QinetiQ

Analysis of the Impact of Open Source Software

QINETIQ/KI/SEB/CR010223

Cover + xii + 37 pages
October 2001
Dr Nic Peeling and Dr Julian Satchell

This page is intentionally blank

Authorisation

Prepared by	Dr Julian Satchell
Title	
Signature	
Date	
Location	
Authorised by	Dr Nic Peeling
Title	
Signature	
Date	

Record of changes

None

Executive summary

Background

Open Source Software (OSS) is software whose source code is openly published, which is usually available at no charge, and which is often developed by voluntary efforts. This paper examines how the OSS movement could impact the UK Government's policy towards the worldwide market in software infrastructure[1] and software applications[2].

OSS has leapt to prominence by starting to take a significant market share in some specific parts of the software infrastructure market. For example, since 1998 Linux has been one of the fastest growing server operating systems[3]. Indeed, in a few important market segments, such as storing Web Pages, OSS software is far and away the market leader[4].

The software industry is very fast moving, and frequently throws up promising new developments that initially promise to make great changes in the marketplace, but which ultimately fail to live up to their initial press hype. Our first key conclusion is that OSS is indeed the start of a fundamental change in the software infrastructure marketplace, and is not a hype bubble that will burst. This is perhaps surprising because OSS does at first sight appear to be a bit of a paradox.

Given that OSS software is often developed by largely volunteer efforts, how can software, such as the Linux operating system, compete with software such as Microsoft Windows, which has had, and will continue to have, billions of dollars invested in it? In particular, how is it that the best OSS is perceived by many to be at least as reliable as market leading proprietary software? The body of this paper explains how this has happened. OSS's credibility has been established as market giants such as HP, IBM and Sun have thrown their weight behind it.

---

[1] By software infrastructure, we mean the software that represents the plumbing of IT systems and the Internet. Software infrastructure includes operating systems, databases, Web servers, and all the other major components that enable software applications to run.

[2] By software applications we mean software directly run by an organisation's users such as word processors, spreadsheets, presentation tools, project management tools, financial systems, and the myriad of other applications that organisations use to run their businesses.

[3] See for example, http://idg.com/www/pr.nst/webPRForm?OpenForm&region=WW&unid=5BE05D2B7D29EB7788256A37006C6A3. Since 1998 IDC has had Linux and Windows NT as the two fastest growing server operating systems.

[4] The Netcraft survey (http://www.netcraft.com/survey/) shows that about 60% of web sites use the OSS Apache Web server.

Future Trends

Where will it all end? We now have to move into the area of predictions[5].

Within five years, 50% of the volume of the software infrastructure market could be taken by OSS[6]. We expect that OSS's position in the small server market (file and print servers and Web servers) will grow fastest.

OSS's position in large servers (e.g. those managing massive multi-user databases), such as those that underpin many large Government procurements, will grow from its current position of near zero penetration, to a position where OSS is a viable option, within 2 - 3 years.

Within the developed world, we as yet see no sign that OSS will become a viable alternative to Microsoft Windows, for users' (general purpose) desktop machines in the corporate or home PC markets[7]. However, OSS on the desktop may soon become a significant player on the desktop in the developing world. For these reasons the study recommends against any preference for OSS on the desktop, but also recommends that this issue be reassessed by the end of 2002, by which time early trials of the use of OSS desktops may have generated sufficient evidence to warrant a reassessment.

OSS is already suited to restricted functionality desktops, such as those used in industry for point-of-sales and point-of-service[8] terminals; and in these areas OSS's market share is likely to grow significantly.

We expect OSS to rapidly[9] become the market leader in consumer computing devices[10].

We expect the market for new portable and consumer computing devices (such as set-top boxes and smart mobile phones) to remain very dynamic, with no dominant market leader emerging. OSS is however likely to be a significant[11] player in this market.

We expect that the software infrastructure that is implemented on top of operating systems (so-called middleware) will move gradually[12] from proprietary products towards OSS.

---

[5] The Authors are members of a substantial team funded by the MOD who track market trends, and these are the consensus views of that team.

[6] There is considerable confusion about Linux market share statistics and projections - see for example http://www.osopinion.com/perl/story/11177.html for a discussion of current statistics of Linux server market share.

[7] The first real trials are starting now. For example, see Computing Aug 2 2001, pg 3 which reports that Central Scottish Police and an unnamed local council have adopted Sun's Open Source Star Office suite. A comparable initiative in the City of Largo, Florida, is reported at http://www.newsforge.com/article.pl?sid=01/08/10/1441239

[8] Such as those used for airline bookings, travel agents etc.

[9] Within the next year.

[10] We refer here to devices such as Web pads, set-top boxes and digital video recorders. We exclude hard real-time embedded computers and those driving a range of special-purpose hardware peripherals (industrial process control, washing machines etc.).

[11] Greater than 20% in the next 3 years.

[12] We anticipate OSS taking 25% of the middleware market in 2 years, and 50% in 5 years.

All of the above predictions relate to the software infrastructure market. As yet, it is not possible to predict that, within the developed world, OSS will make such a major contribution to the software applications market. There are a few OSS applications[13] that are becoming significant, but it is too early to say if a trend is developing.

Major impact of OSS on UK Government

There are a number of major issues that the UK Government needs to be aware of:

Any discussion of Open Source Software must include comparisons and contrasts with proprietary software. Microsoft is the world's leading software company, and their products now dominate the office and home markets. They also have a strong presence in the server market, especially for small and middle-sized systems. When we comment about Microsoft in this report we are seeking a comparison with the industry leader; no criticism of Microsoft as a company, or any of their products, is intended or implied.

This report argues that many of the Government's risks that arise from over-dependence on proprietary free protocols and data formats for interoperability can be controlled by the selective use of open data standards. There are many such standards within the Internet world, and the Government can develop its own standards for use within communities-of-interest (e.g. MOD, Health etc.). This report concludes that the existence of an OSS reference implementation of a data standard has often accelerated the adoption of such standards, and recommends that the Government consider selective sponsorship of OSS reference implementations.

The rise of OSS offers the possibility that non-US players will find it easier to influence the future direction of IT infrastructure technology. This may be a significant reason for the enthusiasm of the CEC (Commission of the European Communities) and some Member States for OSS, and explains the prominence that OSS plays as an exploitation route within the CEC's Framework 6 R&D programme[14]. There has been a long and successful history in the US of using OSS as an exploitation route for Government-funded software, and this report concludes that the Government should consider using OSS as the default exploitation route for UK Government-funded software.

There has been a heated debate about the benefits and risks that OSS poses to the vulnerability of a nation's IT infrastructure. This report concludes that the differences between OSS and proprietary software are not a major factor in either improving or degrading the vulnerability of a nation's IT infrastructure.

This report concludes that OSS has shown that access to software's source code is a major enabler of flexibility, and hence reduces legacy problems considerably. The report recommends that the Government obtain full rights to bespoke software that it procures – this includes any customisation of off-the-shelf software packages.

The Open Source model offers a new paradigm for funding software in communities-of-interest (e.g. Health and Education). There is some indication that such projects are developing in other countries and the Government could consider running pilot projects to test the viability of the OSS approach to such software. One particular project came to

[13] Such as the GIMP image manipulation application.
[14] ftp://ftp.cordis.lu/pub/ist/docs/b_wp_en_200101.pdf

light during the study which was the development of a Medical Records data standard[15], which we recommend is examined by appropriate domain experts for possible inclusion in the e-Government Interoperability Framework (e-GIF[16]).

[15] http://sourceforge.net/projects/medrec/
[16] http://www.govtalk.gov.uk/interoperability/egif.asp

Summary of conclusions and recommendations

With just one exception, all the conclusions and recommendations have already been discussed in this Executive Summary, and they are listed below. The exception is the recommendation that the authors of this report had the most difficulty framing. This recommendation relates to the issue of whether the Government should make any policy statements regarding the use of OSS in the server infrastructures of Government procurements.

We have already said that we think that the current immaturity of OSS on the desktop means that there is a clear reason to not express any preference for OSS on the desktop, at the current time. This reason does not apply to many parts of the server infrastructure where OSS is a technically viable choice. The authors could see no benefit that the Government would gain from expressing a general preference for OSS within server infrastructures. However, the authors thought there were two areas where lack of guidance from Government might prejudice bidders from offering OSS even if it was the most cost-effective option:

Recent press reports[17] might give bidders the impression that the Government has a preference for Microsoft solutions, and the Government could clarify its position as to whether there are circumstances in which Microsoft products are to be preferred;

Some proprietary products make the subsequent deployment of products from other suppliers difficult, and the Government could consider publishing policy as to how the risk of lock-in to proprietary protocols is to be managed.

The full list of conclusions and recommendations:

1. OSS is indeed the start of a fundamental change in the software infrastructure marketplace, and is not a hype bubble that will burst.

