Why I hate Microsoft

"A personal, lengthy, but highly articulate outburst"

by F.W. van Wensveen

Table of contents

Introduction

"One OS to bring them all and in the darkness bind them..."

From the title of this paper you may have guessed that I am not very impressed with the guys in Redmond. One might even say that my dislike for Microsoft is a pet hate gone out of control in an almost quixotic fashion. Why is this?

Of course I have been accused of personal antipathy, of being jealous of Bill Gates and his billions, and of being prejudiced against all things Microsoft without any reason whatsoever. None of this is true. I have nothing personal against Bill Gates. Why should I? I don't know the man, I've never met him. I agree with those who say he might be the most successful salesman in history. And I've always thought that even one billion in almost any currency is more than I could reasonably spend.

No. It's rather his business practices, and that of his company, that I am opposed to, for a large and still growing number of reasons, most of which are plain, verifiable facts.
This paper explains why Microsoft is bad for us.

Disclaimer

The following represents my own personal point of view. It does not necessarily represent the points of view of my employers, clients, associates, friends, relatives, pets or houseplants. If they want a point of view, they can go and grow their own, so there. If you read this document and a tree falls on your house as a result of any inaccuracy on my part, I shall not be held responsible.

Abstract

Microsoft controls the current PC software market and has a de facto monopoly on the desktop. This monopoly has not been achieved and is not being maintained by offering the user community better products than Microsoft's competitors can offer. On the contrary, Microsoft has earned a reputation for selling unreliable products, thrown together from third-party technology, full of bugs and security holes, and in need of constant maintenance and repair. Windows is a technically inferior operating system with a seriously flawed architecture, a weak security model and sloppy code, while other Microsoft applications are equally kludgy. New Microsoft products are essentially re-wrapped bits of old technology which offer no essential improvements over previous or competing products, and with a Return On Investment between small and zero. In spite of this Microsoft boasts about about being innovative and customer-driven.

Instead of making better software, Microsoft has focused on using brilliant but doubtful marketing tactics to force their products upon the user community in order to establish and maintain their monopoly. These methods include a tight integration of applications into the operating system, the bundling of applications with Windows to force competing application vendors out of the market, the mandatory bundling of Windows with new computer equipment, deliberate limitations in the compatibility of their own software with competing products, contracts that keep third parties from doing business with anyone but Microsoft, and retaliatory practices against non-cooperating vendors. In addition to this, third-party developers are induced, through cheap or free development programs and the sabotaging of alternatives, to develop applications based upon proprietary methods of interfacing with the operating system. This results in third-party applications that are virtually non-portable, which in turn locks both developers and users into the MS-Windows platform.

These methods only serve to further inflate Microsoft's already obscene profit margins, at the price of the interests of the user community, the IT market and the field of computer technology as a whole.

1. From the people who brought you EDLIN

"640k should be enough for anyone."
-- Attr. to Bill Gates, Microsoft CEO, 1981

In 1975 Bill Gates and Paul Allen, who were students at Harvard University at the time, adapted BASIC to run on the popular Altair 8800 computer and sold it to the Altair's manufacturer, MITS. Although BASIC had been developed by Kemeny and Kurtz in 1963, the Altair BASIC interpreter was the first "high language" program to run on the type of computer that would later become known as the microcomputer or home computer. While the BASIC programming language itself was already in the public domain by then, there was no interpreter that could run it on the first microcomputers, and the small microprocessor systems typically developed by hobbyists and researchers were still being programmed in machine code and often operated via switches.

Gates and Allen, ca. 1968

Thus Gates and Allen could be said to have created an original product. One might even call it a true innovation.

It would be one of their last.

Developing BASIC on borrowed time

Gates and Allen initially met at Lakeside School (an exclusive private school for rich boys) where Gates became an adept at BASIC on a General Electric Mark II computer. Shortly thereafter they got access to a PDP-10 run by a private company in Seattle. The company offered free time to the Lakeside school kids to see if they could crash the system. Gates proved to be particularly good at doing so. When the free time ran out, Gates and Allen figured out how to continue using the PDP-10 by logging on as the system operator. About a year later the company that owned the PDP-10 went bankrupt.

This left Gates and Allen without a source of unpaid computing resources. Therefore Allen went over to the University of Washington and began using a Xerox computer by pretending to be a graduate student. Gates soon followed, and this went on until they were caught and removed from the campus. They continued to break into university and privately owned computer systems until about 1975. By that time Gates was a student at Harvard University, and HP had been selling the 9830 calculator (an expensive system for scientific and industrial math applications) for three years. The 9830 had a BASIC interpreter, which opened up a whole new range of applications outside the field of mathematical calculation. Whether or not Gates and Allen had actually seen a 9830 before they coded up their BASIC interpreter for the Altair (with the help of Monte Davidoff, who wrote the floating point arithmetic routines) is not known, but it is quite possible.

In any case, the BASIC that Gates sold to MITS had been developed and tested on a PDP-10 computer owned by Harvard, using an 8080-emulation program that Allen had adapted from earlier code. In fact, by the time Gates contacted MITS to announce their product, it had never seen an actual 8080 CPU. The demonstration that Allen put up for MITS in New Mexico was the first time the product actually ran on the system it was intended for. Gates sold it by announcing a product that didn't exist, developing it on the model of the best version available elsewhere, not testing it very seriously, demonstrating an edition that didn't fully work, and finally releasing the product in rather buggy form after a lengthy delay. From then on this modus operandi became Microsoft's trademark.

The controversy begins

After Gates sold the 8800 BASIC interpreter to MITS he left Harvard University, and went into business for himself with Allen as a partner. Allen was also an MITS employee at the time, which made his position somewhat questionable.

Gates' departure from Harvard appears to be somewhat controversial. Some say he dropped out, others say he was expelled for stealing computer time. Whatever the case may be, the fact is that Gates did most of the work on his BASIC version in a Harvard computer lab without having been authorized to use the computing resources for the project. The legal aspects are murky, as Harvard did not have a written policy regarding the use of their DARPA-funded PDP-10 computer. So perhaps Gates did not exactly steal unauthorized computer capacity (a limited and valuable commodity in those days) to develop his first commercially successful product. Yet he has never offered a different explanation. He did however send his now-infamous "Open Letter To Hobbyists" to every major computer publication in February 1976, in which he decried the copying of commerical software (especially Altair BASIC) by home computer hobbyists as simple theft. He also claimed to have spent $40,000 in computer time developing BASIC, but neglected to mention where that computer time came from and whose money it was that he spent.

Be that as it may, Gates was brilliant enough at the time to realize that he was sitting on a goldmine. While MITS demanded, and got, the exclusive rights to the software, Gates insisted on a clause in the contract where MITS agreed to "commercialize the product". These "best efforts" never panned out and Microsoft's income began to dry up. In 1977 Gates and Allen sent a letter of protest to MITS, whereupon MITS got a judge to restrain Microsoft from disclosing 8080 BASIC code to any third party. Microsoft was saved from bankruptcy only by payments for the 6502 BASIC from Apple Computer. (MITS only had the rights to 8080 BASIC, so Microsoft was allowed to port it to other CPU architectures and sell it all over again.) Then Microsoft sued their first customer MITS over the exclusive rights on 8080 BASIC, and won. They immediately went on to sell BASIC over and over again, to any other hardware manufacturer who would have it, from Commodore in Europe to Radio Shack in the US. Thus Gates' vision was one of the important factors in the creation of the home computer market of the late 1970's and early 1980's. (The other was the increasing availability of affordable VLSI computer chips.)

Microsoft ~~develops~~ buys MS-DOS

It went more or less the same when IBM came to Microsoft in 1980, for an operating system (OS) for their "Project Chess" which we now know as the development of IBM's new Personal Computer. Microsoft was still a small-scale operation in those days, making mostly software for the hobby and home computer market, and a few computer language products. IBM had another preferred supplier at the time: they went to Digital Research to discuss their needs for an OS for their upcoming PC. Gary Kildall, the founder of Digital Research, was the author of CP/M, the first operating system for microcomputers. This made Digital Research, and not Microsoft, the logical first choice for IBM, as CP/M would have fit their requirements. However, common lore has it that Kildall wasn't in Pebble Beach the day the IBM representatives arrived there for their appointment, and his wife and lawyer wouldn't sign the non-disclosure agreement until Kildall had returned. (That mistake has gone on record as perhaps the most capital blunder in the entire history of the PC industry.)

This, and time restrictions, led up to IBM's visit to Gates and his friends who, as rumor has it, were in the picture only because Gates' mother happened to know someone at IBM. This last detail may or may not be true; in any case it's a fact that Microsoft was a small company without management, without much administration or bookkeeping, with employees who slept on the floor behind their keyboards, and with a corporate culture based on shouting matches that were usually won by Gates. Microsoft had only worked on home computer software and programming languages at the time, and was not a supplier of operating systems or other system software. (Kildall himself has later added to this story that he did manage to contact the IBM representatives upon his return, discussed the deal with them, and was left with the impression that he had an agreement with IBM. Shortly thereafter he learned that IBM had signed contracts with Microsoft. This may or may not be true, but in any case it's hardly relevant here.)

When the IBM representatives showed up on his doorstep, Gates recognized this lucky break for what it was, and promised them an OS. Because he didn't have one and couldn't make one (at least not well enough and quickly enough) he bought the rights to a CP/M clone named QDOS from Seattle Computing Products, and filed off the serial numbers. Again Gates demonstrated his commercial genius at that point. He realized that although the PC was far from superior from a technological standpoint, IBM's position as a hardware manufacturer would go a long way to unifying the personal computer market, which had always been rather fragmented. Gates saw visions of minor investments resulting in huge sales figures. Innovation did not come into it at all; at the time the world's buildings, bridges and aeroplanes were mostly developed on VAX and Unix workstations.

So when IBM demanded exclusive rights to PC-DOS, Gates was adamant: IBM was prohibited from licensing Microsoft's software to third parties, but Microsoft itself was free to do so. Microsoft would sell MS-DOS to all interested clone manufacturers, just as they did with BASIC when MITS lost their exclusive rights. Thereby Gates created most of the basis for the PC market as we know it today.

A changing market

This is Microsoft's contribution to the field of computer technology: before they sold BASIC and later DOS to any hardware vendor who would buy it, end users were completely dependent on a hardware manufacturer not only for hardware, but also for platform-specific operating systems and application software. Microsoft's marketing strategy put an end to that, and contributed to changing the vertical computer market into a horizontal one. For that the company deserves due credit.

But that is all. Microsoft applied the right leverage at the right time, and the market's natural inertia did the rest. The IBM PC happened to be based on an Intel 8086 processor. The CP/M-descended products now sold by Microsoft standardized on the 80x86 processor architecture and weren't portable to other platforms. That in turn caused Intel to continue the 80x86-based architecture. This symbiotic relationship popularly known as Wintel still continues today.

The demise of innovation

While Microsoft was the first to market (but not create) a more-or-less functional operating system for the IBM-PC platform, the company has never originated any significant technological improvement since Altair BASIC. At best they've modified and adapted existing technology, but nothing original or particularly innovative has been created by Microsoft ever since. The first version of PC-DOS (later MS-DOS) was little more than a revamped version of QDOS (or DOS-86), the code for which they bought from Seattle Computing Products (SCP). QDOS, which stands for "Quick & Dirty Operating System", was the work of Tim Paterson, a friend of Paul Allen's. It was derived from CP/M, to such an extent that it has been called pirated. Most specifically, it had almost exactly the same basic interface layers and the same function set, and even used identical function call numbers. It differed from CP/M in only one significant aspect: it used a different way to store files on a disk. The first version of this file system, known as FAT (File Allocation Table) had been developed by Bill Gates and Marc McDonald in 1977 as part of a disk based version of Microsoft BASIC which, ironically, was in fact a remote descendant of Altair BASIC.

After the initial release of MS-DOS numerous features, including suspiciously Unix-like but rather broken support for subdirectories, I/O redirection and pipelines, were hacked into subsequent versions. This resulted in two or more incompatible versions of many system calls in the DOS kernel, and MS-DOS programmers could never agree on basic things like what character to use as an option switch or whether to be case-sensitive, or what file descriptor to use. Not much has changed in the twenty years that followed. When WIndows ME came out in 2000, all you had to do was look under the hood and the QDOS and CP/M legacies from elder days would stare you in the face.

As an interesting aside, while Gary Kildall had worked on CP/M for years, Tim Paterson of SCP compiled QDOS in under 6 weeks. He left SCP in 1981 and joined his friend Paul Allen at Microsoft. Later Kildall allegedly went to IBM and pointed out where his own copyright statement was still embedded in PC-DOS, but he did not dare fight it out with the full force of IBM's legal division. To forestall legal action, IBM offered consumers the choice of either CP/M or MS-DOS. But at $240, six times the price of MS-DOS, CP/M was quickly headed for extinction.

Kildall's allegations of theft by SCP, and the fact that the differences between QDOS and CP/M are minute at best, can't have escaped Microsoft's attention at the time. This leads to the interesting conclusion that if this is true, then Microsoft and IBM knowingly acted as fences, and Microsoft founded a global empire on a crime.

The market lock-in

Microsoft team, 1978

In any case MS-DOS thrived. It remained the only PC operating system on the market for years, in spite of the fact that it was rather restrictive. In fact the restrictions it imposed upon the application developers prolonged its success: few developers were really happy with it, but they were stuck with it. MS-DOS offered way too little functionality, so that application builders were forced to make their application code carry out tasks that should have been performed by the OS. Some early applications (such as Lotus-123 and, more commonly, computer games) bypassed DOS entirely. In other products most peripheral access, video and printer communications (I/O) had to be done by having the application access the hardware directly in order to get a decent performance. Users had to know (and remember) the 's I/O port addresses, IRQ numbers and DMA channel settings for each hardware component when installing and configuring applications.

This lack of proper OS functions in MS-DOS resulted in application software less portable than the Rocky Mountains, which effectively forced software developers to stick with the MS-DOS platform in order to maintain their applications and protect their investments. DOS itself was non-portable as well, being largely written in Assembly language and containing a lot of low-level code and little structure. I've personally seen the DOS 6 source code. It's not a pretty sight.

Gates: don't develop, copy

By the time PC-DOS took hold, Gates had already shown that Microsoft's future would hold very little innovation indeed. Gates' views on development are probably best illustrated by the following:

> *From: 'Programmers at work', Microsoft Press, Redmond, WA [1986]:*
> *Interviewer: "Is studying computer science the best way to prepare to be a programmer?"*
> *Gates: "No, the best way to prepare is to write programs, and to study great programs that other people have written. In my case, I went to the garbage cans at the Computer Science Center and I fished out listings of their operating system."*

Seldom have both Microsoft's lack of innovation and their kludgy, ad-hoc approach to software design been explained so concisely. It's also interesting to note that while many people have called Microsoft products copycat, trash or garbage, most of them probably had no idea how close to the truth they really were.

Indeed MS-DOS has seen little innovation in the two decades or so when it dominated the PC market. The most important improvement in DOS 2.0 was the addition of subdirectories and device drivers, ideas that were borrowed from Unix. Later versions came with a few extra functions in the kernel, and they boasted more tools and utility programs, initially written by Microsoft but later bought from third parties. Except for the additions in DOS 2 (subdirectories, device drivers) and DOS 5 (extended and enhanced memory management on 80286 and 80386 CPU's based on technology from Quarterdeck) DOS itself has only seen minor development. In the meantime Microsoft briefly sold Xenix (a rather unimpressive Unix port for the PC, which they bought outright from SCO) but when it failed to sell in huge volumes they soon lost interest and concentrated on DOS.

The Windows coup

When Windows came into existence, Microsoft had been collaborating with IBM on OS/2 1.x for some time. This collaboration sprung from the insight that with the advent of the 80286 CPU and Intel's plans for the 80386, DOS had become obsolete. IBM worked mainly on the OS/2 kernel, which in its first incarnation was basically a 16-bit successor to DOS with a command line interface. Microsoft concentrated on the Graphic User Interface (GUI).

The idea for a Graphic User Interface was neither new nor original. Years before, Xerox had demonstrated a mouse-controlled GUI in their Palo Alto Research Center. This demonstration featured the Alto computer, which in 1973 sported a GUI, a mouse, graphic WYSIWYG technology and an Ethernet network interface. The demo was attended by Steve Jobs (Apple) and Bill Gates, among others. Jobs saw the possibilities of the GUI and went on to implement the idea into Apple's OS and application software, while Gates decided to stay with the text-based user interface. Later Gates was forced to revise his opinion about the GUI when it turned out to be successful on the Apple platform. Thus it was decided that OS/2 would have a GUI.

Soon Microsoft's code began to diverge from IBM's (especially from Presentation Manager, IBM's GUI component of OS/2) and became increasingly incompatible with it. Meanwhile Gary Kildall of Digital Research had already released the first version of GEM, a Graphic Environment Manager for DOS. In order to sabotage this, Microsoft announced that they were working on their own, much better, graphic environment. Eventually they took the GUI portion of what should have become OS/2 and sold it as a separate DOS product called MS-Windows. In its initial form it was mainly text based and hardly useful, but they claimed to work on it in preparation for the upcoming OS/2. In the meantime, application developers (e.g. Word Perfect Corp., Ashton-Tate and Lotus) spent huge R&D budgets on rewriting their applications for OS/2, assuming that the IBM/Microsoft partnership would deliver as promised.

MS-Windows could have been a new start, but (mainly for strategic and marketing reasons) it wasn't. It tightly clung to the mistakes of the past, being based upon the underlying MS-DOS architecture for basic OS functions such as file system access. It added a simple cooperative multitasker to MS-DOS, in a manner strangely like that of DesqView (a multitasker for DOS that had been available from Quarterdeck for years). It also sported a GUI that was so close to the one used by Apple that it kept lawyers occupied for over half a decade. But as far as innovation was concerned, that was it.

Initial versions of Windows were very bad, but Microsoft kept promising that a better product would come out Real Soon Now, still as part of their joint OS/2 efforts with IBM. Until one day, that is, when suddenly they turned their backs on OS/2. They cried "innovation" and went back to DOS in spite of earlier having admitted it to be obsolete. Then they went and dropped out of the collaboration with IBM entirely, taking with them a lot of IBM technology that had ended up in Windows, which they now suddenly positioned as the operating system of the future. They never even mentioned their earlier promises about OS/2 again.

Hijacking the applications market

Microsoft already sold applications for the Apple Macintosh. This gave them a good look under the hood of Apple's operating system software, and enabled them to muscle Apple into granting them a license for portions of the MacUI. (They threatened to withdraw all Mac applications, unless Apple would grant them a license to use MacUI code to port Macintosh apps to the PC.) They then raided MacUI for extra ideas. The remaining few bits (e.g. the font technology they later called TrueType) they bought, occasionally bartering vaporware that later failed to materialize. They also threw in a random collection of small applications, completely unrelated to an operating system (e.g. Paintbrush) which they had bought from various sources to flesh things out a bit. The resulting mixed bag of bits and pieces was massaged into an end product and released as Windows 3.0.

It was not too difficult for Microsoft to adapt the Apple versions of Word and Excel to run on Windows 3. There is some indication that Windows was adapted to Word and Excel as much as Word and Excel were adapted to Windows. By the time Windows 3.0 hit the market, competing application developers had already put their R&D money into OS/2 versions of their products, on the assumption that OS/2 would be delivered as promised by the IBM/Microsoft partnership. And now OS/2 did not materialize. But a blown R&D budget was only half the problem. Even if most of the application manufacturers had been wealthy enough to fund two separate development efforts to upgrade their DOS products, there was not enough time to do the Windows version before Windows' projected release date. The fact that the Windows API had not been published in any permanent form yet didn't help either. Without a good Application Program Interface (API) specification, an application developer is not able to interface with the operating system or with other software products. This essentially prevents application development. And Microsoft was the only application vendor at the time who knew enough about the Windows API to come up with a market-ready product.

So Microsoft shipped both an OS and an application suite, several months before their competitors in the applications market had a chance to catch up with Microsoft's last-moment switch to Windows. And that was that. All those who had expected to sail with the IBM/Microsoft alliance missed the boat, when Microsoft suddenly and deliberately decided to cast off earlier and in another direction than they had originally promised. Most of the independent application vendors never recovered.

The demise of OS/2

IBM eventually went on to release their own version of OS/2, and botched it completely. This is partially due to the fact that by the time OS/2 hit the market, that market had already been taken away from them by Microsoft, especially because most application developers had committed themselves to Windows by then. They used Windows development tools, so their code had become extremely hard to port to another OS. Native OS/2 application software remained scarce, and hardware support was even a bigger problem.

Even so, IBM remains responsible for much of the demise of OS/2. Although it had an infinitely better architecture than Windows, OS/2 was killed off by some of the worst strategic and marketing decisions in the history of the industry. Its brief and unhappy existence was marked by a lack of drivers and hardware support, a lack of development tools, and a lack of applications. In typical IBM fashion the end user was expected to manually edit a 4-page CONFIG.SYS file to configure the system. Partnerships with hardware vendors to ship OS/2 with systems that didn't have the power to run it properly made the problem even worse. Lack of good marketing drove the final nail into OS/2's coffin.

After this debacle IBM withdrew from the desktop software market which they had never really understood, in spite of having created the original IBM PC.

The non-innovation continues

Creating a better software platform would have been a real innovation, but that would have meant to abandon DOS, which was all that Microsoft had at the time. Since DOS applications were practically non-portable, a new and better OS would have broken the ties that bound developers (and therefore users) to Microsoft. In order to maintain their market share, Microsoft chose not to innovate. So for reasons of marketing, Windows 3.x ran on top of DOS as little more than a hybrid multitasking shell.

The Windows 95 architecture was merely a continuation of Microsoft's uninnovative strategy. When Windows 95 was released no less than three years later (Windows 93 was planned but never made it) it still turned out to be a disappointing rehashed DOS-based product. It still ran on top of DOS as an application-level shell, although DOS and Windows were now installed from a single bundle rather than as separate products. Basically Windows 95 was nothing but plain old Windows 3.x with a new GUI and a souped-up memory manager, and the formerly separate DOS code integrated in the bundle. This did not stop Microsoft from marketing it as a completely new 32-bit OS, which of course it wasn't. Granted, portions of the code were 32-bit, but there was still a lot of 16-bit code running under the hood, and memory protection was partially functional at best. Windows 95 and its successors still relied heavily on obsolete DOS code. Windows 98 (Windows '97 was planned but again never made it) was not a significant improvement in this respect either, and Windows ME was just more of the same tired old stuff, plus a lot of new bugs. It was still DOS-based, although Microsoft had gone to great pains to hide that fact, through many cosmetic changes and the bundling of application software with the OS. Basically there's nothing new to the whole Windows 95/98/ME product line; most design flaws from previous Windows versions right back to 3.0 are still present, and many new flaws have been introduced. When you get right down to it, Windows ME wasn't much more than the repackaged Windows 3.x descendant that Windows 95 was, full of architectural ineptitude and based upon technology that has been obsolete for decades, with a lot of extra bells and whistles thrown in to confuse the issue.

None of this has stopped Microsoft from presenting all these minor updates as new products and pushing them as recommended upgrades.

"Windows [n.] - A thirty-two bit extension and GUI shell to a sixteen bit patch to an eight bit operating system originally coded for a four bit microprocessor and sold by a two-bit company that can't stand one bit of competition."

(Anonymous USEnet post)

NT: Not-so-new Technology

Windows NT finally appeared to be a step in the right direction. At least the NT product line (which includes Windows 2000, XP and Vista) is the better one. 'NT' stands for 'New Technology', presumably because Windows NT is one of the few keystone products in the history of Microsoft that they didn't buy outright. Instead they hired David Cutler, who had played an important role in the development of VMS at DEC (Digital Equipment Corporation). VMS was a successful and innovative industrial OS in its days, and Digital had been working on it since the 1970's. Cutler took some 20 former Digital employees with him, and he and his team began the development of NT. The project eventually involved hundreds of other coders and testers, but Cutler and his core team of VMS engineers provided most of the know-how that went into NT's kernel code.

As a result, many design principles found in the VMS kernel ended up in Windows NT. (The number and splitting of priority levels in the scheduler, the use of demand-paged virtual memory and the layered driver model are only a few examples of many, many similarities.) The first version of VMS was released in 1977. Without trivializing the efforts of Cutler and his team (they did a lot of work on the project) one has to wonder what Microsoft really means with "New Technology". To illustrate, in a little known out-of-court settlement Microsoft paid DEC $150 million in compensation for using portions of old Digital OS code in Windows NT.

Ehm... New Technology...??

Marketing prevails over engineering

Even though its roots go back to the 1970's, the Windows NT product line is a big improvement over Microsoft's DOS-based products. Unfortunately that doesn't automatically mean that it's a well-designed operating system.

Cutler's team had to operate within Microsoft's additional design restrictions, and the result was a tradeoff. Cutler took a number of design principles from VMS, which was good. They expanded on that, so in a way NT can be said to contain at least some "New Technology" and perhaps Cutler's work even represented (dare I say it?) some innovation, in that it brought robust design principles to the IBM PC platform. Had that been all, the end result could have been a good, efficient and robust OS. But Gates needed a vehicle that would further Microsoft's marketing strategies, rather than a robust OS. And of course much of the eventual coding on NT was done by Microsoft engineers, so in the end the quality of NT's final code wasn't even in the same league as VMS.

VMS was an industrial-strength operating system with native clustering, but NT was to be a single-user desktop operating system. Account and data management were rudimentary; the user home directory resided on the workstation's local harddisk, under the subdirectory that held the bulk of the operating system code. Applications and user settings were system-based rather than account-based. Separation between OS code, user settings, application code and configuration data became all but impossible; application and GUI settings were stored along with vital operating system information in an insecure central registry that was also system-based. Therefore network-based user accounts could only be implemented with complex and cumbersome workarounds. One of the biggest design mistakes in the history of Windows (the design of the DLL subsystem) was perpetuated, and networking was initially based on the hopelessly inadequate NetBEUI protocol. Even though NT followed a peer-to-peer networking model, a separate "NT Server" version was shipped. (NT Server contained exactly the same code as NT Workstation, with a few additions that amount to only a fraction of the product's total code set.) Initially there had been intentions of portability to non-Intel hardware, the incorporation of a Hardware Abstraction Layer, and versions of Windows NT on Digital and other platforms, but as the market became more and more monolithic these good intentions fell by the wayside. Eventually Digital did the same.

So at the end of the day Microsoft's marketing prevailed over Cutler's engineering. The result wasn't pretty. NT became an OS based on a set of old VMS design principles that were made compatible with everything that Microsoft had ever done wrong. It was full of legacy API's, it was kludged up to run applications written for OS/2 1.0 (but not very well), it paid lip service to POSIX but never offered anything more than fractional POSIX compliance, and it sported a Windows 3 GUI that had its roots in both Apple's and IBM's user interfaces. It even contained the entire Windows 3 kernel and the bulk of its accompanying code (and Windows XP still does) in the original 16-bit executables, as well as the complete set of decades-old DOS code. In short, it was a real Microsoft product. All later versions of Windows that descended from this piece of "New Technology", right up to Windows Vista, suffer from this legacy.

Sic transit gloria Fenestrae.

Consolidation rather than innovation

It's rather ironic that Microsoft prides itself on their "innovative role" in the IT market. The sad truth is that Microsoft has rarely been an innovator. They purchased a CP/M ripoff and named it MS-DOS, and they cobbled Windows together from various bits and pieces that they bought, stole or borrowed. The graphic user interface for Windows was based on IBM know-how and the user interface of the Apple Macintosh, which was in turn derived from technology developed by Xerox ages ago. NT was based on good but old VAX VMS design principles. In short, all Microsoft OS products only implement features and ideas that have been around for as much as a quarter of a century.

Later versions of Windows contain no significant improvement over previous versions. Windows 98, ME, 2000, XP and Vista are in fact 'point releases'; they're relatively minor updates that contain mostly fixes, new bugs, and a few small extras that used to be sold separately but are now bundled into the package. For example: Windows XP comes with application software for scanners and digital cameras, or the "remote desktop" feature that was formerly sold separately by Citrix. The rest is little but cosmetics. The whole product line remains riddled with serious design flaws, kludgy code to work around those flaws, and tons of bugs. There's been little reason to switch from Windows 95 to 98 (except perhaps the discontinuation of support and maintenance on '95) and none at all to switch to ME. Windows 2000, XP and Vista contain mostly bug fixes and work-arounds. Neither 2000 nor XP or Vista offered a proper Return On Investment to users of previous versions, and there's little or no demand for any of the extras that come with

them. In fact in August 2005 a significant percentage of Windows services was still based on NT4, while Windows 2000 was still the most common version on the desktop. Especially the latter was interesting, as Microsoft had discontinued support for this version by then. Windows XP was around for about five years and by the end of that period had become the most common version, but few XP users have found reason to upgrade to Vista.
Nevertheless Bill Gates called Windows XP "a very big thing" and Steve Ballmer said that "Windows XP is a more significant advance forward than anything since Windows 3.0". Microsoft's rhetoric on Vista was even more unrealistic.

A better Windows? Or just better marketing?

Windows XP was the next version of the Windows NT/2000 product line, but it was marketed as a replacement for Window 9x/ME. By default it sports a seriously dumbed-down user interface. This insulting toy box, apparently aimed at users aged 1 - 4 and technophobes who are scared off even by Macintosh desktops, can fortunately be overridden but is always installed by default. Under the hood XP has a Windows-2000 kernel. There are a few slight improvements to the kernel code, but nothing dramatic. Of course there is also a lot of additional application software bundled with it, especially third-party multimedia products that MS bought and re-branded.

XP's release was timed to coincide with the discontinuation of the 9x/ME line, as part of Microsoft's repositioning of their Windows product lines. Through this admittedly clever marketing trick, end users were encouraged to compare XP with Windows 9x/ME and think of it as a new product, which was of course rather misleading. XP was just an overpriced point update of Windows 2000 and nothing more.

Incidentally, 'XP' stands for 'eXPerience'. Apparently Microsoft thinks we need a new 'experience' with our operating systems and applications, and that we sit at our computers expecting to be entertained by OS features and a spreadsheet or two. And indeed most of the 'improvements' in XP are on the presentation level. If you look in some executables in the Windows directory, you find internal labels like "ProductName: Microsoft Windows (TM) operating system, ProductVersion: 3.10". There's even DOS 5.0 code with a 1981-1991 copyright date. What a great new product. Of course it makes sense to provide compatibility modes for old Windows applications, but to find the bulk of Windows 3.10 and DOS 5 (all of it 16-bit code) up to and including EDLIN, installed under the hood of Windows XP makes you wonder about the design principles that have gone into each "new" version of Windows.

Microsoft released XP on a marketing budget of half a billion dollars to promote it. None of the new cosmetic bells and whistles in XP actually made it any more stable than Windows 2000 was, but that hasn't stopped Microsoft from marketing XP as the OS that "keeps on running" instead of crashing, and that protects the users from viruses. How's that again? Vista was also marketed as a "multi-media operating system" in spite of the fact that there's nothing multi-media about the OS itself. It comes bundled with a few applications for digital photos and video (which Microsoft bought and put into the box) and of course with Microsoft's own MediaPlayer application, but that has nothing to do with the operating system itself.

XP's successor, Vista, turned out to be similarly long on empty marketing rhetoric and short on innovation. According to Microsoft Vista will "Bring Clarity To Your World!" and "enhance your confidence in PC technology and give you a new outlook on the digital world around you". Vista was also said to "help you to organize information intuitively and to stay in touch with information, people and resources, so you can enjoy life more!" The truth is of course that this new version of Windows is mostly an 'XP 2nd edition' release. No surprises there. Vista has the usual many small improvements, several of which have something to do with security, and some attempts to work around Windows' most gaping shortcomings. None of these really address any real design flaws, except perhaps the improved access privileges. Vista also has a heavy dose of features related to Digital Rights Management (DRM) and more options for the integration of (and dependency on) Internet based services. There's a lot of extra gadgetry in the user interface and on the application level. Some menus and features have been restructured a bit more conveniently (for example wireless networks are now grouped with the other network setup options and no longer separate) and everything looks very slick.

There are no significant, major or structural improvements in Vista to justify an expensive upgrade, though. Many of the announced features have failed to materialize, and what's left is mostly a new search facility, a few extra features for laptop computers, a parental control feature, several features for remote access, and a downright bizarre set of hardware requirements. But in Vista the windows now have rounded corners, semi-transparent backgrounds, and zooming and fading effects! Oh yes! Vista also has shiny glassy buttons, a sidebar with a calendar and a photo slideshow, all of which is strangely reminescent of Apple's UI design. Buttons and icons now can show a tiny representation of a window or document. And of course the system folder icons now show a three-dimensional representation of a folder, standing vertically on a horizontal surface, complete with shadow effects! How... innovative.

Applications: more of the same

In the application market things aren't much better. MS Word isn't quite the word processor that Word Perfect was, a fact that MS attempted to gloss over by adding functions that really belong to desktop publishing software (but cannot replace it for serious applications). As a result, Word lacks many features that users would like to have (such as the option to view markup codes) but at the same time it has become so loaded with other features that its complexity is actually counter-productive. Excel, originally developed on the Apple platform, doesn't really do anything that Lotus-123 couldn't do in the nineteen eighties (although it has a fancier user interface and more graphic capabilities) and comes loaded with macro bugs and version problems to boot. Microsoft Access is something halfway between a 'flat' database and a relational database system, combining the advantages of neither with the disadvantages of both. The first line in the 'About' window in Internet Explorer says "Based on NCSA Mosaic" (which was the very first web browser to be used during the Stone Age of the World Wide Web) and PowerPoint merely duplicates the functionality that other presentation packages already offered in the late nineteen eighties. (Unless of course you count the Visual BASIC hooks that virus authors and hackers are having such a ball with.)

In fact, none of these products use any significant technology invented by Microsoft. Sure, they're all dressed up like maypoles with tons of gadgetry and flashy colors, and the implementation of the old technology has become more streamlined, especially when it comes to exchanging data between applications. They've been ported to Windows so their user interfaces have a uniform look-and-feel (but are still inconsistent) and IBM's data exchange techniques such as OLE give the impression of integration. But in fact it's all old technology. This isn't innovation; it's recycling. To illustrate: several of the files that came with Word 97 (and perhaps with later versions as well) still contained the text "Copyright WordPerfect Corporation 1994. All rights reserved." I rest my case.

ASP: old technology rewrapped

Microsoft's future plans are full of the same kind of "innovation". Their long-term strategy involves client systems that will be used to access server-based or network-based applications and services. This idea is known as ASP (Application Service Providing). It moves applications from the workstation to a central server, and does away with the need to install, maintain and run application software locally on workstations.

Of course Microsoft claims that this approach is innovative. In truth there's very little innovative about it. Essentially it's a step back to the decades-old host-with-terminals approach. Microsoft will almost certainly be able to rewrap it in a more attractive package, but that's as far as their innovation is likely to go. All you need to offer network-based applications and services today (as well as twenty years ago) is a server (which would typically run Unix) with a bunch of applications and some graphic terminals. Granted, the X protocol (the most popular graphic terminal standard on Unix systems) is more than a little ugly and unsuited for anything but LAN's. However, the implementation of a more elegant and efficient client/server protocol layer (e.g. ICA or something similar) would be rather trivial. At that point all that Microsoft's developers need to do is to recode their system and application products so that resources are used efficiently (as they should have done in the first place) and move the applications back to the server where they originated decades ago. Given the current sorry state of affairs on the Windows platform, that might even be an improvement... but not innovative.

