknesses of Linux as you see them in the context of your organisation, answer questions, correct misconceptions, suggest opportunities, provide realistic adoption and deployment scenarios, and make sensible policy recommendations. You may well recommend

TOC

against using Linux, but your recommendation will be based on verifiable facts and undisputable personal experience, not on hearsay, preconceptions or plain ignorance.

If this document can get you started on that journey, it will have accomplished its purpose.

References:

Linux Today has links to breaking news affecting Linux and Open Source: http://www.linuxtoday.com
Slashdot is where the "nerds" hang out. The threaded discussion groups are raucous, but can be very educative. http://www.slashdot.org
Freshmeat has links to the very latest versions of free software as they are released. http://www.freshmeat.net
Samba installation guide: http://www.sfu.ca/~yzhang/linux/samba/index.html
An interactive firewall configuration tool on the Internet, for use with ipchains: http://rlz.ne.mediaone.net/linux/firewall
Firewall-building tool "Mason" licensed under GNU GPL: http://users.dhp.com/~whisper/mason
Apache home page: http://www.apache.org
Sendmail home page: http://www.sendmail.org
SSLeay home page: http://www.psy.uq.oz.au/~ftp/Crypto
SSLeay download page: ftp://ftp.pca.dfn.de/pub/tools/net/ssleay
mod_ssl download page: http://www.engelschall.com/sw/mod_ssl/distrib
ServletCentral website for server-side Java: http://www.servletcentral.com
PHP (ASP-style HTML-embedded scripting language): http://www.php.net
The GNU C/C++ compiler gcc: http://www.gnu.org/software/gcc/gcc.html
The GIMP home page: http://www.gimp.org
Review of the GIMP: http://webreview.com/wr/pub/1999/03/05/studio/index.html
The Artists' guide to the GIMP: http://www.ssc.com/ssc/gimp.html
Squid home page: http://squid.nlanr.net/Squid
Linux-based routers: http://www.zdnet.com/sr/stories/column/0,4712,381687,00.html
Enhydra application server home page: http://www.enhydra.com
OpenLDAP project home page: http://www.openldap.org
Mailman mailing list manager: http://www.list.org
PostgreSQL website: http://www.postgresql.org
Hylafax home page: http://www.hylafax.or

A downloadable version of this entire document is available "here."

Authors background:

"Ganesh C. Prasad"

is an MBA with a degree in Computer Science, a combination that gives him a unique perspective on computers in business. He has 12 years of software development experience, and has worked on a variety of platforms and technologies, ranging from IBM mainframe COBOL, through 'C', VAX/VMS and Unix-based RDBMS applications, client/server systems, to Web and Java apps. Like many other IT professionals, he uses Windows NT at work, and Linux at home.

Ganesh has worked in leading consultancy firms in India and Australia, and in the IT department of a large bank

TOC

in Dubai. He is a keen follower of technology and business trends in the IT industry, and an amateur studen economics. Linux interests him more for its social and economic impact than for its technical features.

Ganesh is currently employed as a Senior Information Specialist in the Internet Development Services group of EDS Australia. The views expressed in this document are his own and do not represent the views of EDS.