2. Within five years, 50% of the volume of the software infrastructure market could be taken by OSS.

3. OSS's position in large servers (e.g. those managing massive multi-user databases), such as those that underpin many large Government procurements, will grow from its current position of near zero penetration, to a position where OSS is a viable option, within 2 - 3 years.

4. Within the developed world, we as yet see no sign that OSS will become a viable alternative to Microsoft Windows, for user's (general purpose) desktop machines in the corporate or home PC markets. However, OSS may soon become a significant

[17] For example, http://www.theregister.co.uk/content/4/19239.html, http://www.linuxuser.co.uk/articles/issue11/gateway.html, http://www.guardian.co.uk/internetnews/story/0,7369,504403,00.html and Andrew Pinder's response at http://www.guardian.co.uk/internetnews/story/0,7369,508041,00.html

player on the desktop in the developing world. For these reasons we recommend against any preference for OSS on the desktop, but also recommend that this issue be reassessed by the end of 2002, by which time early trials of the use of OSS desktops may have generated sufficient evidence to warrant a reassessment.

5. We see no benefit that the Government would gain from expressing a general preference for OSS within server infrastructures.

6. The Government could clarify its position as to whether there are circumstances in which Microsoft products are to be preferred.

7. The Government could consider publishing policy as to how the risk of lock-in to proprietary protocols is to be managed.

8. As yet it is not possible to predict that OSS will make a major contribution to the software applications market.

9. Many of the Government's risks that arise from over-dependence on proprietary protocols and data formats for interoperability can be controlled by the selective use of open data standards.

10. The existence of an OSS reference implementation of a data standard has often accelerated the adoption of such standards, and we recommend that the Government consider selective sponsorship of OSS reference implementations.

11. The rise of OSS offers the possibility that non-US players will find it easier to influence the future direction of IT infrastructure technology.

12. The Government should consider using OSS as the default exploitation route for UK Government funded software.

13. The differences between OSS and proprietary software are not a major factor in either improving or degrading the vulnerability of a nation's IT infrastructure.

14. We recommend that the Government obtain full rights to bespoke software that it procures – this includes any customisation of off-the-shelf software packages.

15. The Open Source model offers a new paradigm for funding software in communities-of-interest (e.g. Health and Education). The Government could consider running pilot projects to test the viability of the OSS approach to such software.

16. We recommend that the Medical Records data standard be examined by appropriate domain experts for possible inclusion in the e-GIF.

List of contents

This page is intentionally blank

1 The Open Source Phenomenon

1.1 Introduction

1.1.1 Since the second half of 1999 Open Source Software in general, and the Linux operating system in particular, has seldom been out of the press. In November 1999 a Microsoft paper (the "Halloween memo") on Open Source was leaked which showed that Microsoft viewed the Open Source phenomenon as a major threat to their business. At the time that Open Source started hitting the headlines in 1999, it was already being used to run a significant part of the infrastructure of the Internet. Since then there has been a sustained growth in the role of Open Source software in the IT industry:

- Almost all of the major enterprise software vendors sell versions of their software on Linux;
- Almost all of the major computer vendors sell their computers with Linux pre-installed;
- Linux is a strategic operating system for many of the large computer vendors (e.g. IBM, HP and SGI);
- Linux is the fastest growing server operating system[18] ;
- Linux is taking a major part of the operating system market for consumer computing devices;
- The OSS Apache is the dominant Web server product;[19]
- A number of Open Source projects are starting to have a major impact on the market – e.g. the GIMP (image manipulation), SAMBA (Microsoft file and print services), many program development tools (Perl, Python, emacs etc.), and Postgres and MySQL (databases);
- New, potentially significant, Open Source projects are springing up everywhere.

1.1.2 There is also a growing interest amongst Governments about the significance of Open Source. In the US, the NSA (National Security Agency) has funded Secure Computing inc. to develop a secure variant of Linux[20]. Open Source is playing an increasingly prominent role within the European Commission and EU Member States. The 2001 Work Programme for the Framework 6 IST (Information Society Technologies) Programme has as one of its nine priorities "to foster the development and use of Open Source Software". The Information Society for All eEurope Action Plan includes a specific action on the European Commission and Member States to "promote the use of Open Source Software in the public sector and e-government through exchange of experiences across the Union". There are persistent press reports[21] that France, Germany and Denmark are moving towards a preference for Open Source software

[18] See for example, http://idg.com/www/pr.nst/webPRForm?OpenForm&region=WW&unid=5BE05D2B7D29EB7788256A37006C6A3

[19] The Netcraft survey (http://www.netcraft.com/survey/) shows that about 60% of web sites use Apache.

[20] http://www.nsa.gov/selinux/

[21] http://www.sans.org/infosecFAQ/code/open_source.htm reports French legislative proposals, see also text of speech by German Secretary of State in the Federal Ministry for Economy and technology reported at http://ww.internetnews.com/intl-news/article/0,,6_408271,00.html

within Government funded R&D and/or Government procurements. It is our opinion that the European Commission and some Member States are promoting Open Source as a means of reducing lock-in to dominant proprietary (US) suppliers, and as a new, credible means of increasing European influence on the IT marketplace.

1.1.3 In many ways the Open Source phenomenon is counter-intuitive. How can software that is often written by unpaid volunteers, in their spare time, possibly destabilise the market for software applications written by multi-billion dollar corporations like Oracle, IBM or Microsoft? This section looks at the history of Open Source software, and examines the properties that Open Source software exhibits, which help to explain its rapid spread.

1.2 The History of the Free Software / Open Source Community

1.2.1 UNIX grew to prominence in the early 1970s after its originators AT&T licensed the source code on very favourable terms to US Universities. This tapped into a substantial vein of creativity at centres such as the University of California at Berkeley, turning UNIX into a software development environment that was more portable, and much cheaper and easier to write software for, than the proprietary mainframe and mini-computer operating systems of the day.

1.2.2 UNIX was the first significant manifestation of the hacker culture of technology enthusiasts that was centred in Universities and Research Institutes (such as the Massachusetts Institute of Technology (MIT) and CERN). Note that in this community, the term “hacker” is not a term of abuse but is used to describe a skillful programmer, and this is what we mean by the term “hacker” in this study. UNIX was the first indication that this community could produce software as (or more) robust and “professional” than purely commercial products.

1.2.3 The Open Source community's nervous system has always been the Internet, indeed it was the this community that laid the foundations for the Internet. It was the Internet that allowed such a strong sense of community to develop amongst Open Source software developers. A consequence of this is that the Open Source community is truly international with only a minority of the most active OSS developers being American.

1.2.4 There was an early split in the community with one strand being the Free Software Foundation (FSF), headed up by Richard Stallman. Stallman worked in the 1970s at the MIT Artificial Intelligence Laboratory and in 1990 he received a $240,000 MacArthur “genius” grant which allowed him to devote all his time to the FSF. The FSF believe that denying software users access to the source code of software programs is morally wrong[22] and they have developed a free but restrictive copyright license, called "copyleft" or GPL[23] - The GNU General Public License (where GNU is recursively defined as GNU is Not UNIX). The GPL states that any GPL software can be used or amended without any payment to the copyright owners, but that any software that incorporates GPL software must be published under the GPL license. The GPL allows distributors of GPL software to make a charge to cover their distribution costs and also allows companies to sell services to support free GPL software.

[22] The FSF’s supporters often use the phrase “free as in *free speech*, not free as in *free beer*”.
[23] GNU General Purpose License - http://www.linux.org/info/gnu.html

1.2.5 The other strand of the Open Source software community uses variants of Berkeley's copyright license which is similar but less restrictive than GPL in that it does not mandate the publication of the source code of commercial software products that include Berkeley copyrighted source code.

1.2.6 The Open Source community is still actively debating the GPL vs. Berkeley license issues. There is a fuller discussion of OSS licenses in Appendix B. The widespread use of the term “Open Source” instead of “free software” is a visible sign that the mainstream of the Open Source community is trying to distance itself from the FSF camp. The upsurge in pragmatism has been greatly assisted by the moderate personality of Linus Torvalds (the originator of the Linux project) who has positively encouraged the commercial distribution and usage of Linux[24]. There is even a voluntary marketing effort that is attempting to sell the Open Source model to the commercial world (see the Open Source web page[25] for further details).

1.2.7 The FSF's most stunning successes were to produce the most widely used portable editor, emacs, and the dominant portable compilers for C and C++ (called gcc and g++), plus a host of lesser known, but widely used, software development tools. During the 1980s, UNIX vendors vied with each other to publish the fastest compiler figures (measured using the industry standard benchmarks called the SPEC tests - SPEC stands for System Performance Evaluation Corporation). Gcc effortlessly stayed up with, or ahead of, commercial C compilers.