Innovation? What innovation?

The machine in Redmond lumbers on. More gadgets, more flashy colors, more overhead, more old stuff with a new paint job, all marketed as new technology which they claim to have personally invented from scratch. They dress up their "technological innovations" with flashy names like Single Instance Store, to disguise the fact that Single Instance Store is nothing but a slightly souped-up version of the symbolic links that have been around on Unix systems for about three decades. Another "innovation" is the addition of the Narrator text-to-speech converter as an aid for the visually impaired. A useful feature, granted... but innovative? We've had commercial text-to-speech conversion since the early nineteen eighties. Even most of the cosmetic changes in Windows Vista were "inspired" by Apple's desktops, and Internet Explorer 7 was mostly an attempt to copy some of the most popular features from Mozilla Firefox.

Microsoft apparently thinks that R&D stands for 'Rewrap & Disguise'. A baroque excess of features presents itself to the user, mainly to hide the fact that the software contains nothing that rightly could be called innovative. In spite of a marketing budget of some five billion dollars a year, the best Microsoft has managed to do is repackage various ideas as their own, list TCP/IP under 'Microsoft protocols' in Windows, tout that they've "assisted with IPv6" (they did what, exactly?) and of course they came up with an animated paper clip. Windows hasn't added one basic service to the PC that wasn't available on, say, a Sun workstation in 1990. Yes, hardware has become cheaper, smaller, faster and more powerful (just like all other electronics on the market) so today's PCs look much better than those old workstations. But basically no new technology has been invented by Microsoft that really adds new capabilities to a personal computer.

Microsoft Research, in spite of an astronomic budget, hasn't come up with any truly useful technology so far. Name one, just one, major piece of useful technology that's ostensibly been invented or developed by Microsoft. One single original concept, that's all I ask. Name it, and I'll tell you where they got it from.

Innovation? Yeah, right.

2. The not-so-good, the bad and the ugly

"... it is easy to be blinded to the essential uselessness of them by the sense of achievement you get from getting them to work at all. In other words ... their fundamental design flaws are completely hidden by their superficial design flaws."

-- The Hitchhiker's Guide to the Galaxy, on the products of the Sirius Cybernetics Corporation.

Let's be honest: there's no such thing as bug-free software. Initial versions of programs may occasionally crash, fail to de-allocate memory, or encounter untested conditions. Developers may overlook security holes, users may do things nobody thought of, and not all systems are identical. Software developers are only human, and they make mistakes now and then. It happens. But of all major software vendors Microsoft has the worst record by far when it comes to the quality of their products in general.
Outlining the battle field

Microsoft boasts a rather extensive product range, but in fact there's less here than meets the eye. Microsoft has forever been selling essentially the same software over and over again, in a variety of colorful new wrappers.

Microsoft products can be divided into three categories: applications, operating systems, and additional server products. The applications include the Microsoft Office suite, but also Internet Explorer, Media Player, Visio, Frontpage, etc. The operating systems involve desktop and server versions of Windows. On the desktop we find Windows 9x/ME, NT Workstation, Windows 2000, XP and Vista, and at the server end we have Windows NT Server, Windows 2003 Server and varieties such as Datacenter. The additional server products, e.g. Internet Information Server (IIS) and SQL Server, run on top of one of the Windows server products. They add services (e.g. webserver or database server functions) to the basic file, print and authentication services that the Windows server platform provides.

Two different Windows families

Windows on the desktop comes in two flavors: the Windows 9x/ME product line, and the Windows NT/2000/XP/Vista product line. The different versions within one product line are made to look a bit different, but the difference is in the details only; they are essentially the same. Windows '95, '98 and ME are descended from DOS and Windows 3.x, and contain significant portions of old 16-bit legacy code. These Windows versions are essentially DOS-based, with 32-bit extensions. Process and resource management, memory protection and security were added as an afterthought and are rudimentary at best. This Windows product line is totally unsuited for applications where security and reliability are an issue. It is completely insecure, e.g. it may ask for a password but it won't mind if you don't supply one. There is no way to prevent the user or the applications from accessing and possibly corrupting the entire system (including the file system), and each user can alter the system's configuration, either by mistake or deliberately. The Windows 9x/ME line primarily targets consumers (although Windows '95 marketing was aimed at corporate users as well). Although this entire product line was retired upon the release of Windows XP, computers running Windows '98 or (to a lesser degree) Windows ME are still common.

The other Windows product line includes Windows NT, 2000, XP and Vista, and the server products. This Windows family is better than the 9x/ME line and at least runs new (i.e. post-DOS) 32-bit code. Memory protection, resource management and security are a bit more serious than in Windows 9x/ME, and they even have some support for access restrictions and a secure filesystem. That doesn't mean that this Windows family is anywhere near as reliable and secure as Redmond's marketeers claim, but compared to Windows 9x/ME its additional features at least have the advantage of being there at all. But even this Windows line contains a certain amount of 16-bit legacy code, and the entire 16-bit subsystem is a direct legacy from Microsoft's OS/2 days with IBM. In short, all 16-bit applications share one 16-bit subsystem (just as with OS/2). There's no internal memory protection, so one 16-bit application may crash all the others and the the entire 16-bit subsystem as well. This may create persistent locks from the crashed 16-bit code on 32-bit resources, and eventually bring Windows to a halt. Fortunately this isn't much of a problem anymore now that 16-bit applications have all but died out.

While Windows has seen a lot of development over the years, relatively little has really improved. The new features in new versions of Windows all show the same half-baked, patchy approach. For each fixed problem, at least one new problem is introduced (and often more than one). Windows XP for example comes loaded with more applications and features than ever before. While this may seem convenient at first sight, the included features aren't as good as those provided by external software. For example, XP insists on supporting DSL ("wideband Internet") networking, scanners and other peripherals with the built-in Microsoft code instead of requiring third-party code. So you end up with things like DSL networking that uses incorrect settings (and no convenient way to change that), scanner support that won't let you use your scanner's photocopy feature, or a digital camera interface that will let you download images from the camera but you can't use its webcam function. Wireless (WiFi) network cards are even more of a problem: where manufacturers could include their own drivers and client manager software in previous versions of Windows, users are now reduced to using XP's native WiFi support. Unfortunately XP's WiFi support is full of problems that may cause wireless PCs to lose their connection to the wireless access point with frustrating regularity. Also XP's native WiFi support lacks extra functions (such as advanced multiple-profile management) that manufacturers used to include in their client software.

Even basic services are affected. Windows 2000 and later have built-in DNS (Domain Name System) caching. DNS is the mechanism that resolves Internet host and domain names (e.g. www.microsoft.com) into the numerical IP addresses used by computers (e.g. 194.134.0.67). Windows' DNS caching basically remembers resolved hostnames for faster access and reduced DNS lookups. This would be a nice feature, if it weren't for the blunder that failed DNS lookups get cached by default as well. When a DNS lookup fails (due to temporary DNS problems) Windows caches the unsuccessful DNS query, and continues to fail to connect to a host regardless of the fact that the DNS server might be responding properly a few seconds later. And of course applications (such as Internet Explorer and Outlook) have been integrated in the operating system more tightly than ever before, and more (formerly separate) products have been bundled with the operating system.

Design flaws common to all Windows versions

All versions of Windows share a number of structural design flaws. Application installation procedures, user errors and runaway applications may easily corrupt the operating system beyond repair. Networking support is poorly implemented. Inefficient code leads to sub-standard performance, and both scalability and manageability leave a lot to be desired. (See also appendix A.) In fact, NT and its successors (or any version of Windows) are just not comparable to the functionality, robustness or performance that the UNIX community has been used to for decades. They may work well, or they may not. On one system Windows will run for weeks on end, on another it may crash quite frequently. I've attended trainings at a Microsoft Authorized Education Center, and I was told: "We are now going to install Windows on the servers. The installation will probably fail on one or two systems [They had ten identical systems in the classroom] but that always happens - we don't know why and neither does Microsoft." I repeat, this from a Microsoft Authorized Partner.

Be that as it may... Even without any installation problems or serious crashes (the kind that require restore operations or reinstallations) Windows doesn't do the job very well. Many users think it does, but they generally haven't experienced any alternatives. In fact Windows' unreliability has become commonplace and even proverbial. Tthe dreaded blue screen (popularly known as the Blue Screen of Death or BSOD for short) has featured in cartoons, screen savers and on t-shirts, it has appeared at airports and on buildings, and there has even been a Star Trek episode in which a malfunctioning space ship had to be switched off and back on in order to get it going.

Even if Windows stays up it leaves a lot to be desired. On an old-but-still-good desktop PC (e.g. a 450MHz PII CPU with 256MB RAM, something we could only dream of fifteen years ago) four or five simultaneous tasks are enough to tax Windows' multitasking capabilities to their limits, even with plenty of core memory available. Task switches will start to take forever, applications will stop responding simply because they're waiting for system resources to be released by other applications (which may have halted without releasing those resources), or kernel and library routines lock into some unknown wait condition. Soon the whole system locks up entirely or becomes all but unusable. In short, Windows' process management is as unimpressive as its memory protection and resource management are, and an operating system that may crash entirely when an application error occurs should not be sold as a serious multi-tasking environment. Granted, it does run several processes at once - but not very well. Recent versions of Windows (i.e. XP and Vista) are somewhat better in this respect and more stable than their predecessors, but not spectacularly so. Although they have been patched up to reduce the impact of some of the most serious problems, their multitasking still depends to a large degree on the application rather than on the operating system. That means that a single process may still paralize the entire system or a process can become impossible to terminate without using dynamite. The basic flaws in the OS architecture remain; a crashing application (e.g. a video player or a communications package) can still lock up the system, crash it into a BSOD or cause a sudden and spontaneous reboot.

Code separation, protection and sharing flaws

Windows is quite fragile, and the operating system can get corrupted quite easily. This happens most often during the installation of updates, service packs, drivers or application software, and the problem exists in all versions of Windows so far. The heart of the problem lies in the fact that Windows can't (or rather, is designed not to) separate application and operating system code and settings. Code gets mixed up when applications install portions of themselves between files that belong to the operating system, occasionally replacing them in the process. Settings are written to a central registry that also stores vital OS settings. The registry database is basically insecure, and settings that are vital to the OS or to other applications are easily corrupted.

Even more problems are caused by the limitations of Windows' DLL subsystem. A good multi-tasking and/or multi-user OS utilizes a principle called code sharing. Code sharing means that if an application is running n times at once, the code segment that contains the program code (which is called the static segment) is loaded into memory only once, to be used by n different processes which are therefore instances of the same application. Apparently Microsoft had heard about something called code sharing, but obviously didn't really understand the concept and the benefits, or they didn't bother with the whole idea. Whatever the reason, they went and used DLLs instead. DLL files contain Dynamic Link Libraries and are intended to contain library functions only. Windows doesn't share the static (code) segment - if you run 10 instances of Word, the bulk of the code will be loaded into memory 10 times. Only a fraction of the code, e.g. library functions, has been moved to DLLs and may be shared.

The main problem with DLL support is that the OS keeps track of DLLs by name only. There is no adequate signature system to keep track of different DLL versions. In other words, Windows cannot see the difference between one WHATSIT.DLL and another DLL with the same name, although they may contain entirely different code. Once a DLL in the Windows directory has been overwritten by another one, there's no way back. Also, the order in which applications are started (and DLLs are loaded) determines which DLL will become active, and how the system will eventually crash. There is no distinction between different versions of the same DLL, or between DLLs that come with Windows and those that come with application software. An application may put its own DLLs in the same directory as the Windows DLLs during installation, and may overwrite DLLs by the same name if they exist.

What it boils down to is that the application may add portions of itself to the operating system. (This is one of the reasons why Windows needs to be rebooted after an application has been installed or changed.) That means that the installation procedure introduces third-party code (read: uncertified code) into the operating system and into other applications that load the affected DLLs. Furthermore, because there is no real distinction between system level code and user level code, the software in DLLs that has been provided by application programmers or the user may now run at system level. This corrupts the integrity of the operating system and other applications. A rather effective demonstration was provided by Bill Gates himself who, during a Comdex presentation of the Windows 98 USB Plug-and-Play features, connected a scanner to a PC and caused it to crash into a Blue Screen. "Moving right along," said Gates, "I guess this is why we're not shipping it yet." Nice try, Mr. Gates, but of course the release versions of Windows '98 and ME were just as unstable, and in Windows 2000 and its sucessors new problems have been introduced. These versions of Windows use a firmware revision number to recognize devices, so an update of a peripheral's firmware may cause that device to be 'lost' to PnP.

Another, less harmful but most annoying, side-effect of code confusion is that different language versions of software may get mixed up. A foreign language version of an application may add to or partially overwrite Windows' list of dialog messages. This may cause a dialog window to prompt "Are you sure?" in English, followed by two buttons marked, say, "Da" and "Nyet".

Peripheral drivers also use a rather weak signature system and suffer from similar problems as DLL's, albeit to a lesser degree. For example, it's quite possible to replace a printer driver with a similar driver from another language version of Windows and mess up the page format as a result. Printer drivers from different language versions of Windows sometimes contain entirely different code that generates different printer output, but Windows is unaware of the difference. This problem has been addressed somewhat with the release of Windows 2000, but it's still far from perfect.

Mixing up OS and application code: why bundling is bad

Designing an OS to deliberately mix up system and application code fits Microsoft's strategy of product bundling and integration. The results are obvious: each file operation that involves executable code essentially puts the entire OS and its applications at risk, and application errors often mean OS errors (and crashes) as well. This leads to ridiculous "issues" such as Outlook Express crashing Windows if it's a "high encryption" version with the locale set to France. Replying to an e-mail message may crash the entire system, a problem which has been traced to one of the DLLs that came with Outlook. (Are you still with me?)

In a well-designed and robustly coded OS something like this could never happen. The first design criterion for any OS is that the system, the applications, and (in a multi-user environment) the users all be separated and protected from each other. Not only does no version of Windows do that by default, it actively prevents you from setting things up that way. The DLL fiasco is just the tip of the iceberg. You can't maintain or adequately restore OS integrity, you can't maintain version control, and you can't prevent applications and users from interfering with each other and the system, either by accident or on purpose.

Integrating applications into the OS is also not a good idea for very practical reasons. First of all it's a mistake from a standpoint of efficiency and reliability. Think of the OS as a delivery van and of the applications as the parcels you want to deliver with it. Imagine that, when you buy your van, it comes with a number of large parcels already in it. These parcels are welded in place so that it is all but impossible to remove them without damaging your van. You have to live with them, drive them around to wherever you go and burn up extra fuel to do so, and quite often these built-in parcels get in the way of the items you wanted to deliver when you bought your van for that purpose. Even worse; when one of the parcels is damaged, quite often your van has to be serviced in order to fix the problem! But when complain about it, the manufacturer of the van tells you that this actually makes it a much better delivery van. Ridiculous? Yes, of course. It's just as ridiculous as Windows using up valuable system resources for bundled applications that more often than not get in the way of what you need to do, and add points-of-failure to boot.

The second practical issue, related to the first one, is control, from the user's point of view. A basic operating system allows the user to install, configure and run applications as required, and to choose the right application based on their features and performance, and the user's preference. Bundling applications in Windows removes their functions from the application level (where the user has control over them) to the operating system (where the user has no control over them). For example, a user application such as a web browser can be installed, configured and uninstalled as necessary, but Internet Explorer is all but impossible to remove without dynamite. The point here is not that Internet Explorer should not be provided (on the contrary; having a web browser available is a convenience much appreciated by most users) but that it should be available as an optional application, to be installed or uninstalled as per the user's preference without any consequence for how the rest of Windows will function.

Beyond repair

Then there's Windows' lack of an adequate repair or maintenance mode. If anything goes wrong and a minor error or corruption occurs in one of the (literally) thousands of files that make up Windows, often the only real solution is a large-scale restore operation or even to reinstall the OS. Yes, you read correctly. If your OS suddenly, or gradually, stops working properly and the components which you need to repair are unknown or being locked by Windows, the standard approach to the problem (as recommended by Microsoft) is to do a complete reinstallation. There's no such thing as single user mode or maintenance mode to do repairs, nor is there a good way to find out which component has been corrupted in the first place, let alone to repair the damage. (The so-called 'safe mode' merely swaps configurations and does not offer sufficient control for serious system repairs.)

Windows has often been criticized for the many problems that occur while installing it on a random PC, which may be an A-brand or clone system in any possible configuration. This criticism is not entirely justified; after all it's not practically feasible to foresee all the possible hardware configurations that may occur at the user end. But that's not the point. The point is that these problems are often impossible to fix or even properly diagnose, because most of the Windows operating system is beyond the users' or administrators' control. This is of course less true for Windows 9x/ ME. Because these are essentially DOS products, you can reboot the system using DOS and do manual repairs to a certain degree. With Windows NT and its successors this is generally impossible. Windows 2000, XP and Vista come with an external repair console utility on the CD, that allows you some access to the file system of a damaged Windows installation. But that's about it.

The inability to make repairs has been addressed, to a certain degree, in Windows XP. This comes with a 'System Restore' feature that tracks changes to the OS, so that administrators may 'roll back' the system to a previous state before the problem occurred. Also, the 'System File Check' feature attempts to make sure that some 1000 system files are the ones that were originally installed. If a "major" system file is replaced by another version (for example if a Windows XP DLL file is overwritten by a Windows '95 DLL with the same name) the original version will be restored. (Of course this also prevents you from removing things like Outlook Express or Progman.exe, since the specification of what is an important system file is rather sloppy.) Windows Vista takes these features even further, by incorporating transaction-based principles. This enhances the chances of a successful roll-back from changes that have not been committed permanently yet.

Most of these workarounds are largely beyond the user's control. While some of them may have adverse effects (e.g. File System Check may undo necessary modifications) their effectivity is limited by nature. There are many fault conditions possible that prevent automated repair features from working correctly in the first place. When Windows breaks, the automated features to recover from that fault generally break as well. Also the number of faults that the automated repair options can deal with are limited. At some point manual intervention is the only option, but that requires the adequate maintenance mode that Windows doesn't have. The inability of a commercial OS to allow for its own maintenance is a good demonstration of its immaturity.

Even so, even the added options for system restore in XP and Vista are an improvement over the previous situation, in that at least a certain amount of recovery is now possible. On the other hand, this illustrates Microsoft's kludgy approach to a very serious problem: instead of implementing changes in the architecture to prevent OS corruption, they perpetuate the basic design flaw and try to deal with the damage after the fact. They don't fix the hole in your roof, they sell you a bucket to put under it instead. When the bucket overflows (i.e. the system recovery features are insufficient to solve a problem) you're still left with a mess.

Wasted resources, wasted investments

The slipshod design of Windows does not only reflect in its flawed architecture. The general quality of its code leaves a lot to be desired as well. This translates not only in a disproportionately large number of bugs, but also in a lot of inefficiency. Microsoft needs at least three or four times as much hardware to deliver the same performance that other operating systems (e.g. Unix) deliver on much less. Likewise, on similar hardware competing products perform much better, or will even run well on hardware that does not meet Microsoft's minimum system requirements.

Inefficient code is not the only problem. Another issue is that most bells and whistles in Microsoft products are expensive in terms of additional hardware requirements and maintenance, but do not increase productivity at all. Given the fact that ICT investments are expected to pay off in increased productivity, reduced cost or both, this means that most "improvements" in Microsoft products over the past decades have been a waste of time from a Return On Investment standpoint. Typical office tasks (e.g. accounting, data processing, correspondence) have not essentially changed, and still take as much time and personpower as they did in the MS-DOS era. However the required hardware, software and ICT staff have increased manifold. Very few of these investments have resulted in proportional increases in profit.

Only 32 kilobytes of RAM in the Apollo capsules' computers was enough to put men on the moon and safely get them back to Earth. The Voyager deep space probes that sent us a wealth of images and scientific data from the outer reaches of the solar system (and still continue to do so from interstellar space) have on-board computers based on a 4-bit CPU. An 80C85 CPU with 176 kilobytes of ROM and 576 kilobytes of RAM was all that controlled the Sojourner robot that drove across the surface of Mars and delivered geological data data and high-resolution images in full-color stereo. But when I have an 800MHz Pentium III with 256 Megabytes of RAM and 40 Gigabytes of disk space, and I try to type a letter to my grandmother using Windows XP and Office XP, the job will take me forever because my computer is underpowered! And of course Windows Vista won't even run on such a machine...

Server-based or network-based computing is no solution either, mainly because Windows doesn't have any real code sharing capability. If you were to shift the workload of ten workstations to an application server (using Windows Terminal Server, Citrix Server or another ASP-like solution) the server would need a theoretical ten times the system resources of each of the workstations it replaced to maintain the same performance, not counting the inevitable overhead which could easily run up to an additional 10 or 20 percent.

Then there's the incredible amount of inefficient, or even completely unnecessary code in the Windows file set. Take the 3D Pinball game in Windows 2000 Professional and XP Professional, for example. This game (you'll find it under \Program Files\Windows NT\Pinball) is installed with Windows and takes up a few megabytes of disk space. But most users will never know that it's sitting there, wasting storage and doing nothing productive at all. It doesn't appear in the program menu or control panel, and no shortcuts point to it. The user isn't asked any questions about it during installation. In fact its only conceivable purpose would be to illustrate Microsoft's definition of 'professional'. No wonder Windows has needed more and more resources over the years. A few megabytes doesn't seem much, perhaps, but that's only because we've become used to the enormous footprints of Windows and Windows applications. Besides, if Microsoft installs an entire pinball game that most users neither need nor want, they obviously don't care about conserving resources (which are paid for by the user community). What does that tell you about the resource-efficiency of the rest of their code? Let me give you a hint: results published in PC Magazine in April 2002 show that the latest Samba software surpasses the performance of Windows 2000 by about 100 percent under benchmark tests. In terms of scalability, the results show that Unix and Samba can handle four times as many client systems as Windows 2000 before performance begins to drop off.

Another example is what happened when one of my own clients switched from Unix to Windows (the reason for this move being the necessity to run some webbased accounting package with BackOffice connectivity on the server). Their first server ran Unix, Apache, PHP and MySQL and did everything it had to do with the engine barely idling. On the same system they then installed Windows Server 2003, IIS, PHP and MySQL, after which even the simplest of PHP scripts (e.g. a basic 100-line form generator) would abort when the 30 second execution timeout was exceeded.

Paradoxically, though, the fact that Microsoft products need humongous piles of hardware in order to perform decently has contributed to their commercial success. Many integrators and resellers push Microsoft software because it enables them to prescribe the latest and heaviest hardware platforms in the market. Unix and Netware can deliver the same or better performance on much less. Windows 2000 and XP however need bigger and faster systems, and are often incompatible with older hardware and firmware versions (especially the BIOS). This, and the fact that hardware manufacturers discontinue support for older hardware and BIOSes, forces the user to purchase expensive hardware with no significant increase in return on investment. This boosts hardware sales, at the expense of the "dear, valued customer". Resellers make more money when they push Microsoft products. It's as simple as that.

Many small flaws make a big one

Apart from the above (and other) major flaws there's also a staggering amount of minor flaws. In fact there are so many minor flaws that their sheer number can be classified as a major flaw. In short, the general quality of Microsoft's entire set of program code is sub-standard. Unchecked buffers, unverified I/O operations, race conditions, incorrectly implemented protocols, failures to deallocate resources, failures to check environmental parameters, et cetera ad nauseam... You name it, it's in there. Microsoft products contain some extremely sloppy code and bad coding practices that would give an undergraduate some well-deserved bad marks. As a result of their lack of quality control, Microsoft products and especially Windows are riddled with literally thousands and thousands of bugs and glitches. Even many of the error messages are incorrect!

Some of these blunders can be classified as clumsy design rather than as mere sloppiness. A good example is Windows' long filename support. In an attempt to allow for long filenames in Windows '9x/ME, Microsoft deliberately broke the FAT file system. They stored the extension information into deliberately cross-linked directory entries, which is probably one of their dirtiest kludges ever. And if that wasn't enough, they made it legal for filenames to contain whitespace. Because this was incompatible with Windows' own command line parsing (Windows still expects the old FAT notation) another kludge was needed, and whitespace had to be enclosed in quotation marks. This confused (and broke) many programs, including many of Microsoft's own that came with Windows.

Another good example is Windows' apparent case-sensitivity. Windows seems to make a distinction between upper and lower case when handling filenames, but the underlying software layers are still case-insensitive. So Windows only changes the case of the files and directories as they are presented to the user. The names of the actual files and directories may be stored in uppercase, lowercase or mixed case, while they are still presented as capitalized lower case file names. Of course this discrepancy causes no problems in a Windows-only environment. Since the underlying code is essentially case-insensitive, case is not critical to Windows' operation. However as soon as you want to incorporate Unix-based services (e.g. a Unix-based webserver instead of IIS) you discover that Windows has messed up the case of filenames and directories.

But most of Windows' errors and glitches are just the result of sloppy work. Of course there is no such thing as bug-free software, but the amount of bugs found in Windows is, to put it mildly, disproportionate. For example, Service Pack 4 for Windows NT 4.0 attempted to fix some 1200 bugs (yes, one thousand two hundred). But there had already been three previous service packs at the time! Microsoft shamelessly admitted this, and even boasted about having "improved" NT on 1200 points. Then they had to release several more subsequent service packs in the months that followed, to fix remaining issues and of course the additional problems that had been introduced by the service packs themselves.

An internal memo among Microsoft developers mentioned 63,000 (yes: sixty-three thousand) known defects in the initial Windows 2000 release. Keith White, Windows Marketing Director, did not deny the existence of the document, but claimed that the statements therein were made in order to "motivate the Windows development team". He went on to state that "Windows 2000 is the most reliable Windows so far." Yes, that's what he said. A product with 63,000 known defects (mind you, that's only the known defects) and he admits it's the best they can do. Ye gods.

And the story continues: Windows XP Service Pack 2 was touted to address a large number of security issues and make computing safer. Instead it breaks many things (mostly products made by Microsoft's competitors, but of course that is merely coincidence) but does not really fix any real security flaws. The first major security hole in XP-SP2 was described by security experts as "not a hole but rather a crater" and allowed downloadable code to spoof firewall information. Only days after XP-SP2 was released the first Internet Explorer vulnerability of the SP2-era was discovered. Furthermore SP2 leaves many unnecessary networking components enabled, bungles permissions, leaves IE and OE open to malicious scripts, and installs a packet filter that lacks a capacity for egress filtering. It also makes it more difficult for third-party products (especially multimedia plugins) to access the ActiveX controls, which in turn prevents the installation of quite a bit of multimedia software made by Microsoft's competitors. XP-SP2's most noticeable effect (apart from broken application compatibility) are frequent popups that give the user a sense of security. Apart from this placebo effect the long-awaited and much-touted XP-SP2 doesn't really fix very much.

In the summer of 2005 Jim Allchin, then group VP in charge of Windows, finally went and admitted all this. In a rare display of corporate honesty, he told the Wall Street Journal that the first version of Longhorn (then the code name for Windows Vista) had to be entirely scrapped because the quality of the program code had deteriorated too far. The root of the problem, said Allchin, was Microsoft's historical approach to developing software (the so-called "spaghetti code culture") where the company's thousands of programmers would each develop their own piece of code and it would then all be stitched together at the end. Allchin also said to have faced opposition to his call for a completely new development approach, firstly from Gates himself and then the company's engineers.

MS developers: "We are morons"

Allchin's revelations came as no great surprise. Part of the source code to Windows 2000 had been leaked onto the Internet before, and pretty it was not. Microsoft's flagship product turned out to be a vast sprawl of spaghetti in Assembly, C and C++, all held together with sticky tape and paper clips. The source code files contained many now-infamous comments including "We are morons" and "If you change tabs to spaces, you will be killed! Doing so f***s the build process".

There were many references to idiots and morons, some external but mostly at Microsoft. For example:

In the file private\ntos\rtl\heap.c, which dates from 1989:
```
// The specific idiot in this case is Office95, which likes
// to free a random pointer when you start Word95 from a desktop
// shortcut.
```

In the file private\ntos\w32\ntuser\kernel\swp.c from 11-Jul-1991:
```
// for idiots like MS-Access 2.0 who SetWindowPos( SWP_BOZO )
// and blow away themselves on the shell, then lets
// just ignore their plea to be removed from the tray.
```

Morons are also to be found in the file private\genx\shell\inc\prsht.w:
```
// We are such morons. Wiz97 underwent a redesign between IE4 and IE5
```

And in private\shell\shdoc401\unicpp\desktop.cpp:
```
// We are morons. We changed the IDeskTray interface between IE4
```

In private\shell\browseui\itbar.cpp:
```
// should be fixed in the apps themselves. Morons!
```

As well in private\shell\ext\ftp\ftpdrop.cpp:
```
We have to do this only because Exchange is a moron.
```

Microsoft programmers also take their duty to warn their fellow developers seriously against unsavory practices, which are apparently committed on a regular basis. There are over 4,000 references to "hacks". These include:

In the file private\inet\mshtml\src\core\cdbase\baseprop.cxx:
```
// HACK! HACK! HACK! (MohanB) In order to fix #64710
// at this very late date
```

In private\inet\mshtml\src\core\cdutil\genutil.cxx:
```
// HACK HACK HACK. REMOVE THIS ONCE MARLETT IS AROUND
```

In private\inet\mshtml\src\site\layout\flowlyt.cxx:
// God, I hate this hack ...

In private\inet\wininet\urlcache\cachecfg.cxx:
```
// Dumb hack for back compatibility. *sigh*
```

In private\ispu\pkitrust\trustui\acuictl.cpp:
```
// ACHTUNG! HACK ON TOP OF HACK ALERT:
// Believe it or not there is no way to get current height
```

In private\ntos\udfs\devctrl.c:
```
// Add to the hack-o-rama to fix formats.
```

In private\shell\shdoc401\unicpp\sendto.cpp:
```
// Mondo hackitude-o-rama.
```

In private\ntos\w32\ntcon\server\link.c:
```
// HUGE, HUGE hack-o-rama to get NTSD started on this process!
```

In private\ntos\w32\ntuser\client\dlgmgr.c:
```
// HACK OF DEATH!!
```

In private\shell\lib\util.cpp:
```
// TERRIBLE HORRIBLE NO GOOD VERY BAD HACK
```

In private\ntos\w32\ntuser\client\nt6\user.h:
```
// The magnitude of this hack compares favorably with that
// of the national debt.
```

The most worrying aspect here is not just how these bad practices persist and even find their ways into release builds in large numbers. After all, few things are as permanent as a "temporary" solution. Nor is it surprising how much ancient code still exists in the most recent versions of Windows (although it is somewhat unsettling to see how decades-old mistakes continue to be a problem). No, the most frightening thing is that Microsoft's developers obviously know they are doing terrible things that serious undermine the quality of the end product, but *are apparently unable to remedy the known bad quality of their own code.*

As you may remember, Windows XP was already out by the time that the above source code got leaked. In fact, back in 2004, Microsoft had been talking about Longhorn (Windows Vista) for three years. Just a few months after the source code leaked out, it was announced that WinFS, touted as Microsoft's flagship Relational File System Of The Future, would not ship with Vista after all. The reason isn't hard to guess: Windows' program code has become increasingly unmaintainable and irrepairable over the years.

In the long years since XP was launched, Apple have come out with five major upgrades to OSX, upgrades which (dare I say it?) install with about as much effort as it takes to brush your teeth in the morning. No nightmare calls to tech-support, no sudden hardware incompatibilities, no hassle. Yet Microsoft has failed to keep up, and the above example of the state of their program code clearly demonstrates why.

Unreliable servers

All these blunders have of course their effects on Windows' reliability and availability. Depending on application and system load, most Windows systems tend to need frequent rebooting, either to fix problems or on a regular basis to prevent performance degradation as a result of Windows' shaky resource management.

On the desktop this is bad enough, but the same flaws exist in the Windows server products. Servers are much more likely to be used for mission-critical applications than workstations are, so Windows' limited availability and its impact on business become a major issue. The uptimes of typical Windows-based servers in serious applications (i.e. more than just file and print services for a few workstations) tend to be limited to a few weeks at most. One or two server crashes (read: interruptions of business and loss of data) every few months are not uncommon. As a server OS, Windows clearly lacks reliability.

Windows server products aren't even really server OSes. Their architecture is no different from the workstation versions. The server and workstation kernels in NT are identical, and changing two registry keys is enough to convince a workstation that it's a server. Networking capabilities are still largely based on the peer-to-peer method that was part of Windows for Workgroups 3.11 and that Microsoft copied, after it had been successfully pioneered by Apple and others in the mid-eighties. Of course some code in the server products has been extended or optimized for performance, and domain-based authentication has been added, but that doesn't make it a true server platform. Neither does the fact that NT Server costs almost three times as much as NT Workstation. In fact we're talking about little more than Windows for Workgroups on steroids.

In November 1999, Sm@rt Reseller's Steven J. Vaughan-Nichols ran a test to compare the stability of Windows NT Server (presumably running Microsoft Internet Information Server) with that of the Open Source Linux operating system (running Samba and Apache). He wrote:

> *Conventional wisdom says Linux is incredibly stable. Always skeptical, we decided to put that claim to the test over a 10-month period. In our test, we ran Caldera Systems OpenLinux, Red Hat Linux, and Windows NT Server 4.0 with Service Pack 3 on duplicate 100MHz Pentium systems with 64MB of memory. Ever since we first booted up our test systems in January, network requests have been sent to each server in parallel for standard Internet, file and print services. The results were quite revealing. Our NT server crashed an average of once every six weeks. Each failure took roughly 30 minutes to fix. That's not so bad, until you consider that* ***neither Linux server ever went down.***

Interesting: a crash that takes 30 minutes to fix means that something critical has been damaged and needs to be repaired or restored. At least it takes more than just a reboot. This happens once every six weeks on a server, and that's considered "not so bad"... Think about it. Also note that most other Unix flavors such as Solaris, BSD or AIX are just as reliable as Linux.

But the gap between Windows and real uptime figures is even greater than Vaughan-Nichols describes above. Compare that half hour downtime per six weeks to that of Netware, in the following article from Techweb on 9 April 2001:

> *Server 54, Where Are You?*
> *The University of North Carolina has finally found a network server that, although missing for four years, hasn't missed a packet in all that time. Try as they might, university administrators couldn't find the server. Working with Novell Inc. (stock: NOVL), IT workers tracked it down by meticulously following cable until they literally ran into a wall. The server had been mistakenly sealed behind drywall by maintenance workers.*

Although there is some doubt as to the actual truth of this story, it's a known fact that Netware servers are capable of years of uninterrupted service. Shortly before I wrote this, I brought down a Netware server at our head office. This was a Netware 5.0 server that also ran software to act as the corporate SMTP/POP3 server, fax server and main virus protection for the network, next to providing regular file and print services for the whole company. It had been up and running without a single glitch for more than a year, and the only reason we shut it down was because it had to be physically moved to another building. Had the move not been necessary, it could have run on as long as the mains power held out. There's simply no reason why its performance should be affected, as long as nobody pulls the plug or rashly loads untested software. The uptimes of our Linux and Solaris servers (mission-critical web servers, database servers and mail servers, or just basic file and print servers) are measured in months as well. Uptimes in excess of a year are not uncommon for Netware and Unix platforms, and uptimes of more than two years are not unheard of either. Most OS updates short of a kernel replacement do not require a Unix server to be rebooted, as opposed to Windows that expects a complete server reboot whenever a DLL in some subsystem is updated. But see for yourself: check the **Netcraft Uptime** statistics and compare the uptimes of Windows servers to those of Unix servers. The figures speak for themselves.