1.2.8 Even in the 1980s there was a strong strain of pragmatist Open Source developers as typified by Bob Scheifler, from MIT, who could with some justification claim to be one of the fathers of Client/Server computing. He was one of the inventors of the X Window System and formed a commercially supported consortium, the MIT X Consortium (now part of The Open Group), to maintain a free, but commercialisable, source code for X. X's ability to allow the Graphical User Interface of an application to run on a different computer from the application was one of the main features that propelled UNIX to the front of enterprise computing in the 1980s.

1.2.9 AT&T was broken up in 1984 by anti-trust decree, and this allowed UNIX to become a mainstream commercial operating system. However in becoming a commercial product, UNIX cut itself off from its roots in the Open Source community, and consequently the rate of innovation in UNIX dropped and the number of subtly incompatible commercial UNIX variants increased. These factors, amongst others[26], allowed a proprietary competitor (Microsoft), controlling a single de-facto standard (Windows), to easily take market share from UNIX.

1.2.10 The Open Source community has attempted to try to regain some control over UNIX. In 1994 the team at U-Cal Berkeley won a two year legal fight to distribute a UNIX free of AT&T source code (BSD4.4 - from which has spawned FreeBSD, OpenBSD and NetBSD) distributed under the Berkeley copyright. For a considerable length of time the FSF have been working on a free, post-UNIX operating system kernel called Hurd. The most successful Open Source operating systems project is Linux.

---

[24] This has been possible even though Linux is distributed under the GPL. The use of the GPL is seen by many Linux developers to be essential in order to prevent the proliferation of numerous incompatible variants of Linux.

[25] Open Source web page - http://www.opensource.org/

[26] Windows commitment to the PC platform as opposed to proprietary UNIX workstations, was probably the single most important factor.

1.2.11 Since its inception, many key parts of the Internet infrastructure have depended on Open Source products - for example BIND (which implements the Domain Name Server (DNS) which translates host names into network addresses), INN which runs the Usenet Newsgroups (often referred to as Internet Newsgroups) and *sendmail* (used for email delivery).

1.2.12 In more recent times the most spectacular development from the Open Source community is Tim Berners Lee's invention of the Web, whilst working at CERN in 1989. Even today, the leading web server is the Open Source Apache product.

1.2.13 There have been a number of other Open Source success stories which, for want of space, we only mention a few of in passing - *TeX* and *LaTeX* (publishing tools), *Perl* and *Tcl* (scripting languages), *Python* (advanced object-oriented programming language), *SAMBA* which provides file and print services using Windows protocols, *The GIMP* (GNU Image Manipulation Program), and many more.

1.2.14 An indication of the perceived power of Open Source software is that in 1999 Netscape made their browser available as Open Source.

1.2.15 How is it that a large, seemingly disparate, group of volunteer developers can produce such robust, powerful software - especially as the accepted wisdom in commercial software development is that adding very large numbers of programmers to a software project seldom helps, and usually dooms it to failure? Eric Raymond, a well-known advocate for Open Source software, who maintains the Open Source marketing web pages, has authored a number of articles that go a long way to explaining this apparent paradox. His article "The Cathedral and the Bazaar" [27] describes the dynamics by which large communities of Open Source developers produce robust programs; and his article "Homesteading the Noosphere" [28] gives a plausible anthropological model for the Open Source movement.

1.3 A Brief History of Linux

1.3.1 In 1991 Linus Torvalds, then a student at the University of Helsinki, issued a very early release of a UNIX-like kernel. His kernel was strongly influenced by the Minix kernel produced by Andy Tanenbaum (professor of computer science at the Vrije Universiteit in Amsterdam). The kernel contained support for multiple processes, a file system and a few device drivers. By March 1994 Linux was released at version 1 (previous releases had been pre-release version 0's) and the user community had started to grow substantially. Today the kernel is at version 2.4 and it is a stable, high performance, 64-bit kernel with support for symmetric multiprocessing and state of the art networking capabilities. One notable feature of Linux is its modest (by comparison with other UNIX and Microsoft operating systems) hardware requirements - which allows it to run on older hardware configurations.

[27] *The Cathedral and the Bazaar* by Eric Raymond - http://www.redhat.com/redhat/cathedral-bazaar/cathedral-bazaar.html

[28] *Homesteading the Noosphere* by Eric Raymond - http://sagan.earthspace.net/~esr/writings/homesteading/index.html

1.3.2 The user base of Linux is estimated to have grown as follows: - 1993 - 100,000, 1994 - 500,000, 1995 - 1,500,000, 1996 - 3,500,000, to a current figure well in excess of 15 million[29]. In particular we are starting to see significant growth coming from Eastern Europe, the former USSR, China, the Indian subcontinent, the Pacific Rim and the poorer third world countries. Linux's growth outside of the highly developed countries will be greatly aided by its ability to run on older PC configurations.

1.3.3 On top of the Linux kernel are mounted the Free Software Foundation's (FSF's) program development tools, the FSF's basic UNIX tools, plus a raft of other software from other Open Source projects. A Linux operating system with a complete set of the tools on top of the Linux kernel is many millions of lines of source code. To download a Linux operating system from the Internet is a time-consuming business, and as a result a number of commercial companies (of which Red Hat, Caldera, SuSE, Mandrake and TurboLinux are the best known) have sprung up to sell *Linux Distributions* on CD - containing both source and binaries. There are Distributions for virtually every computer made. Many of the distributors sell a variety of levels of support with their distributions.

1.3.4 What is Linux good for?

- Linux, with Apache (an Open Source web server), has a strong presence amongst ISPs (Internet Service Providers), and it has a growing presence as a server operating system for Intranets;
- Linux, and the other Open Source UNIXs (FreeBSD / OpenBSD / NetBSD), are heavily used for Internet firewalls;
- It is finding favour as a print and file server for UNIX and Windows networks;
- Open Source developers are using it as their Internet access device and also a program development environment;
- In the server market, Linux initially made a larger penetration amongst small and medium sized companies than multi-nationals - although this is changing as most of the main enterprise software applications are now available on Linux, and major system vendors such as IBM are pushing Linux hard to their customers;
- At the moment it has almost zero penetration on the corporate desktop;
- Linux is one of the leading, if not the leading, operating systems for consumer computing devices;
- Some technically challenging areas, such as the movie industry, are adopting Linux for their specific needs;
- Linux is also one of the major players in supercomputer clusters of computers.

[29] Estimating the usage of a free system is difficult, as copying is legal, even encouraged. A good discussion is found at http://www2.linuxjournal.com/enterprise/linuxmarket.html

1.4 The Open Source Software Market Model

1.4.1 In this section we explore the advantages and disadvantages of the Open Source model, both to developers and users of Open Source software. This section goes a long way to explaining why Open Source has been so successful. Starting with the advantages to software developers:

- Open Source software has a unique advantage in "crossing the chasm" [30]. The reason that products drop into the chasm (i.e. they fail to establish a sustainable market share) is that companies choose to cut their losses rather than keeping on funding the product in the hope that it achieves acceptance in a niche or mainstream market. Proprietary developers solve this problem by having the resources and management commitment to continue pushing products that they believe in (e.g. Microsoft with Windows and Windows NT) for as long as it takes for them to take off. Open Source solves this by having a zero cost base - so running out of money is not a problem - as long as the group of developers maintain their interest they can keep on going;
- A second major issue in crossing the chasm is the need to offer customers a total solution to their IT problems, not just a component of the solution which the customer has to integrate themselves. Typically, a company trying to get a commercial product across the chasm needs to recruit a significant number of service providers who perform the necessary integration for the customers. A useful analogy here would be that the manufacturers of central heating boilers do not, in general, sell directly to customers - customers buy the boilers recommended by the heating engineers who are installing or upgrading their central heating systems. Service providers will like the fact that Open Source software, such as Linux, is free (helping their margins); that it can be tailored to the requirements of the installation; that it is very reliable; and that if it goes wrong they are not dependent on the manufacturer to fix it;
- The ability of users to deploy the software without having to sign licenses, or make financial cases to their management, aids initial take up;
- Open Source developers have access to the existing body of Open Source software to include in their programs (the Open Source community have been one of the first communities to be able to exploit the potential offered by widespread software reuse);
- The Open Source community attracts very bright, very motivated developers, who although frequently unpaid, are often very disciplined. In addition, these developers are not part of corporate cultures where the best route to large salaries is to move into management, hence some Open Source developers are amongst the most experienced in the industry. In addition all users of Open Source “products” have access to the source code and debugging tools, and hence often suggest both bug fixes and enhancements as actual changes to the source code. Consequently the quality of software produced by the Open Source community sometimes exceeds that produced by purely commercial organisations;
- The size of the Open Source developer community is very large. We would estimate that there are many tens of thousands of active Open Source software developers; hundreds of thousands of active beta testers; and a non-commercial user base of about 5 million Open Source supporters (and many more millions of