Microsoft promises 99.999% availability with Windows 2000. That's a little over 5 minutes of downtime per year. Frankly I can't believe this is a realistic target for Windows. Microsoft products have never even approached such uptime figures. Even though most of the increased availability of Windows 2000 must be provided through third-party clustering and redundancy solutions (something that the glossy ads neglect to mention) it's highly unlikely that less than five minutes of downtime per year for the entire Windows cluster is practically feasible.

Perhaps even more serious is the fact that, short of clustering, there is no adequate solution for the many software glitches that jeopardize the availability of a typical Windows server. A typical NT or 2000 server can spontaneously develop numerous transient problems. These may vary from network processes that seem healthy but ignore all network requests, to runaway server applications that lock up the entire operating system. Usually the only solution in these cases is to power cycle and restart the system. I remember having to do that three times a week on a production server. Believe me, it's no fun. Perhaps it's understandable that some network administrators feel that the best way to accelerate a Windows system is at 9.81 meters per second squared.

More worries, more cost, or both

Does all this make Windows an entirely unusable product that cannot run in a stable configuration anywhere? No, fortunately not. There are situations where Windows systems (both workstations and servers) may run for long periods without crashing. A vendor-installed version of Windows NT of 2000 on an HCL-compliant, A-brand system, with all the required service packs and only certified drivers, should give you relatively few problems (provided that you don't use it for much more than basic file and print services, of course). The rule of thumb here is to use only hardware that is on Microsoft's Hardware Compatibility List (HCL), to use only vendor-supplied, certified drivers and other software, and to use third-party clustering solutions for applications where availability is a critical issue.

Another rule of thumb is: one service, one server. Unix sysadmins would expect to run multiple services on one server and still have resources to spare. Good Windows sysadmins generally don't do that. If you need to run a file/print server, a web server and a mail server, all under Windows, use three servers. This will minimize the risk of software conflicts, and it will help prevent overload. On the other hand, you now have to maintain three servers instead of one, which in turn requires more IT staff to keep up with the work.

A diligent regime of upgrading and running only the latest versions of Microsoft products may help as well. Such a policy will cost a small fortune in license upgrades, but it may help to solve and even prevent some problems. To be honest, Windows XP and Vista on the desktop, and Windows Server 2003 in the server room, are somewhat better (or rather, less bad) than NT4 was. These versions are at least more stable, and less prone to spontaneous crashes, than NT4 was. Some of NT's most awkward design blunders have been fixed. For example, the user home directories are no longer located under the WINNT directory. On most systems (especially on notebook computers) XP and Vista are considerably less shaky (albeit by no means perfect) and hardware support is certainly a lot better. Which goes to show that a few relatively trivial changes may go a long way..

But still, given the general quality of Microsoft products and Windows in particular, there are absolutely no guarantees. And of course Microsoft introduced a whole new set of glitches and bugs in Windows XP, which largely undid many of the improvements in Windows 2000. So now Windows XP is less stable in some situations than Windows 2000 was, and Vista will stumble on issues that didn't bother XP, starting with the inability to copy files in less than a few days, an issue that even Service Pack 1 didn't solve on most PCs. But that's innovation for you, I suppose.

Denial will see us through

One frightening aspect about all this is that Microsoft doesn't seem to realize how serious these problems are. Or rather, they probably realize it but they don't seem to care as long as sales hold up. While the core systems of large companies still run on either mainframes or midrange Linux systems in order to provide sufficient reliability and performance, Microsoft sales reps pretend that Windows is good enough to compete in that area.

Microsoft likes to pretend that Windows' huge shortcomings are only minor. Their documents on serious problems (which are always called 'Issues' in Microsoft-speak) are very clear on that point. Take the classic 'TCP/IP Denial Of Service Issue' for example: a serious problem that was discovered a few years ago. It caused NT servers to stop responding to network service requests, thus rendering mission-critical services unavailable. (This should not be confused with deliberate Denial Of Service attacks to which most operating systems are vulnerable; this was a Windows issue only.) At the time there was no real solution for this problem. Microsoft's only response at the time was to state that "This issue does not compromise sensitive data in any way. It merely forces a server to become unavailable for a short time, which is easily remedied by rebooting the server." NT sysadmins had to wait for the next service pack that was released several months later before this problem was addressed. In the meantime they were expected to accept downtime and the rebooting of mission-critical servers as a matter of course. After all no data was lost, so how bad could it be?

And Microsoft thinks that this stuff can compete with Unix and threaten the mainframe market for mission-critical applications?

Uh-huh. I don't think so.

In September 2001 Hewlett-Packard clustered 225 PCs running the Open Source Linux operating system. The resulting system (called I-cluster) benchmarked itself right into the global top-500 of supercomputers, using nothing but unmodified, out-of-the-box hardware. (A significant number of entries in that top-500, by the way, runs Linux, and more and more Unix clusters are being used for supercomputing applications.) Microsoft, with a product line that is descended solely from single-user desktop systems, can't even dream of such scalability - not now, not ever. Nevertheless Microsoft claimed on a **partner website** with Unisys that Windows will outperform Unix, because Unisys' server with Windows 2000 Datacenter could be scaled up to 32 CPU's. This performance claim is of course a blatant lie: the opposite is true and they know it. Still Microsoft would have us believe that the performance, reliability and scalability of the entire Windows product line is on par with that of Unix, and that clustered Windows servers are a viable replacement option for mainframes and Unix midrange systems. I'm not kidding, that's what they say. If you're at all familiar with the scalability of Unix midrange servers and the requirements of the applications that mainframes are being used for, you will realize how ludicrous this is.

Microsoft lacks confidence in own products

Dog food is sold to the dog owners who buy it, not to the dogs who have to eat it. "Eating your own dog food" is a metaphor for a programmer who uses the system he or she is working on. Is it yet functional enough for real work? Would you trust it not to crash and lose your data? Does it have rough edges that scour your hand every time you use a particular feature? Would you use it yourself by choice?

When Microsoft acquired the successful Hotmail free Email service, the system had roughly 10 million users, and the core systems that powered Hotmail all ran Unix. A few years later the number of Hotmail users had exceeded 100 million, but in spite of Microsoft's claims about the power of Windows and their previous intentions to replace Hotmail's core systems with Windows servers, Hotmail's core systems still run Unix. This was discussed thoroughly in a leaked-out internal paper by Microsoft's Windows 2000 Server Product Group member David Brooks. It mentioned the proverbial stability of the Unix kernel and the Apache web server, the system's transparency and combination of power and simplicity. Windows on the other hand it considered to be needlessly GUI-biased (Brooks wrote: "Windows [...] server products continue to be designed with the desktop in mind") and also complex, obscure and needlessly resource-hungry. (Brooks: "It's true that Windows kequires a more powerful computer than Linux or FreeBSD [and treats a server] reboot as an expectation".)

Hotmail is not the only example of Microsoft's refusal to eat their own dog food. The "We have the way out" anti-Unix website that Microsoft (along with partner Unisys) put up in the first months of 2002, initially ran Unix and Apache. (It was ported to IIS on Windows 2000 only after the entire ICT community had had a good laugh).

For many years Microsoft's own email servers have protected by third-party security software. This amounts to a recognition of the fact that Exchange on Windows needs such third party assistance to provide even a basic level of system security.

Microsoft's SQL Labs, the part of the company that works on Microsoft's SQL Server, purchased NetScreen's 500-series security appliance to defend its network against Code Red, Nimda and other worm attacks. Apparently the labs' choice was made despite the fact that Microsoft then already sold its own security product touted as a defense against such worms. The Microsoft ISA [Internet Security and Acceleration] Server was introduced in early 2001 and was hailed by Microsoft as their first product aimed entirely at the security market. In fact, the most important reason businesses ought to switch to ISA Server, according to Microsoft, was that "ISA Server is an [...] enterprise firewall and secure application gateway designed to protect the enterprise network from hacker intrusion and malicious worms". Still Microsoft's SQL Labs prudently decided to rely on other products than their own to provide basic security.

Microsoft's own accounting division used IBM's AS/400 midrange platform for critical applications such as the payroll system, until well in the late nineties.

The most recent example of Microsoft's awareness of the shortcoming of their own products thus far is how some of Microsoft's own top executives had trouble getting Windows Vista to work in the weeks after its release. The officials, including a member of the Microsoft board of directors, voiced some of the same complaints about missing drivers and crippled graphics that users have raised since Vista debuted in January 2007. Steven Sinofsky, the Microsoft senior vice president who took charge of Windows development the day after Vista's retail release, complained that some of his hardware wouldn't work with the new OS. "My home multi-function printer did not have drivers until 2/2 and even then [they] pulled their 1/30 drivers and released them (Brother)" said Sinofsky in an e-mail dated Feb. 18, 2007. Sinofsky's e-mail was one of hundreds made public in February 2008 by U.S. District Court Judge Marsha Pechman as part of a lawsuit that claimed Microsoft deceived buyers when it promoted PCs as "Windows Vista Capable" in the run-up to the 2006 holiday season. Mike Nash, vice president for Windows product management, was nailed by the Vista Capable debacle more than a year later when he bought a new laptop. "I know that I chose my laptop (a Sony TX770P) because it had the Vista logo and was pretty disappointed that it not only wouldn't run [Aero], but more important wouldn't run [Windows] Movie Maker" Nash said in an email on Feb. 25, 2007. "Now I have a $2,100 e-mail machine."

Network pollution

It should also be mentioned that Microsoft doesn't know the first thing about networking. A Windows system in a TCP/IP environment still uses a NetBIOS name. Microsoft networking is built around NetBEUI, which is an extended version of NetBIOS. This is a true Stone Age protocol which is totally unroutable. It uses lots of broadcasts, and on a network segment with Windows PCs broadcasts indeed make up a significant portion of the network traffic, even for point-to-point connections (e.g. between a Microsoft mailbox and a client PC). If it weren't for the fact that it is possible to encapsulate NetBIOS/NetBEUI traffic in a TCP/IP envelope, connecting Windows to the real world would be totally impossible. (Note that Microsoft calls the IP encapsulation of NetBEUI packets 'native IP'. Go figure.) The problem is being made worse by the ridiculous way in which Microsoft applications handle file I/O. Word can easily do over a hundred 'open' operations on one single file, and saving a document involves multiple write commands with only one single byte each. Thus Windows PCs tend to generate an inordinate amount of garbage and unnecessary traffic on the network.

Microsoft's design approach has never shown much understanding of of computer networking. I remember reading a document from Microsoft that stated that a typical PC network consists of ten or at most twenty peer-to-peer workstations on a single cable segment, all running Microsoft operating systems. And that explains it, I suppose. If you want anything more than that, on your own head be it.

Here's a simple test. Take a good, fast FTP server (i.e. one that runs on Unix). Upload and download a few large files (say, 50MB) from and to a Windows NT or 2000 workstation. (I used a 233MHz Pentium-II.) You will probably see a throughput in the order of 1 Mbit/s for uploads and 2 to 3 Mbit/s for downloads, or more on faster hardware.
Then boot Linux on the same workstation (a quick and easy way is to use a Linux distribution on a ready-to-run CD that requires no installation, such as Knoppix). Then repeat the upload and download test. You will now see your throughput limited only by the bandwidth or your network connection, the capacity of your FTP server, or by your hardware performance, whichever comes first. On 10 Mbit/s Ethernet, 5 Mbit/s upload and download throughput are the least you may expect. To further test this, you can repeat it with a really slow client (e.g. a 60 or 75MHz Pentium) running Linux. The throughput limit will still be network-bound and not system-bound. (Note: this is not limited to FTP but also affects other network protocols. It's a performance problem related to the code in Windows' IP stack and other parts of the architecture involved with data throughput.)

New Windows versions bring no relief here. Any network engineer who uses PPPoE (Point-to-Point Protocol over Ethernet) with ADSL will tell you that the MTU (a setting that limits packet size) should be set to 1492 or less. In XP it's set by default to 1500, which may lead to problems with the routers of many DSL ISPs. Microsoft is aware of the problem, but XP nevertheless persists in setting up PPPoE with an MTU of 1500. There is a registry hack for PPPoE users, but there is no patch, and XP has no GUI-based option which enables the user to change the MTU conveniently.

The above example is fairly typical of XP. It tries to do things itself and botches the job, rather than give you control over it to do it properly. But all versions of Windows share a number of clumsily designed and coded network features, starting with Windows file sharing. This service uses fixed ports, and can't be moved to other ports without using dynamite. This means that routing the essentially insecure Windows file sharing connections through a secure SSH tunnel is extremely cumbersome, and requires disabling (or, on XP, uninstalling) file sharing services on the local client, so that using both a tunneled and a non-tunneled file sharing connection at the same time is impossible, and switching back and forth between the two requires rebooting. Yes, you could conceivably solve this with a VPN configuration, but that's not the point. The point is that any self-respecting network client will let you configure the ports it uses but, apparently for reasons of user-friendliness, Microsoft hard-coded the file sharing ports into their software, thereby making it impossible to extend file sharing beyond insecure connections on a local LAN.

While many Windows' networking limitations are rapidly phasing out now that the '9x/ME product line has been abandoned, others persist. Set an XP or Vista box or a Windows 2003 server to share files, and then try to get Windows networking to 'see' those shares over a VPN or from the other end of an Internet router. You can't, or at least not without cumbersome and unnecessarily expensive workarounds, due to Windows Networking still being based on a non-routable IBM protocol from the 1970's.

On top of all this, readers of this paper report that according to John Dvorak in PC Magazine, the Windows kernel maxes out at 483 Mbps. He remarks that as many businesses are upgrading to 1 Gigabit Ethernet, Windows (including XP) just can't keep up.

Now go read what Microsoft writes about Windows 2000 and XP being the ideal platform for Internet applications...

Denial of Service vulnerabilities

The sloppy nature of Windows' networking support code and protocol stacks also makes the system more vulnerable to Denial of Service attacks. A DoS attack is a form of computer vandalism or sabotage, with the intention to crash a system or otherwise render it unavailable. In a typical DoS attack a deliberately malformed network packet is sent to the target system, where it triggers a known flaw in the operating system to disrupt it. In the case of Windows, though, there are more ways to bring down a system. For example, the kernel routines in Windows 2000 and XP that process incoming IPsec (UDP port 500) packets are written so badly that sending a stream of regular IPsec packets to the server will cause it to bog down in a CPU overload. And of course Windows' IPsec filters cannot block a 500/udp packet stream.

Another way to render a system unavailable is a Distributed Denial of Service attack. A DDoS attack involves many networked systems that send network traffic to a single target system or network segment, which is then swamped with traffic and becomes unreachable. There's very little that can be done against DDoS attacks, and all platforms are equally vulnerable.

With all these DoS and DDoS vulnerabilities, it's a worrying development that Windows 2000 and XP provide new platforms to generate such attacks. The only real 'improvement' in Windows 2000's and XP's IP stacks is that for no good reason whatsoever, Microsoft has extended the control that an application has over the IP stack. This does not improve Windows' sub-standard networking performance, but it gives applications the option to build custom IP packets to generate incredibly malicious Internet traffic. This includes spoofed source IP addresses and SYN-flooding full scale Denial of Service (DoS) attacks. As if things weren't bad enough...

Cumulative problems on the server

So far we have concentrated on Windows. Most of the problems with Microsoft products originate here, since Windows is by far the most complex Microsoft product line, and there are more interactions between Windows and other products than anywhere else. But unfortunately most server and desktop applications are cut from the same cloth as Windows is. The general quality of their code and design is not much better.

The additional server products generally run on a Windows server. This means that all the disadvantages of an insecure, unstable platform also apply to the server products that run on those servers. For example, Microsoft SQL Server is a product that has relatively few problems. Granted, it suffers from the usual problems, but nothing unexpected. It's basically a straightforward implementation of a general SQL server, based on technology not developed by MS but purchased from Sybase. Prior to V7, SQL Server was mostly Sybase code. It wasn't until V7 that SQL Server saw major rewrites.

While SQL Server causes relatively few problems, it is not a very remarkable or innovative product. Not only does it bear all worst of Microsoft's hallmarks (things like Service Pack 4 for MS SQL Server 2000 having to mostly fix problems caused by Service Pack 3) but if I had waited until 2005 to implement database partitioning, I think I'd be covering it up, not trumpeting it to the world...

Still SQL Server is not a bad product as far as it goes, certainly not by Microsoft standards. However, no database service can perform better or be more reliable than the platform it's running on. (This goes of course for any software product, not just for a database server.) All vulnerabilities that apply to the Windows server directly apply to the database service as well.

Other additional server products come with their own additional problems. Microsoft's webserver product, Internet Information Server (IIS) is designed not just to serve up web pages written in the standard HTML language, but also to provide additional authentication and links to content databases, to add server and client side scripting to web pages, to generate Dynamic HTML and Active Server Pages, et cetera. And it does all these things, and more, but often not very well. IIS is outperformed by all other major webserver products (especially Apache). IIS' authentication is far from robust (the general lack of security in MS products is discussed below) and the integration of an IIS webserver with a content database server is far from seamless. Dynamic HTML, ASP and scripting require the webserver to execute code at the server end, and there Microsoft's bad process management comes into play: server load is often excessive. Running code at the server end in response to web requests creates a lot of security issues as well, and on top of all that the web pages that are generated do not conform to the global HTML standards, they are only viewed correctly in Microsoft's own web browser products.

Microsoft's mail server product, Exchange, has a few sharp edges as well. To begin with, its performance is definitely sub-standard. Where one Unix-based mail server will easily handle thousands of users, an Exhange server maxes out at about one hundred. So to replace large Unix-based email services with Exchange generally requires a server farm.
A much bigger problem is Exchange's lack of stability and reliability. To lose a few days worth of corporate E-mail in an Exchange crash is not uncommon. Most of these problems are caused by the hackish quality of the software. Exchange is designed to integrate primarily with other Microsoft products (especially the Outlook E-mail client) and it doesn't take the Internet's global RFC standards too seriously. This limits compatibility and may cause all kinds of problems. Outlook Express also has a strange way of talking IMAP to the Exchange server. It makes a bazillion IMAP connections; each connection logs in, performs one task, sets the connection to IDLE-- and then drops the connection. Since OE does not always close the mailbox properly before dropping the connection, the mailbox and Outlook do not necessarily sync up. This means that you may delete messages in OE that magically return to life in a few minutes because those deletions did not get disseminated to the mailbox before the connection terminated.

Just like other Microsoft applications, the additional server products are tightly integrated into Windows during installation. They replace DLLs belonging to the operating system, and they run services at the system level. This does not improve the stability of the system as a whole to begin with, and of course most of the code in the additional server products is of the same doubtful quality as the code that makes up Windows. The reliability and availability of any service can never be better than the server OS it runs on. However most of Microsoft's additional server products add their own complexity, bugs and glitches to the system, which only makes it worse. The resulting uptime and reliability figures are rather predictable. The inefficiency that exists in Windows is also present in the additional server products, so as a rule of thumb each additional service needs its own server platform. In other words: if you need a file and print server, a web server and a mail server, you need three separate systems whereas Unix or Netware could probably do the job on only one system.

Desktop: bigger but not better

Microsoft desktop applications (like Word and Excel) are largely more of the same. They're in the terminal stages of feature bloat: they're full of gadgets that don't really add anything useful to the product, but that ruin productivity because of their complexity, and that introduce more errors, increase resource demands, and require more code which in turn leads to increased risks. After years of patching and adding, the code base for these products has become very messy indeed. Security, if any, has been added as an afterthought here, too. For example, a password-protected Word document is not encrypted in any way. Inserting a 'protected' document into another non-protected document (e.g. an empty new document) is enough to get around the 'protection'. And if that fails, a simple hex editor is enough to change the 'Password To Modify' in a document. Microsoft is aware of this, but now claims that the 'Password To Modify' is only intended to "prevent accidental changes to a document" and not to offer protection from modifications by malicious third parties. Uh-huh.

Animated paper clips don't really make Word a better word processor. We'd be better off with other things, such as a consistent behavior of the auto-format features, the ability to view markup codes, or a more complete spell checking algorithm and dictionary. But in spite of all the "new" versions of Office and persistent feature requests from their users, Microsoft still hasn't gotten around to that. Instead we have multi-language support that tends to 'forget' its settings occasionally, and an 'auto-correct' feature that's limited to the point of being more annoying than useful. Word documents have become excessively large and unwieldy, and occasionally they are corrupted while being saved to disk. When that happens, Word cannot recover these documents and will crash in the attempt to open them.

In fact it's hilarious that the latest version of Office, well into the 21st century, still can't handle multiple users reading and writing the same data. It's stuck in the eighties, when multiple users might have been able to read the same data, but all but the best systems couldn't properly handle writing to the same files, let alone database records. This problem, referred to as record locking, was fixed in modern software over a decade ago.

We can be brief about Excel: it has similar problems, and calculation errors in formula-based spreadsheets on top of that. Excel is full of frills and spiffy graphics and animations, but essentially it's still a spreadsheet that cannot count and that requires many formulas and macros to be rewritten for each new version of Excel.

The database component of Office, Microsoft Access, isn't exactly a stellar piece of work either. Access, apart from its quirky way of interfacing with backend databases, can still lock out an entire (possibly mission-critical) database, just because one user hasn't shut down the application used to write or modify data. Access is actually supposed to be able to properly handle this condition, but it doesn't. And in a stunning display of lack of understanding, Access 2007 introduced the use of multi-valued data types in SQL databases, in an attempt to make the product easier for power users to drive. The development team felt that power users find the creation of many-to-many joins using three tables conceptually very difficult, and will find multi-valued data types a much easier solution. In this they are correct; users certainly do struggle with the concept of creating many-to-many joins using three tables as is the 'classic' way in SQL. However the reason for doing it the old-fashioned way is that this is totally accurate and predictable, and that every bit of data (every atomic value) will always be accessible, which was one of main design principles (perhaps even the whole point) of SQL's design around atomic values. The multi-valued approach is like putting cruise control on a back hoe or a bullldozer in an attempt to make it easier for unskilled operators to use, and it will result in a similar mess.

Menu interfaces in all Microsoft applications, even in those that are bundled in MS Office, are inconsistent and non-intuitive. For example, the menu option to set application preferences, which may be titled 'Preferences' in some products but 'Options' in others, may be found under the 'File' menu, under the 'Edit' menu, under 'View' or somewhere else, depending on what product you're currently using. To create even more confusion, the same options in different applications do not work identically. For example the 'Unsorted list' button (to create a bullet list) handles indentation correctly in Word but ignores it completely in PowerPoint (PowerPoint adds bullets but messes up the left outline of the text). And the 'F3' key activates the 'search again' function for string searches in practically all Microsoft products, except in Internet Explorer and Excel where it brings up something totally different for no apparent reason.

Microsoft does the Internet
(In a manner not unlike Debbie did Dallas)

Microsoft's most important application outside MS Office is without doubt Internet Explorer. In its first incarnation IE was a very unremarkable web browser; e.g. version 2.0 as it was shipped with Windows NT 4 was so backward that it didn't even have frame capability. This soon changed as Microsoft began integrating the web browser with Windows as a part of their integration and bundling strategies (which are discussed in detail below).

In all honesty it must be said that recent versions of IE (starting with version 6) aren't really bad web browsers. That is, from the end users' point of view they mostly do what they're supposed to do. They do have many nasty bugs and problems, but these mainly cause headaches for web developers and not for end users. IE has more than its share of annoying errors in the implementation of style sheets and some strange discrepancies in the rendering of tables. It also tends to become confused by some rather elementary things, such as submitting a page that contains more than one button, in which case IE erroneously returns values for multiple button names. It has its own ideas about the Domain Object Model (DOM) and does not support the DOM Level 2 Events module, even though Microsoft participated in the definition of this module and had ample time to implement it. IE6 also boasts Jscript behavior that is different from any other browser and full of implementation quirks, and lacks proper support for xHTML support and character encoding negotiations. And of course PNG support in IE6 is so bad that it has singlehandedly delayed the acceptation of this image format by many years, and perhaps forever.

Even so, on the whole IE6 and 7 do the job well enough for most users. At least they display standards-compliant HTML as more or less correctly rendered web pages, at a speed that is by all means acceptable. Previous versions of IE weren't nearly this good, and even contained deliberate deviations from the global HTML standards that were intended to discourage the use of standardized HTML in favor of Microsoft's own proprietary and restrictive ideas.

The main drawbacks of Internet Explorer lie in the fact that it tries to be more than just a web browser. It adds scripting support (with the ability to run Visual BASIC or Jscripts that are embedded in web pages) and it hooks directly into the Windows kernel. I've seen web pages that would shut down the Windows system as soon as the page was viewed with Internet Explorer. Microsoft doesn't seem to have bothered very much with basic security considerations, to put it mildly. And of course the installation of a new version of Internet Explorer replaces (overwrites) significant portions of the Windows operating system, with all the drawbacks discussed above.

Similar problems are found in Outlook, Microsoft's E-mail client. Outlook is in fact a separate application, but it isn't shipped separately. There are two versions: one is bundled with Internet Explorer (this version is called Outlook Express) and the other is part of MS-Office (this version is named 'Outlook' and comes with groupware and scheduler capabilities). In itself Outlook is an acceptable, if unremarkable, E-mail client; it allows the user to read and write E-mail. It comes with a few nasty default settings, but at least these can be changed, although the average novice user of course never does that. (For example, messages are sent by default not as readable text but as HTML file attachments. When a user replies to an E-mail, the quoting feature sometimes uses weird formatting that won't go away without dynamite. And there's often a lot of junk that accompanies an outgoing E-mail message.) More serious is the fact that both Outlook and its server-end companion Exchange tend to strip fields from E-mail headers, a practice that is largely frowned upon. This also makes both network administration and troubleshooting more difficult.

The most worrying problem with Outlook is that it comes with a lot of hooks into Internet Explorer. IE code is being used to render HTML file attachments, including scripts that may be embedded into an HTML-formatted E-mail message. Again Microsoft seems to have been completely unaware of the need for any security here; code embedded in inbound E-mail is by default executed without any further checking or intervention from the user.

Basic insecurity of MS products

Which brings us to another major weakness of all Microsoft products: security, or rather the lack thereof. The notorious insecurity of Microsoft software is a problem in itself.

It all begins with Windows' rather weak (not to say naive) security models, and it's apalling quality control.. The number of reports on security holes has become downright embarrassing, but it still keeps increasing regularly. On the other hand, Windows security holes have become so common that they hardly attract attention anymore. Microsoft usually downplays the latest security issues and releases another patch... after the fact. If Microsoft really wanted to resolve these software problems, they would take greater care to ensure such problems were fixed before its products went on sale-- and thus reverse the way it traditionally conducts business. Doing so would mean less resources wasted by its customers each year patching and re-patching their systems in an attempt to clean up after Microsoft's mistakes, but it would also decrease the customers' dependency on what Microsoft calls 'software maintenance'.

In the meantime, hackers are having a ball with Microsoft's shaky security models and even weaker password encryption (which includes simple XOR bitflip operations, the kind of 'encryption' that just about every student reinvents in school). Hackers, script kiddies and other wannabees get to take their pick from the wide choice of elementary security weaknesses to exploit. Some historic and highly virulent worms, for example, spread so rapidly because they could crack remote share passwords in about twenty seconds. This did not stop Microsoft from running an advertising campaign in spring 2003 that centered on hackers becoming extinct along with the dodo and the dinosaur, all because of Microsoft's oh so secure software. Unsurprisingly this violated a few regulations on truth in advertising, and the campaign had to be hastily withdrawn.

In an attempt to clean up their image somewhat, Microsoft made sure that Windows Vista was launched with a lot of security-related noise. For starters, Vista has a better set of default access privileges. Well, finally! Ancient commercial OSes like Univac Exec, CDC Scope and DEC VMS all had special accounts with various permissions ages ago as a matter of course and common sense. On Windows every user needed administrator rights to do basic tasks. But apart from being decades too late, this basic requirement has been met in a typical Microsoft fashion: it creates a security hole so large that it might more properly be called a void. In Windows Vista the need for certain access privileges are now tied to... program names! For example, if Vista sees that an application developer has created a Microsoft Visual C++ project with the word "install" in the project name, then that executable will automatically require admin rights to run. Create exactly the same project but call it, say, Fred, and the need for elevated access permissions magically disappears. In short, all that malicious software has to do is to present a harmless-looking name to Vista, and Vista will let it through.

Apart from the fact that proper access control should have been implemented in Windows NT right from the start, and that Microsoft botched it when it finally did appear in Vista, the rest of Vista's security is the usual hodge-podge of kludges and work-arounds that often attempt to patch one hole and create another one in the process. For example let's look at Vista's "PatchGuard" service. PatchGuard crashes the computer when it detects that specific internal data structures have been "hooked", which is a common way that malicious software starts doing its damage. Not only does this work-around still not amount to proper protection of operating system code in the first place, but it also prevents third-party security products (e.g. anti-virus and anti-spyware programs) from working correctly. In order to remedy this, Microsoft released API's that essentially enable a user-level program to shut down Vista's Security Center. Uh-huh.

An important part of the problem is Windows' lack of proper separation between code running on various system and user levels. Windows was designed around the basic assumption that code always runs with the highest privilege, so that it can do almost anything, including malicious intent. This makes it impossible to prevent malicious code from invading the system. Users may (inadvertently or deliberately) download and run code from the Internet, but it's impossible to adequately protect system level resources from damage by user level code.

Integrated vulnerabilities

The tight integration between the various Microsoft products does little to improve overall security. All software components are loaded with features, and all components can use each other's functions. Unfortunately this means that all security weaknesses are shared as well. For example, the Outlook E-mail client uses portions of Internet Explorer to render HTML that is embedded in E-mail messages, including script code. And of course IE and Outlook hook into the Windows kernel with enough privileges to run arbitrary malicious code that happens to be embedded in a received E-mail message or a viewed web page. Since Outlook uses portions of IE's code, it's vulnerable to IE's bugs as well. So a scripting vulnerability that exists in Outlook also opens up IE and vice versa, and if IE has a hook into certain Windows kernel functions, those functions can also be exploited through a weakness in Outlook. In other words, a minor security leak in one of the components immediately puts the entire system at risk. Read: a vulnerability in Internet Explorer means a vulnerability in Windows Server 2003! A simple Visual BASIC script in an E-mail message has sufficient access rights to overwrite half the planet, as has been proven by Email virus outbreaks (e.g. Melissa, ILOVEYOU and similar worms) that have caused billions of dollars worth of damage.

A good example are Word viruses; these are essentially VBS (Visual BASIC Script) routines that are embedded in Word documents as a macro. The creation of a relatively simple macro requires more programming skills than the average office employee can be expected to have, but at the same time a total lack of even basic security features makes Word users vulnerable to malicious code in Word documents. Because of the integrated nature of the software components, a Word macro is able to read Outlook's E-mail address book and then propagate itself through the system's E-mail and/or networking components. If Windows' security settings prevent this, the malicious virus code can easily circumvent this protective measure by the simple expedient of changing the security settings. How's that for security?

Similarly, VBS scripts embedded in web pages or E-mail messages may exploit weaknesses in IE or Outlook, so that viewing an infected web page or receiving an infected E-mail is enough to corrupt the system without any further action from the user (including manually downloading a file or opening an attachment). Through those weaknesses the malicious code may access data elsewhere on the system, modify the system's configuration or even start processes. In March 2000, a hacker wrote (of course anonymously) on ICQ:

```
21/03/2k: Found the 1st Weakness: In Windows 2000 [...] there is a Telnet daemon service,
which is not started by default. It can be remotely started by embedding a COM object into
HTML code that can be posted on a web page, or sent to an Outlook client. Following script
will start the Telnet service:
<SCRIPT LANGUAGE=VBScript> CreateObject("TlntSvr.EnumTelnetClientsSvr")</SCRIPT>

We've tried it and it really works. Only a Problem... we've put it into a html page. When
opening the page... our best friend "IE5" shows an alert msg saying that "you're going to run
some commands that can be dangerous to your PC...Continue?" We must fix it! No problem using
```

```
    Outlook... [sic]
```

Note that after patching no fewer than seven different security holes in the Windows 2000 telnet code (yes, that's seven security leaks in telnet alone!) Microsoft released another patch in February 2002, to fix security issue number eight: another buffer overflow vulnerability. Somehow I don't think this patch will be the last. If you don't succeed at first, try seven more times, try, try (and try some more) again. Seen in this light, it's not surprising that J.S. Wurzler Underwriting Managers, one of the first companies to offer hacker insurance, have begun charging clients 5 to 15 percent more if they use Microsoft's Windows NT software in their Internet operations.

Microsoft knows exactly how bad their own product security is. Nevertheless they wax lyrical about new features rather than accept their responsibility for their actions. To quote Tom Lehrer:

> *"The rockets go up, who cares where they come down?*
> *That's not my department, says Werner von Braun."*

Microsoft can't be unaware of the risks and damages they cause. After all they prudently refuse to rely on their own products for security, but use third party protection instead. (See above.) And while they try to push their user community into upgrading to new product versions as soon as possible, Microsoft can hardly be called an early adopter. In the autumn of 2001 they still did not run Windows and Exchange 2000 on their own mail servers yet, long after these versions had been released to the market. On other internal systems (less visible but still there) a similar reluctance can be seen to upgrade to new versions of MS products. Only after many security patches and bug fixes have been released will Microsoft risk upgrading their own critical systems.

Sloppiness makes the problem worse

Many security problems are caused by the sloppy code found in many Microsoft products. The many buffer overrun vulnerabilities can be combined with scripting weaknesses. You don't need to open E-mail attachments or even read an incoming E-mail message to risk the introduction of malicious code on your system. Just receiving the data (e.g. downloading E-mail from a POP3 server or viewing a web page) is enough. Yes, stories like this have long been urban legend, but Outlook has made it reality. Microsoft explains: "The vulnerability results because a component used by both Outlook and Outlook Express contains an unchecked buffer in the module that interprets E-mail header fields when certain E-mail protocols are used to download mail from the mail server. This could allow a malicious user to send an E-mail that, when retrieved from the server using an affected product, could cause code of his choice to run on the recipient's computer." This vulnerability has been successfully exploited by Nimda and other malicious worm programs. Other worm programs (e.g. Code Red) combine vulnerabilities like this with creatively constructed URL's that trigger buffer overruns in IIS. Even without the Frontpage extensions installed it is relatively easy to obtain unencrypted administration passwords and non-public files and documents from an IIS webserver. Furthermore, this "E-commerce solution of the future" contains a prank (a hardcoded passphrase deriding Netscape developers as "weenies") in the code section concerned with the access verification mechanism for the whole system. And there are many more weaknesses like this. The list goes on and on and on.

IIS is supposed to power multi-million dollar E-commerce sites, and it has many backend features to accomplish this application. But each and every time we hear about a large online mailorder or E-commerce website that has spilled confidential user data (including credit card numbers) it turns out that that website runs IIS on Windows NT or 2000. (And that goes for adult mailorder houses too. I'm not quite sure what kind of toy a Tarzan II MultiSpeed Deluxe is, but I can probably tell you who bought one, and to which address it was shipped. Many E-commerce websites promise you security and discretion, but if they run IIS they can only promise you good intentions and nothing more. Caveat emptor!)

The Code Red and Nimda worms provided a nice and instructive demonstration of how easy it is to infect servers running IIS and other Microsoft products, and use them for malicious purposes (i.e. the spreading of malicious code and DDoS attacks on a global scale). Anyone who bothers to exploit one of the many documented vulnerabilities can do this. Some of the vulnerabilities exploited by Code Red and Nimda were months old, but many administrators just can't keep up with the ridiculous amount of patches required by IIS. Nor is patching always a solution: the patch that Microsoft released to counter Nimda contained bugs that left mission-critical IIS production servers non-operational.

On 20 June 2001, Gartner vice president and analyst John Pescatore wrote:

> *IIS security vulnerabilities are not even newsworthy anymore as they are discovered almost weekly. This latest bug echoes the very first reported Windows 2000 security vulnerability in the Indexing Service, an add-on component in Windows NT Server incorporated into the code base of Windows 2000. As Gartner warned in 1999, pulling complex application software into operating system software represents a substantial security risk. More lines of code mean more complexity, which means more security bugs. Worse yet, it often means that fixing one security bug will cause one or more new security bugs.*
>
> *The fact that the beta version of Windows XP also contains this vulnerability raises serious concerns about whether XP will show any security improvement over Windows 2000.*

On 19 September 2001, Pescatore continued:

> *Code Red also showed how easy it is to attack IIS Web servers [...] Thus, using Internet-exposed IIS Web servers securely has a high cost of ownership. Enterprises using Microsoft's IIS Web server software have to update every IIS server with every Microsoft security patch that comes out - almost weekly. However, Nimda (and to a lesser degree Code Blue) has again shown the high risk of using IIS and the effort involved in keeping up with Microsoft's frequent security patches.*
>
> *Gartner recommends that enterprises hit by both Code Red and Nimda immediately investigate alternatives to IIS, including moving Web applications to Web server software from other vendors, such as iPlanet and Apache. Although these Web servers have required some security patches, they have much better security records than IIS and are not under active attack by the vast number of virus and worm writers. Gartner remains concerned that viruses and worms will continue to attack IIS until Microsoft has released a completely rewritten, thoroughly and publicly tested, new release of IIS. Sufficient operational testing should follow to ensure that the initial wave of security vulnerabilities every software product experiences has been uncovered and fixed. This move should include any Microsoft .Net Web services, which requires the use of IIS. Gartner believes that this rewriting will not occur before year-end 2002 (0.8 probability).*

As it turns out, Gartner's estimate was overly optimistic. Now, several years later, still no adequately reworked version of IIS has been released yet.

So how serious is this?

In all honesty it must be said that Microsoft has learned to react generally well to newly discovered security holes. Although the severity of many security problems is often downplayed and the underlying cause (flawed or absent security models) is glossed over, information and patches are generally released promptly and are available to the user community without cost. This is commendable. But then the procedure has become routine for Microsoft, since new leaks are discovered literally several times a week, and plugging leaks has become part of Microsoft's core business. The flood of patches has become so great that it's almost impossible to keep up with it. This is illustrated by the fact that most of today's security breaches successfully exploit leaks for which patches have already been released. In fact the sheer volume of patchwork eventually became sufficient to justify the automated distribution of patches. For recent versions of Windows there is an automatic service to notify the user of required "critical updates" (read: security patches) which may then be downloaded with a single mouseclick. This service (which does work fairly well) has become very popular. And for good reason: in the year 2000 alone MS released about 100 (yes, one hundred) security bulletins - that's an average of one newly discovered security-related issue every three to four days! The number of holes in Microsoft products would put a Swiss cheese to shame.

And the pace has increased rather than slowed down. For example, once you install a "recommended update" (such as Media Player) through the Windows Update service, you discover immediately afterwards that you must repeat the whole exercise in order to install a "critical update" to patch the new security leaks that were introduced with the first download! It's hardly reasonable to expect users to keep up with such a rat race, and not surprising that most users can't. As a result, many E-mail viruses and worms exploit security holes that are months or years old. The MSBlaster worm that spread in the summer of 2003 managed to infect Windows Server 2003 using a vulnerability that was already present in NT4!

In an age where smokers sue the tobacco industry for millions of dollars over health issues, all Microsoft products had better come with a warning on the package, stating that "This product is insecure and will cause expensive damage to your ICT infrastructure unless you update frequently and allocate time on a daily basis to locate, download, test and install the patch-du-jour". Unfortunately they don't, and Windows-based macro and script viruses emerge at a rate of several hundreds a month, while the average time for an unpatched Windows server with a direct Internet connection to be compromised is only a few minutes.

Patch release as a substitute for quality

An interesting side effect of the ridiculous rate with which patches have to be released is that some users now get the impression that Microsoft takes software maintenance very seriously and that they are constantly working to improve their products. This is of course rather naive. If they'd bought a car that needed serious maintenance or repairs every two weeks or so, they probably wouldn't feel that way about their car dealer.
Redmond has exploited this misconception more than once. In recent comparisons of Windows vs. Linux they quoted patch response times, in an attempt to show that Windows is more secure than Linux. They had of course to reclassify critical vulnerabilities as non-critical, misinterpret a lot of figures, and totally ignore the fact that Windows develops many times the number of vulnerabilities than any other product.

Even so, if Microsoft's patching policy was effective we'd have run out of security holes in most MS products about now, staring with IE. Obviously no amount of patching can possibly remedy the structural design flaws in (or absence of) Microsoft products' security. A patch is like a band-aid: it will help to heal a simple cut or abrasion, but it won't prevent getting hurt again, in the same way or otherwise, and for a broken leg or a genetic deficiency it's totally useless, even if you apply a few thousand of them. The obvious weak point in the system is of course the integration of application software into the OS. Microsoft likes to call Windows "feature-rich" but when they have to release an advisory on a serious vulnerability in Windows Server 2003 that involves MIDI files, it becomes obvious that the set of "features" integrated in Windows has long since passed the limits of usefulness.

Microsoft's solution: security through obscurity

Lately Microsoft lobbyists are trying to promote the idea that free communication about newly discovered security leaks is not in the interest of the user community, since public knowledge of the many weaknesses in their products would enable and even encourage malicious hackers to exploit those leaks. Microsoft's Security Response Center spokesman Scott Culp blamed security experts for the outbreak of worms like Code Red and Nimda, and in an article on Microsoft's website in October 2001 he proposed to restrict circulation of security-related information to "select circles". And it's all for our own good, of course. After all, censorship is such a nasty word.

In August 2002, during a court hearing discussing a settlement between Microsoft and the DoJ, Windows OS chief Jim Allchin testified how cover-ups are Microsoft's preferred (and recommended) course of action:

> *"There is a protocol dealing with software functionality in Windows called message queueing, and there is a mistake in that protocol. And that mistake, if we disclosed it, would in my opinion compromise a company who is using that particular protocol."*

In the meantime things are only getting worse with the lack of security in Microsoft products. The latest incarnation of Office (Office XP) provides full VBA support for Outlook, while CryptoAPI provides encryption for messages and documents, including VBS attaches and macro's. In other words, anti-virus software will no longer be able to detect and intercept viruses that come with E-mail and Word documents, rendering companies completely defenseless against virus attacks.

Clearly this is a recipe for disaster. It's like a car manufacturer who floods the market with cars without brakes, and then tries to suppress all consumer warnings in order to protect his sales figures.

Count the bugs: 1 + 1 = 3

Another worrying development is that leaky code from products such as IIS or other products is often installed with other software (and even with Windows XP) without the system administrators being aware of it. For example: SQL Server 2000 introduced 'super sockets' support for data access via the Dnetlib DLL. It provides multi-protocol connectivity, encryption, and authentication; in other words a roll-up of the different implementations of these technologies in past versions of the product. A system would only have this DLL if SQL Server 2000, the client administration tools, MSDE, or a vendor-specific solution was installed on the box. However, with XP this DLL is part of the default installation-- even on the home edition. One has to wonder how a component goes from "installed only in specialized machines on a particular platform" to "installed by default on all flavors of the OS." What other components and vulnerabilities are now automatically installed that we don't know about?

And the Windows fileset is getting extremely cluttered as it is. Looking through the WINNT directory on a Windows 2000 or XP system, you'll find lots of legacy executables that are obsolete and never used: Media Player 5, 16-bit legacy code from previous Windows versions as far back as version 3.10 (in fact the bulk of the original Windows 3.10 executable code is there), files that belong to features that are never used in most cases (e.g. RAS support) or accessibility options that most users fortunately don't need (such as the Narrator and Onscreen Keyboard features). Dormant code means possible dormant security issues. The needless installation of such a roundup reflects the laziness of Microsoft's developers: if you just install everything but the kitchen sink, you can just assume it's there at a later time and not bother with the proper verifications. Of course this practice doesn't improve quality control at all, it merely adds to the bloat that has plagued Windows from day one.

Expect no real improvement

The future promises only more of the same. Since Microsoft is always working on the next versions of Windows, it seems a safe assumption that we're stuck with the current flawed Windows architecture and that no structural improvements are to be expected. So far Microsoft has never seemed capable of cleaning up their software architectures. Instead they concentrate on finding workarounds to avoid the real problem.

A prime example was Microsoft's recommendation that PCs "designed for Windows XP" should no longer accept expansion cards but only work with USB peripherals. The reason for this recommendation was that XP and its successor Vista still suffer from the architecture-related driver problems that have caused so many Windows crashes in the past. In an attempt to get rid of the problem, Microsoft tried to persuade PC manufacturers to abandon the PCI expansion bus. The fact that this recommendation was immediately rejected by the hardware industry is irrelevant; the point is that Microsoft tried to get rid of expansion bus support rather than improve Windows' architecture to make it robust. When this attempt failed, they resorted to their second option, which was to discourage (in XP) and prevent (in Vista) the user from installing any device driver that hasn't been tested and certified not to cause OS instabilities.

In fact Microsoft does recognize the need for structural improvements in Windows' architecture. However, in spite of years of effort it has proven impossible for them to deliver any. This is perfectly illustrated by the early announcements of the .Net initiative, shortly after the turn of the century. Eventually .Net materialized as a framework for programmers to facilitate the development of network applications. During the first year or so of its initial announcement, though, .Net was presented as the future of desktop computing that was going to solve all deficiencies in Windows once and for all. Its most touted "innovation" was "Zero Impact Install" which amounted to doing away with the tight integration between application and operating system. Instead of the current mess of DLL's being inserted into the OS and settings spread throughout an insecure registry database, applications would live in their own subdirectories and be self-contained. Code would be delivered in a cross-platform format and be JIT-compiled (Just In Time) for the platform it was to run on.

While these things would have meant a dramatic improvement over the current situation, their innovation factor was of course close to zero: Windows' need for an adequate separation between OS and application code makes sophisticated ICT professionals long for Unix, mainframe environments or even DOS. JIT-compilation is nothing new either; it wasn't even a new idea when Sun Microsystems proposed Java in the mid-1990's. But more importantly, none of these changes ever materialized. Windows XP and Vista were going to be the first step to accomplish this enlightened new vision, but after the release of both Windows versions no trace of these bold plans remains. Instead the release of Vista was delayed by several years, because it was impossible to make even minor improvements without massive code rewrites.

So what we ultimately got instead was Windows Vista: a version of Windows so bloated with useless and annoying features that its sales have slumped in a manner never seen before in the history of Windows, and almost universially hated by the user community. As an illustration of how seriously Microsoft botched the release of Vista, Service Pack 1 for Vista involved a complete kernel replacement! Shortly after releasing SP1, though, Microsoft had to retract it in a hurry because it caused "updated" PCs to get stuck in an endless loop of reboots.

Microsoft will soon (read: sometime before the end of the decade) release Vista's succesor (Windows version 8) but especially after the dog's breakfast that Vista turned out to be, there's not much to hope for. Microsoft announced early in the development stage that Windows 8 is based on the same kernel and code base as Windows Vista.

Trustworthy computing? Not from Microsoft

Lately Microsoft has been making a lot of noise about how they now suddenly take security very seriously, but the bad overall quality of their product code makes it impossible to live up to that promise. Their Baseline Security Analyzer (which they released some time ago as part of their attempts to improve their image) was a good indication: it didn't scan for vulnerabilities but merely for missing patches, and it did a sloppy job at that with a lot of false positives as a result.

Let's face it: Microsoft's promises about dramatic quality improvement are unrealistic at best, not to say misleading. They're impossible to fulfil in the foreseeable future, and everyone at Microsoft knows it. To illustrate, in January 2002 Bill Gates wrote in his "Trustworthy computing" memo to all Microsoft employees:

> *"Today, in the developed world, we do not worry about electricity and water services being available. With telephony, we rely both on its availability and its security for conducting highly confidential business transactions without worrying that information about who we call or what we say will be compromised. Computing falls well short of this, ranging from the individual user who isn't willing to add a new application because it might destabilize their system, to a corporation that moves slowly to embrace e-business because today's platforms don't make the grade."*

Now, for "today's platforms" read "a decade of Windows, in most cases" and keep in mind that Microsoft won't use their own security products but relies on third party products instead. Add to that the presence of spyware features in Windows Media Player, Internet Explorer 7 and XP's Search Assistant (all of which contact Microsoft servers regularly whenever content is being accessed), the fact that Windows XP Home Edition regularly connects to a Microsoft server for no clearly explained reason, and the hooks for the Alexa data gathering software in IE's 'Tools/Show Related Links' feature... and the picture is about complete. Trustworthy? Sure! Maybe Big Brother isn't watching you, and maybe nothing is being done with the information that's being collected about what you search for and what content you access... and maybe pigs really can fly. For those who still don't get it: in November 2002 Microsoft made customer details, along with numerous confidential internal documents, freely available from a very insecure FTP server. This FTP server sported many known vulnerabilities, which made gaining access a trivial exercise. Clearly, Microsoft's recent privacy-concerned and quality-concerned noises sound rather hollow at best. They don't even have any respect for their customers' privacy and security themselves.

As if to make a point, a few weeks after Gates' memo on Trustworthy Computing, Microsoft managed to send the Nimda worm to their own Korean developers, along with the Korean language version of Visual Studio .Net, thus spreading an infection that had originated with the third-party Korean translators. How 'trustworthy' can we expect a company to be, if they aren't even capable of basic precautions such as adequate virus protection in their own organisation?

Of course nothing has changed since Gates wrote the above memo. Security holes and vulnerabilities in all MS products, many of which allow blackhat hackers to execute arbitrary code on any PC connected to the Internet, continue to be discovered and exploited with a depressing regularity. Microsoft claims to have put 11,000 engineers through security training to solve the problem, but all users of Microsoft products continue to be plagued by security flaws. It's obvious that real improvement won't come around anytime soon. Windows Server 2003 ws marketed as "secure by design" but apart from a couple of improved default settings and the Software Restriction Policies not much has changed. Right after Windows Server 2003 was released, its first security patch (to plug a vulnerability that existed through Internet Explorer 6) had to be applied, to nobody's surprise.

Lip service will do

Following Gates' memo on Trustworthy Computing, Microsoft has made a lot of noise about taking security very seriously. However, Stuart Okin, MS Security Officer for the UK, described security as "a recent issue". During an interview at Microsoft's Tech Ed event in 2002, Okin explained that recent press coverage on viruses and related issues had put security high on Microsoft's agenda. Read: it was never much of an issue, but now it's time to pay lip service to security concerns in order to save public relations.

And indeed Microsoft's only real 'improvement' so far has been an advertising campaign that touts Windows XP as the secure platform that protects the corporate user from virus attacks. No, really - that's what they said. They also made a lot of noise about having received "Government-based security certification". In fact this only means that Windows 2000 SP3 met the CCITSE Common Criteria, so that it can be part of government systems without buyers having to get special waivers from the National Security Agency or perform additional testing every time. CC-compliance does not not mean the software is now secure, but merely means the testing has confirmed the code is working as per specifications. That's all -- the discovery of new security holes at least once a week has nothing to do with it. But even so, Windows 2000 SP3 was the first Microsoft product ever that worked well enough to be CC-certified. Go figure.

Gates' initial launch of the Trustworthy Computing idea was much like the mating of elephants. There was a lot of trumpeting and stamping around the bush, followed by a brief moment of activity in high places, and then nothing happened for almost two years. Eventually Steve Ballmer made the stunning announcement that the big security initiative will consist of... a lot of minor product fixes (yes, again), training users, and rolling up several minor patches into bigger ones. Microsoft's press release actually used the words, quote, "improving the patch experience", unquote. So far this "improvement" has mainly consisted of monthly patch packages, which had to be re-released and re-installed several times a month in a 'revised' monthly version more than once.

Another sad aspect of Microsoft's actual stance on security is neatly summed up by Internet.com editor Rebecca Lieb, who investigated Microsoft's commitment on fighting the epidemic flood of spam. She concludes:

> *"[Microsoft] executives are certainly committed to saying they are [committed to helping end the spam epidemic]. These days, Bill Gates is front and center: testifying before the Senate; penning a Wall Street Journal editorial; putting millions up in bounty for spammer arrests; building a Web page for consumers; and forming an Anti-Spam Technology & Strategy Group, "fighting spam from all angles-- technology, enforcement, education, legislation and industry self-regulation."*
>
> *When I meet members of that group, I always ask the same question. Every version of the Windows OS that shipped prior to XP's release last year is configured --by default-- as an open relay. Millions have been upgraded to broadband. Ergo, most PCs on planet Earth emit a siren call to spammers: "Use me! Abuse me!" Why won't Microsoft tell its millions of registered customers how to close the open relay?"*

True enough, in 2004 over 75% of all spam is distributed via Windows PCs (on DSL and cable Internet connections) that have been compromised by email worms and Trojan Horse infections. But rather than fix the vulnerabilities in their products, Microsoft so far has concentrated on high-profile actions such as a collaboration with the New York State Attorney General and a highly publicized crusade against Internet advertising companies. Bill Gates' reckless prediction that in the year 2006 the spam problem would be solved has only served to demonstrate the value of Microsoft's promises on quality and security.

Neither is Microsoft's own implementation of Sender Policy Framework even remotely effective. Microsoft touted their use of SPF as a significant step in spam reduction, and introduced it with so much fanfare that you'd think they'd developed it themselves. However, Security appliance firm CipherTrust soon found that spammers adopted the new standard for email authentication much faster than legitimate emailers, and shortly after its introduction more spam than legitimate email was sent using Sender Policy Framework. While this is going on, implementors are balking at MS's licensing policy for the Sender ID system, which amounts to creating great dependency on Microsoft's permission to (continue to) use Sender ID.

Meanwhile Microsoft is branching into new markets and so far runs true to form. They have already shipped their first cellphone products. Orange, the first cellnet operator to run Microsoft Smartphone software on their SPV phones, has already had to investigate several security leaks. On top of that the phones are reported to crash and require three subsequent power-ups to get going again, call random numbers from the address book and have flakey power management. I shudder to think what will happen when their plans on the automotive software market begin to materialize.

Bottom line: things are bad

In spite of what Microsoft's sales droids would have us believe, the facts speak for themselves: developments at Microsoft are solely driven by business targets and not by quality targets. As long as they manage to keep up their $30 billion plus yearly turnover, nobody's posterior is on the line no matter how bad their software is.

Microsoft products are immature and of inferior quality. They waste resources, do not offer proper options for administration and maintenance, and are fragile and easily damaged. Worse, new versions of these products provide no structural remedy, but are in fact point releases with bugfixes, minor updates and little else but cosmetic improvement. Recent versions of Microsoft products are only marginally more secure than those that were released years ago. In fact, if it weren't for additional security products such as hardware-based or Unix-based filters and firewalls, it would be impossible to run an even remotely secure environment with Windows.

MS products are bloated with an almost baroque excess of features, but that's not the point. The point is that they are to be considered harmful, lacking robustness and security as a direct result of basic design flaws that are in many cases over a decade old. They promise to do a lot, but in practice they don't do any of it very well. If you need something robust, designed for mission-critical applications, you might want to look elsewhere. Microsoft's need for compatibility with previous mistakes makes structural improvements impossible. The day Microsoft makes something that doesn't suck, they'll be making vacuum-cleaners.

Besides, 63,000 known defects in Windows should be enough for anyone.

3. The power to bind

"One OS to rule them all
One OS to find them
One OS to bring them all
And in the darkness bind them.."

-- With apologies to J.R.R. Tolkien

Although an apology to Tolkien is probably in order here, the similarities between the Rings of Power and the various Microsoft products are in fact striking. They will subtly try to take control of you, and every time you give in to the temptation to use one, the Dark Lord's power increases.

And in this respect Microsoft's control over document formats and standards for data exchange are most certainly the One Ring of Power.

A bad record

When it comes to supporting of the global standards used in today's IT market, Microsoft's record has never been good. They have always been extremely sloppy in following the standards' specifications, they have attempted to appropriate the standards for HTML, Java, E-mail and more, and they have tried to push proprietary standards that are only supported by Microsoft applications. Fortunately, the Internet community has resisted most of these attempts so far, although their efforts of recent years to "extent" XML (starting with the 'Global XML Web Services Architecture' initiative, and more recently through the .Net framework) doesn't bode well. Neither do Microsoft's applications to patent the "XML-Office" format. Microsoft obviously does not consider XML a format that should promote any increase in document interchangeability.

In the hardware market, especially where peripherals are concerned, compatibility is also deliberately being limited. Far too often the label 'Designed for Windows' means 'incompatible with anything else'. We've seen modems and printers that had no standard interface or hardware API but a proprietary Windows driver instead, and we'll see more of it. For example, before Windows XP was released Microsoft tried (but fortunately failed) to persuade PC manufacturers to discontinue the PCI bus and to support USB devices only. Older versions of Windows, as well as Linux and other Open Source products either have limited or no USB support, or drivers for USB devices are either unavailable or difficult to obtain.

But it's the application market where things are most serious. Microsoft's huge market penetration has flooded the world with documents in all kinds of proprietary formats. According to calculations by Gartner, switching from MS Office to the OpenSource alternative OpenOffice or StarOffice alone will cost, on average, $1200 per user, mainly for document and macro conversion, learning a new user interface and lost productivity during the migration.

Deliberate hurdles

The stranglehold that MS has on the IT market is a major problem for those who work in a multi-vendor environment. Microsoft applications always produce documents in a Microsoft-proprietary format, and they never run on anything but the Wintel platform. (OK, Microsoft ported Internet Explorer to Solaris and the Mac during their marketing war against Netscape, and they resurrected MacOffice as part of their anti-DoJ strategy with a few half-hearted versions for the Mac that are full of compatibility issues. But that's about it.) The net result of this is that when someone sends you a recent PowerPoint or Excel document and you don't have Windows, you may have to switch operating systems in order to view or edit it.

(You may fail to see how ridiculous this really is. After all, it's only reasonable to expect office personnel to run Windows with MS-Office, isn't it? Well... Think about it. Things have deteriorated to a point where many people actually look at it this way.)

Here's what **Forrester Research** said about Microsoft and standards:

Why Microsoft "Standards" Do Not Help They:

- *Work only for Windows (thus leaving out all other systems that do not run Windows and are unlikely to do so in the future)*
- *Increase Support Demands (since techies still must load, update and maintain proprietary code on every computer)*
- *Restrict Creativity (since Bill Gates' troops are defining the generic software layer, MS can tailor the interface to match its own technology biases -- and shut out competing ideas*
- *This is hardly in the best interest of IT.*

Apart from the obvious inability to process documents from a Microsoft application in a non-Microsoft environment, there's also the limited inter-operability with Microsoft products that makes life in a multi-vendor environment difficult. Microsoft's deliberate use of proprietary standards, proprietary extensions to global standards, and their own incorrect implementations of global standards, often makes it extremely complicated to use Microsoft products in anything but a Microsoft-only environment.

Microsoft 'standards': sloppy work

Some of Microsoft's standardization efforts are relatively harmless and merely irritating. The Joliet CD-ROM format extensions, for example, are annoying but not really a problem. In the initial Joliet specifications Microsoft claimed that Joliet was needed in order to enhance the interchangeability of CD-ROM based data, since the bare ISO-9660 specs were too restrictive. Granted, ISO-9660 is far from luxurious, but MacOS and Unixen already had extensions to ISO-9660 that worked quite well and that could have been adopted right away. For a long time Windows was the only platform that could handle Joliet CD-ROM's. Joliet support doesn't even extend to the boot disks that come with Windows 9x/ME: since these are DOS-based they can only handle ISO-9660. That means that even Windows' own installation software cannot read Joliet CD-ROM's. In order to make their own installer work, Microsoft still has to ship Windows 9x/ME on ISO-9660 CD-ROM's. How's that for Joliet's enhanced data interchangeability?

Another annoying example is the use of MS-TNEF file attachments in E-mail. This is Microsoft's idea of using 'Rich Text' in E-mail messages, so that extensive formatting (different fonts, headers, italics, etc.) becomes possible. If you tell MS-Word to E-mail a document, this is what you get. Unfortunately, if the recipient of such a document happens not to use Outlook to read E-mail, his or her PC doesn't recognize the MS-TNEF format, and trying to use MS-Word to read it won't work either. In other words, if a Microsoft application uses 'Rich Text' to send a nicely formatted E-mail message, the recipient of that message is forced to use Outlook (and therefore Windows) to read it.

Outlook also sends messages that contain all kinds of embedded bells and whistles, MIME-attached business cards, message text in HTML format, embedded VBS, et cetera ad nauseam. This nonsense will be attractively displayed if the message recipient also uses Outlook (i.e. Windows) but is only so much garbage when viewed with any other RFC-compliant message editor.

More than just annoying

Other issues are more serious. For example, you cannot format a volume larger than 32 GB in size using the FAT32 file system in Windows 2000 and later. The Windows FastFAT driver can mount and support volumes larger than 32 GB that use the FAT32 file system, but you cannot create one using the Format tool. This behavior is by design. If you need to create a volume larger than 32 GB, use the NTFS file system instead. When attempting to format a FAT32 partition larger than 32 GB, the format fails near the end of the process with the following error: Logical Disk Manager: Volume size too big.

There are good reasons why one would want large FAT32 partitions (for example, it might be necessary to share data on such partitions with other operating systems) but Microsoft forces you to use their own, proprietary NTFS instead. This creates an artificial barrier for non-Microsoft products for no good reason whatsoever and overrules the customers' needs.

A side-effect of Microsoft's indiscriminate use of non-standards and their sloppy implementation of standards they happen to support, is that third-party developers are under increasing pressure to support Microsoft's deviations. If software is confused by the way Outlook or Exchange treat message headers, customers who use Outlook tend to demand that all products must handle mail from Outlook correctly, reasoning that since Outlook is the most widely used client today, all the world must be expected to support its quirks.

Things get worse

While these impediments to information exchange are extremely annoying, they're not all that bad in comparison to other issues. Microsoft violates a number of 'best practices' that aren't really standards but that nevertheless ensure interoperability. For example, it's common practice not to use whitespace in filenames, since many operating systems use whitespace for command parsing and not all file systems can handle it. But Windows uses filenames that contain whitespace by default, which often leads to problems when Windows systems send or share files through non-Microsoft networking technology. For example, using FTP on a Unix file system that also acts as an SMB share for Windows clients often causes problems at the Unix end.

There are also deviations from the official, stricter standards. The sloppy implementation of the SMTP protocol in the Microsoft Exchange mail server, for example, can affect the flow of E-mail in other parts of the Internet and in non-Microsoft E-mail servers. Microsoft's Directory Services products tout being LDAP and ODBC compliant, but if you want an X.500 directory or a sync engine (other than MS DirSync) to talk to it, you'll need custom meta agents.

These issues are more than just annoying; they are serious hurdles in a multi-vendor environment (such as the

Internet) and mean a possible disruption to the interoperability of both Microsoft and non-Microsoft systems.

Standards as a means of sabotage

Then there are the really nasty deviations from global standards that Microsoft has deliberately introduced to sabotage interoperability and freedom of choice. Take HTML and Java for example. The Frontpage web editor, the IIS webserver and the various backend E-commerce products all generate proprietary extensions to HTML and scripting languages that only Internet Explorer on Windows will handle correctly, and renders all other web browsers and platforms unusable. The same goes for Java support: Microsoft Java does not follow the Java specifications. Again this means that applets in this particular dialect can only be executed by Microsoft's own Java engine on Windows. Yet Microsoft used the Java label for products that were incompatible with the Java standard. This caused Sun to file suit. Microsoft then dropped the global Java standard entirely and now only supports their own Java dialect. The net result of this whole procedure is that Microsoft web server products and development tools generate code that needs Windows, Internet Explorer and the Microsoft Java engine at the user end in order to work properly.

So the use of a simple consumer-level HTML editor like Frontpage can be the start of complete vendor-dependence. Frontpage is mainly intended for consumer use, and at the professional end of the scale we have IIS, ASP, scripting and other dynamic technologies, and the backend and development tools to create them. The World Wide Web becomes flooded with non-compliant HTML and JavaScript code that generates error messages, or that works incorrectly or not at all, with Netscape Navigator or other non-Microsoft browsers. Only with Internet Explorer on Windows can these websites be displayed correctly. Recent versions of Frontpage, IIS and the assorted E-commerce solutions increasingly use this proprietary scripting code for menus and navigation. This makes correct support of these dialects (read: the use of IE on Windows) essential to the usability of a website.

And this is not a transient problem, because competing browser manufacturers can never keep up. Shortly after Microsoft releases an updated version of Frontpage, IIS or other backend or development tools, older browsers will begin to show more and more error messages, and users will be urged to upgrade to the latest version of Internet Explorer. Some Microsoft web server products even use Active-X. That means that if you access a website that uses Active-X (another Microsoft 'standard'), the server sends commands to your browser which then makes system calls directly into your Windows operating system code. These websites can only be fully and correctly accessed and displayed by clients running MS-Windows. Non-Wintel systems (e.g. workstations running Unix) are excluded from those web-based services.

Also, in 2003 Microsoft participated in the Web Services workgroup of the W3C (the World Wide Web Consortium, a committee that maintains and guards the global web standards that we all use) with hopes of getting some Microsoft proprietary technology ratified as a global standard. Since the W3C was unwilling to do this (read: to promote technology to be used on a royalty basis) the Microsoft representatives picked up their marbles and left, stating that the purposes of the W3C did not match those of Microsoft. Shortly thereafter Microsoft said that no major new versions of Internet Explorer as a separate product are to be expected,, and announced that future major releases of IE will be an integrated part of future (post-XP) Windows versions. Given the dependencies they've created, this means that in order to access information on a global network, we'll need to buy the latest version of Microsoft Windows.

Deliberate censorship

Microsoft's contempt for HTML and related global standards is nicely illustrated by the way Internet Explorer 5.5 and 6 alter HTML code before presenting it to the user. For example, a web page header might contain a directive that an ISO (i.e. platform-independent) character set be used, with the following command:
```
<META HTTP-EQUIV="Content-Type" CONTENT="text/html charset=ISO-8859-1">
```

But after downloading and saving the web page source code with Internet Explorer 5.5 or later, this line looks quite different:
```
<META http-equiv=Content-Type content="text/html; charset=windows-1252">
```

IE's default behavior is to quietly modify the content of third-party web pages. This default can be changed by doing something in the View->Encoding menu that many users will find obscure and few will bother with, but that's not the point. The point is that without such tweaking a user will never see the original directive! And since many web developers reuse code or at least look at it (many of us built their first web page by modifying existing code found on the web) this platform-specific nonsense will propagate. Granted, it's a small detail, but it's another typical step in spreading the notion that "all the world is a Windows PC".

And then there's of course the principle of the whole thing. If Microsoft will insert code into IE to modify a character set declaration (that the author of the web page supposedly put in there for good reason) this proves that they're not above using their products to manipulate the information that eventually reaches the end user.

In fact this is a form of censorship, intended to manipulate the general perception on what practice is customary and what's not. Microsoft has become the IT market's demagogue, not by controlling the media, but by taking control of global protocol standards. Proprietary standards and products are presented so that they appear to have been a general standard all along. And where that's not possible or convenient, Microsoft encourages the notion that global standards are Microsoft's inventions. For example, look in the Windows 9x/ME control panel for the TCP/IP protocol. You'll find it listed under 'Microsoft protocols', along with NetBEUI. How's that again? (TCP/IP is a global standard that's older than the Microsoft Corporation.)

Microsoft not only seizes global standards but also attempts to rewrite history -- a practice not uncommon among those who want to control public opinion.

IE6, 7 and 8 - a change of policy?

With the release of Internet Explorer 6, Microsoft finally seemed to respond to criticism from the Internet community. In what looks like an attempt to play by the rules, IE6 can be made to be less forgiving about the HTML and CSS errors that earlier versions of IE would take in stride. However this is not its default behavior, but has to be enforced with the right document type declaration in the webpage. And of course IE6 is still full of old code (which, for example, creates "web archive" MTHML files that carry the label 'Saved by Microsoft Internet Explorer 5'. So much for version management and quality assurance...

It's also fairly significant that MS had to put an entire second rendering engine in IE6 to achieve an acceptable level of standards compliance, which says a lot about the rendering engine in previous versions of IE. Then there's the annoying fact that the Windows and Macintosh builds of Internet Explorer are based on different rendering engines, and therefore react differently to the same web page. Also some new bugs are only present in the new second rendering engine, so ironically enough the only workaround for these bugs is to write HTML code that deviates from the standards enough to throw IE into 'quirk mode' so that the older, non-standard rendering engine is used.

IE7 is much the same in this regard. It has a few extra features that its chief competitors FireFox, Safari and Opera have had for years, and boast as many bugs and security vulnerabilities as its predecessors, several of which surfaced within hours of its initial release. Not much else is new.

Still the improved standards compliance in IE6 and 7 might be a good sign. On the other hand, Microsoft has also submitted proposals to the World Wide Web Consortium (W3C) for ratification as a global standard. At first sight this seems to be a Good Thing, until you read the small print in the submissions: Microsoft reserves the right to charge royalties for use of their technology even after it's been ratified by the W3C as a global Internet standard. In other words, they're now trying to stick an 'independent standard' label on their own protocols and formats, and still charge for their use as proprietary technology. This would effectively enable them to charge royalties on Internet traffic that uses a globally standardized protocol proposed by Microsoft. Will they try and get away with this? Only time will tell.

But it doesn't look good. Even during the development stage of Internet Explorer 7, new issues presented themselves. Microsoft has announced that IE7 will never conform to WaSP standards, because this would render websites that have been developed for IE compatibility inoperable. Furthermore Microsoft never planned to fully support the latest CSS standard in IE 7.0. Instead of using well-established Web standards, IE 7.0 will continue to foist proprietary technologies on Web developers, forcing them to choose between two competing and mutually incompatible ways of creating Web sites. To top it all off, during the beta stage of IE8 Microsoft issued several warnings about compatibility, and urged web developers to "prepare their websites" for IE8. Translation: after having battled to work around the quirks in IE6 and be punished for it when IE7 came along, now web developers are presented with a whole new set of headaches because IE8 breaks the websites that had to be "adapted" for IE7.