[30] *Inside the Tornado* by Geoffrey A. Moore, HarperCollins 1995.

users). As we have said earlier the size of the development community is growing all the time. A successful Open Source project can attract (and use effectively) a size and quality of developer and testing community that no company (even giants like Microsoft or IBM) can hope to match. An example of this effect was that when Oracle announced in July 1998 that they were going to port their database to Linux, they had 20,000 developers sign up to Oracle's development programme - an unprecedented amount of interest for a future port of the Oracle RDBMS;

- Open Source developers are not constrained by corporate product development processes or ISO 9001-style software development and Quality Assurance processes. Although some Open Source software is unreliable, many of the most popular OSS products have a rate of evolution, robustness and responsiveness to bug reports that much commercial software can only dream of;
- Open Source is an obvious subject for projects in Universities and research institutes. There is growing interest amongst Governments in using Open Source as a mechanism for exploiting research results. The Research Community gives Open Source developers free access to a large community of the brightest and freshest minds. This provides a major (free) source of analysis and incremental enhancements to Open Source developers;
- Many students get involved with Open Source software at university and when they go into industry and obtain positions of power they have a natural tendency to favour the software they worked on in their student days. This effect is widely credited with being one of the reasons why UNIX first made an impact in the enterprise market. It would also help explain the current surge of commercial interest in Linux.

1.4.2 The disadvantages to software developers who produce software within the Open Source model are:

- Commercial companies have to find new places to make money - e.g. selling services or books. For some markets (e.g. games) it is difficult to see where commercial companies can "make a buck". At the current time investors do not really understand the Open Source model. The possible relationships between the Open Source culture (motivated by individuals desire for recognition) and commercial models (where there will be a profit motive) are still developing and are consequently unpredictable;
- There is no marketing budget to push the product;
- There is no funded product development budget. In practice, this means that Open Source software "products" tend to get ease-of-use features and user-oriented documentation significantly later in their lifecycle than commercial products. For example, early Linux distributions attracted justifiable criticism for the complexity and lack of standardisation of their installation and management processes;
- Open Source developers tend to be very passionate about technical issues. Consequently, without a project leader with good people skills an Open Source project can break up in acrimony.

1.4.3 The advantages to users of Open Source software are:

- Although, in reality, software licensing costs are not a major part of the Total Cost of Ownership (TCO) of IT, the attraction of the software being free should not be underestimated. A corporate license for even cheap software can, in absolute (not relative) terms be serious money, and niche software can be dauntingly expensive. In addition, users are not faced with user-unfriendly license managers that can lock out users when more than the licensed number of users try to use a

product simultaneously. Users do not face costly management of licenses – including: the legal costs and risks of checking and signing licenses, ensuring that license conditions are adhered to, and ensuring that all relevant licenses have been purchased and are up to date. Users are also not locked into having to buy future upgrades;

- The market greatly values robustness, and the Open Source model, particularly as practiced by Linux, encourages a large market of early adopters (compared to the size of the early market for commercial products) who actively help debug the software. Consequently much Open Source software becomes highly robust at a surprisingly early stage of its development, and mature Open Source products are setting new industry standards for bulletproofness;
- Paradoxically, the evolution of Open Source software can often be much more responsive to user requirements than commercial software. The upgrade cycle for Open Source is usually much faster than the typical 12 - 18 month cycle of commercial products. In addition Open Source products such as Linux often provide the most rapid turnaround on urgent issues, such as patches against newly found vulnerabilities to external attack;
- Open Source software tends to be written portably and hence is available on a wide range of platforms. In addition, because the source code is openly available, people interested in availability on another platform can do the port themselves or pay someone else to do it. As a result, Open Source allows a wider choice of computing platforms and potentially easier upgrade to new technology;
- Open Source allows the user (and their service providers) to control vulnerabilities themselves.
- Open Source software tends to be free of dependency on related products. Purchasers often perceive that the product works best with other products from the same manufacturer. Open Source software offers its users greater freedom to purchase other products, avoiding lock-in to particular manufacturers;
- Open Source software means that there is no single proprietary source of software support and upgrades. This has a double advantage - firstly, there is no risk that the one company that supports the software stops supporting it or goes out of business; secondly, there can be a competitive market in companies offering support services (reducing cost and increasing quality of service). Users also have access to an Internet community, which includes both the developers and users of the software, so that in-depth advice (and possibly source code fixes) can often be obtained rapidly, and at no charge. An additional advantage is that often there are many people available for recruiting who are capable of supporting the software. In addition the source code is available to system integrators making system integration much easier and cheaper than having to rely on the originator to make cosmetic or interfacing changes;
- Companies developing in-house applications on Linux, and service providers who use Linux, will like the fact that if they hit a bug in Linux there is a possibility that they can go into the source code and fix it. Encountering a bug in a proprietary operating system can stop a project or service supplier dead in their tracks;
- Commercial software may become unmaintainable once its originators leave the company. By comparison, Open Source software often maintains a vibrant life for much longer as it is, in effect, the property of a community. This property is enhanced by the fact that Open Source software is often better structured and with better program documentation than commercial software - after all, everyone can see an Open Source developer's code so personal pride (and the need to maintain

the respect of one's peers) usually ensures it looks pretty. Indeed large, geographically dispersed teams can only work well if the software design is highly modular.

1.4.4 The disadvantages to users of Open Source software are:

- There is no single organisation with a vested interest in supporting it;
- As mentioned above, ease-of-use features tend to arrive later than for commercial products;
- There are lots of negative perceptions that Open Source still has to overcome (note: in our opinion most of these perceptions have little actual substance, but it will require lots of additional publicity about deployment of Open Source in large, respected companies to overcome them). These perceptions include:
    - Senior managers in companies are likely to equate "free" with "unreliable";
    - There is no commercial organisation who you can sue if something goes wrong;
    - Because I do not pay the software developers I have no control over them;
    - Because the developers are motivated by recognition rather than money, they are unpredictable, for example they might all rush off and work on a new, more exciting Open Source project;
    - Open Source developers will not understand commercial imperatives like backwards compatibility, and the need for interoperability.

2 Detailed Analysis of the Impact of OSS

2.1 Future Trends

2.1.1 To place this analysis in context we will give some more detailed predictions about the market uptake of Open Source Software (OSS).

2.1.2 Within five years, 50% of the volume of the software infrastructure market could be taken by OSS. We expect that OSS's position in the small server market (file and print servers and Web servers) will grow fastest.

2.1.3 OSS's position in large servers (e.g. those managing massive multi-user databases), such as those that underpin many large Government procurements, will grow from its current position of near zero penetration, to a position where OSS is a viable option, within 2 - 3 years.

2.1.4 Within the developed world, Open Source solutions are unlikely to have a major impact on the dominance of Windows applications for use on general-purpose desktop computers within the foreseeable future. In this context, the foreseeable future is until the end of 2002. There are two main factors responsible for the *de facto* monopoly of Microsoft Office. None of the competing suites (proprietary or Open Source) are completely compatible with MS Office; indeed very few can handle fast-saved files at all. The user interfaces are subtly different, and so training and support costs favour a single vendor solution. Microsoft Office is the key reason why Windows is unlikely to be displaced from corporate desktops in the foreseeable future. It is likely that for the next two years the duality of Windows applications (particularly Microsoft Office) and Windows operating systems on client / desktop PCs is likely to be part of any mainstream market solution to Government- procured IT systems that are deployed into office environments. Any attempt to ignore this market reality would be potentially risky to the UK Government, and consequently we would recommend against any immediate preference in Government procurements for use of OSS on the desktop. Appendix A contains more details about the status of Open Source desktop software.

2.1.5 There are reports[31] of Open Source desktops being used in organisations like Police Authorities and councils. A closed community may have a reduced need to interoperate with the standard Microsoft applications, and the number of desks offers potential savings in up-front costs. Early adopters are also likely to be organisations that possess the in-house expertise to manage what is not yet a mainstream IT configuration. The experiences of these early adopters will form important evidence to see if there is a Total Cost of Ownership (TCO) advantage in using an Open Source desktop in these scenarios. We recommend that the status of Open Source desktops should be re-considered at the end of 2002, as the software will be significantly more mature, and there should be results from these early adopters.

[31] See Computing Aug 2 2001, pg 3 which reports that Central Scottish Police and an unnamed local council have adopted Sun's Open Source Star Office suite. A comparable American initiative in the City of Largo, Florida, is reported at http://www.newsforge.com/article.pl?sid=01/08/10/1441239

2.1.6 The analysis in 2.1.4 is only true in the developed world. In the developing world the cost of software licenses is much more significant, and the existing investments in training much less significant. Consequently we expect OSS on the desktop to start being a major player in the developing world within the next year.