In the meantime Microsoft continues to flood the market with products that use proprietary technology rather than support global standards. We've got DNS on all platforms and NIS on Unix -- Microsoft creates WINS. We've got NDS on Novell NetWare, SUN Solaris, (Free)BSD or Linux, supported by major peripheral vendors -- Microsoft comes up with Active Directory, only available on recent versions of Windows. We have printers and modems that will work with any system -- Microsoft comes up with an API specification that enables hardware manufacturers to build slightly cheaper devices but requires a proprietary interface in the form of a Windows driver. And the list goes on and on.

The bigger picture

Microsoft doesn't mess with standards for nothing. Microsoft's interests go way beyond controlling the application market.

A good example is the "Browser War". Microsoft ignored the Internet completely until Netscape made sudden and huge profits, whereupon Microsoft decided they wanted that market share for themselves. Being the biggest fish in the pond, they just took what they wanted, by bundling their own Internet Explorer with Windows. This killed off the innovative Netscape in short order. A takeover by AOL didn't save Netscape, and in 2003 Microsoft bought off the whole conflict for a mere $750 million; big money to most of us but chump change for Microsoft. Less than a week later Microsoft announced the discontinuation of IE for the Mac.

This browser war was not just a battle between two competing application developers (Microsoft vs. Netscape) for the biggest market share. It went much further than that: it was a conflict between two philosophies, between vendor-dependence and vendor-independence. On one side we had Microsoft, pushing a product that was (and is) tightly bound to the Windows platform, and on the other side there was Netscape, promoting a product that was (and is) available for many different environments. Microsoft intended (and largely succeeded) by adding MS-specific features to Internet Explorer, Java, scripting and HTML, to commit the Internet community to the Windows platform. They've also used product bundling and other doubtful and illegal methods, and finally succeeded in forcing Netscape out of the browser market. As a result, more and more Internet services now expect the user to run IE and Windows on their Internet client systems. If you access E-commerce websites based on Microsoft IIS and backend products with anything else but Internet Explorer, you can expect rendering and scripting problems.

This has led to an interesting side effect: intimidated by the many browser dependencies in Microsoft-related web page code, web designers have taken to displaying an error message instead of a web site whenever their products are accessed with anything but Internet Explorer. Competing browser manufacturers soon reacted to that with masquerading: browsers like Opera just tell the server that they're another Internet Explorer. Of course this has boosted the statistics to the point where 95% of the world is said to use IE, which is exactly what Microsoft wants everyone to hear. More advanced browser detection however shows that the real figure is closer to 60%.

It's also interesting to note that the User Interface in Windows XP is based on a mix of XML and proprietary HTML-derivatives, that can be understood only by portions of Microsoft's own Internet Explorer application code. This ensures the need for a fully integrated version of IE that cannot be removed or replaced by a standards-compliant browser without losing important UI functions (such as online help pages). Support for the platform-independent Java programming language on the other hand has been dropped entirely from Windows XP, which leaves only Microsoft's own scripting and language support. Microsoft claimed that this was done in response to legal actions from Sun, but in truth the legal agreements allow them to ship Java with their products for several years to come -- provided they follow the Java standard. Rather than doing this, Redmond's marketeers decided to remove Java from Windows XP entirely. Even Microsoft hasn't been able to explain how this benefits the user community. First they attempted to corrupt Java by creating their own incompatible variety, but Sun filed suit and won. Under the conditions of that legal settlement, Microsoft was forbidden to ship their own incompatible product and call it Java, but they were allowed to support 'pure' Java. In an open letter to their customers, Microsoft claimed that "Sun resorted to litigation to stop Microsoft from shipping a high performance Java virtual machine that took optimal advantage of Windows" and that "Sun's idea of choice is that you can have any language you want, as long as it is Sun's version of Java under Sun's control." So now Microsoft tries to kill of Java (and platform independence) by removing Java support from Windows. After accusing Sun of restrictive policies, they now allow their own customers to use only Microsoft's version of Java under Microsoft's control. Of course the latter is available on Windows only, while Sun's Java may be implemented on any platform by any developer, provided that the Java standard is followed and compatibility issues are respected.

Corrupt standards rather than innovate

These are all examples of how Microsoft attempts to limit the users' freedom of choice rather than compete by making better software that respects global standards and that can interact and coexist with third-party products in a multivendor environment. While most other major software developers contribute to Open Source Software and thereby create new markets, Microsoft continues to slander OSS. Open Standards are commercially unacceptable for Microsoft. Open Standards and cross-platform technology bring people together. Microsoft on the other hand imposes deliberate barriers between them; artificial walls that only exist to create an artificial need for Gates and Windows.

In November 1998 an internal memo leaked out of Microsoft which clearly stated that Open Source software not only performs and scales much better than Microsoft Products (it discussed especially the quality and availability of Linux), but also proposed that Microsoft attack these superior products by "de-commoditizing protocols". In other words, when faced with a superior competitor, Microsoft's preferred approach is to corrupt global standards and to introduce proprietary protocols that bind the user to the Microsoft environment.

Don't believe me, see for yourself. Read the Halloween documents that have been made available by The Open Source Foundation. (Microsoft has acknowledged the authenticity of these documents.) It's interesting reading. Very.

A good example of this policy in action (apart from the HTML and Java deviations described above) is Microsoft's attempt to appropriate the Kerberos protocol. Kerberos is an authentication protocol developed by MIT, distributed as Open Source software. Microsoft added an "innovative improvement" to Kerberos, by misusing a reserved field to specify whether or not an NT machine was allowed to authenticate another Kerberos system, rendering this corrupted version of Kerberos incompatible with Open Source versions in the process. (The misuse of a reserved field, or any field for that matter, is of course a gross violation of protocol standards.) Then Microsoft went on to state that they had "created" an "improved version of Kerberos", called the result their own intellectual property, and threatened to sue anyone who would dare to put it in their software, including Kerberos' inventor MIT. Only the global uproar that followed caused Microsoft to reconsider this nonsense.

With the above and other standards deviations in mind, it's rather ironic to read what Microsoft wrote on a web page that opposed the Open Source initiative: "The next generation Internet can only come into existence if we have standards that everyone adheres to." (From the white paper "Shared Source" by Michel van der Bel, General Manager Microsoft Netherlands.) Apparently we can only have a workable Internet if Microsoft exclusively dictates proprietary standards, and everyone obediently follows them. Is this a developer license agreement which I see before me?

Converting the world to Microsoft standards

This is where Tolkien and the Rings of Power come to mind. Use any of the above products or technology, and you're automatically committed to Windows. Use Windows, and Microsoft applications and protocols will automatically come with it, and more will follow. Slowly but surely, and often unnoticed, you are bound to Microsoft. Appropriating standards for data exchange is only part of the job. Application developers are bound to Microsoft just as much as the user community is. Only in this case development standards, methods and programming environments play the crucial role.

Microsoft has always taken good care of application software developers. They learned a lot about the care and feeding of developers when they were working on Windows 3.0 and few third party developers were willing to adapt their application code to the new platform. This was not surprising: Microsoft's strategy was rather muddy at the time, IBM was preparing lawsuits against MS because the new Windows code was full of technology that IBM felt MS had stolen, and few third parties felt there was any percentage in partnerships. The initial Windows specifications weren't too clear, and what its future would bring was anyone's guess. So nobody knew what kind of a life cycle a partnership and the products resulting therefrom would have. All that was clear to application developers, in 1989, was that 90% of their existing DOS application code would have to be rewritten entirely. This meant that developing applications for Windows meant a huge investment with doubtful returns. As a result there were almost no native third-party Windows applications when Windows 3.0 was released. Initially all that kept Windows alive was its backward compatibility with DOS.

Microsoft learned from all this, and Gates inspired trust by stating Microsoft's total commitment to Windows (and, it must be said, by putting his money where his mouth was). From then on, application development preceded each new release of Windows instead of following it. Long before the first beta version of Windows '95 became available, developers had been working with freely available SDK's and other development tools in order to have their new application code ready by the time Windows '95 hit the market. Long before Windows '95 had reached beta stage, developers would get all the inside information they needed about the operating system, API's would be readily available, pre-release versions of Windows could be used for testing, and all development tools could be obtained for a song.

Corrupting the developer

Unsurprisingly, the wealth of cheap tools and information attracted many developers to the Windows environment to create new applications. Of course those applications became tightly bound to the Microsoft platform, because of Window's closed architecture and API's; Windows code is even less portable than DOS code was. But by the time that realization took hold, most developers had already made a huge investment in development costs and efforts, which they were reluctant to write off.

In all honesty it must be said that Microsoft development tools are attractive. They do allow you to integrate different components of Microsoft products with relative ease, and they offer a framework that offers many options to develop applications rapidly. Writing, say, a Visual Basic front end for an Access database, or web-enabling that same database (or an MS-SQL database) withouth having to invent the wheel first, is largely a matter of using the tools and following the manuals. But in doing so, you have accept certain limitations. These techniques only work well in a Microsoft environment (which means that you have to take performance and availability issues for granted) and they tie your investments to the Microsoft platform indefinitely. Migrating to other platforms would require you to abandon your investment and start over.

Unfortunately most people don't pay much attention to software development tools. While the world argues, debates and even sues over Microsoft's dominance over applications and operating systems, the software we use every day is created by developers. And Microsoft quietly controls the developers' hearts, minds and digital tool chests. This is a most insidious way to stamp out competition. As time goes by, it becomes more and more difficult for developers to make a competing product or to support competing technology.

Because of the stranglehold that MS has on the IT market, the majority of both users and developers don't even realize that not all the world is a PC. Users rarely see anything but Microsoft applications and documents in some Microsoft format, and yesterday's users have become today's developers.

Lately, MS is trying to ram ADSI down developers' throats for creating new directory service interfaces, and the poor schmucks don't even realize what's going on. All they see is a toolkit that's easy to use and that comes with a user interface like all the other MS development products. They get their project done in half the time, and before they know it they've produced a Microsoft "standard" interface that borders on the proprietary. Then they're stuck with that non-portable code forever, or they'll lose their investment.

As a result, companies are forced to implement technologically challenged operating systems in order to do trivial things like exchanging documents, and to buy frequent updates of application software for the same purpose. But even diligently updating Microsoft applications is a two-edged sword: Service Pack 3 for Office 2003 removed the product's capability to read certain older document formats. It's a nice catch-22: you're stuck if you don't update, and you're stuck if you do.

Media standards

Microsoft's utter disregard for any standard that they haven't created themselves keeps getting in the way of real technological improvement. This is not surprising, of course. After proprietary standards have gained a solid foothold in the IT market, they'll provide Microsoft with more leverage to push Windows and Windows applications, and the cycle repeats itself. For example, Windows Media Player 7 and up didn't run on Windows NT4, and soon most of the content available on the World Wide Web will require Media Player 8 or later in order to be accessed. Microsoft will make sure of that through "strategic" partnerships and developer tools that crank out only the latest file formats. This effectively forces customers to upgrade to the latest version of Windows (and perhaps buy new and bigger PCs as well) in order to continue their access to multimedia content.

May 2003 saw Microsoft introducing their own DVD format, with movies only playable on Windows Media Player 9. Microsoft also commented that "...the forthcoming Office 2003 productivity software suite will enable users to designate who can open a document or email message, and specify the terms of use - for example, whether they can print, copy or forward the data. A rights management add-on for Internet Explorer will extend these protections to Web content." Great. In other words, you won't be able to do business with people using Microsoft's rights management unless you're using Microsoft's rights management too. And why did Microsoft bother with MSN and MSNBC for years and years, while all these divisions did was add red ink to the books? I'll give you a hint: after putting Media Player and their own brand of Digital Rights Management into Windows, Microsoft has now begun to sell music and other content.

Obstructing portability

None of this is actually conducive to technological innovation. Applications could run on any platform, and data formats could be documented and freely usable. The Open Source community and much of the Unix application market thrives on such platform independence. (And so does most of the Internet.) And not only Open Source software is platform-independent. A web browser like Netscape Navigator, for example, runs on all flavors of Windows, OS/2, a dozen flavors of UNIX, and MacOS. The Opera browser is available for several OS'es as well. DivX video tools exist on Windows, Unix, Mac and more. And many other serious applications, from database servers and clients to office suites, are available for multiple operating systems. These applications are typically developed on a non-Windows platform (thereby avoiding Windows' inherent portability issues) and then ported to Windows in response to market demands.

I generally use about 20 different applications. At the office I may run them on several systems, varying from desktop Unix workstations to the bigger midrange systems (Solaris, AIX), the occasional Apple Macintosh, and even Windows PCs. When I'm working at home or with customers, I run the same applications on Linux, in DOS boxes or on Windows. None of these applications, needless to say, were developed by Microsoft or Microsoft Partners, and few were originally developed on the Windows platform. And of course most of them can't handle documents in one of the many proprietary and barely documented formats used by Microsoft applications.

Recently some attempts have been made to enable Microsoft applications to run on other operating systems than Windows, with varying degrees of success. However, there is a clause in the Office XP End-User License Agreement, which stipulates that Office XP be used only on top of a Microsoft operating system. Apparently, Microsoft is unwilling to rely on technical impediments to interoperability only, and has decided to throw in legal obstructions as well to limit the users' freedom of choice.

Divide and conquer

It's a simple strategy: divide and conquer. Prevent information exchange through proprietary protocols and formats. Encourage or force users to use your applications and bind them to your own protocols. Make them use software that will only run on your own platform. Pollute existing global standards as much as possible, and induce service providers to use proprietary protocol extensions. Force your business partners to do the same. Do not provide new opportunities for the user community by offering better technology, but instead sabotage the interoperability of other products and standards, and smother any progress and development of competing technology.

Then make the users believe that they they benefit from "these exciting new innovations" and point out that adherence to existing standards would impede progress. Control the tools used by the developers who create the applications that we use every day, and make it more and more difficult for them to follow a competitor's path. Bind users and developers to the Windows platform, slowly but surely, one step at a time, until it's impossible for them to escape.

That's how it's done... in the Land of Redmond, where the Shadows lie.

4. World domination

"I am Billgatus of Borg. Resistance is futile."

Microsoft has been compared to the Borg Collective more than once. Indeed, you don't have to be a hard-core Star Trek fan to notice the similarity between Microsoft and the Borg. Microsoft's marketing methods have always shown a certain hunger for power, but lately an undisguised megalomania has set in.

"WE ARE MICROSOFT. LOWER YOUR FIREWALLS AND SURRENDER. WE WILL ADD YOUR TECHNOLOGICAL DISTINCTIVENESS TO OUR OWN. YOU WILL BE ASSIMILATED. RESISTANCE IS FUTILE."

Competitors beware

It's long been known that to oppose Microsoft means certain death (commercially speaking of course). Microsoft's marketing division just tramples the corpse of anyone who thinks he can shift it left or right. In fact, during its recent competitive struggles with information provider Google, Microsoft has stated that they want to destroy Google. Not compete with it, but destroy it. Google provides some of the most popular and useful services currently available on the Internet, but Microsoft would like to see it destroyed. The fact that they have not been able to carry out this threat hardly matters; their position is clear.

Microsoft's stragegy (and, therefore, its technological developments) are directed only at extracting more and more money from the customer, and at continuing to do so in the future. The customers' needs are irrelevant. Microsoft's preferred way of accomplishing this prime directive is to sabotage alternatives to Microsoft products and to use any means available to eliminate competitors, rather than to bring real technological innovation.

Microsoft's sheer marketing power has grown to the point where it can hurt competitors even by merely threatening them. We see this curious effect throughout the entire software market: as soon as Microsoft targets a certain part of the market, something happens to the competing market leaders. They start to falter. The value of their stock market shares drop. Their strategy becomes erratic and looses focus, and ultimately the quality of their products suffers. Novell and Netscape, to name a few good examples, have lost a good deal of their market share this way. Of course these companies have made mistakes. Of course they have ruined the potential of superior products with bungled marketing and disastrous commercial strategies. Of course Apple, IBM and all the others have done the same. Of course they only have themselves to blame for an inadequate reaction to a threat they should and could have expected. Of course the ability and the guts to deal with competitive pressure is a required part of doing business: if you can't stand the heat, stay out of the kitchen. But even so, their fairly typical behavior is a good illustration of the blind fear that Microsoft's business practices have managed to instill in would-be competitors, because everyone knows that Microsoft's use of FUD campaigns, corruption of standards, forced partner agreements, product bundling and other monopolist practices have become almost impossible to counter.

This fear is not without ground. Competitors who offer alternative and sometimes better technology are ruthlessly crushed, not because MS offers a better product but because Microsoft can manipulate the users and the software market to cut off anyone's oxygen supply without even making a dent in their profits, and still have their marketing division make enough noise to drown out all the other players in the market.

Forced sales

Microsoft has had PC manufacturers by the short-and-curlies for years: if integrators wanted to pre-load an OEM version of Windows on the computers they sold, then they had to discontinue all products from Microsoft's competitors. If they wouldn't sign such a contract to bundle a pre-loaded version of Windows with all their shipped systems, they'd face a hefty increase in Windows license fees. In other words: either they had to sell a copy of Windows with all their shipped systems and nothing else, or they would face retaliatory measures from Microsoft, which meant that they wouldn't be able to offer a copy of Windows at a competitive price. Only in recent years the largest PC manufacturers have been allowed to support Linux to a certain degree. (Note that the above may not apply in all countries and to all OEM manufacturers. Local policies may differ, and smaller system integrators pay different prices and have different contracts with Microsoft than huge companies do. Your mileage may vary.)

This strategy of forced sales is an old one: the same has been done in the past when PC vendors were forced to bundle Windows 3.1 with new PCs in order to be allowed to ship MS-DOS. Even before they modified Windows 3.x to crash when it detected the presence of DR-DOS instead of MS-DOS, Microsoft adopted several tactics to destroy DR-DOS. The most damaging of these was tying PC makers into secret per-processor license agreements, which meant that they paid for Microsoft's MS-DOS whether they shipped it with the PC or not, foreclosing the most important route to market.

Also note how Microsoft makes is cumbersome to legitimately re-use a Windows XP or Vista license. The license is tied to the computer's hardware, both with the activation system and with the requirement to affix the serial number of the license to the computer's exterior. To remove the software from one machine and transfer it to another one is a pain, even though such a transfer is normal practice with other software products, and the Windows End User License Agreement (EULA) does not prohibit it. Yes, it is possible to legally remove Windows from one PC and install it on another one, but that requires a telephone call to Microsoft to have the transfer authorized.

As if this wasn't bad enough, Microsoft arbitrarily changed the OEM license agreement several years after the release of Windows XP, to state that replacing the mainboard of a computer essentially creates a new computer, which requires a new license for the operating system, and they sent a memo to its OEM partners requesting to enforce this new policy.

It is interesting to note is that second hand and refurbished personal computers meke up about 10% of the world's global PC sales. Many of these second hand computers are shipped with the Windows license that they were originally sold with when they were new, but during the refurbishing process (which includes wiping the harddisk) a new, and often generic, version of Windows is installed. Microsoft does not allow this; the EULA demands that the original disks be used for the re-install of Windows and that the original disks be shipped with the refurbished PC. In practice this is often impossible, which gives traders in refurbished PCs only three options: violate the EULA and risk legal repercussions; buy a new Windows license and thereby drive up the price of a refurbished PC to a point where it won't sell, or just close up shop right away.

Another needless restriction is that an OEM version shipped with one brand of computer will not accept a (legal) serial number issued with another hardware brand. This can make it difficult or impossible to modify (e.g. add specific drivers to) an OEM version of XP, or to re-install XP if the original CD (but not the license) shipped with the hardware has been lost. In spite of the fact that you have paid for a legal end user license, the licensing system effectively prevents you from using it.

The EULA for Office is just as brutal in its own way: if you want to use just Word, PowerPoint or Excel on a single PC, you still have to license the entire Office package. Nor are you permitted to use Word on one PC and Excel on another under the same license; you must buy a separate license for the entire Office suite for each PC.

But hey -- if you're working in education, Microsoft wants to be your friend! For a few bucks per seat you'll get all the licenses you want. Since budgets in the educational sector are usually tight, a batch of almost free software is a godsend. Or is it? Maybe not. The small print in these "education-friendly" licenses prohibits running anything but Microsoft products on the systems that run under an educational license, including free Open Source alternatives (e.g. Star Office). They'd make it illegal to mention non-MS products to students at all if they could find a way to pull that off. Nor is this the only example of Microsoft meddling with the curriculum. In August 2002 Microsoft made a controversial donation of 2.3 million dollars to the University of Waterloo, Canada, on the condition that the university would teach their students Microsoft's new C# programming language as a mandatory subject for students entering the university's Electrical and Computer Engineering programme.

With these things in mind it's rather ironic that as part of their settlement with the DoJ for anti-competitive practices in November 2001, Microsoft agreed to supply schools with software, hardware and services. What a great chance for Microsoft to kill two birds with one stone. They get to control the curriculum and expose the students to a Microsoft-only environment before they enter the work force, and they meet the conditions of the settlement at the same time!

Needless to say that Microsoft's efforts to rigidly control PC suppliers have been very effective. If you're a consumer, you'll find that it's nearly impossible to buy an A-brand PC without a bundled Windows license. Recently some large PC manufacturers offer Linux to the corporate market, but these exceptions are still few and far between, and an A-brand PC without an operating system (which would in itself be quite legitimate) is generally unavailable. On a propaganda webpage aimed squarely at OEM resellers, Microsoft went to considerable lengths to blacken the reputation of what it terms "Naked PCs". A Naked PC is a PC that you can (or rather, you can't) buy without an operating system. Try it, you'll find it's really quite difficult in any case, and Microsoft wants OEMs to make it even more difficult by refusing to sell you one. "Think of selling a house without a roof - selling your customers Naked PCs leaves them equally exposed", says Microsoft. "If you allow your customers to buy Naked PCs - placing them at risk of acquiring pirated operating systems elsewhere - you expose them to legal risks, viruses, and frustrating technical troubles." In other words the customer has to buy Windows, it's for his own good. In fact, it should be made illegal to buy a PC without Windows, because Microsoft continues with: "And even if your customer manages to illegally acquire and install operating systems elsewhere..." Apparently it's either inconceivable or immoral to consider alternatives for Windows, and installing products such as Linux or FreeBSD is a crime.
No Windows? We'll get you

In May 2001, Microsoft took this idiocy even further. Several local hardware integrators in the US were offered rewards for reporting their customers who buy PCs without a Windows user license. Yes, I'm serious. If you buy a PC and you plan to run Linux or FreeBSD on it, you automatically become a suspect and Microsoft puts a price on your head.

And there's no way out of this nonsense. In cases where large customers build their own PCs in order to avoid putting too much money in Microsoft's pocket, they can't use their volume license programs as the basis for installing

Windows. Those volume programs only allow upgrades of systems that have been purchased with original Windows licenses. They can't even save some money and build their own computers or buy them from a local whitebox shop without also tacking on a Windows license. By contract the OEMs are required to report any customer that requests 'naked PCs' and it often triggers a software contract audit by Microsoft, sometimes followed by seven figure surprise bills.

Fear campaign

Microsoft will enforce the conditions in their license agreements with a heavy hand, if need be. Or rather, they use the BSA (Business Software Alliance) as their enforcer. The BSA is a trade group that helps enforce copyrights and licensing provisions for large business software manufacturers. Steve Ball, CEO of the famous guitar string manufacturing company Ernie Ball, said in an interview:

> *"I became an open-source guy because we're a privately owned company, a family business that's been around for 30 years, making products and being a good member of society. We've never been sued, never had any problems paying our bills. And one day I got a call that there were armed marshals at my door talking about software license compliance. [...] They basically shut us down. We were out of compliance I figure by about 8 percent (out of 72 desktops). [...]*
>
> *How did this happen? We pass our old computers down. The guys in engineering need a new PC, so they get one and we pass theirs on to somebody doing clerical work. Well, if you don't wipe the hard drive on that PC, that's a violation. Even if they can tell a piece of software isn't being used, it's still a violation if it's on that hard drive."*

Similar practices abound in Europe. Many companies in Holland have received threatening letters from Microsoft (and Microsoft lawyers) with thinly veiled accusations of software piracy. Apparently Microsoft assumes that large companies should have at least a certain number of Windows and Office licenses, and at least as many Office licenses as they have Windows licenses. Large companies with a smaller number of licenses than Microsoft thought they should have were ordered to present complete and accurate information about their numbers of servers, PCs and software licenses. Failure to comply with this order in full would result in audits and legal procedures. Apparently Microsoft considers it unthinkable that PCs can be used for purposes other than running Windows or Office.

A few months later Microsoft hired a law firm to target an even broader selection of small businesses, who were more or less ordered to submit a complete and comprehensive list of all Microsoft products in their possession. Again there was the thinly veiled threat that failure to comply with this order would have "legal consequences".

What other type of company but an utterly ruthless monopolist would have the arrogance to threaten and intimidate their own customers like this?

Killing off the competition

If you're a competing software developer, things are even worse. A Microsoft version of the software that keeps you in business could be integrated with the next release of Windows, or given away for free as a separate product. Microsoft has used this and other tactics (such as deliberate vaporware announcements) many times in the past to smother innovation and break innovative developers.

If you're too big to be eliminated like that, Microsoft still controls whether or not your software will be compatible with future releases of Microsoft products. A classic example is MS Office on OS/2 Warp: several components (most noticably Word) were tailored to crash on OS/2. This strategy has continued ever since: when Windows XP came out, it wouldn't run the then-current versions of RealPlayer and Quicktime... but of course XP did come with an integrated MS Media player. Several years earlier, when I subjected a brand-new Compaq Deskpro (running NT Workstation) to the Windows 2000 Upgrade Compatibility Check, guess what happened: all installed Novell products were found to be incompatible with Windows 2000. What a surprise. Fortunately my trashcan was Windows 2000 Ready...

Apart from the above measures, there's always brute force. The blind fear that Microsoft's legal department has managed to instill in some independent developers (especially the smaller companies) is nicely illustrated by what happened to Ghisler & co, a small Swiss developer. Ghisler's primary product, a file manager that is essentially a Windows version of previous DOS-based file managers such as Norton Commander, is especially popular among power users and administrators. Ghisler had shipped Windows Commander for no less than nine years, when a letter from Microsoft claimed ownership of the word 'Windows' in the product name 'Windows Commander' and demanded that the name be changed. Ghisler not only immediately complied wih the demand to avoid legal repercussions, but also put Microsoft trademark notices on the homepage of their website, released a bulletin that avoided the word 'Microsoft' entirely but only referred to "the owner of the trademark 'Windows'", and even requested their users not to make negative comments in their forums. Such is the reputation of Microsoft's lawyers.

Nor is this reputation undeserved. Shortly after Ghisler & co required a change of underwear, the seventeen year old Canadian Mike Rowe decided, mainly as a lark, to put 'soft' behind his name and register his own domain. Microsoft's lawyers then demanded that Rowe cease and desist his "copyright infringements" and hand over his domain name. Rowe suggested compensation. Microsoft's lawyers offered Rowe 10 (ten) dollars. Rowe did not consider that a serious offer and demanded more. The Microsoft lawyers then hit him with a 25-page document that accused him of price gauging and promised legal actions. As said, Rowe was all of 17 years old at the time.

By now the field is littered with the carcasses of software companies that held a share of the market that Microsoft decided they wanted. For example, does anyone remember an upstart company named Argonaut? They were one of the few small companies that made excellent 3D rendering software in the early nineties, years before the technology became widely available on the PC. We had to wait for it all those years, though, because Microsoft bought Rendermorphic, one of Argonaut's their competitors, and started to give away their software licenses for free. This killed off all developments at Argonaut and the other small 3D developers of those days in short order, and it meant the end of another piece of innovation.

Vaporware works

Selling vaporware is one of Microsoft's favorite tactics to sabotage their competitors. The idea is simple: announce a revolutionary, new product or technology that will make your competitors' products obsolete right away, and everyone becomes reluctant to invest in those competing products. By the time you eventually release something (that may or may not resemble whatever you announced) the competition will be gone, or at least on the way out. And if truth be told, Microsoft has this technique down to a fine art. Few others are as good at it as Redmond's marketeers.

A good example of their masterful control of vaporware selling was the initial announcement of the .Net initiative. .Net was essentially announced as a whole new product line, to which all existing products were going to be converted. It was going to be the future of computing, if we were to believe Microsoft. And it was hard not to believe them, because they were already advertising ".Net Connected Software" as if it were an available product instead of a concept that hadn't even laid down a set of final specifications yet. And it worked: in an attempt to capitalize on the hysteria, third parties were falling over themselves to jump on the .Net bandwagon. It rapidly became a fashionable buzzword that CEO's hastily declared commitment to. The press especially paid a lot of lip service to .Net, and all major book and magazine publishers were in a hurry to flood the market with .Net publications. Whole series of books about .Net were being released, regardless the fact that .Net hadn't materialized yet and even the exact specifications did not yet exist!

Eventually all .Net turned out to be was a framework for the development of network applications. As such it's a typical Microsoft product: it's an attractive environment for application developers, but with serious drawbacks. It offers powerful features that often don't perform very well, and it ties developers firmly to the Microsoft platform.

Sales: from promises to lies

Microsoft Products are sold not on their technical merits, but by brute force and sheer marketing violence. IT Managers read in their investment magazines that Microsoft Is The Future. They attend a few management seminars or other sponsored events, they are exposed to a few sales presentations that are long on promises and short on facts, and so they become convinced that they have to switch to Microsoft products. After all, everyone is using Windows so it must be a good thing. Of course the same thing could be said about pot, with as much validity. The only difference is that you can't Just Say No to Windows.

Microsoft products are peddled to the corporate sector mainly through high-level selling. Large-account managers directly approach the top executives of the companies they wish to target. During tasteful lunch meetings they spin a glorious tale about how more investments in the Microsoft platform would have "strategic advantages" for the whole company. They make sure to use terms like "installed base" and "target threshold" repeatedly. They cite success stories, they mention Fortune 500 companies, they emphasize the importance of keeping strategic decisions on the executive level. They mention in passing that Windows has removed the need for computer techies in making informed decisions about computing, so now boardroom executives are qualified to select operating systems as part of their corporate strategy planning. And if the technical staff happens to disagree about the wisdom of switching to Windows, well, that's only because the techies feel that their turf is being threatened by the introduction of an operation system that removes the need for skilled personnel, and because they lack insight into strategic matters.

Of course these marketroids never even mention such unimportant details as the need for more and bigger servers than other products would require, or the fact that uptime and availability are only a fraction of that of competing products. Oh no. They also gloss over what people in the field think about what goes on under the hood of Microsoft products (after all, techies have never been realists) and they blissfully ignore the numerous implementation problems (excuse me, I mean 'challenges') that come with each new version of any Microsoft product you care to mention. Instead they emphasize that all large companies have "switched to Windows", so it has to be a Good Thing. They promise that the latest Windows Server will speed up the business and save millions of dollars per year, but of course they forget to mention that they're comparing it to Windows NT4, released in 1996. And if all this doesn't do the job, they cinch the deal with an offer that the customer can't refuse, such as a 50% discount on software licenses, and if it's about a choice between a Windows environment and Open Source software, they'll even happily give away licenses for free. Not that those licenses are so overpriced that they'd still make a profit at half the price or less, oh no, of course not. The offer should merely be seen as a quantity discount for an especially valued customer.

A real world example

I'm not making this up. I've seen it happen in large companies all around me. This is how the game is being played. The following response from a British reader (a corporate user who wishes to remain anonymous) illustrates this fairly well:

We decided to use FreeBSD, Apache, mySQL+PostgreSQL, Perl+PHP [as Open Source alternatives to Microsoft products]. The company I am working with is a pure-Microsoft company, i.e. they only used to use Microsoft software, and they even didn't know anything about Open Source. [...] When the local Microsoft rep "heard" about it (someone inside the company tipped them off), they asked to meet my team(!) and discuss the reasons for our Open Source use.

> *In fact, it was a meeting of 2 1/2 hours with 3 Microsoft sales/consulting reps trying to persuade us not to use Open Source (mainly they talked about Linux until we told them that we don't use Linux and that we don't understand what they are talking about :-) because "it is inherently insecure, unreliable" and, what was their biggest argument, "there is nobody in this country who could give you any support for Open Source", etc. Also, they wanted (actually they required!) us to tell them the reasons why we are using Open Source instead of the already introduced and long-time proven Microsoft Software in this company. I started explaining [...] and when we came to the point of 'Licensing Costs', they offered us to give the Windows server licenses for free.*
>
> *I am not kidding. When I told them that I'd need at least ten licenses and at $400/each, too much for me to begin with, they offered to give us the license for free - and not only for now, but also for the future when we kept working on Microsoft.*

Commercial brute force is not the way to introduce new software standards. If software retail stores open at midnight so that people can rush off with a new Windows '95 package the very minute it is released, it's obvious that OS implementation is no longer based on common sense or rational decisions, but merely on a stampeding software market that has been hyped into hysteria. It's obvious that something here is very, very wrong.

Keeping the customer ignorant

I'm not into conspiracy theories, but still I think it's interesting to note how Microsoft has progressed from an upstart software company to a party that attempts to control not only the market but even public opinion.

Educating the masses was an important step in Microsoft's strategy. It had long been common knowledge that "computers are difficult to use". Indeed, a system like Unix or DOS has never been known for its user-friendliness, requiring the user to use a keyboard to type commands like 'ls', 'rm' (Unix), 'REN' or 'DRIVPARM' (DOS). The steep learning curve ensured that users would eventually be fairly computer-literate (which was good) but also that few would succeed in or even start the time-consuming, difficult and expensive learning process (which was definitely bad).

The Graphic User Interface (GUI) in MS Windows put an end to all that. It offered an attractive, accessible and friendly-looking interface, designed so that it wouldn't scare the novice user. This has played a large role in making the PC available to the masses, and Microsoft deserves due credit for that, even though GUI's aren't and have never been Microsoft technology.

But in their zeal to shield the poor novice users from confusing or intimidating glimpses at the underlying technology that might frighten them, Microsoft has actually oversimplified the interface. Users simply drag and drop, unable to determine the difference between local 'folders' and those on network servers. They don't know that a local 'folder' is not the same as a server mapping, and they're unaware that 'My Documents' is in fact a subdirectory that may reside on a local disk or on a network server. In fact, usually they have no idea what a subdirectory is. So they simply right-click a document (which is represented by an icon) to 'send it' to a 'mail recipient' without knowing that they are in fact pushing an uncompressed 12 megabyte BMP file through an E-mail server and an Internet link.

Even worse: not only are users ignorant of what happens in response to a simple mouse click, but the Windows environment actually makes it difficult for them to find out. At least the pre-Windows user interfaces eventually stimulated the user to gain some insight in what he or she was doing, and what the results of seemingly innocent actions could be. Nowadays even the computer-literate have trouble understanding what goes on behind the facade of the Windows GUI. Users are actually being conditioned to associate daily tasks with Window GUI elements. By the time they have managed to change their preference settings so that Windows displays filename extensions and they can see what kind of file they're dealing with instead of just seeing 'documents', they're no longer average users.

Apart from all that, expecting a systems or network administrator or an experienced user to work with the user interface that comes with Microsoft products is a bad joke at best. Imagine an operating system that won't let you tell it what you want, but lets you point at a picture instead and then does for you what it thinks best. I can imagine a three-year-old preferring it that way, but not a mature ICT professional. Of course GUI-based system administration has its advantages, at least from a certain perspective, i.e. a Windows-using ICT manager's perspective. Large-scale, properly set up Windows networks with a ton of hardware and GUI management tools all over the shop needn't cost a lot in terms of machine minders, whereas a Unix or Open Source based network without these tools will need the requisite number of skilled geeks to mumble incantations over shell prompts. But this is comparing apples and pears: the geeks will serve you better than the deskilled machine minders will when something goes badly wrong (which it will). The GUI has put on a lot of weight in recent years, but in the end it serves more to restrict than to enhance, limiting the users' understanding of, and control over, their computers and software. The GUI removes all transparency from the system, so that power users and sysadmins no longer have access to the underlying processes.

Limitations perceived as ease of use

It's interesting to see how a lack of options is often confused with ease of use. Granted, any appliance that only has one big red button marked 'On' is easy to use. But don't expect it to be useful for more than one major purpose, or otherwise flexible.

Windows advocates often argue that only Windows (and certainly not its main rival Linux) has understood the users' needs to use a computer as a tool. They say: "Could it be perhaps that Microsoft got to be a multi million dollar company, precisely because it set out to build a simple to use, easy to understand operating system? One that just works, out of the box. Without the need to be a geek and spend all day configuring complicated services and settings every time you want to make something happen."

One enraged reader of this paper even wrote:

> *"I own a computer repair centre and deal with literally thousands of home users a year. I would say 80% of my customer base are exactly that, 'Users'. They know how to turn the thing on, they know what the big blue 'e' in the middle of the desktop is for. Some of them can even word process. But for the vast majority of them that's as far as it goes.*
>
> *The problem with geeks is, they seem to inhabit their own little world, where everyone is a computer 'expert' and all the answers are black and white. Meanwhile here in the real world companies like Microsoft understand that the majority of their customers are not. That they view their computer as a functional item, a means to an end and base their software purchasing decisions on which product will allow them to do what they need to do, as simply and as quickly as possible. Not for its technical merits or because they get aroused at the thought of tweaking their system to perfection."*

This is essentially true. A computer should be a tool, a means to an end. However, Windows advocates often confuse a lack of options with the lack of a need for options. They are right in that most users just want an appliance, rather than a complicated assembly of software that requires fine-tuning. On the other hand, an assembly of software is all they get, and turning it into a simple and reliable appliance, through fine-tuning or other means, is barely possible. And that is a real problem.

The world is full of people who make you realize why an electric hand mixer needs a warning label advising users to switch it off before licking icing from the beaters. These geniuses are far better off with, say, a well-configured Linux box, installed straight off some CD with a few mouse clicks. This will let them do their jobs and still prevent anyone without a root password from doing any damage. Windows on the other hand can be damaged by an end user through the mere installation of an application. Yet it does not offer many options, neither for the novice user nor for the professional, to track down and fix the problem, since that would expose options that are more complex than a one-click "wizard" feature. Windows is like a car with the hood welded shut. If you don't know any better it may give you the idea that it requires no maintenance or repair... until it breaks down, and that's when the problems really start.

Windows pulls the wool over your eyes, and it does that very well. Its many limitations are not apparent to the novice user. This is an important factor in guiding the users' perceptions. Most average Windows users are not aware that what they perceive as simplicity is in fact a lack of sophistication. They just click on an icon, and when things do not behave as advertised they enlist the help of a support technician. If the technician is unable to adequately solve the problem due to Windows' lack of transparency and manageability on the system level, they tend to blame the technician and not the software. Most Windows advocates (who generally call themselves power users and therefore should know better) do the same.

Closing the curtains on Windows

Contrary to popular belief, a GUI is not ergonomic. For example it requires users to take their hands off the keyboard and their eyes off the screen in order to operate the mouse during word processing, and graphic fonts and black-on-white text cause more eye strain than the old text-based equipment used to do. Neither is a GUI conducive to productivity; although the learning curve of a command line environment is steeper, after some training many users can perform most operations faster through keyboard commands than with a mouse.

Another headache for sysadmins is that GUI operations are essentially impossible to script, so that with large numbers of servers it is impractical to use the GUI to carry out installation tasks or regular maintenance tasks. Desktop users face the same problem: in the early nineties it was possible to produce large amounts of personalized correspondence using nothing but Word Perfect macros, a simple database and a few batch files. In Windows most of these jobs have to be done by hand, over and over again. In short, Microsoft tried to create products that even a fool could use, but they ended up with something that only a fool would want to use, given the chance to make an informed choice.

But then again, Microsoft's regard for their user community is best illustrated by the useful tips in MS Word, my favorite of which has alway been "Don't run with scissors". And of course there was the 'log-on help' in Windows 2000 Professional: an explanation on how to press the Ctrl, Alt and Delete keys, complete with a graphic animation of those keys being indicated and depressed. The animated question mark icon ("Any time you need help, click me with the mouse or press the F1 key. I'll be right here if you need me!") in the Windows XP Professional installer is even more annoying, and bears an uncanny resemblance to the animated help in MS Office. These 'professional versions' target the 'professional' user, who is apparently assumed to be unable to handle complex operations such as accepting defaults in an install program or even logging on without animated graphics as a guideline.

Ignorant users are happy users

In all fairness, technophiles have always been exasperated by the 'ignorance' of non-techies. But these days we're dealing with a generation of users that can't even understand the need to know the basics. All they have to do is double-click on a document, and things start to happen. Of course as soon as the document's file extension (which is hidden by default in the first place) isn't properly associated with an application, the average user is immediately lost. Users have never been invited to learn. They've been told that they no longer need to know about the basics of driving, so they just expect their cars to take them wherever they want to go today.

As a result of all this, average users don't even realize that computers and Windows aren't a necessary combination, or that there is a distinction between operating systems and the applications that run on it. They've been taught to think of Windows as something that comes with your PC, or even as something that is part of your PC. They have been told that Windows XP is a multimedia environment. The idea that Windows XP is an operating system that could, but not necessarily should, run multimedia applications is completely beyond them. The thought that Windows is one of the many operating systems that could be installed on a computer is just as alien to most of them. To them all computers in the world are PC's running Windows.

Today the user, tomorrow the world

The rot has now spread so far that this misconception affects many software and content developers. Web designers automatically assume that their web sites will be viewed on a PC, and if you're lucky they'll write code that runs on both Mozilla/Netscape and Microsoft browsers. (As if those were the only ones around.) Application developers usually aren't much better either: they write software for Windows, period. Even they just don't know any better. Even in Windows itself you can see that portions of the code have been created by junior programmers who have never known a more robust environment. Nor is this surprising. Most IT students only encounter Windows these days. Most of them have never seen a text-mode interface, they don't know that there are other OS's than Windows out there or how they work, and their understanding of what lies beneath the Windows GUI is rudimentary at best. They've never seen robust software, let alone learned how to write any. Still these students are supposed to become tomorrow's IT workers.

Quality standards have steadily dropped. The average user routinely endures buggy software, computer crashes and loss of data. Think about it: To have several computer crashes or forced reboots a week is considered normal and acceptable! That is, by those who have never known anything but a PC with Windows, which is most of today's user community. The thought that it's not normal and acceptable for computers to crash or require rebooting regularly never enters their minds.

Most computer users know computer technology only through Microsoft products. They no longer learn about computing; the Windows user interface discourages anything beyond point-and-click actions. Like toddlers they point at small pictures and they think they are knowledgeable about computers, while the marketeers wax lyrical about how easy and exciting it all is, as long as we all keep buying more and more of the same junk.

And that is the basis on which many IT managers choose the platforms for their future investments! That, and the comforting knowledge that "nobody ever got fired for buying Microsoft."

God help us.

Power corrupts, absolute power is even better

I used the word 'megalomania' above. You'll understand why if you take a good look at Microsoft's plans for the future. Controlling the PC market is not enough for Microsoft. In the near future we can expect to see them move into different markets.

They're well on their way to flooding the market for handhelds with Windows CE. They're trying to get Windows on the road by embedding it in automotive electronics. They've briefly courted TV networking. They spun up their marketing machine to take over the cellphone software market, starting with Ballmer's claim that 25% of all multimedia cellphones will run Microsoft's Smartphone 2002 operating system within three to five years. Fortunately the first releases of Smartphone were such a disaster that most cellphone manufacturers soon lost interest.

One of the most interesting new initiatives is Windows Media Center. This is a special pre-installed version of Windows XP or Vista. Windows Media Center won't be sold separately but comes with Multimedia Entertainment Systems (which are essentially PCs with a TV tuner and a remote control). This Windows version incorporates entertainment features like DVD-playing, recording TV programmes, and an application to manage and view digital photos. It comes with a simplified user interface that can be read from across the room. None of this is very innovative, but Media Center PC is likely to be just the opening salvo in Microsoft's bid to control home entertainment in the same way it already dominates home computing. In a few years a personal computer (or something essentially like one but with a more purpose-specific design) could be the heart of many families' entertainment centers, and Microsoft will attempt to exercise control over it just like they do with the PC market. At WinHEC 2003 Gates presented further plans to integrate your TV, stereo, VCR, phone etc. (all of which are devices that switch on immediately and then just work) into the Windows PC (the device that doesn't).

Microsoft has also begun to sell their own gaming hardware with the release of the Xbox gaming console. The reason that Microsoft is getting into games is not readily apparent. Their explanation that they wanted to save the world from Playstation domination is of course not to be taken seriously. As far as domination is concerned, it's an interesting fact that IBM was Apple's sole supplier of Power-PC chips, on which Apple's hardware architecture was based, and which IBM produces in limited quantities. The Xbox uses several of these IBM Power-PC chips. Now convincing IBM that it would be more profitable to do business with Microsoft than with Apple was not very difficult. Fortunately for Apple the company proved agile and resilient enough to adapt, and it continues to thrive after a timely but forced switch to Intel chips. Still the way in which the Xbox forced Apple through a major change in hardware platforms is an interesting one.

Even more interesting is the simple but often overlooked fact that the Xbox is not a PC. It's Microsoft's first attempt at widely deploying a device that offers home entertainment, Internet access, multimedia functions and (with a few software updates) any other recreational or home application that you care to think of. Currently Microsoft's survival is tied to the technological life cycle of the PC and Windows as a platform. The Xbox offers Microsoft a valuable opportunity to play with technology that could be the future of home entertainment and the ultimate replacement for the home PC. The fact that Microsoft has rigidly tied the Xbox to its own internet-based Xbox services (including in option to permanently disable the Xbox hardware if Microsoft detects that it has been tampered with) bear this out.

But Microsoft's primary reasons to venture into the hardcore gaming market are actually rather simple. PCs and hardware have gotten faster and more powerful all the time, but the only applications that really tax those resources are games (and lately, but to a far lesser degree, digital video). Gamers tend to keep their hardware and the supporting operating systems up to date, and therefore games are a powerful contribution to the update frenzy that Microsoft thrives on. But game consoles have always been a competitor to the PC, and therefore a threat to Windows. Microsoft has always tried to exterminate all competition with fire and sword, but in order to do this they needed to enter the market for game consoles themselves. In the late 1990's, through a little known and rather half-hearted deal with Sega, they tried to push Windows CE as an OS for console games. The unsurprising lack of success of this idea and the subsequent demise of Sega went largely unnoticed, but they did prove that just putting Microsoft software on a third party game box doesn't work very well.

Therefore the Xbox was released and, being a Microsoft product, it gives MS full control over what will and what won't run on it. Furthermore, Microsoft attempts to further control the gaming community through online services, on which more and more Xbox features will become heavily dependent in the future. This fits in with Microsoft's plans to tie their customers down into Internet-based subscription services to protect revenues. That is why the Xbox exists. That is why Microsoft introduced the Xbox in the US on a 500 million dollar PR budget, and why they continue it in spite of the fact that the Xbox has only netted a loss since day one. Half a billion US dollars to introduce a gaming console that doesn't even turn a profit -- think about it.

Controlling the Internet

Perhaps the most important new business for Microsoft is web services. Microsoft is really getting into web content with MSN, its search engine, their ill-fated and fortunately short-lived Passport services and other, related projects. Windows XP and Vista come loaded with features designed to lure the user into buying music online (from Microsoft and their partners), have digital photos printed at the click of a mouse (through a Microsoft online service), to browse

MSN (which boosts Microsoft's advertising revenues) and to shop online (using Microsoft's passport and payment services in the process).

With these first steps, Microsoft has begun a gradual but deliberate change. Microsoft the software monopolist is trying to become Microsoft the web services monopolist. Also note that MSN does not make any profit. Instead Microsoft needs to spend in the order of half a billion US$ each year (!) to keep it operational. Obviously this investment contributes to inflated profits elsewhere.

After more than a decade of having milked Windows for all it was worth, it's becoming increasingly obvious that Windows revenues won't last forever. The answer is both simple and complex: Microsoft needs to find a new way of ensuring revenues in future years. Since Microsoft Windows and server products are an excellent means of tying the user community to proprietary protocols and services, it stands to reason to use it to leverage the user community into a new dependency. Enter Microsoft's new Internet strategy.

The idea is simple. Start partnerships with large information and service providers on the Internet, and plan to hurt competing information providers (such as Google) as much as possible if they won't co-operate. Then set up a bunch of web services, and bundle clients that use those web services with Windows, so that the user will get it "for free". Gradually discontinue PC-based support for these services in software. Start with trivial things like software activation and registration, user authentication and software maintenance, and then move on to things like payment services, address books and appointment schedulers, and eventually to full-fledged web-based applications. Initially offer the new services for free or for a low entry fee, and when user dependency is at a sufficiently high level, start charging serious subscription fees. And there you are.

This future has already begun. The first implementations of this new strategy are already visible in Windows XP, and even more so in Vista.

Control every keystroke

Microsoft already controls the kind of software we buy and use. The next step into the future is to seize control of the work that we do and the way that we do it.

A major component of Microsoft's long-term future plans revolves around Application Service Providing (ASP). The idea is to offer the applications that we now use as an internet-based service. Microsoft or its partners will host our Office applications for us, and we'll access them using only a (thin) client system. Microsoft promises huge reductions in TCO, mainly because the installation, management and maintenance of server and applications will be outsourced with this concept.

While ASP is of course touted as being innovative, basically it's a step back to the decades-old mainframe-with-terminals approach. In fact, all you need to become an ASP today is a Unix server, a bunch of applications and some graphic terminals. Granted, the X protocol is ugly and unsuited for anything but LAN's, but the implementation of a more elegant and efficient client/server protocol layer (e.g. ICA or something better) is relatively trivial. Still, notwithstanding the fact that it's essentially retro-technology, at first sight ASP might not seem such a bad idea. After all, we won't have to bother with local software maintenance, and we'll only be charged for the actual use of services and not for software licenses. This should simplify things no end, right?

Well... Think about it. The whole idea is that Microsoft will take the application software that we now run locally, and host it for us on their own Windows-based servers. First of all this raises questions about reliability: will Microsoft's technology be up to a job that is mission-critical to large parts of the planet? With incidents of some 30 million users having problems with the MSN Messenger instant messaging service, caused by a malfunctioning disk controller on a buddy list database server that took Microsoft over a week to fix, the prospects aren't all that good.

Secondly, Microsoft will take the application software that we now buy, and rent it out to us on a subscription or per-use basis. Yes, we'll save money on one-time licenses and on local administration. How very decent of Microsoft - after they inflated the costs of licensing and ownership themselves. But will we actually save money in the long run? We'll have to buy and run local client software from Microsoft. You can say what you want about Microsoft products, but Lean & Mean is not the way to describe any of them. They'll need serious hardware, and bugs and implementation problems are common. On top of that, ASP will only shift the workload (and cost center) from local server and application administration to Internetworking and network administration, simply replacing one problem with another.

But the most worrisome aspect of a shift from Microsoft as a software vendor to Microsoft as an Application Service Provider is that it means our complete and utter day-to-day dependence on Microsoft for earning our daily wages. We'll be forced to keep paying whatever subscription fees Microsoft chooses to charge us. Even better, Microsoft will also be able to control and monitor our daily work. Microsoft will control whether or not our applications will run, Microsoft will control which services and software products will be available to us, and Microsoft will know about it each and every time we use an application (i.e. request a service from Microsoft). If Microsoft wants to monitor each and every keystroke in said applications or even look into our own corporate data, they'd have no problem doing so. And If Microsoft isn't interested in our corporate data, I'm sure someone will be. And that someone will be very happy with the appalling lack of security in any Microsoft product so far.

ASP and User lock-in

If ASP ever takes off, we'll of course be forced to buy the client software from Microsoft (most likely bundled with an advanced PC) since adherence to open standards is something not even the most naive optimist has reason to expect. Microsoft's application service is going to be a closed system. Microsoft will control it, and therefore will control the operational costs. Instead of having to pay an admittedly steep, but one-time, license fee we'll now regularly pay a subscription fee, to be set by Microsoft. After all, Microsoft's office application division is facing a revenue problem, as more and more users refuse to buy yet another version of MS Office for the sake of a few trivial "improvements". And we'll keep paying, because once we've switched from locally administered software to the ASP model, we'll be committed to it. Trust me: a back-out from a shift to Microsoft's new scheme will be costly.

But we'll have little choice: the ASP platform will be gradually incorporated in all new versions of major Microsoft products. Each time we're forced to buy another upgrade in order to maintain compatibility with the rest of the world, a piece of the new framework comes with it, and eventually the whole scheme will be forced upon us. Microsoft has announced that the extensions to implement this new framework in existing OS products will be free. Right.

Where have we heard this before? Microsoft has given away products for free in the past: web browsers and media players come to mind. Each and every time they gave away free software their ultimate purpose was to kill off a competing product that might have offered a viable alternative to the user. And now the ASP framework extensions will be free? Sure... Timeo Danaos et dona ferentes.

Forced updates and Trojan Horses

Financial ties aren't the only kind of control that Microsoft will have under the new ASP scheme. Currently we may choose to purchase software for a one-time license fee and decide not to upgrade it. We may choose not to embrace dubious concepts or empty hypes. We may choose to wait, or to skip certain products or versions entirely. Under the ASP concept, Microsoft won't allow us the freedom to do that. Microsoft controls our software, period.

Take the auto-update features in the client software, for example. Our client software will automatically be updated whenever Microsoft wants it to, installing new drivers, patches-du-jour and additions, and in the process of course uninstalling everything that has to go. Apart from doing away with most of the distribution channel and thereby inflating Microsoft's revenues as an added bonus, auto-update has enormous possibilities:

- Microsoft will control which drivers are present on our client computers. Soon we'll see 'strategic partnerships' emerge between Microsoft and peripheral manufacturers, and if we want to connect a printer from a brand that competes with one of Microsoft's favored partners we're stuck.
- Microsoft controls document compatibility and portability. As part of an update, the software may helpfully convert your existing documents and files to the new format. Today you may be able to export data to Oracle, tomorrow you might be limited to MS SQL.
- Exclusive control over the driver and application software will make it that much easier to appropriate open standards.
- Applications that worked well in 512 megabytes of RAM yesterday suddenly need twice that much memory tomorrow.
- In order for auto-update to work, Microsoft will need serious access to all the files on your harddisk. Of course they'll promise us that that access will be limited to the files that make up the operating system... just like all the other spyware manufacturers do.

All of this will be completely automatic. We won't have to worry about it... meaning that we won't have any control over it. Essentially, the auto-update feature is a trojan horse. We won't even have to wait until ASP really takes off. In Windows 2000, XP and Vista the first incarnations of the auto-update scheme are already hard at work. The WGA (Windows Genuine Advantage) feature is a good example. Microsoft's initiative to prevent their products being pirated through the WGA feature can of course hardly be criticized. What is worrying, however, is that they slipped in WGA through Windows auto-update disguised as a critical security update that ended up breaking some quite legitimate OEM installations of Windows. Microsoft has also on at least one occasion quietly installed updates on the PCs of XP users even if the latter had set their XP not to download and install any updates. This stealth update (forced and without notification) then turned out to break about 80 patches and a lot of other things. This is a prime example of a "service" that is solely designed to benefit Microsoft and not the user community.

Even more interesting is Microsoft's announcement that in the future this service is designed to provide not only automatic updates to Windows, but also to take care of virus and spyware protection, network security and other essentials. Of course this service merely lets the user pay Microsoft to clean up their own rubbish, while there is no reason to expect it to be any more reliable and secure than is normal practice for Microsoft. Even more importantly, this all-in-one service is primarily a vessel for software distribution and control, disguised as a maintenance process. The software it distributes, through an integrated Windows service, directly competes with all anti-virus and anti-spyware manufacturers in the market.

Update your software, downgrade your rights

A major advantage of auto-updating (at least from Microsoft's standpoint) is that it gives them tremendous control over the users' rights and ability to use their software.

In June 2002 Microsoft injected a critical security patch for Windows Media Player into the auto-update channels. The patch itself was harmless enough (though of course it destroyed RealPlayer's ability to play audio CD's) but during the automatic installation process the user was quietly required to agree to a brand new clause in the software End User License Agreement.

This new clause in the EULA gives Microsoft the right to "provide security related updates to the OS Components that will be automatically downloaded onto your computer [and] may disable your ability to copy and/or play Secure Content and use other software on your computer." In other words, by installing the patch (which is critical to the security of your system) you have agreed to give Microsoft deed and title to your personal property, to disable functions on your computer whenever they feel like it, and to leave them immune from legal repercussions if they damage your system, livelihood or worse.

Whenever this happens, Microsoft promises to make a "reasonable effort" to post notices somewhere on a website. It's clear from their wording that MS has absolutely no intention of bringing this behavior to our attention. Instead, Microsoft just assumes the right to surreptitiously install code of their choosing on your computer. You will not be warned; you will not be offered an opportunity to examine the download or refuse it. MS will simply connect remotely and install or disable whatever they will, or do so secretly when your computer contacts any of their servers. Microsoft will have administrator privileges on your personal computer. What they feed you may be infected with viruses; it may break your applications, corrupt data files, destroy weeks or months or even years of work, but you'll have no recourse if it does. Their responsibility ends with "Sorry".

As if to illustrate that this was more than incidental, a few weeks later Microsoft released Service Pack 3 for Windows 2000 with a similar clause in the EULA. This essentially gave Microsoft the right to go into your systems, gather whatever information they think they need, including an inventory of what software you're running, and "disclose this information to others, but not in a form that personally identifies you". Similar things are going on with recent updates of Internet Explorer, during the installation of which you grant Microsoft permission to collect information about OS version numbers and product identification numbers, IE version number, version numbers of other software, and Plug-and-Play ID numbers of hardware devices.

It's interesting to note that the Computer Incident Advisory Capability office (CIAC) has issued an official warning against Windows XP and Office XP. (CIAC bulletin M-005c.) CIAC officials were displeased with the error reporting feature in these products. After a crash, Windows and Office XP send information (i.e. memory dumps) to Microsoft so that developers may do a 'post-mortem' on the data to see what went wrong. These memory snapshots are likely to contain (possibly sensitive) user data, e.g. the document or spreadsheet that the user was working on at the time. Microsoft's promise that any "accidentally" received sensitive data would not be used in any way did not impress the CIAC.

Windows 1984

All these changes are to a great extent exercises in fixing flaws in a product you have already bought. But the hidden control features that they come with are an outrageous imposition for Microsoft to seize more rights for itself as a condition of those fixes being applied. If you want to keep your systems working properly, you are forced to give Microsoft control over your personal and corporate information. Scary? Try 'Orwellian'...

But wait -- it gets better. Shortly after announcing their planned future shift to Internet-based application services, Microsoft launched a new scheme: Microsoft Passport. This was presented as a simpler authentication system that would effectively enable us to log in to the whole planet with one single password. Personal information, passwords, a virtual identity, credit card information and many other types of data would be bundled in one system (codename Hailstorm). The Passport "Wallet" system was the first step in this plan, and while it was operational it allowed us to log on to all affiliated websites (including E-commerce sites) with one and the same password. Or at least, that was the idea.

Apart from the huge security weakness that this single-point authentication implied, the terms of use left little to the imagination: you had the right to use the service, period. Microsoft reserved all other rights, including the right to use the information you provided as they saw fit, the right to change conditions without notice, and the right "to exploit any proprietary rights" that you might hold. It was interesting to note that they used the word "feedback" for all user-supplied information (which included each and every mouseclick). This legally gave them the right to monitor and track everything you did on the web. In their Passport privacy statement they stated a commitment (in less than legally airtight terms) to provide secure user interfaces and transmissions for your data, but little more. In fact, they explicitly stated that they would "disclose Personal Information if required to do so by law or in the good-faith belief that such action is necessary..." In return, they continued to state that "If Microsoft becomes aware of ongoing site-specific consumer concerns or problems with Passport participating sites, we will take these issues seriously..." Well, that should protect our privacy and legal rights.

Microsoft and your wallet

Given Microsoft's penchant for apallingly bad security standards, it was only a matter of time before the Passport Wallet system would be cracked and spill its (or rather, your) secrets. And indeed it didn't take long. Shortly after Passport became operational, credit card information became available for unauthorized access. Microsoft product manager spokesman Adam Sohn said there was "no evidence" that data security was compromised, but the fact that Microsoft took the entire Passport Wallet service offline until the largest security holes had been patched up is a fair indication that things just might have been a little bit more serious. Sohn also stated that Windows XP users were not affected because of XP's "improved security". What he in fact meant was that cross-site scripting is a little harder to do with XP, and his statement illustrates Microsoft's naive ideas about security models rather well.

Meanwhile Microsoft continued to push Passport. Features in Windows XP nagged mercilessly, offering all sorts of goodies to get you to divulge your name, address, age, phone number, and the like, as an incentive. Then, less than a month after the security breach, Hotmail users were required to sign up for Passport, and in so doing were added to the Passport database. Microsoft Messenger suddenly came with compulsory Passport subscription too.

Then all users who had signed up for Hotmail (or anything else linked to Passport) before December 2001 got a big surprise. Suddenly Microsoft quietly changed the rules, and unilaterally decided to pass along personal information to other companies that used Passport on their Web sites. This personal information included the user's email address, birthday, country and zip code, gender and occupation. They did this by the simple expedient of adding check boxes to the users' personal options to indicate whether or not data may be shared, and checking those boxes by default. Microsoft also quietly changed their policy about sharing your personal Passport information, essentially abandoning most privacy-related clauses in their earlier policy, and thereby stripping their Passport customers of all rights to privacy.

God's own address book

This was only the beginning. ZDnet's David Coursey, a self-admitted "non-MS hater", wrote:

> *[Passport] will start simply and helpfully as online services learn to interact with your desktop computer. It will become easier to log on: A single password will give you access to many more services, and you will only enter it once. You'll ask to be notified of events that are important to you--and the notification will just appear on your desktop, or perhaps on a cell phone or pager. The system will know where you are and how to reach you.*
>
> *It will link things together that have never been linked before and it will seem like magic. Or maybe not. Most of what [Passport] wants to do can already be done, but not as flexibly and certainly not on an anything-to-anything basis across multiple vendors or systems.*
>
> *Think of it as God's address book. To accomplish this ultimate linkage, Microsoft will create, perhaps with partners, a giant database to collect, manage, and dispense information from what amounts to God's address book: Everything you might want to know about everyone will be in there.*
>
> *Which is to say Microsoft wants to have all your personal information, like calendars, contact lists, E-mail inbox, credit card information, banking data, and so forth, in this giant database, so that applications can use the information to do your bidding. You won't reveal it all at once, of course, but as you ask it to do more for you, more will be revealed.*

Imagine: God's own database... with your private E-mail address, your private cellphone number, your bank account and credit card numbers, your financial administration, who your doctor is, what prescription medication he gave you... This should be good! A database that knows where you live and what you recently purchased, or whether or not you have received treatment for any venereal diseases. A database that could cause possible rejections from your health insurance company because of genetic defects in your family that you yourself might not even know about. A database that can tell telemarketeers where and how to reach you and where to send their unsolicited E-mail. A database that could get you fired without even knowing why. A database that provides a wealth of useful details on you including your social security number, age, occupation, credit record, income... You name it, it's in there, maintained by Microsoft and "protected" from the eyes of the ungodly by the ridiculous kind of security schemes that Microsoft has become rightly notorious for. Not to mention the US government's demand for wide open backdoor access into such a database.

Forget Orwell, forget 1984 -- this is much better!

Eventually the entire Hailstorm project was put on hold. This was not only in response to widespread criticism concerning security and the ownership of privacy-sensitive data. The main reason for the holdup (and fortunately the eventual demise) of the concept was that Microsoft didn't manage to inspire enough trust in potential implementation partners. The intended adopters of Hailstorm feared that control over the accumulated data would enable Microsoft to interpose themselves between the partners and their customers. Initial negotiations with five interested companies had already taken place, but even those potential early adopters couldn't bring themselves to trust Microsoft enough to do business with them on such a scale.

By the end of 2004 Microsoft was forced to discontinue Passport. In spite of Microsoft's best marketing efforts and greatest sales pitches, nobody trusted them enough to participate in, or even pay lip service to, the Passport initiative. Given the fact that there are generally partners to be found for just about any venture with Microsoft, and the fact that at least some decision makers would have based their decision on Microsoft's track record rather than on sentiment, this should tell us a thing or two about how bad the state of affairs actually is.

Microsoft spyware

Another indication of where Microsoft is going with regard to privacy breaches is the spyware embedded in Windows Media Player (WMP). Computer Bytes' **Richard M. Smith** explains:

> *"Each time a new DVD movie is played on a computer, the WMP software contacts a Microsoft Web server to get title and chapter information for the DVD. When this contact is made, the Microsoft Web server is given an electronic fingerprint which identifies the DVD movie being watched and a cookie which uniquely identifies a particular WMP player. With these two pieces of information Microsoft can track what DVD movies are being watched on a particular computer."*

This nonsense started with Media Player 8, but the Microsoft privacy policy that came with it did not disclose any of this. Media Player 9 came with even bigger backdoor options for Microsoft. Internet Explorer 7 boasts a "security feature" that contacts a Microsoft server whenever you access a website. The advertised purpose of this feature is to protect you from ending up on fraudulent websites. However it has the additional benefit of informing Microsoft exactly what information on the Internet you are trying to access, which is the exact definition of spyware. The Windows XP search assistant also contacts Microsoft servers on a regular basis for no sufficiently explained reason.

But not only separate applications have been deliberately compromised. Windows XP Home Edition regularly connects to a Microsoft server as well. There are several processes running on all versions of Windows XP and Vista that generate unexplained network traffic to IP addresses owned by Microsoft. The US government has a hand in it, too: during an investigation of Windows by **Cryptonym Corporation**, Chief Scientist Andrew Fernandes discovered a backdoor for the National Security Agency (NSA) in every flavor of Windows, from 95 to XP, no matter what country you're in. There is no reason to assume that this backdoor has been closed in Windows Vista. It is part of the 'CryptoAPI' code, the foundation of cryptographic security in Windows. Apart from the question of whether or not the US government should have backdoors into the cryptography on all Windows computers in other countries, this means that any backdoor (not to mention other flaws) in the CryptoAPI module will open up all of Windows to electronic intrusion.

Next Generation Control Secure Computing

Where is all this going? Well... Microsoft has taken to putting some very odd language in some of their updates: things like requiring that you agree not to benchmark their software, or publish the results if you do. This should give us pause. And of course there's also the ridiculous clause in the Office XP EULA that prohibits you from running it on anything but Windows (without actually mentioning the words "Linux" or "OSX"). Then, too, XP and Vista require "activation," which gives Microsoft some information about what you're running, and is the first step toward letting them into your system as a "trusted" associate. Which itself wouldn't be a big problem if weren't for the fact that activation is tied to the identity of several hardware components in the computer, and for Palladium chips and similar hardware being put onto motherboards. In fact, many hardware manufacturers (including major ones) have been quietly putting Palladium chips into their motherboards for years.

The Palladium chip runs a system that, when you boot up, decides what software is trusted and legitimate and thus allowed to run, and what is forbidden. After its introduction, Palladium has been renamed into 'Next Generation Secure Computing Base'. Well, that should help. NGSCB, having attempted to shed the stains of Palladium's negative publicity, was promised to be released as an integral part of Windows Vista. That didn't happen, and analists had already warned not to expect any adequate security and privacy improvements before 2008. So far they have been proven correct.

Whatever part of NGSCB is going to materialize within the next few years is more likely to focus on digital rights management and extending control over the user's desktop than on security. The first thing Palladium (excuse me, NGSCB) will do is to enable software manufacturers to decide when their products will run and when not. It will allow them to bind software products to a single PC, which means that you'll have to get their permission to replace your hardware. It will allow them to make their software run only for a certain time, which will enable them to enforce regular payments for "subscription renewal". It will enable them to limit or prevent the making of backups. It will enable them to track versions of products on your system, link your Internet access to your hardware identity and later to your own (their infamous and ill-fated Passport system comes to mind) and keep track of what data you download, use and distribute. Possibly the same can be done to hardware in the not-so-distant future. After all there is already a "feature" in Microsoft's Xbox gaming console, that remotely and permanently disables it whenever Microsoft's servers detect it has been tampered with.

In short: control, control and more control. Apparently Microsoft's definition of 'secure' has more to do with securing their own interests and extending their control over the user than with actual system security. Their current plans only extend that control further and further, under the guise of enhancing security, protecting third party copyrights and working for the common good.

It's food for thought.

5. Bad practice, foul play

"The greatest joy a man can know is to conquer his enemies and drive them before him. To ride their horses and take away their possessions. To see the faces of those who were dear to them bedewed with tears, and to clasp their wives and daughters to his arms."

-- Genghis Khan

Doing business with Microsoft has never been without risk. They have earned a reputation for dirty deals and backstabbing their business partners whenever that happens to suit them. Time and again small but innovative high-tech developers have entered into partnerships with Microsoft, only to find that Microsoft broke agreements, stole their technology, did not deliver, then dumped them on the edge of bankruptcy. It's also not uncommon for Microsoft to take over a small but innovative company and to put it in the trash can right away, just to keep new and competing bits of technology entirely off the market.

Of course there are plenty of other companies who play dirty tricks in order to get ahead in the market. Microsoft is merely one of the biggest companies who have routinely used foul tactics. In fact that's one of the normal risks of doing business. "If you can't stand the heat, stay out of the kitchen", as the saying goes. Still the risks of a partnership with Microsoft must be considered.

The customer on the other hand has not chosen to be part of this particular fight for commercial domination. Microsoft's clients, and the clients of their business partners, are being promised, pay for, and are thus entitled to expect, good products. Instead they get bad products, or none at all, and they end up as pawns in Microsoft's foul play to establish a complete monopoly. They start out with receiving sub-standard products, and eventually they find themselves tied into the deal indefinitely.

~~Bug fix~~ New product

As has already been discussed, there are many things wrong with Microsoft products. Bugs and design flaws are common. Of course, all 'issues' with MS software will be dealt with in the next release... But not right now. That's the whole point: instead of fixing bugs, Microsoft actually uses these flaws as an excuse for an aggressive update policy. Microsoft doesn't fix bugs, but continuously releases new versions. They call this "innovation", but in truth the only purpose of this strategy is to inflate Microsoft's already obscene profits even further.