2.1.7 Some desktop machines only require very restricted functionality, typically when used as part of a tightly integrated, server-based system. This is the domain of the so-called "green-screen" systems. Open Source systems can deliver this type of functionality now, and are in limited use today for applications like point-of-sale systems in the retail sector, and property management. Web-delivered systems need no more than a fully functional browser on the client, and there are several modern browsers available for Open Source systems. We predict that the functionality available on Open Source desktops will increase steadily, and an increasing fraction of client functionality will become available.

2.1.8 We expect OSS to rapidly[32] become the market leader in consumer computing devices[33].

2.1.9 We expect the market for new portable and consumer computing devices (such as set-top boxes and smart mobile phones) to remain very dynamic, with no dominant market leader emerging. OSS is likely to be a significant[34] player in this market.

2.1.10 We expect that the software infrastructure that is implemented on top of operating systems (so-called middleware) will move gradually[35] from proprietary products towards OSS.

2.1.11 In looking at the role that OSS might play in future Government IT policy it is difficult to separate the role OSS might play from other aspects of policy that might be adopted. We have found that the best way to structure the analysis in the remainder of this section is to start from the viewpoint of the benefits that the Government might seek from policy initiatives that might involve OSS.

2.1.12 The next five sections all examine different aspects that relate to the through-life costs of IT holdings:

2.2 Managing COTS obsolescence:

2.2.1 It is too early to tell whether OSS has a markedly different lifetime to COTS software. Early indications would seem to suggest that OSS may, in general, have a significantly longer useful life than proprietary COTS. There are a number of effects that can already be seen:

- If a company decides to drop support for a COTS product, that is usually the end of it. If a major supporter of OSS drops out then the license terms allow other

[32] Within the next year.
[33] We refer here to devices such as Web pads, set-top boxes and digital video recorders. We exclude hard real-time embedded computers and those driving a range of special-purpose hardware peripherals (industrial process control, washing machines etc.).
[34] Greater than 20% in the next 3 years.
[35] We anticipate OSS taking 25% of the middleware market in 2 years, and 50% in 5 years.

groups to take the software up. This happened recently to the Python programming language;

- Once the initial developers of a piece of proprietary COTS software leave the company that sells it, the company can often struggle to keep updating the product. OSS belongs to a community that (generally) seems better able to evolve with time;
- OSS is usually based on open data standards, which often have a longer useful lifetime than proprietary protocols;
- OSS is often better structured and documented than proprietary products (because most OSS projects involve geographically distributed development by large teams, and this is only possible with good design and internal documentation. In addition, the programmers are motivated by the esteem of their peers).

2.2.2 If it becomes clear that, in general, OSS has a longer lifetime than proprietary COTS, the Government may wish to favour OSS *if there is an OSS software package that is a viable competitor to the dominant COTS products*.

2.3 Data Dungeons

2.3.1 An issue that is somewhat related to COTS obsolescence is managing data dungeons. We use the phrase data dungeon to refer to data that is stored in a proprietary format within a system that cannot be read by other systems, or a replacement system. This is a key issue where the data held within a system is of considerable value. There would be benefit in the Government favouring software that stores its data in open (non proprietary) formats. If there is OSS that manipulates the open data format (which there often is), then it will help in manipulating the Government's data held in that format. A related issue is that the long term archival of data in proprietary formats is unlikely to be appropriate.

2.4 The Monogamy Vortex

2.4.1 By monogamy vortex, we mean that COTS suppliers use proprietary protocols to integrate their different products together. For example, functionality such as single sign-on and authentication of browsers to Web servers, may well depend on such protocols. Consequently, it is difficult for other product vendors to produce products that integrate as well as the equivalent products sold by the dominant supplier. Organisations can find each purchase from the dominant supplier makes it more likely that subsequent purchases will tend to favour the dominant supplier, resulting in the organisation being sucked into a monogamous relationship with the dominant supplier. At the current time there is a significant issue that Government IT infrastructure must face as to the extent to which they allow themselves to be sucked into a monogamous relationship with Microsoft.

2.4.2 There are issues relating to the use of OSS to control the monogamy vortex that are worth discussing. We have said earlier that we think that the current immaturity of OSS on the desktop means that there is a clear reason to not express any preference for OSS on the desktop, at the current time. This reason does not apply to many parts of the server infrastructure where OSS is a technically viable choice. *The authors could*

see no benefit that the Government would gain from expressing a general preference for OSS within server infrastructures. However, the authors thought there were two areas where lack of guidance from Government might prejudice bidders from offering OSS even if it was the most cost-effective option:

2.4.3 Firstly, recent press reports[36] might give bidders the impression that the Government has a preference for Microsoft solutions, and the Government could clarify its position as to whether there are circumstances in which Microsoft products are to be preferred;

2.4.4 Secondly, some proprietary products make it difficult to deploy products from other suppliers, and the Government could consider publishing policy as to how the risk of lock-in to proprietary protocols is to be managed.

2.5 Licensing Costs

2.5.1 The received wisdom is that licensing costs are a small fraction of TCO (Total Cost of Ownership). If one includes upgrade license costs this may be less true. Some suppliers (for example Microsoft and Oracle) have announced an intention to move to a software rental model, with continuous charges for use. In some cases, for example large RDBMS systems, license fees are substantial, and may become comparable to or even exceed hardware costs. OSS does not yet have direct equivalents to the leading RDBMS products, but the feature gaps are closing. We predict that OSS will exert downward pressure on prices of infrastructure components. OSS obviously saves licensing costs, and will probably offer a wider (and hence cheaper) range of suppliers for support.

2.6 Gaining control of Government-procured bespoke software

2.6.1 The Government is one of the largest purchasers of bespoke IT systems in the country. Several of these procurements have been embarrassing and public failures, prompting the recent CITU review[37], and the reforms that it recommends. This review makes many relevant points, but one of these is particularly germane:

2.6.2 “Recommendation 20: Departments and agencies must ensure that they put in place processes that will actively encourage co-operation and an open dialogue between supplier and client. Projects already under way should immediately re-examine their communication mechanisms to ensure appropriate processes are in place."

2.6.3 We wish to suggest that Open Source may be one of the mechanisms by which this can be achieved, although it has not yet been tried in the UK to our knowledge. The process

[36] For example, http://www.theregister.co.uk/content/4/19239.html, http://www.linuxuser.co.uk/articles/issue11/gateway.html, http://www.guardian.co.uk/internetnews/story/0,7369,504403,00.html and Andrew Pinder's response at http://www.guardian.co.uk/internetnews/story/0,7369,508041,00.html

[37] http://www.citu.gov.uk/itprojectsreview/index.htm

might work by invitations to tender specifying that the Government required full rights to any bespoke code that was written for the project, and that the Government intends release the code as Open Source. This would open the contractor's work up for general inspection, and greatly ease a change of prime contractor if the project did not go well. We expect the effect would be to make costs transparent, which might raise initial costs, but reduce maintenance and upgrade costs, as the initial supplier has a greatly reduced advantage when bidding for follow-on contracts. A process of this sort has been used by the European Environment Agency (EEA), which mandates Open Source for all software developments. We quote conditions 9.6 through 9.8 from their general terms and conditions:

"(6) Notwithstanding what is said in article 1, results of software developments will be freely available as open source products as specified in Mozilla Public License version 1.1[38]. Each source file must have a header that it has been prepared under Mozilla Public License.

(7) Distribution of the results shall be managed through a publicly available code repository as required by the EEA. Deliveries shall be synchronised with regular reporting activities of the contract.

(8) When making information technology developments, the Contractor will adhere to the standards stated in the document "Software Standards of EEA and EIONET". The version of that live document that was effective at the date of contract or specific agreement signature prevails. Possible exceptions from the standards are handled through the provisions stated in the said document."

2.6.4 This type of approach will not be suitable in all cases, but we believe it should be considered for systems that consist of multiple components that can be procured independently. The EEA have used it for their Web Portal, as well as office automation. A portal wraps up information from multiple components, and is very suited to piece-wise procurement. The use of Open Source may allow the benefits of both competition and co-operation to be realised.

2.6.5 Even if full Open Sourcing of Government-procured software is not appropriate, we believe that there would be significant benefits if the Government required that full source code rights be vested in the Crown. At the current state of software technology, source code modification is the only credible technique that can deliver system flexibility and software reuse. As a means of controlling legacy problems we see no other credible strategy. It is a simple mechanism for avoiding supplier lock-in by allowing the Government to have the software it has procured to be modified by companies other than the one that produced it. The authors have often heard it argued that companies will not bid under such conditions. Frankly, we do not believe this argument and recommend that the Government adopt the policy of demanding full source code rights to Government-procured bespoke software as a matter of course. We would also recommend that the Government demand full rights in all customisations of COTS packages.