Picture this: you buy a newly-built house from a real estate company. As soon as it starts to rain, you discover that the roof leaks. When you complain about it, the real estate company either ignores you or they tell you that this kind of roof is a brand-new innovation; the sort of house they used to sell never had such a beautiful roof. Instead of fixing your roof they promise that the next house they'll build won't leak. Eventually they complete their next house, three years or so behind schedule, and you have to pay a hefty price for it... only to find that it comes with a patched roof, and now the water seeps through the walls instead. The new house has an extra wing added to it that you didn't ask for, but as soon as you enter it the floor collapses, and if you try to save yourself you find door jammed.

Would you accept such nonsense? Of course not! You'd file complaints, you'd sue! But this is what Microsoft has been doing with their software products, and the user community has been taught to accept this.

The "innovation" upgrade treadmill

Of course new releases are necessary if software is going to evolve at all. But are these new versions indeed as innovative as Microsoft would have us believe? Or is it merely a chance to integrate the separate Microsoft products more tightly and to increase the users' vendor dependence?

Let's say that a new version of a Microsoft application comes with documents in a Microsoft proprietary format (e.g. the .CHM format, a Microsoft-proprietary version of HTML, used for help files and such). This is something that has happened many times in the past and will continue to happen. The use of this proprietary file format means that we're now suddenly forced to install (or update to) a recent version of Internet Explorer (which means installing or updating Outlook as well) because no other application will correctly support this file format. Since earlier version of IE under Windows were designed to conflict with other browsers (e.g. Netscape Navigator) something as simple as online documentation in a "product update" could mean to discontinue competing products in favor of a Microsoft-only environment. These days, in the post-IE5 era, such conflicts aren't the problem they used to be, but still IE and Outlook represent additional overhead, additional security vulnerabilities, and additional maintenance. And the user has no choice but to accept that.

And what good is such an update, really? What are the innovations in, say, Office XP over previous versions? The most significant 'improvement' is that new versions of Office produce documents that are incompatible with older versions of the same applications. If I create a document in a current version of MS-Word (e.g. Word 2000 or XP) and mail it to a friend who still uses an older version (say, Word '95), he cannot read it, view it, print it or anything else. He's forced to let Microsoft ram an unwanted and expensive upgrade down his throat before he can use my document.

And it doesn't stop there. Let's take a look at Office 2000. Quoting PC-World, June '99:

> *"An interesting issue is that Office2000's HTML format is incompatible to some extent with almost any other program, including MS's. So you can create all sorts of groovy Word, XL, or PP documents, and only people with Office2000 can read them. Even IE4 and 5, and Front Page 2000 can't read XL or PP files in HTML format without significant distortions in the display."*

New releases of Microsoft products generally don't contain any significant innovations whatsoever. As far as improvements are concerned, a "new" release like Office 2003 has barely made more than a ripple: it doesn't do anything better than versions five years old do. It's the same with Office 2007: the only real difference with Office 2007 is the user interface, and most users don't consider that an improvement anyway. In fact you could take versions of Word and Excel from 10 years ago and they'd do the same job just as well. But still we have to keep buying new versions at steep prices in order to maintain document compatibility with our fellow users. Microsoft calls this "the freedom to innovate", and waxes poetic about "all this exciting new technology that has been invented by Microsoft".

As if the document's version-dependence wasn't enough, Microsoft also discontinues serious support for each product version soon after a new version is released. Although by the end of 2002 Microsoft announced prolonged support in the future for their major products, support for older versions remains limited. Several months after Windows Vista's release the new version proved less than popular, mainly due to its many driver issues, bizarre hardwere requirements and strangely expensive licenses. Microsoft's response to the continuing demand for Windows XP was to announce that by the end of 2007 XP licenses will no longer be available.

If you have problems with any Microsoft product, you are invariably encouraged to buy a new version, usually at a rather steep price. But when you do, you'll find that the bugs haven't been fixed, that new bugs have been introduced, and that all design flaws have been perpetuated.

Proprietary lock-in pushing bad products

Microsoft forces us to buy substandard, proprietary technology along with Windows. Take ADSI for example. The Active Directory capabilities in Windows 2000/XP are much harder to integrate into a multi-platform environment (e.g. in combination with Novell Netware or Unix systems) than the more primitive domain services (which could be taken care of by means of a simple redirection mechanism). Of course Microsoft has done little to facilitate the integration of ADSI with other products; ADSI is engineered to promote Microsoft-only environments; it's an immature product that can't handle multi-vendor or multi-platform environments and scales poorly.

After only a few months of use in the larger corporate environments, ADSI's limitations were already painfully obvious: only 5000 users per group, single points of authentication (which means that remote offices are dependent on the availability of WAN links for local log-on) and most painful of all: the lack of adequate record locking, which means that two simultaneous updates of one record will result in serious loss of data.

ADSI's limited scalability was again confirmed by the Gartner Group in August 2000: large corporations will suffer from excessive overhead and network load when implementing ADSI over, say, 300 offices or more (something that can be, and has been, successfully done with NDS). Microsoft's counterargument was that their own ADSI-based network contains some 39,000 PCs, but they neglected to mention that those PCs are scattered across multiple non-integrated domains. And of course this "metadirectory solution" touts being LDAP and ODBC compliant, but fails to mention that it requires custom meta agents to talk to an X.500 directory or sync engine (other than MS DirSync).

Draw your own conclusion; mine is that Microsoft has again released a half-baked product that hasn't been seriously thought through by qualified networking software architects, and no amount of patching or service packs will be able to remedy all the basic design flaws.

And of course real Active Directory support is available only on the Windows 2000 platform which doesn't have native NDS support, but still users will have to adopt it eventually. (Soon most applications and drivers will demand it, and of course Microsoft has already discontinued serious support and code maintenance for NT versions previous to XP.) Also, Microsoft has shamelessly admitted that the flaws in ADSI mentioned above would not be fixed before Windows XP, thereby effectively forcing large customers to accept another mandatory "upgrade" and of course even more vendor-dependent features.

So much for freedom of choice.

Bundling

Microsoft has always claimed that the bundling of application software with Windows was only intended to improve quality, and that consumers are better served by the fact that both operating system and applications are produced by the same company. Well, we've seen how that goes. Word Perfect was a better word processor than MS-Word ever was (read: it delivered a better quality of word processing, whereas Word only contains more gadgetry). But when Windows 3 was released, few application developers had caught up with the need to entirely rewrite their application code. Only the Microsoft Applications Group was ready.

As it turned out, the latest release of Word at the time just happened to be fully compatible with Windows as soon as it hit the market, while WP Corp. struggled to get their DOS-version ported to Windows - with an unsurprising lack of success, as they had previously been forced to write DOS-dependency into their program code, due to DOS's lack of decent device support. And WP Corp. wasn't the only one: when Windows was first released most competing software vendors soon discovered that porting their existing DOS applications to Windows looked easier than it was, and that it took a complete re-write to produce efficient and stable code.

It's also common knowledge that MS applications perform much better under Windows than competing products ever can, since MS controls the API (Application Program Interface) and uses undocumented features to enhance their own products. Compare Internet Explorer, for example, with other browsers: IE hooks directly into Windows' internals while others are limited to documented API calls. And since IE and Windows share major chunks of code, firing up IE is much faster since when you start Windows you already preload most of IE. But Microsoft still denies having an unfair advantage over competing developers of application software. Instead they call this bundling "the freedom to innovate".

Can a monopoly innovate?

Let's take a good, hard look at this idea. Are a monopoly and the bundling of products really conducive to innovation?

Imagine for a moment that Standard Oil hadn't been stopped by the Sherman Anti-trust Act at the beginning of the 20th century, and had gone on to seize complete control of the fuel market. Let me stretch your imagination even further: suppose Standard Oil had also bought the Ford Motor Company. Owning virtually every gas station around, SO could have switched to a type of fuel uniquely suited for their own automobile products, and less well suited (and eventually unsuited) for competing cars. Consumers would have had no choice but to switch to SO-powered Ford automobiles. Both competing fuel vendors and automobile manufacturers would have been history.

If this had actually happened, what would car traffic look like these days? Think about it: would we have had a wide choice from affordable, safe and dependable mass-produced cars running 50 miles to a gallon? Or would we be driving a glorified Model T instead, in any color as long as it's black, at the original price or more, corrected for a century of inflation? (Today most people can afford a car. A century ago automobiles were far beyond most peoples' budgets.) And what would we pay for a full tank of gas, with oil prices being whatever the sole supplier says they are?

That is why monopolies and the wholesale bundling of products are bad. Bundling does not lead to innovation. Instead it merely lends power and control over the masses to those who practice it. This is interestingly, and maybe more than incidentally, reflected in the root of the word 'fascism', which is derived from the Latin 'fascio', or 'bundle'.

To illustrate: In November 2003 (during a major slump of the IT market) Microsoft declared a quarterly turnover of $2.81 billion for their Windows division, with a $2.26 billion net profit. Read: an 80.5% net profit margin on a multi-billion dollar turnover. MS-Office did slightly less well, with a mere $1.63 billion net profit, on revenues of $2.29 billion. On 30 September 2003 Microsoft had $51.62 billion on the books. And this at a time when the entire ICT industry couldn't afford to spend any money that can possibly be saved! And as if that wasn't enough, subsequently Microsoft's fourth fiscal quarter showed not only a 15% increase in turnover, but over 80% increase in net profit. In 2005 Microsoft's annual turnover had reached $40 billion with an annual net profit of $12 billion. The first quarter of 2006 showed $2.98 billion on a turnover of $10.9 billion, and the first quarter of 2007 netted a stunning $4.93 billion on a $14.4 billion turnover. That's almost five billion US dollars net profit in three months! The second quarter of 2007 continued the trend with another 13% increase in turnover, and subsequent financial figures proved this growth to be persistent.

This is what happens if a monopolist has been allowed to eliminate all serious competitors. By contrast, other Microsoft divisions such as Home Entertainment still have to compete with other players in the market, and these divisions declared losses up to a few hundred million dollars.

Perception is reality

A central principle in Microsoft's marketing is that it's far more important what the customer thinks he's getting rather than what's actually being delivered. Another major strategy is to hide the fact that Microsoft can never deliver what they promise, so that the customer will keep purchasing new versions over and over again.

Of course hardball sales tactics have never been the exclusive domain of any one corporate software company. And all is fair in love, war and marketing... Or is it? What about foul play? What about the distinction between competition, the prevalence of sales targets over ethics, and illegal practices?

For most of its history Microsoft has been involved in legal actions, the most important of which has been the investigation by the US Department of Justice (DoJ). This has culminated in the late nineteen nineties with the so-called anti-trust trials. The testimony from Microsoft's competitors was especially interesting to hear. Intel Vice-President Steven McGeady, called as a witness, quoted Paul Maritz, a senior Microsoft vice president as having stated an intention to "extinguish" and "smother" rival Netscape Communications Corporation and to "cut off Netscape's air supply" by giving away a clone of Netscape's flagship product for free. IBM representatives testified that IBM had been forced to drop OS/2 when Microsoft threatened to raise their prices for Windows OEM licenses. Digital Research demonstrated how Windows 3.11 was tweaked to crash when running on top of DR-DOS instead of MS-DOS. These are only a few examples; the list goes on and on.

Microsoft claimed in defense that this was all "innovation" and that the integration of Internet Explorer was a technical necessity. The Department of Justice then went on to demonstrate that this was a blatant lie, and made short work of Microsoft's entire defense plea. Microsoft in turn didn't even manage to present credible witnesses; all those who testified either had a significant interest in Microsoft or could be put at a significant disadvantage by Microsoft. Ironically, Microsoft's own witnesses, and even Gates' own testimonies, did their own case more harm than good. When Gates was summoned to testify in the case as the chairman of Microsoft, he was called "evasive and nonresponsive". He argued over the definitions of words such as "compete", "jihad", "concerned", "ask", and "we". BusinessWeek reported, "Early rounds of his deposition show him offering obfuscatory answers and saying "I don't recall" so many times that even the presiding judge had to chuckle. Worse, many of the technology chief's denials and pleas of ignorance have been directly refuted by prosecutors with snippets of E-mail Gates both sent and received."

US DoJ: Findings of Fact

On 5 November 1999 the DoJ published their Findings of Fact, and concludes, to condense the original document into a nutshell, that Microsoft has used foul play, has manipulated the market, has impeded progress, has harmed the IT market, the user community and consumers, and has violated anti-trust regulations:

> *"The ultimate result is that some innovations that would truly benefit consumers never occur, for the sole reason that they do not coincide with Microsoft's self-interest."*

On 3 April 2000, the DoJ went on to state their Conclusions of Law and Final Order, leaving even less to the imagination:

> *"[Software bundling] cannot truly be explained as an attempt to benefit consumers and improve the efficiency of the software market generally, but rather as part of a larger campaign to quash innovation that threatened [Microsoft's] monopoly position. [...]*
>
> *In essence, Microsoft mounted a deliberate assault upon entrepreneurial efforts that, left to rise or fall on their own merits, could well have enabled the introduction of competition into the market for Intel-compatible PC operating systems [...] thereby effectively guaranteeing its continued dominance in the relevant market. More broadly, Microsoft's anticompetitive actions trammeled the competitive process through which the computer software industry generally stimulates innovation and conduces to the optimum benefit of consumers."*

I won't bore you with the rest of the legalese (which you can read for yourself, if you're so inclined, in the original document) but the bottom line is that Microsoft was found guilty as charged.

This ruling was in part (not in whole) reversed by the U.S. Court of Appeals and sent back to a lower court for reevaluation, not because the facts that led to the initial ruling were invalid (the court unanimously found that Microsoft engaged in unlawful conduct to maintain its dominant position in the operating systems market) but mainly on the grounds of unprofessional conduct by judge Thomas Penfield Jackson, who discussed his personal feelings about the case in the press immediately after the ruling. Although Microsoft claimed victory after this partial reversal of the original ruling, the essence of that ruling still stands, and Microsoft has still been found guilty of illegal monopolist practices and other unlawful conduct, such as:

- Exclusive agreements with PC manufacturers to bundle Microsoft products
- Overruling the users' decision to use Netscape Navigator
- Mixing Windows and browser code to prevent the removal of Internet Explorer
- Agreements with ISP's to exclusively promote Internet Explorer
- Exclusive agreements with developers to create software that forces Internet Explorer to be the default

browser
- Making Internet Explorer the exclusive browser on the Apple platform, by threatening to halt the development of MS Office for MacOS
- Lying to Java developers about Microsoft Java being cross-platform
- Pressuring Intel to discontinue development of their own cross-platform Java

Microsoft's response to anti-trust: business as usual

Shortly after this ruling Microsoft suddenly lifted the ban on the shipping of competing application software with Windows by PC manufacturers. This gesture was part of their attempts to mend their fences with the DoJ, but it's far too little and much too late to make any real difference as far as the IT market is concerned.

The release version of Windows XP proved to be nothing more than a continuation of Microsoft's monopolistic and anti-competitive practices. After being found guilty of forcing Netscape (later AOL Time Warner) out of the market with Internet Explorer and HTML, they continued the practice against Real Networks, and they are thumbing their nose at the DoJ and at AOL Time Warner by bundling MSN Messenger and MediaPlayer with the OS. You cannot have Windows XP without MSN Messenger, and it is cumbersome to install either of the other Instant Messaging Services into Windows XP. RealPlayer has already lost much of its market share to the bundled MediaPlayer, and is likely to be the next victim of Microsoft's product bundling strategies and suffer the same fate as Netscape did. How long do we have before people totally give up on AIM or ICQ?

Even consumer organizations have become worried now, as shown in a **study** by **four major consumer organizations** in the US in September 2001.

In the aftermath of the legal wrangling that followed the antitrust trials in the US, both Gates and Ballmer claimed repeatedly that it is technically impossible to remove application software like Internet Explorer and Media Player from the Windows distribution, and that Microsoft would have to take Windows off the market if a court order forced them to remove it anyway. This is of course nonsense. Microsoft has always claimed to be able to make anything that can be called software, and they have never refrained from doing so. They can put anything into Windows they want, and now they're unable to remove something from it and to implement a workaround to deal with any side effects? That's hard to believe.

The European Commission didn't believe it either, and the European Union's court ruled against Microsoft in 2004. Microsoft complied with the EU ruling and produced a Windows version without Media Player, thereby proving that Gates and Ballmer had been willingly and knowingly lying in their teeth when they said it couldn't be done. Of course by that time such proof was hardly necessary anymore. By the end of May 2002 Microsoft had already announced, in response to earlier legal settlements, that Windows XP Service Pack 1 would incorporate changes to allow consumers and PC makers to override Microsoft's default media products, and replace them with competing products. In other words, after Steve Ballmer's earlier statements of having to take Windows off the market if this ever came to pass, Microsoft fixed it with no more than a service pack. This would make Ballmer's hyperbole rather laughable... if the matter weren't so serious. In fact, it should tell us two things.

First, Microsoft admits having lied about this for years. So what else have they lied about? They said Media Player could not be overridden. They also said, in court, that Internet Explorer couldn't be overridden. Perjury is such a nasty word.

Second, Microsoft's claims about how this puts an end to unfair conduct and anti-competitive monopolist practices should be seen against a long history of consistent lying. We should not be surprised to discover other fraudulous practices. For example, much of Microsoft's removal of offending components could be limited to hiding the associated icons. Or code could be moved from application executables and hidden in DLLs or other obscure modules. After proof of the blatant lies we've seen recently, anything is possible. Whatever the case, we'd better not expect miracles.

At least not if Windows Media Player is any indication: after installing an update of WMP in Windows ME or XP, the application cannot be removed since it replaces parts of the OS. Even a service pack can be uninstalled, but WMP can't. Uh-huh. Furthermore, Windows XP Service Pack 1 made other changes to the system as well, to achieve 'further compliance' with several court rulings. This apparently included changes made to Outlook Express, that caused Outlook to label Microsoft's competitors' documents as dangerous, in particular Adobe Acrobat documents. However, the number one virus carriers in the world --Microsoft Office Documents with macros-- were not blocked.

Meanwhile the EU continues to take exception to Windows Media Player, which Microsoft still bundles with all major versions of Windows. The EU holds that this gives Microsoft an unfair advantage over competing media formats and is essentially a continuation of the same anti-competitive strategies that the DoJ objected to. The EU demands that Microsoft allow fair competition from other content providers and software manufacturers, but at this time of writing Microsoft's usual obstructive and delaying tactics have continued to prevent the EU from enforcing any legislation upon the company.

In short, it's still business as usual. The lies go on and on and on. After having testified in April 2002 that too many versions of Windows would be bad for consumers and for competition, Microsoft essentially doubled --to about two dozen-- the number of "current" versions of the operating system software. Between November 2002 and April 2003, for example, Microsoft released three new versions of Windows XP alone. This, and everything else, should should tell us a lot about the trustworthiness of Microsoft's testimonies.

Little to fear

Let's face it: even after officially being found guilty, Microsoft has little to fear. The sad truth is that the company has grown too big to be seriously affected by something as trivial as law and order. Microsoft could buy large portions of the United States if they wished to. Microsoft could buy several small countries. Microsoft knows the inside of the software that powers much of the world's economy; software that runs on government computers, including those used by the DoJ, the CIA, FBI and KGB...

But I digress. Suffice it to note that, according to a study by Common Cause, Microsoft has spent millions on political lobbying, doling out large sums across a wide spectrum of political activities since 1997. These millions are of course a mere pittance for Microsoft. Should the political climate turn unfavorable, I'm sure a few hundred million (which is only a small percentage of Microsoft's annual net profit) can easily be reallocated from the marketing budget to politics.

As much as I applaud the efforts of the DoJ to expose the practices of Microsoft for what they really are, I have to admit it's always been unlikely that effective action against Microsoft would ever be taken. The DoJ's ideas on how to make Microsoft cease and desist (e.g. the proposal that the company be broken up into different business units) could not have had the desired result, even if such measures could actually be enforced. So when the dust settled, Microsoft was still standing, grinning and raising a finger at the world, the user community and a powerless DoJ.

In fact, Microsoft has managed to appropriate some of the legal aftermath and turn it to their own advantage. For example, in January 2003 Microsoft settled one of their big anti-trust cases in California. Under the terms of the settlement, Microsoft agreed to pay back $1.1 billion to their customers. If part of that sum is not claimed, Microsoft will donate 2/3 of the remainder to schools, under the condition that at least half of the donation be spent on Microsoft products.

Hang on -- how's that again? Microsoft promises not to do it again, as they have done so many times before, never keeping their promise even once. Then they'll spend 1.1 billion dollars-- a sum that won't make a big dent in their annual revenues. But since usually less than 25% of the money is claimed in cases like this, they'll end up giving most of it to schools, half of it in the form of free Microsoft products, thereby eliminating the competition in just about the only market that Microsoft hasn't managed to monopolize yet. And they get away with it!

Something's very wrong here.

Expect no change

So far nothing has changed. And nothing will change. Things will just continue to get worse. Even after the DoJ's ruling, all available evidence suggests that Microsoft persists in practices that have been found unlawful. Next to R&D and marketing, Microsoft has now taken to budgetting major money (over $700 million in 2005 and more in subsequent years) for antitrust claims, rather than to clean up their act.

No matter how spectacular the innovations by competing vendors may be, the chances that they will be able to offer us these innovations so that we may benefit from them are practically zero. After all, Microsoft versions of similar products will already have been forced upon us with the installation of Windows, and Microsoft products will generally conflict with competing products. For example, when Windows XP was released, the media players from Apple (Quicktime) and RealNetworks (RealPlayer) wouldn't work any more, for no apparent reason. Users had to download patches or updates from Apple or RealNetworks in order to get these players to work again. And Windows XP came bundled with tons of multimedia applications to start with.

Even during many trials, Microsoft showed no signs of cleaning up their act. Arguing that conduct remedies were insufficient to stop Microsoft's anti-competitive and unlawful conduct, the DoJ reported that on July 11, 1999, "Bill Gates wrote an E-mail directing that Microsoft redesign its software to harm competitors" who make personal digital appliances. It indicated "a willingness to change the details of its Office applications to favor devices that run on Windows, even if doing so would disadvantage other customers who now rely on the Palm Pilot", officials said. The department noted that this was less than 30 days after the company's 78-day trial ended, in which it was accused of using similar tactics against Netscape and others. Microsoft went on to release PocketPC, the successor to the Windows-CE operating system for hand-held devices. Initially the above allegation of unlawful conduct was denied, but then Microsoft requested that Gates' E-mail be placed under court's seal.

Moving right along

The future doesn't look very promising either. Leaked-out beta versions of Internet Explorer contained hardcoded links to Microsoft websites, and have increasingly been designed as an integral part of Windows. And this will only become worse as more and more Microsoft products become Internet-based.

Microsoft's long-term strategy will target (read: attempt to appropriate) Internet Services, and promises to introduce even more proprietary standards than before. Microsoft still controls all major proprietary API's. The documentation they released as part of a legal settlement in August 2002 was incomplete and virtually worthless, not to mention full of errors that were obviously the result of sloppiness rather than of malicious intent. No adequate API documentation has been released since. That means that, with a shift to Internet-based computing, Microsoft essentially controls what will work or not for third-party software. During several early announcements of Microsoft's new Internet-based strategy, Bill Gates conceded that "while all .Net devices will have access to Microsoft's .Net infrastructure, those based on the Microsoft Windows platform will work better". Sounds familiar, doesn't it?

If you haven't got the full picture by now: Gates told the audience at the MS Developer Conference on 13 July 2000 that, quote, "the next two releases of Windows is where you'll see .Net built into the user interfaces.", unquote. He went on to outline the most profound changes in the User Interface that can be expected in the foreseeable future. One of Microsoft's software partners, who requested to remain anonymous, said:

> *Remember it's Microsoft we're talking about. Microsoft's number one priority in the post-Windows-2000 era is the same -- to make sure all devices are Microsoft-based.*

As a taste of things to come, MSN users noted as early as October 2001 that MSN suddenly required the use of Internet Explorer. Users of other (competing) browsers were redirected to a webpage where they could download IE. After considerable public pressure Microsoft dropped this requirement, thereby proving that there was no technical necessity for such a browser-dependency, but that it was a commercial issue only. However, a Microsoft spokesman warned that users of competing browsers would have a "slightly diminished experience" because non-MS browsers "do not support MSN's HTML standard". Go figure.

Now, several years later, many of Microsoft's earlier attempts to bind the user community to proprietary Internet-based technology have not yet panned out. .Net was eventually released as a development framework for network applications. Their attempts to seize control over third party authentication services have crumbled under public pressure, and the announced changes in XP and Vista have failed to materialize. However the long-term strategy to shift from desktop-based to Internet-based computing is still in effect, and will prove to be yet another attempt at exerting control over the user community.

MS marketing: the anti-truth

Some examples of untruth in Microsoft marketing are almost funny. In the first months of 2002, Microsoft (along with partner Unisys) put up a website with the title "We have the way out". This website was part of a campaign that used slogans such as "Unix makes you feel boxed in. It ties you to an inflexible system." The ICT community was vastly amused: this website ran on Apache and Free BSD Unix. Then, in August 2003, Microsoft changed their DNS so that requests for www.microsoft.com no longer resolved to machines on Microsoft's own network, but instead were handled by Akamai's caching system... which ran Linux.

But most of all, Microsoft continues to, how shall I put it, adhere to rather peculiar ideas of what's true and what isn't. Spreading FUD (Fear, Uncertainty and Doubt) and other forms of misinformation has always been normal business routine for Microsoft. For example, on 22 December 1999 the Microsoft website blandly stated that:

> *"These are just a few of the features Windows 2000 Server offers that aren't found in [Novell] NetWare: Integrated namespace support, file compression, configurable block size, mirroring, duplexing, striping with or without parity, removable device support, link tracking, integrated content indexing, user-definable file properties and a tracking log to audit storage services utilization."*

They went on to state that NDS (Netware Directory Services) is known for its poor scalability, then they emphasized that Active Directory supports LDAP and DNS (yes, for Microsoft that's a novelty all right) and to cap it all they called Active Directory "secure", I kid you not.

While I'm not sure what they mean with buzzwords like "link tracking" or "integrated content indexing", I've found practically all the above features in Netware 4 since 1993 (!) while NDS has already been scaled to handle billions of objects. On the other hand, Microsoft failed to mention important weaknesses in Active Directory, such as the inability to adequately protect sensitive data from the Administrator account. Granted, Administrator's access rights to an Organizational Unit can be revoked, but the Administrator account can retake those access rights at any time. In other words, it's not possible to adequately shield sensitive data from the Administrator. OK, it is possible to detect unauthorized access (e.g. through a security audit) after the fact. But that's about it. This weakness was also present in Netware's earlier bindery-based architecture, which is one of the reasons why Novell abandoned in 1993 with the release of Netware 4.0 and the switch to NDS.

In April 2001, Microsoft spread the rumor that Novell was moving out of the software business and even managed to get it published by third parties. Microsoft eventually modified the statement in response to demands from Novell. However they repeated this nonsense on 1 October 2001 in a direct mailing to Novell customers. This marketing piece suggested that Novell server products would "expire" at some unknown date in the near future. This is not true; there is no "expiration date" on Novell products, they keep working indefinitely. They also claimed that Novell, after its merger with Cambridge Technology Partners, would discontinue software development and shift to consultancy. They implied that Novell customers would soon be left with a server platform without the full support of its manufacturer. Novell of course filed suit, but much of the damage had been done.

Even in their own certifications Microsoft attempts to misinform their audience. A reader of this paper reports:

> *I have recently been forced to acquire A+ certification. The content of the exam was weak at best and tested a minimum of skills (after using their prep materials I STILL hadn't found a good explanation of memory timings, but that didn't stop me from getting a perfect score on the exam). It wasn't the sloppiness of the exam that bothered me, though... it was the fact that the whole curriculum is used to peddle MS as the champion of computing (or as the only existing option) and Windows as the only OS a "professional" would consider.*
>
> *MS through their puppet companies (CompTIA, PrepLogic, etc.) use these exams as written infomercials, and they LIE to do it. For example: The PrepLogic Network+ practice exam had (has?) a question on which Network Operating Systems are X.500 (LDAP) compliant-- apparently Windows NT 4's NTDS and Windows 2000'S Active Directory are considered FULLY X.500 compliant, while "Linux in any flavor is supposed not to have a directory service of this type"... Go figure-- the LDAP daemon I run daily doesn't exist!!!*

Rigged tests, distorted reality

In November 2001 Microsoft spread more lies when they published a whitepaper on their website that compared Embedded Windows XP with embedded Linux. Among other inaccuracies, the paper touted the superiority of embedded XP, called it "proven performance and reliability" It blithely ignored the fact that XP is Windows and therefore known for its unreliability, and that XP was brand new and barely tested at the time. It claimed that Linux is "a follower, not an innovator", based on the fact that Microsoft continues to integrate support for new "standards" in their products that the Open Source community struggles to keep up with. The opposite is true, and we all know it.

Nor is this the only example Microsoft's attempts to distort reality. In February 2001 Microsoft's Windows Operating System chief, Jim Allchin, stated that freely distributed software code such as Linux "could stifle innovation" and that "legislators need to understand the threat". The result of Open Source initiatives will be the demise of both intellectual property rights and the incentive to spend on research and development, Allchin claimed. He went on to call Open Source an intellectual-property destroyer, and stated that nothing could be worse than this for the software business and the intellectual-property business.

And it goes on and on: in October 2004 Steve Ballmer wrote an edition of 'Executive E-mail' titled "Comparing Windows with Linux and Unix" in which he stated, among other things, that an MS customer who ran Linux "migrated to Microsoft Windows Server System, and reduced Total Cost of Ownership by 25 percent, consolidated the server population by 50 percent, reduced maintenance time by 50 percent, and boosted developer productivity by 200 percent." I suppose it is possible to replace a freely available, robust and independent Open Source environment with a proprietary, expensive and unreliable product from a vendor known for a sales-driven development strategy... but I can't see how. In any case you'd spend more money rigging the comparison than you'd eventually save.

Ballmer also writes that "A number of third-party reports have questioned how safe the Linux platform really is" and he continues to suggest that Windows is at least as secure as Unix, quoting another success story in which "the core reason for selecting Microsoft was the increase in network security, complemented by the ability to reduce patch-deployment time by 50 percent while cutting unsolicited e-mail by half."

I have to admit that Ballmer (or his ghost writer) has worded it brilliantly. He even manages to pass off the ridiculous amount of security patches released by Microsoft as an indication of how well Microsoft's products are being kept secure. Rather than, say, as an indication of how many security holes there are still being discovered on a daily basis in products that have been in maintenance mode for a long time.

After this nonsense, Ballmer's conclusion comes almost naturally: "it's pretty clear that the facts show that Windows provides a lower total cost of ownership than Linux; the number of security vulnerabilities is lower on Windows, and Windows responsiveness on security is better than Linux; and Microsoft provides uncapped IP indemnification of their products, while no such comprehensive offering is available for Linux or open source." However the actual truth is different. There are lies, damn lies, and Microsoft "facts". Unfortunately whatever nonsense Microsoft publishes is generally repeated indiscriminately in the press.

Microsoft's marketing machine continues to make groundless promises. Windows Server 2003 is being marketed as a huge cost saver. Advertising campaigns promise (without qualification) that you'll "save a nickel on every transaction" just by switching to Windows Server 2003, or that you'll reduce the complexity of your infrastructure by implementing Active Directory and "save two million dollars a year". Of course such unqualified promises are meaningless without being put into any context (such as what kind of infrastructure and ICT environment you have) but that's all irrelevant when Microsoft salesmen step into the boardroom.

Another distorted aspect of the whole Windows Server 2003 campaign to "do more with less" is Microsoft's announcement that their latest and greatest will save you "millions of dollars" because now users can restore their own accidentally deleted documents. Wow. What a great and innovative feature! Users of Novell Netware especially will appreciate it. After all, they've been using Netware's SALVAGE command for exactly this purpose since the 1980's. And this simple feature, long present in a competing product, will save millions on user support now that it's finally been introduced in Windows? Microsoft would have us believe that this makes Windows 2003 a must-have... rather than recognize it as proof that the competing product has been the better one for over 20 years, or as proof that Windows users have wasted millions on user support for lack of such a basic feature. In fact they'd rather not mention this at all. Neither do they mention the fact that no 64-bit version of Windows Server 2003 or XP Professional existed until May 2005, which made Linux the only operating system available on PC-grade hardware (e.g. LAN servers) that fully utilized the power of 64-bit CPUs. Even today, the 64-bit version of Windows consists mostly of 32-bits code, just like earlier 32-bits versions of Windows were mostly 16-bits code under the hood. Yet Microsoft would have us believe that, in comparison to true 64-bit products with a proven track record, Windows is the superior one.

While all this is going on we keep seeing Microsoft-financed "research" that pronounces Windows both superior to and cheaper than Linux, in spite of independent research, daily experience and common sense proving the opposite time and again. Why is it that Microsoft has to pay researchers and analysts before they'll come up with conclusions that favor Microsoft?

Censorship

Releasing biased or distorted reports on the ostensible benefits of Microsoft products is not the only tactic by which Microsoft attempts to manipulate mass opinion. Occasionally an objective (and highly damning) report on the real state of affairs is being released, in which case Microsoft's preferred response is to have it quietly removed from public view.

For example, Bloor Research once compared Microsoft's database engine with its main competitors, and tested DB2 on AIX, DB2 on NT and MS SQL Server 6.5 on Windows NT. The report was published under the title "The Realities of Scalability" in March 1997. It is still, today, an impressive body of work. Over 130 detailed pages of complex tests really put the three database engines through the wringer. Bloor tested their performance under many different conditions and performed a rigid statistical analysis on the results to determine their significance. The resulting conclusion was highly critical of MS SQL Server 6.5, especially in comparison with the other two contestants. It found SQL Server to be seriously lacking both in scaleability and reliability, cited a number of repeatable failure states, and used the words "dramatically worse" when comparing it to the alternatives.

The report was quietly suppressed. The acid test is looking on Bloor's own website. You will find an archive there that does, indeed, go back to 1997, but there is no record of any database scalability report.

Unfounded accusations

In May 2001 Microsoft CEO Steve Ballmer stated in an interview that "Linux is a cancer that attaches itself in an intellectual property sense to everything it touches". Even better; the user license for the second beta version of Microsoft's Mobile Internet Toolkit came with a condition that the product not be used "in combination with potentially viral software". The document went on to name examples of what Microsoft considers "potentially viral software": any software distributed under the GNU Public License (the most common license for Open Source software) and also the Lesser General Public License, the Mozilla Public License and the Sun Industry Standards License.

Spreading FUD (Fear, Uncertainty and Doubt) is a staple ingredient of Microsoft's ongoing attempts to damage competing products. A good example is the rather shameful affair of Microsoft **hiring SCO** for over $60 million to **make bogus claims of copyright infringement by Linux** and threaten Open Source and the Gnu Public License in general. Of course SCO proved unable to produce a single shred of evidence to back up their accusations, and the whole matter was swept under the carpet in the usual fashion... until 2007, when Microsoft did it again, when Steve Ballmer accused Linux of violating hundreds of Microsoft patents, and urged Novell users to purchase a license from Microsoft in order to prevent legal repercussions. If at first you don't succeed, spread some more FUD...