[38] http://www.mozilla.org/MPL

2.7 Interoperability

2.7.1 Many current systems are stand-alone monoliths, but the vision of "joined up Government" will make this rarer in the future. Large sections of the Governmental IT infrastructure will become interconnected and interdependent in a vast "system of systems". This raises many questions about interoperability, which is at least partly the domain of the e-GIF. One solution to interoperability between components has been to procure the complete system from one prime contractor, and to make interoperability the responsibility of that contractor. This approach cannot scale to cover the whole of government, and instead the e-GIF mandates a range of data interchange standards.

2.7.2 We would advocate that these standards should be openly defined standards as far as practical (as is currently the case for the e-GIF), rather than proprietary ones. This is a weaker recommendation than mandating Open Source, as proprietary implementations of the standards would conform. In many cases (e.g. jpeg or png image formats), an Open Source reference implementation is available, which has helped the dissemination and uptake of the standard. The use of proprietary *standards* (for example, the Microsoft Office file formats) locks in dependency on proprietary implementations, and would limit the choices in all connected components. If the interchange standards between components are open, it is much easier to replace components, allowing stronger competition as components and the business processes they serve advance. The components may be very large, in this vision something as large as the whole of the tax system might be a component.

2.7.3 There is a close linkage between the uptake of OSS and the role that Open Standards have played in the evolution of the Internet. One of the reasons that standard protocols promoted by the International Standards Organisation (ISO) in their OSI (Open Standards Interconnect) 7-layer model failed, and the TCP/IP based protocols of the Internet succeeded, was the availability of reference implementations of the Internet protocols which were made available as OSS. It is very difficult to write a paper specification of a protocol that is detailed and precise enough to make different implementations interoperate, whereas a reference implementation sets a *de-facto* standard that resolves most detailed implementation issues. A reference implementation also allows interested parties to perform early trials with new protocols, which assists with early adoption.

2.7.4 As mentioned earlier, the current version of the e-GIF takes an open approach, and suggests open standards in all cases where they are practical. If there were cases in the future where the e-GIF needs to promulgate an interoperability standard, publishing an OSS implementation is an efficient mechanism for encouraging uptake. The use of a BSD or LGPL license is appropriate in a case like this, as the intention is to allow vendors to incorporate the reference implementation in new products. At present, we can see no example where the e-GIF would need to take such a standards-making role.

2.7.5 There is also a close linkage between open data standards and protocols and the Government's ability to avoid the Government and the Citizen being sucked into the *monogamy vortex (see Section 3.4)*. It is the use of proprietary standards and protocols that effectively mandates the purchase of further products from the same supplier. Mandating the use of open internet standards (as in the e-GIF) rather than proprietary formats, and developing XML-based data definitions, for intra-Government, and

Government-to-Citizen interoperability, is a practical approach to controlling the monogamy vortex.

2.8 A mechanism for developing IT support in critical areas

2.8.1 Two specific areas where OSS may be important in the future are Health and Education.

2.8.2 The impact of OSS in education is very limited at present. The most highly visible education project is Red Escolar in Mexico. Each school will have a teaching laboratory with a Linux server and between 7 and 27 Linux desktop machines running the Gnome desktop software. This choice may be at least partly because the founders of the Gnome project are Mexican citizens! The school servers are connected to central servers from which lesson materials can be downloaded. The initial rollout is in San Luis Potosí. If this were a success, it would eventually grow to involve between 600,000 and a million computers. A quotation from this project follows (note: This text has been translated from a Spanish original at http://www.mexicoextremo.com.mx/noticias/redesc-linux.php3)

"Considering some numbers, the basic education system has some 120,000 schools and if each one were to have a complete computer laboratory, which would require a server connected to a LAN. The estimated cost, based on Windows, was approximately $55 US for the computer (including operating system and Office), plus the WinNT license, around $500 US, that is allowing 6 workstations per laboratory… it's better not to do the maths, it's truly a lot of money.

The first obstacle was the cost; but the second, no less important reason, was the difficulty in administering centrally so much equipment, to distribute updates and to support the users. Additionally, we have the problem of obsolescence of the equipment due to application updates, attacks by virus, instability of the operating system, etc."

2.8.3 There are many much smaller education projects, often involving an enthusiastic teacher determined to save money. For example, St Johns School at Northwood, London uses a Linux server to support a network of 24 Windows PCs with web serving and email. An Australian school has recycled otherwise unusable 386 PC's by using them as diskless display terminals running Linux, with the applications running on a server. This is cheap to administer, as the terminals have no disks or applications to be managed. There are many other examples; the main attraction at present is that technically capable teachers can save money and get improved security. We expect uptake will be fastest in the developing world, where cost is an even bigger driver.

2.8.4 Open Source health software is much more widely used than we had expected. The US Government funded the development of DHCP (Decentralized Hospital Computer Program). The project was started in 1982 by the U.S. Department of Veterans Affairs (VA). As of 1994, the VA operated 173 medical centers, 389 outpatient clinics, 131 nursing homes, and 39 domiciliaries. In 1995, DHCP was nominated for the Smithsonian award for best use of information technology. Because this software was developed by and for the U.S. Government, the source code is largely in the public domain. Several dozen other institutions have implemented systems based on the VA source code. Development has actively continued.

2.8.5 There are several Open Source projects that provide practice administration for medical, dental or veterinary practices; there are also commercial products that run on an Open Source infrastructure. There are also a variety of special purpose projects ranging from statistics for decision support, obstetrics, audiometry and translation between medical image formats. Many older medical systems were implemented in a computer language

called Mumps, and a free implementation of this for the PC has been developed to solve the problem of maintaining these legacy systems when the old hardware can no longer be supported.

2.8.6 The chief executive of the National Health Service Information Authority (NHSIA) is quoted[39] as saying:

"The NHS is well suited to shared software writing because it comprises many different but collaborating organisations.

It is the open-source development model which has aroused NHS interest, rather than products such as Linux"

2.8.7 A project that seems worthy of Government interest is the Medical Record DTD (Document Type Definition). We quote from http://sourceforge.net/projects/medrec/

"The goal is to develop XML DTDs and software to facilitate the secure transfer of personal health record information from notebooks, PDAs, and other local databases to websites that specialize in archiving health record information."

2.8.8 We are not medical IT experts but this seems a highly desirable goal and fits in with the e-GIF strategy of open data standards. We recommend that this project should be examined by domain experts; if they think it appropriate it might form part of some future release of the e-GIF.

2.8.9 We conclude that the Open Source model offers a new paradigm for funding software in communities-of-interest (e.g. Health and Education). The Government could consider running pilot projects to test the viability of the OSS approach to such software.

2.9 Influencing the COTS market

2.9.1 There are two situations we can think of where the Government might wish to influence the COTS marketplace:

2.9.2 The first is when there is no open (non-proprietary) approach to a Government need, and the Government does not wish to mandate a single vendor's proprietary approach on the Citizen or across Government. In such a situation, the Government might wish to sponsor the development of an OSS solution to that requirement.

2.9.3 The second is the situation where COTS products do not meet a key Government requirement (for example the NHS requirements for patient confidentiality). Many such requirements can be met by writing add-on software – hypothetically, software could add

[39] Computing, 27 September 2001, page 17.

confidentiality labelling to COTS word processors and presentation tools used in the health service. Sponsoring the development of OSS software that meets such requirements may be a good way to make such add-on products available for health IT contractors to build into NHS procurements.

2.10 Improving security

2.10.1 Strong assertions have been made both for and against the security of Open Source software. We will briefly restate the official position, some arguments in both directions, and then look at some empirical evidence.

2.10.2 The official rules about software security are set by CESG (part of the Government Communications Headquarters (GCHQ) at Cheltenham), and presented as Memo-10, now known as InfoSec-1. This is the definitive statement of current software security policy, and would override any incompatibilities in the interpretation we make here. A key idea in all security policy is trust. Memo-10 talks about the degree of trust in a system vendor. We presume that the use of Open Source software by a vendor does not in itself prohibit a trust relationship. If this is the case, then it should be possible for trusted vendors to offer appropriate Open Source software.

2.10.3 Systems for use at the higher levels of assurance are subject to an expensive and time-consuming audit of the development process. Passing such an audit requires a major commitment of resources by the developers, and precludes almost all off-the-shelf software, both commercial and Open Source. Modern security architectures try to minimise the number of components that require such assurance, but it is likely that some critical components will have to be procured from specialist commercial suppliers.

2.10.4 The arguments for and against the security of Open Source are impassioned and controversial. We will reproduce some typical arguments, and then move on to the empirical evidence.

2.10.5 Microsoft has argued[40] against the security of Open Source software and has raised a number of issues.

2.10.6 In a proprietary product the vendor's eyes in a security review tend to be dedicated, trained, full time and paid.

2.10.7 Network administrators are better off spending their time reading log files and installing patches than poring over source code looking for security holes, and the system of 'peer review' that works well for vetting encryption algorithms, doesn't work to evaluate large pieces of software for flaws. Microsoft's view is that an encryption algorithm is relatively simple, compared to a 40 million line operating system, and the discovery of an individual software flaw is not an attractive challenge to Open Source developers.

2.10.8 Microsoft believe that making source code public also increases the risk that attackers will find a crucial security hole that reviewers missed.

[40] See for example: http://www.securityfocus.com/archive/12/176723

2.10.9 Microsoft states that it does extensive testing on every product, and on every patch, and hence one of the reasons they take so long to distribute their security patches is because of that testing.

2.10.10 Microsoft warn that, in their opinion, the nature of open source development may lend itself to abuse by malicious coders, who could place clever 'trapdoors' in the code that escapes detection, hidden in plain sight.

2.10.11 Open Source software advocates would take issue with many of Microsoft's views. For example, almost all of the significant Open Source projects have extensive regression test suites, and use advanced frameworks for automated testing. Indeed the status of the project is frequently published on the Web in real time (for example see http://tinderbox.mozilla.org/showbuilds.cgi?tree=SeaMonkey which shows the status of the Mozilla code).

2.10.12 As recognised by Microsoft, peer review is universally practiced for cryptography algorithms and their implementations. Open Source proponents claim that by publishing the source code more people check it and so accidental security defects are more likely to be found. Deliberate security holes are sometimes built in to commercial products (so-called back-doors), rarely maliciously, more often for developer or installer convenience. Back-doors are much less likely in Open Source code, and in one recent case involving the Interbase database, a significant backdoor was discovered when the previously proprietary code was published as Open Source[41].

2.10.13 Empirical evidence about software security is not easy to find. Anecdotal evidence must be discounted, as it is probably biased. We have found four sources of evidence that seem credible: premiums for hacking insurance, the BugTraq vulnerability statistics, virus counts and a log of web site defacements.

2.11 Insurance Premiums

2.11.1 Insurance against attack by crackers (malicious hackers) has only recently become available. J.S.Wurzler[42] is one of the companies that offers policies, and charge a higher premium (by about 25%) for companies that use Windows NT, as they assess the risk of a payout to be greater. They do not distinguish between Open Source operating systems and other proprietary systems. Other insurers have not stated any predefined policy to our knowledge, and seem to assess risks on a case by case basis.

2.12 BugTraq vulnerability statistics

2.12.1 The BugTraq website[43] contains up to date reports on security problems that afflict operating systems and application software. They produce a year by year statistical summary, based on these reports but raise several important warnings about interpretation. We quote:

[41] http://www.theregister.co.uk/content/archive/16023.html
[42] http://www.jswum.com
[43] http://www.securityfocus.com/frames/?content=/vdb/stats.html

2.12.2 "There are many factors that should be considered while trying to interpret these numbers. The numbers do not distinguish between vulnerabilities discovered in the wild and those found proactively by developers or security researchers. Nor do they say anything about how quickly the vulnerabilities were fixed by the vendors. They do not take into account the popularity or impact of a vulnerability. A root shell vulnerability is treated the same as a disclosure of sensitive information.

2.12.3 It is possible some operating systems and applications have more known vulnerabilities because they are more popular and have undergone more scrutiny, or because their source code is available. Also an operating system or application with more features is more likely to contain vulnerabilities than those with less features, but the latter may not be suitable for some applications.

2.12.4 We consider a vulnerability to affect an application or operating system if the vulnerability affects a component that is part of the application or operating system when brought or downloaded."

2.12.5 Their summary data shows no significant difference in vulnerability counts between any of the major Linux distributors and Windows NT. For the last complete year of records (2000), Redhat Linux had 85 vulnerabilities, against 144 for Windows NT. This comparison is slightly flawed as Redhat Linux ships with a very large number of applications, while many of the comparable NT applications are separately licensed.

2.13 Virus counts

2.13.1 Virus infection has been a major cost to computer users. The LoveLetter virus is estimated to have cost $960million in direct costs and $7.7billion in lost productivity[44].

2.13.2 The anti-virus software industry is large, with sales totalling nearly $1billion a year. We have found several sources on the Internet that contain estimates of infection by viruses, classified by platform[45]. The numbers differ in detail, but all sources agree that computer viruses are overwhelmingly more prevalent on Windows than any other system. There are about 60,000 viruses known for Windows, 40 or so for the Macintosh, about 5 for commercial Unix versions and perhaps 40 for Linux. Most of the Windows viruses are not important, but many hundreds have caused widespread damage. Two or three of the Macintosh viruses were widespread enough to be of importance. None of the Unix or Linux viruses became widespread – most were confined to the laboratory.

2.13.3 Why is Windows disproportionately vulnerable? There are three reasons, one social and two technical. Windows is much the most attractive target for virus writers, simply because it is in such widespread use. For a virus to spread, it has to transmit itself to other susceptible computers; on average, each infection has to cause at least one more. The ubiquity of Windows machines makes it easier for this threshold to be reached. Finally, Windows has had a number of design choices over the years (*e.g.* execution of start-up macros in Word, execution of attachments in Outlook, lack of write protection on

[44] http://news.cnet.com/news/0-1003-200-5828578.html?tag=owv

[45] For example, http://www.sherpasoft.org.uk/MacSupporters/macvir.html, http://www.kaspersky.com/news.asp?tnews=0&nview=4&id=144&page=0#8, http://vil.nai.com/vil/default.asp

system directories in Windows 3.1/95/98) that have allowed the execution of untrusted code, and this has made it a very easy target.

2.13.4 There have been a number of high profile virus or worm problems this year that have nearly brought the Internet to its knees; these include the CodeRed, CodeBlue and Nimda worms. All of these worms have exploited vulnerabilities in IIS (Microsoft's Web Server), Outlook (e-mail) and Internet Explorer. They have prompted Gartner group to issue a recent recommendation[46] that IIS should not be used, and companies should adopt other servers.

2.14 Web site defacements

2.14.1 Crackers attack insecure web server machines and overwrite the pages with a message of some sort. This type of defacement is frequently reported to various security web sites. A log of web site defacements was maintained at http://www.attrition.org. The defacement log attracted many attacks, and is no longer published, but a statistical summary is still available. This summary is analysed by operating system and server. Note that not all defacements will be reported to attrition.org, so the records are incomplete. Two trends are clear. The total number of defacements is rising rapidly, and now typically of order 1,000 per month. About 60% of the defacements are of machines running IIS on Windows NT.

2.14.2 The Netcraft survey[47] of web-servers shows that only about 25% of sites on the Internet run IIS/NT, so a superficial interpretation of the numbers would suggest that it is disproportionately vulnerable. We do not believe this to be the case, as IIS/NT is typically used on commercial or government sites, which are attractive targets for cracker activity. Looking at a sample of the reports shows that in many cases the sites were poorly configured, or had not had current security patches applied. A proper security architecture, correct configuration, professional penetration testing and efficient application of updates, are needed regardless of the choice of server platform.

2.15 Misc. Security

2.15.1 It is interesting to note that in the US, the NSA has supported and is still supporting several security related Open Source projects, including a project to add security extensions to Linux.

2.16 Security conclusions

2.16.1 It is our view that there is no great security benefit or disbenefit between proprietary and OSS software. Issues such as properly designed, and rigorously maintained security architectures are much more important than the choice between OSS and proprietary systems.

2.16.2 It would be unwise to draw any general conclusions about proprietary software from recent high-profile incidents that have affected Windows. Other proprietary systems have been more fortunate; the problems appear to be specific to some widely deployed

[46] http://www3.gartner.com/DisplayDocument?doc_cd=101034
[47] http://www.netcraft.com/survey/

Windows components. It is outside the scope of this document to consider if this is largely due to the size of the installed base of Windows systems, or whether there are other intrinsic problems.

3 Improving the competitiveness of UK industry

3.1 As an exploitation route for Government-funded R&D

3.1.1 Open Source has been the *de-facto* standard for the exploitation of academic software research in the US for many years. It is hard to over-state the beneficial effect that this has had on the technology and the wider computer industry. Here are some examples:

1. The Berkley Electronics Engineering department released a circuit simulator called Spice in the 1970's. This became an industry standard, and formed the basis of the CAD design industry for electronic circuits. They followed this up with switch level simulators, and layout tools that formed the basis of the vastly sophisticated commercial tools now in use.
2. The Unix operating system was made available to academics by AT&T who had initially developed it. The Berkeley Computer Science department developed it, and their releases (known as BSD - Berkeley Standard Distribution) form the basis of many commercial Unix systems. The resulting sales now run into 10's of billions of dollars a year.
3. The X-consortium based at MIT developed one of the first modern user interfaces. The X window system is standard on modern Unix and Linux distributions[48], and had some influence in the design and implementation of the Microsoft user interface.
4. Another group at MIT developed a security product called Kerberos. This is widely deployed and has been adopted in a modified form by Microsoft for Windows 2000.
5. The DHCP health-care software was developed for the Veterans Administration. It is now used by several other health-care providers.