The truth emerges

Microsoft's track record speaks for itself. Decades of non-innovation and monopolist practices. Legal procedures, lies that border on perjury (and perhaps even cross that line) and finally a ruling by the DoJ that Microsoft mostly ignored. Nothing but rewrapped old technology from competing products, touted as the hottest thing since sliced bread and even marketed as a cost saver. Nothing but misinformation, FUD and outright lies on web pages aimed at the ignorant. Suggestions that TCP/IP is a Microsoft protocol, claims that the integration of Internet Explorer and Media Player in Windows is a technical necessity, unrealistic promises about cost savings and reliability, slanderous untruths about competing companies and products, and attempts to paint the Open Source community as being fascist and full of copyright infringements.

Interestingly, though, Microsoft's own actions show the truth. Internet Explorer 6 was released years ago, and has been in maintenance mode ever since. Only in response to the hugely popular Mozilla Firefox webbrowser (an Open Source project) Microsoft decided to start development of IE7, with the deliberate intention of adding many features that IE6 lacked and Firefox has. Not only does this prove that freedom of competition (and not a monopoly) drives innovation, but it also clearly shows that Open Source drives innovation and even causes Microsoft to follow these innovations. It proves Open Source to be not a cancer but rather a cure. The fact that IE7 only runs on XP and Vista and not on earlier Windows versions or competing operating systems, on the other hand, merely points out one of the ailments in need of such a cure.

Meanwhile the lies continue. In fact Microsoft has been proven guilty of most of the things they accuse their rivals of, and then some. Can you say "dishonest"? It's spelled M-I-C-R-O-S-O-F-T.

6. Caveat Emptor

"If I can have honesty, it's easier to overlook mistakes."

-- James T. Kirk, in Star Trek episode "Space Seed", stardate 3141.9

Obviously Microsoft doesn't have the slightest respect for their customers (note how I try to avoid the word 'contempt' here). Their track record speaks louder than words on this point. Microsoft is a truly digital company: Microsoft is number one, and the millions of consumers who use their products are the zero's.

Before doing business with any company, most customers like to know if they're dealing with a reliable party. Well - to summarize:

User problems not addressed directly

Microsoft rarely fixes user problems. Granted, service packs for Windows '95, NT, Office and such have been released, but only in an attempt to fix blunders that should never have been released in the first place. And each new service pack introduces new (and untested) features to Windows, so the problem is always perpetuated.
Instead of solving problems with new interim releases, MS issues only major new releases with 'additional features' and loads of extra bells and whistles that distract the attention from the main issue: software quality. Could someone tell me what the real structural improvements there were in the latest Windows ME release?
No apparent quality control

Microsoft does not seem to have a quality plan, carries no ISO900x (or any other) quality certification that I know of, and apparently does not intend to acquire any. Worse, Microsoft does not seem to have full control over the contents of their own software. The 'Weenie Issue' in IIS and the 'Gray hair issue' are good examples. Granted, these may be relatively harmless bits of code, but the point is that if these can pass through Quality Control, so can serious flaws, security backdoors and the like. That's assuming that quality control is actually part of Microsoft's production process and that it's intended to do a serious job. If it were serious, it would have frowned upon hidden flight simulator and pinball games in release versions of Office. (Such deliberately hidden features are called 'Easter Eggs' and are usually put in by developers as a prank.)

Millennium bugs forever

In 1998 Microsoft released one of their major products (Windows '98) that turned out not to be millennium-proof. After no fewer than five service packs for NT4, users still needed to install several post-SP Y2K hotfixes by the end of 1999. (Can you say "Quality Assurance"?)

No support on OEM sales

Microsoft refuses to support their own products if those products have been sold to the customer through an OEM distributor.

Products further Microsoft's own interests

Microsoft products are designed to benefit Microsoft. Even in the days of Windows 3.11, they incorporated code to display an incorrect error message if the competing product DR-DOS was detected. How does the customer benefit from this? Microsoft uses their customer base as a pawn in the battle for market domination. Windows '98 forces the user to run Internet Explorer, regardless of the needs, desires and wishes of said user. There is no technical reason to do so, it is a monopoly issue only, as has been proven in the course of legal procedures against Microsoft. And who else but Microsoft would put a feature in MSN Explorer to spam your entire address book with endorsement messages gushing praise about "this exciting new product from Microsoft"? (Incidentally, this spam has your name on it. It's your reputation going down the drain.)

Microsoft sabotages alternatives

Microsoft manipulates the market by making it cumbersome to use competing products instead of offering truly better alternatives, enforces proprietary extensions to otherwise open standards and introduces deliberate version conflicts.
Doubtful business practice

Microsoft is not above playing fast-and-loose with the law when it comes to killing off alternative suppliers. They prefer to use pressure and force to restrict the consumer's free choice, rather than to allow true and healthy competition based on the merits of different products. Their methods to accomplish this can be called doubtful from a legal point of view, to say the least.

Microsoft spreads lies

Microsoft lies to the customer (yes, they LIE) to deny the quality of competing products and to make their own look more favorable. Microsoft will look you straight in the face and tell you that NDS is known for poor scalability, that Netware doesn't support basic file system features such as sub-allocation and compression, and that Windows outperforms Unix.

Forced upgrades

With the introduction of Office XP, Microsoft resorts to a new upgrade policy: force-feeding. You'll upgrade whenever Microsoft tells you to and meet their deadline, or else face a huge cost increase the next time you upgrade. That's the kind of freedom of choice that Microsoft gives you: either pay up now for something you don't really need, or pay much more a little later when (not if) new products will be made incompatible with previous versions.

Still Microsoft hails the free market and tells the customers that they benefit from this.

Uh-huh.

7. Where are you forced to go today?

"Remove me from this land of slaves,
Where all are fools, and all are knaves,
Where every knave and fool is bought,
Yet kindly sells himself for nought.

-- Jonathan Swift

Microsoft has such a nice slogan. "Where do you want to go today?" But in truth Microsoft couldn't care less where you want to go. All they care about is inflating their revenue at your expense. They'll tell you where to go. And you will go along with it. You can go easy or you can go hard...

A good example is the enormous market share that has been conquered by Outlook and Internet Explorer. Well, of course these products are the most widely used in the world! It's practically impossible to buy a PC without Windows these days, and Windows comes with Outlook and IE. Setting up Windows for its initial use involves the procedure for entering account data in Outlook and the use of IE as the system's default browser. Switching from these defaults to alternative products involves a conscious effort on the part of the user, removing Outlook and IE is practically impossible. But coercing the user to stick with Microsoft-supplied Internet applications is only a start.

Price gouging

Microsoft's prices have always been rather steep, but Windows XP and Vista offer striking examples of Microsoft's price gouging policies. Windows XP comes in two flavors: the 'home edition' and the 'professional edition'. Of course they're essentially the same product: the same kernel, the same user interface and the same bundled applications. At least Windows NT and Windows 95 were products of an entirely different caliber. Microsoft had intended to sell Windows NT and 2000 to the corporate sector and Windows 95/98/ME to the home and SOHO markets, but many companies used Windows 9x in the office because they could not justify the much higher price of NT and 2000. So the home edition of Windows XP came without a few features that are required in an office environment, such as network client support, group policies and roaming profiles. You don't really get much more software for your money when you buy the professional edition, but the few parts that are missing from the home edition are exactly the parts you don't want to do without in a corporate environment. No matter how you look at it, Microsoft obviously has decided to remove these portions from the XP home edition deliberately, in order to force the corporate sector to use the professional edition of XP, at about twice the price of the home edition.

XP's successor Vista knows no less than six differently priced versions. For the home market there is Vista Home Basic, Vista Home Premium and Vista Home Ultimate, while the business market gets to choose from Vista Business and Vista Enterprise. For "upcoming markets" (read: impoverished countries, mainly in Africa and Asia) there is the Vista Starter Edition. In all cases the cheaper versions have had certain features (all minor ones from an OS design standpoint) removed to make them unattractive for business use, but the bulk of the core code is essentially the same in all versions. As per usual, Vista is more expensive than its predecessor, too: Vista Home Basic is priced similar to XP Home Edition but far less functional, while Vista Home Premium has functionality comparable to XP Home Edition but comes at a higher price. In the corporate segment Vista Ultimate Edition is significantly more expensive than XP Professional.

Imagine that your automobile dealer wants to sell you a new car. You tell him that you will use it to go to the office in the morning and perhaps to visit a few customers as well. He tells you that for professional use you must buy the professional version of the car you wanted, which is essentially the same car at twice the price but comes with a nice 'professional' sticker on the doors and, if you opt for the "enterprise" version, has perhaps an extra light or two on the dashboard. And you don't have any choice, because the "home edition" of the same car has been modified so that you cannot get into your office parking lot.

Would you do business with him?

Mandatory upgrades

And it gets even better. A fine example of Microsoft's policy of force-feeding their products to their customers, and a fair indication of what Microsoft has in store for us, is the latest initiative to "simplify" their upgrade policy. Instead of having to agonize over the decision when to upgrade and having to choose between CUP, VUP, PUP or other upgrade schemes, we are now reduced to only one simple option: we are required to buy and install an update whenever Microsoft tells us to.

Under the so-called "Software Assurance Program", which became the only game in town when it replaced all existing upgrade policies, users had to upgrade to Office XP before 1 October 2001 (notwithstanding the fact that Office XP didn't hit the market before 31 May 2001!) or else be charged the full price for a new license the next time they upgrade. (This included a necessary hardware upgrade in many cases, since the XP product line doesn't run well on pre-1999 hardware.) Existing upgrade agreements were terminated on 1 October 2001. Just like that.

In fact the new update policy is an enforced subscription model. Software Assurance is only available for 'current' versions of Microsoft products, and Office 2000 was NOT CONSIDERED TO BE A CURRENT VERSION as of 1 October 2001, since Office XP had been released on 31 May, four months earlier. Furthermore, Microsoft has carefully neglected to emphasize that this extortionist scheme applies to all Microsoft products and not just to MS-Office. As of 1 October 2001 all server products, all desktop products and all application software had to be made 'current' and maintained under the Software Assurance Program, at a price of 29% of the original software cost. Microsoft intended to "reevaluate" this percentage after two years, but apparently never did. (One exception: the enforcement of Software Assurance doesn't apply to operating system products -- yet.)

As a result, many corporate customers faced an unexpected upgrade expense (in many cases a large one) to avoid having to pay the full price for their next upgrades. They also had (and will have) to implement brand-new and barely tested "service pack zero" versions of Microsoft products, on only four months notice before Microsoft declared existing upgrade policies on mission-critical application software null and void.

In the summer of 2005 Microsoft announced that Software Assurance would be mandatory in order to obtain an Enterprise license for Windows Vista.

Planned obsolescence

In a normal, healthy business model sales figures are generally a function of demand for the product which in turn reflects, among other things, its quality, value for money, performance in comparison to competing alternatives, and marketing. In Microsoft's case this obvously wouldn't be enough to keep up the sales of new product versions. Their product quality, or lack thereof, has already been discussed at length in this paper. Upgrades to new versions are strangely expensive but do not offer any new features that really pay for such an investment. Competing alternatives are scarce thanks to Microsoft's succesful anti-competitive tactics, but more and more major Open Source Software products owe their growing popularity to their excellent performance. And in spite of Microsoft's succesful marketing a large and growing number of their customers grumbles about never ending software glitches and expensive licenses, to the point where the company has become known for making bizarre profits by selling shoddy products.

Planned obsolescence has been an important factor in Microsoft's continued sales. In short, whenever a new product is being released the previous one (which is generally cheaper both in licensing fees and hardware requirements) is taken off the market and support for it is discontinued a little later. Some ten years after their release, Windows '98 and Office '97 are still being used by small companies and private persons, because they have all the features these users need (and then some) and they run on older, cheaper hardware. The only thing that eventually forces these users to buy an expensive upgrade is the fact that these products are no longer supported, so drivers are no longer available and security updates are no longer being released.

The main reason why large companies are generally quicker to replace discontinued software is that it is cumbersome to have to maintain different versions. That generally means that the introduction of a new product version (which soon becomes the only one available) generally signals the end of previous versions. While Office '97 may still do everything these users need, having to maintain Office '97, Office 2000 and Office XP in a single corporate network becomes such a pain that eventually upgrading to the "current" version of Office becomes the lesser evil.

Windows Vista is a good example of how planned obsolescence works. Some six months after its initial release the market still largely ignored Vista due to its many driver issues, its hardware requirements, its expensive licenses, and lack of a first service pack. But many hardware suppliers (especially in the market for notebook computers) soon shipped their products only with a preloaded OEM version of Vista, and Microsoft announced that XP will no longer be available by the end of 2007. While XP will still be supported with critical security patches for some time, its days are numbered. This leaves the customer no choice. While the market has shown little interest in upgrading to Vista, the planned obsolescence of XP has made Vista the only game in town.

Do the math

The increase in software cost in the next few years will typically be about 35 percent for companies who upgrade once every three years, and can be anywhere between 68 percent and more than 100 percent (!) for those with four year upgrade cycles, as marketing research bureau Gartner has calculated.

Since the maintenance agreement Microsoft wanted customers to purchase after the upgrade (before the October 1 2001 deadline) costs 29 percent of the full software price, you don't have to wait much more than a year to break even by not upgrading but putting the money in the bank instead of giving it to Microsoft.

The new scheme leads to ridiculous situations in which it is often cheaper to buy new software before the deadline and let it sit on the shelf for a few years instead of installing it, rather than to upgrade three years or so from now. Of course we'll eventually have to upgrade anyway, as new releases of Microsoft products introduce incompatibilities with the versions that are current today.

(**Note:** in an unprecedented response to pressure from large customers, Microsoft declared a 'transition period' from 1 October 2001 to 28 February 2002 to ease the pressure a bit. This gave users a bit more time to cough up the money for their mandatory upgrades. But apart from this minor delay, which is essentially nothing more than a nice gesture, the new scheme remained the same.)

At the time **Gartner** did the math, and their response didn't leave much doubt about the matter:

> *"Microsoft believes it has simplified its licensing; Gartner believes Microsoft confuses simplification with the elimination of options. Either way, most enterprises will pay much more. A typical enterprise with 5,000 desktops that upgrades Microsoft Office every four years will have its fees increase from $900,000 to $1.7 million."*

Similarly, the 4,000-member Dutch Network Users Association calculated that 86% of its members face a price increase. NGN chairman Vincent Everts noted that companies will have to pay just to qualify for Software Assurance, because companies must be running the most current version of Windows or Office to get the maintenance agreement. "They are forced to buy this program [...] even though they don't want it, and that's what a lot of people are very angry about," Everts said. And with good reason: by the end of 2006 it was clear that, with a five year interval between the release dates of Windows XP and Vista, none of the updates that so many Software Assurance licencees had ostensibly been paying for, had materialized. Forrester Research concluded that Software Assurance had only driven up the cost of software licenses significantly without actually delivering any benefit whatsoever.
Driving up licensing costs

Yes, Your Honor, here we have Microsoft demanding a revenue spike solely for their own purposes and promising consequences if customers don't cough up on schedule. This policy will of course be enforced through product bundling, tight control of document formats, and the deliberate introduction of incompatibilities of new products with existing ones. No significant improvements in the way of products, services or functionality has been offered to justify such inflated prices. Yet Microsoft insists that this new scheme is intended to benefit users, that 80 percent of their customers won't pay more than they used to (even in the face of simple calculations that show this not to be true) and that they've come up with this extortion scheme in response to demands from their customers. Excuse me?

But it gets better. In the summer of 2005, Microsoft CEO Steve Ballmer told analysts that Microsoft will release new, more expensive versions of Windows and Office. After the current 'Professional' editions of Windows (which raked in a few extra billions already) Microsoft now considers a Windows 'Enterprise' edition and a 'Premium' version of Office. However Ballmer did not clarify what extra features, if any, these versions will offer in order to justify their inflated licensing costs.

Another good example of how Microsoft only wants to protect revenues rather than serve their customers is the licensing technology that will be incorporated in all new products, starting with Office XP. The software license is tied to the PC's hardware, which is identified through the unique characteristics of ten hardware components, e.g. the MAC address of the network interface and the serial numbers of IDE harddisks. Licenses need to be 'activated' (for which you have to contact Microsoft). Licenses automatically become void (read: the software shuts down) after certain hardware modifications. In other words, if you replace a malfunctioning network card or hard disk you have to to contact Microsoft and kindly request that they 'reactivate' your license so that you may continue with your work. The license verification code also contains bugs that may result in Office suddenly shutting down and asking for an original CD for re-activation, which essentially leaves you without a functioning set of Office applications. How's that for enhanced productivity?

Microsoft doesn't care where you want to go today. You'll go wherever Microsoft tells you to go, period.

8. The road ahead

"The future is a race between education and catastrophe."

-- H.G. Wells

Honor where honor's due: Microsoft accomplishments are impressive. Gates and Allen started with practically nothing in the early seventies, and today Microsoft is perhaps the most commercially successful company in the world with a net worth that runs into billions of dollars. Their methods may always have been doubtful from both a moral and a legal standpoint, but they did turn inferior products into a commercial success, and a small startup company into the biggest money-making machine in existence today. This requires commercial genius, which Gates undeniably has, even though he never really contributed much to technology or to life in general. Being a great salesman, he has become perversely rich by selling bad, copycat products. Which only goes to show that for every rule there's an exception; in this case the computer industry's old maxim that wealth is a function of creativity and innovation. Commercial ingenuity will do as well, and Gates' marketing strategies have been nothing less than brilliant.

Even though the company has done little more than disguise various ideas as their own, it cannot be denied that Microsoft products have played an important role in the maturing of the IT industry. Microsoft (in a symbiotic relationship with chip manufacturer Intel, largely thanks to IBM's decision to allow third party manufacturers to clone the IBM PC design) has been one of the factors that made computer technology available to the masses.

When these accomplishments are regarded in their proper perspective, though, the picture that emerges is less than wonderful.

Riding the wave

Microsoft has been credited with being a stimulator of technological development, and has even been called "the jet engine of the new economy". The truth is, however, that corporate ICT investments and efforts have rarely triggered technological development, but at best followed up on it. Nuclear research (which lead to a better understanding of semiconductor materials and ultimately to the modern microchips) and computer research initially took place as part of the war effort during the 1940's. The first real electronic computers (Colossus and ENIAC) were developed at Bletchley Park in the UK and at the Moore School of Electrical Engineering of Pennsylvania, respectively. Both were developed on a defense budget. The cold war and America's urge to outclass the Soviet Union in achievements such as space travel triggered the founding of the Advanced Research Project Agency (ARPA) which developed much technology, including communications technology and ARPAnet. ARPAnet eventually matured and grew as other research institutes began to use it, and ultimately became the Internet. The World Wide Web, powerful as it is, was simply the next step to its inventor Tim Berners-Lee, who had no commercial interests in mind and is said to have described his extending existing hypercard systems with Internet capability as "a logical, perhaps even inevitable next step".

Moore's Law has been in operation for decades: computing power has roughly doubled every 18 months, while hardware integration has increased and hardware prices have dropped accordingly. The first cellphones weighed over 20 pounds, the first GPS receivers filled a desktop. Today we can buy such devices for a song; they fit into a shirt pocket and perform much better than those early dinosaurs. Bill Gates claimed that Microsoft helps keep prices down, saying that modern computing power is "the equivalent of getting a 747 for the price of a pizza". The nonsense of this statement is proven by the fact that similar evolution and price drops have occurred with technology that Microsoft has nothing to do with. The reduced price of computing power is merely a reflection of achievements in the fields of micro-electronics and physics, and of the simple economics of increased mass production.

Microsoft has ridden the wake (but rarely the crest) of these and other developments that stimulated technological innovation. They have followed the trend and capitalized on it, but they haven't driven it noticeably. They have contributed to making new computer technology available to the masses, and they created the foundation for today's IT market by turning available technology into commercially viable products, with great success. But as is so often the case, the initiator has long since become an impediment to further progress.

Two good things...

We owe Microsoft our gratitude for two (and only two) accomplishments. Before Microsoft came along, a computer owner could obtain software only from the manufacturer of that particular computer. If IBM had had it their way with PC-DOS, this would still have been the case. Microsoft changed that, first by selling BASIC interpreters to various competing manufacturers of home computers, later by doing the same with MS-DOS. Their marketing strategy was instrumental in changing a vertical market (where users buy all their products from one supplier) to a horizontal market (where many different suppliers sell components that go into the final end product). Both PC manufacturers and software manufacturers owe their existence in the PC market largely to that single fact. Also, by implementing an entry-level user interface in Windows, they made the PC accessible to the novice user.

These are Good Things, and Microsoft deserves due credit for it. But these accomplishments were mainly a by-product of Microsoft's marketing strategy. Their beneficial effects pale next to the damage that Microsoft has done.

...and countless bad ones

In today's Microsoft-dominated market, customers aren't served. They are being used. Microsoft manipulates the market, with the customers as pawns in their marketing strategies. The needs and wishes of the customers themselves are completely ignored, unless responding to those needs would help Microsoft in their rise to complete market domination. And all the while, Microsoft's marketeers paint this company as the best thing that's ever happened to us.

Microsoft replaced our dependency on different hardware manufacturers by an even stronger dependency on one software company (Microsoft) and one platform (Intel), and the pretty colors of a friendly-looking user interface only hide the very badly designed technology with which they've saturated the market. After changing a restrictive vertical computer market into a less restrictive horizontal one, Microsoft now moves back into a vertical market where all software products in an otherwise horizontal market are again available from (or with the consent of) a single supplier. Microsoft has expanded the market for PC software, but also set the standard for that market: notorious unreliability, sloppy code, and marketing interests that impede the introduction of new ideas. The software market is the only market where products are still accompanied by license agreements that state, often in a few thousand words that are extremely hard to read, that by using the product you indemnify the vendor against any claims, losses, or problems it may cause, even if the vendor knew about the problem before it sold the product. In some cases you even agree to let Microsoft remotely modify your software and you can't hold it liable if something breaks as a result. Can you see the an automobile manufacturer demanding that you sign a waiver before they'll let you drive one of their cars, in case the brakes don't work? Of course not. Yet this is considered a normal standard of conduct in the software market, and while Microsoft is not solely responsible for it, they have contributed greatly to the blind acceptance of this deterioration of quality standards. And if you don't like Ford trucks, you can buy a Jeep instead. PC users generally do not have that option.

Out of control

Microsoft has spun out of control, and rules the field of computer technology and the IT market with an iron fist. Whatever Microsoft's next whim may be, IT customers must follow it, despite forced upgrades, replacements of existing software and hardware, huge costs of ownership and ever-increasing overhead.

Customers in today's IT market no longer have a truly free choice about the products they want to deploy. It has become all but impossible to run an office without using Microsoft products. There are so many MS-proprietary document formats and protocols being used these days that Microsoft products have become the only game in town. Alternatives are no longer a real option, because the companies that used to offer them have either been assimilated or eliminated, and the remaining few are ailing after years of struggling to cope with proprietary standards designed for incompatibility with non-MS products.

The future: more of the same

And nothing will change. The old saying that "Sony will make more compact things, Apple will make more beautiful things, Microsoft will make more money" will hold true as it has done in the past decades. Microsoft is driven by sales targets, not by quality targets. No matter how bad their products are, as long as they continue to sell them nobody will get fired. An sell them they will, by force if necessary.

Microsoft has grown beyond control. Even the DoJ has been unable to come up with effective measures against the company. This demonstrates that Microsoft is now effectively above and beyond the law. Microsoft won't be unhorsed by competition either, because there's virtually no competition left. They'll just go on as planned, tightening their grip on the IT market and the user community, continuing to increase our dependency on their sloppy products, and forcing us to buy useless new releases every other year or so.

Computer hardware is more powerful than ever, but software efficiency and reliability have steadily dropped. PC programmers no longer know any serious quality standards for software development, users still don't do their work any faster or any more efficiently, and therapists are now treating something they call 'Technology Related Anger'.

And we're stuck with all this. That's the whole point: because of Microsoft, we're stuck with Microsoft. In fact we're stuck not only with Microsoft, but also (and especially) with what Microsoft has done to the ICT market. Even if Microsoft would cease to exist today, the PC software market would need years to recover from the huge impediment to innovation that Microsoft has become.

The damage has been done

For years and years Microsoft has stunted the true growth of computer technology under the guise of innovation. In the early years the PC and LAN market was driven by innovation that resulted directly in enhanced productivity. Simple products like DOS and DOS-based applications, even with all their limitations and drawbacks, achieved dramatic boosts in productivity. End users could streamline their business processes with batch files, macros and other forms of scripting, and applications were mostly limited to features that made sense. In those days the investments in automation usually paid off directly in the form of enhanced productivity. But as Windows began to dominate the market this tendency changed, and was ultimately replaced by exploding costs without any further increase in productivity.

Nowadays most desktop tasks actually take longer than they used to ten years ago. Most office applications such as Word and Excel lack features that users would like to have (such as simple macro's and scripting, control over markup codes and freely interchangeable documents) but eat up a lot of production time while users try to find their way through the maze of added bells and whistles and an inconsistent and unergonomic user interface. Windows is designed to be operated manually; the use of scripting and macro-based jobs is no longer possible.

What we need... but won't get

Ideally Windows should be replaced with an OS built around a robust kernel (the Unix approach comes to mind) but with the focus on user-friendliness that has been one of the deciding factors in Windows' popularity. Open API's and toolkits would let application developers create products with a uniform look and feel to the user interface. This could certainly be done with the technology available today. There are a lot of players in the software market who would be eminently capable of such an effort. But nobody in his right mind would invest in such a project, because commercial success requires displacing Microsoft from over 95% of the world's personal computers. The inertia of the market alone guarantees that such a transition will take at least five and maybe as much as ten years, and that's not counting Microsoft's sabotaging such efforts.

Some readers have taken this paper as an advocacy for Linux, MacOS or other products. While those would indeed be better alternatives from a technical point of view, one should not be blind to the fact that for most of us these are not viable alternatives. In order to make our living, we're expected --even forced-- to be compatible with the one brand of products that has been deliberately made incompatible with almost everything else.

Several readers have pointed out that recently OSX has become a more and more realistic alternative, now that Apple has begun to sell computers based on Intel processors. In fact OSX has been made to run succesfully on standard PCs. However these experiments involve illegal versions of OSX, which then have to be massaged and occasionally reverse-engineered (all in violation of the applicable licensing conditions) to make them work. But the point here is not that OSX can now be made to run on a PC, at least from a technical standpoint. The main thing is that Apple is not doing it. If there's anyone who can release such a product and make it popular, it's Apple. But no such product is apparently forthcoming.

At the same time, more and more readers of this paper are wondering what Google is up to. Notwithstanding widespread concerns that Google is getting more and more powerful and could become the next Evil Empire, the fact is that Google is gearing up for something major. They're building humungous datacenters, the locations of which are largely governed by how many Gigawatts of electricity are locally available. That should tell us a few things about Google's visions of the future. Google undeniably packs a punch, and could certainly release, say, a $40 operating system based on a Linux or BSD kernel, and get OEMs to sell it with name recognition. People would definitely buy it. However, again the point is that Google has not done that, and appears to have no plans whatsoever of venturing into that particular market. Google focuses on information providing, online advertising and online application services, which makes sense because that is what they're good at. They're not into operating systems and application software.

So it doesn't look like a change is in the winds anytime soon.

And that is why

Microsoft has stunted the true growth of computer technology under the guise of innovation. They have manipulated technology in order to force their customers to turn to Microsoft as a sole supplier. They have driven up the cost of computing in order to spike their own revenues, and behaved in ways that show contempt for their customers and the law. They have shown a tendency towards lying and general dishonesty. And with a towering display of hypocrisy they keep telling us that everything they have done has only been in response to public demand for innovation and to the needs of the IT professional.

That is why I hate Microsoft. Microsoft isn't the answer. Microsoft is the question. And the answer is "No".

Mr. Gates, Mr. Ballmer, I've upped my standards... so up yours.

Appendix A: A brief overview of Windows' most serious design flaws

The Windows architecture has many design flaws that no amount of patching or updating can ever remedy. Readers of this paper have indicated the need for a brief summary.

The following list is by no means comprehensive, but only summarizes the most serious structural shortcomings that cripple the Windows operating system.

Limited memory protection and memory management.

This problem exists primarily in versions prior to Windows 2000. When an application contains bugs or otherwise runs wild, it may write to memory locations outside its own memory space, thereby crashing the entire system. Attempts to allocate more memory than is available often generates an exception, causing the application to crash instead of allowing it to recover gracefully. Similarly, out-of-boundary reads are also possible, which potentially compromises security.

Insufficient process management.

The OS relies heavily upon the application to release allocated resources. If an application hogs resources or fails to release them for some reason, either while running or upon termination, the OS often cannot reclaim those resources. Nor does the OS offer an administrator full control over processes and resources.

No adequate separation between user-level and kernel-level code.

An application may install DLLs or drivers. This introduces essentially uncertified, third-party code to the system, that may run at kernel-level, i.e. completely unprotected. Applications may also introduce modifications into the registry without any protection or verification whatsoever, which may cause other applications or even the OS to crash. This seriously compromises the reliability of the entire system. In fact Windows is the only major operating system in the market that may break whenever a user installs an application (essentially a user-level operation). In Windows 2000 and later a simple driver signature system has finally been implemented, but in practice this is not sufficient to guarantee stability.

No adequate separation of different kernel-level code types.

Drivers, for example, should contain driver code. They should offer the OS an API to interface with the underlying hardware. In Windows however a video driver may also contain virtual desktop code and other nonsense. Not only does the presence of user interface code in a hardware driver illustrate the messy nature of the Windows code's organisation, but it also leads to ridiculous issues such as system tray icons disappearing due to a bug in the video card's driver. (Nvidia comes to mind.) In a well-structured OS this would never happen.

Lack of meaningful error messages.

Whenever an error message is displayed, it rarely tells you exactly what the problem is. Nor does it give you enough details (e.g. an error or condition number) that would enable a support technician to trace the cause of a problem.

No maintenance mode.

When one or more of the 10,000+ files that make up the OS become corrupted, there is no maintenance mode that allows you to bring up the OS in a controlled state, doing repairs along the way. 'Safe mode' merely swaps configurations but offers little additional control. The OS either runs or crashes. The Windows 2000/XP Repair Console (an external utility on the Windows CD that may be run from the setup routine) does not allow you to run the OS in a maintenance mode, it merely allows you to access the file system of a broken Windows installation. Even finding out which files have been corrupted is often impossible.

No code sharing.

Only DLL code can be shared, which makes up only a tiny fraction of the entire OS and application code.

No version control whatsoever on DLL code.

The OS cannot distinguish one DLL from another one with the same name, even though they may contain entirely different code. Installers can, but generally don't bother with it (beyond warning the user that version x is about to be replaced with version y). The OS however will happily load whatever code is present in the DLL file it happens to find first.

A very rudimentary and weak security model.

Microsoft products have the worst security rating (and track record) in the industry. Their developers seem to have been completely unaware of even basic security issues.

Rudimentary multi-user support.

Being the offspring of a stand-alone, single-user desktop OS, Windows can only be implemented in a LAN (or any other environment where users share computing facilities) by means of cumbersome workarounds and kludges. Multi-user applications (e.g. Citrix, Windows Terminal Server) are even more problematic.

OS code, application code and user data cannot be maintained separately from the OS and from each other.

The OS is actually designed to prevent this. Applications need to be "installed" i.e. integrated into the OS, a procedure that adds to or even overwrites part the OS fileset, may overwrite other application files or (registry) settings, and usually requires an OS reboot. The installation of one application may break another application.

Windows does not follow global protocol standards correctly.

It even deliberately ignores them in favor of proprietary implementations.

Windows' API is only partially documented.

Much of the operating system and the API remain essentially a black box to third-party developers. This causes problems during application development that often cannot be traced. Developers are forced to use workarounds, and may even be tempted to circumvent or ignore the API altogether. This, and the fact that nobody outside Microsoft really knows what goes on in the underlying code, leads to application software that won't run correctly (if at all) on updated or new versions of Windows.

Windows' code is a collection of bad programming practices.

It contains a huge amount of sloppy code and kludgy design, which results in an extremely glitchy and buggy end product. (A good example of sloppy programming is buffer access, which is routinely left unchecked in release versions of all Microsoft products. As a result of this amateur approach, Microsoft products are riddled with buffer overrun vulnerabilities. Of course buffer overruns don't only occur in Microsoft products but are also found in other software on other platforms. But the difference is in the numbers.) Bloated code has made Windows' efficiency the lowest in the market, requiring more resources and yielding less performance than any other OS in existence.

Appendix B: links

- **Microsoft crash gallery** An amusing (or depressing, depending on your point of view) collection of screenshots that illustrate the many ways in which MS products are prone to crash.
- **Windows NT and VMS: the rest of the story** Is NT really New Technology?
- **"Why Open Source Software / Free Software (OSS/FS)?"** This paper by David A. Wheeler provides quantitative data that, in many cases, using open source software / free software is a reasonable or even superior approach to using their proprietary competition according to various measures.
- Jerry Pournelle referred to this article on his **Chaos Manor** website, but disagreed with me about the early history of Microsoft and Windows. Here is a transcript of the discussion that followed. (This is in raw text format because I can't be bothered to convert it to HTML).
- **The Netcraft uptime top-50** lists webservers by average uptime. As I write this, Windows NT/2000 and/or IIS aren't even on the list!
- **Georgi Guninski** is a Bulgarian security consultant, who hunts security leaks for a hobby. Needless to say, Microsoft keeps him quite busy. Don't let his less-than-perfect English fool you; I believe this man holds the unofficial record for discovering MS security leaks. Read it and weep.
- **Windows NT vs. CP/M**
- **This** is how one user felt about Microsoft...
- **Billgatus of Borg** on the cover of Boardwatch magazine
- **A new NT 4.0 logo**
- **What users thought of Bill Gates**
- **Look here** if you thought this page was bad... :-)
- **Microslaves**
- **Windows 2000 buglist**
- **Microsoft Linux** - at last!
- **Microsoft acquires de facto monopolies in education** Catch 'em while they're young...
- **The best bluescreen ever** If the Blue Screen Of Death looked like this, I wouldn't mind seeing it so often! :-))
- **MSBC** The Microsoft Boycott Campaign, including the Superlist of anti-MS websites
- **This** The is probably the best way to install Windows
- **Computer rage** Tales of woe from the field
- **Remarks by Steve Ballmer.** Read a transcript of an interview with Steve Ballmer where he (rather weakly, IMO) tries to defend the new Software Assurance licencing scheme in response to critical questions from his customers.
- **This** is a must-read for anyone who believes that "Linux is too hard to use".
- **Wikipedia** even has an entry on the BSOD. Go figure.
- **10,000 bugs away from world domination** An ex MS developer hails Open Source
- **The Halloween Documents** that leaked out of Microsoft in 1998 clearly state Microsoft's own assessment on how Open Source software not only performs and scales much better than Microsoft Products, but also propose that Microsoft attack these superior products by "de-commoditizing" (i.e. sabotaging) protocols. Recommended reading.