3.1.2 The impact of even this short list of examples shows how effective Open Source is as a method of encouraging industrial exploitation of academic research. The wealth creation impact in the US compares very favourably with the less impressive achievements of the UK software industry in exploiting Government funded software. We would recommend that it be the default exploitation route for Government R&D software.

3.2 Open Source as an Industry

3.2.1 The UK seems to be getting left behind in the market that is growing up around Open Source software. Many major IT companies like IBM, HP, SAP, Apple and Silicon Graphics have adopted the Open Source model for at least part of their software needs. IBM is a very interesting example, as they appear to now view software as a cost, rather than an income source. Instead their income comes predominantly from hardware and services. In addition to these giants, there are several smaller companies that have adopted a pure OSS model. These companies include Redhat, Ximian and Caldera in

[48] Linux uses the XFree86 implementation of the X window system.

the US, and Mandrake and SUSE in Europe. They are not by any means tiny; Redhat had sales of US $84 million in fiscal year 2001, and annualised growth near 100%. We are not aware of any UK companies that are developing and publishing significant Open Source software. Some companies offer consulting, training or support for Open Source platforms. The lack of UK software companies is ironic, as two of the most highly influential developers are UK residents, employed by Redhat and Ximian respectively. If Open Source becomes a significant part of the software industry, the UK is very poorly placed to take advantage. However if the recommendation to use OSS as an exploitation route for Government funded software (3.12.1) were adopted then the situation is likely to improve.

4 Conclusions

1. OSS is indeed the start of a fundamental change in the software infrastructure marketplace, and is not a hype bubble that will burst.
2. Within five years, 50% of the volume of the software infrastructure market could be taken by OSS.
3. OSS's position in large servers (e.g. those managing massive multi-user databases), such as those that underpin many large Government procurements, will grow from its current position of near zero penetration, to a position where OSS is a viable option, within 2 - 3 years.
4. Within the developed world, we as yet see no sign that OSS will become a viable alternative to Microsoft Windows, for user's (general purpose) desktop machines in the corporate or home PC markets. However, OSS on the desktop may soon become a significant player in the developing world. For these reasons we recommend against any preference for OSS on the desktop, but also recommend that this issue be reassessed by the end of 2002, by which time early trials of the use of OSS desktops may have generated sufficient evidence to warrant a reassessment.
5. We see no benefit that the Government would gain from expressing a general preference for OSS within server infrastructures.
6. The Government could clarify its position as to whether there are circumstances in which Microsoft products are to be preferred.
7. The Government could consider publishing policy as to how the risk of lock-in to proprietary protocols is to be managed.
8. As yet it is not possible to predict that OSS will make a major contribution to the software applications market.
9. Many of the Government's risks that arise from over-dependence on proprietary protocols and data formats for interoperability can be controlled by the selective use of open data standards.
10. The existence of an OSS reference implementation of a data standard has often accelerated the adoption of such standards, and we recommend that the Government consider selective sponsorship of OSS reference implementations.
11. The rise of OSS, offers the possibility that non-US players will find it easier to influence the future direction of IT infrastructure technology.
12. The Government should consider using OSS as the default exploitation route for UK Government funded software.
13. The differences between OSS and proprietary software are not a major factor in either improving or degrading the vulnerability of a nation's IT infrastructure.
14. We recommend that the Government obtain full rights to bespoke software that it procures – this includes any customisation of off-the-shelf software packages.
15. The Open Source model offers a new paradigm for funding software in communities-of-interest (e.g. Health and Education). The Government could consider running pilot projects to test the viability of the OSS approach to such software.
16. We recommend that the Medical Records data standard be examined by appropriate domain experts for possible inclusion in the e-GIF.

A Appendix A: OSS Desktops

A.1.1.1 Microsoft Office is the *de-facto* standard for desktop software, and any consideration of Open Source alternatives must recognise this. The suitability of Open Source desktops depends on the deployment scenario. Experienced power users with a large investment in the advanced features of Microsoft Office have very different demands to new users without preconceptions about desktop applications. Replacing an existing Microsoft desktop would almost certainly cause intense user resistance, while using an alternative in an entirely new organisation with newly recruited staff may be accepted.

A.1.1.2 There are five key Microsoft applications that are used by a large fraction of desktop users. They are Word, Excel, PowerPoint, Outlook and Internet Explorer. A viable Open Source alternative for each of these must support all the commonly used features, and work correctly in a stand-alone environment. We will call this viability. Many other users are likely to continue to use Microsoft products for some time, and this means that round trip interoperability is desirable. This means that a Microsoft file can be opened by the Open Source application, changes can be made and the altered version can be re-exported as a file that the Microsoft application can read. The complexity and limited documentation of Microsoft's file formats has made this very difficult.

A.1.1.3 Alternative applications will have their own user interface designs, so an investment in training would be needed to convert Microsoft Office users to an Open Source alternative. If the Open Source version were an exact user interface clone, it would avoid the need for retraining, but this opens questions of copyright infringement on look and feel. In any case, none of the Open Source projects are attempting an exact clone of a Microsoft product.

A.1.1.4 The final question for Open Source projects is the availability of commercial support; are there contractors who can install, support and customise the product? Can they provide end-user training courses?

A.1.1.5 There are three main Open Source projects to develop office suites, Open Office, Gnome Office and KOffice. Gnome Office and Koffice have grown out of the Gnome and KDE desktop projects. Each of these has developed a desktop infrastructure, and a host of more or less specialised applications[49]. Sun has released the source code to Open Office, which is based on a product suite called Star Office. Open Office is a cross-platform project, with downloads available for Windows, Linux and Solaris (Sun's own Unix implementation). Sun have announced the intention to integrate Open Office with Gnome, but there is little evidence that this has started yet, nor is it clear how this will be achieved. It seems likely that integration may merely mean that Open Office components will be embeddable inside Gnome documents and vice-versa. There are also plans to improve interoperation between KDE and Gnome, and eventually allow Gnome components to be embedded in KDE documents and *vice-versa.* This is unlikely to be available for at least a year. All three office suites are internationalised, and most of the major applications have a good choice of language packs. Development is very

[49] There is a simple rule of thumb to identify which desktop an application belongs to; if the first letter is a G it is Gnome, K implies KDE.

rapid, and any statements about the capabilities of a project are only true of the version we tried; new and improved development versions are being released every month or two.

A.1.1.6 We tried recent downloads from all three suites on a PC running Linux (Mandrake 7.2, kernel version 2.2.19), and a Gnome-1.4 desktop. Stability issues were noted, but we did not try even the simplest measures to resolve problems.

A.2 Microsoft Word Alternatives.

A.2.1.1 There are three Open Source word processors that we have tried, OpenWriter, Abiword and Kword.

A.2.1.2 OpenWriter is the word processor component of OpenOffice[50]. It is mature and fully featured; a quick examination of a recent download (build 638) showed every feature of Microsoft Word that the author had ever used. Several Word documents were imported with their formatting intact. The only niggle was that a few characters were converted to question marks. There are some performance and stability problems. On a 500MHz PC with 128Mb of RAM, it took 53seconds for the OpenOffice suite to start; once it had started performance was acceptable, but felt a bit slower than competing products. Large amounts of memory are used; we measured 45Mb with a single document open, but memory is now so cheap that this is not a big problem. The particular build we used clearly has some stability issues; it froze during startup on about one attempt in three, and crashed while trying to save a document. This type of problem has historically been resolved very quickly in Open Source projects. Once the stability issues have been addressed, we would define it as viable and interoperable.

A.2.1.3 Abiword is fast, clean and stable. It has a limited set of features; minimalist tables and no embedded objects. It can interoperate with Word by using rtf files and has a capable Word doc file importer, which can even handle fast saved documents. Word documents that use unimplemented features like tables have their formatting corrupted on import. The Word doc exporter is very limited, and not installed by default. Parts of this document have been written with Abiword. It is slightly unusual in that it is a cross-platform project, and works on Windows, Linux and other Unix flavours, Macintosh (alpha) and several more exotic platforms (QNX, Beos). Abiword works very nicely, the behaviour is intuitive and in some respects it is easier to understand than Word. It interoperates within its limited set of features; the main missing features (full table implementation, embedded objects) seem unlikely to be complete within at least six months. Until these features are complete, we could not define it as viable.

A.2.1.4 Kword is much more fully featured than Abiword. It has embedded objects, tables, footnotes, automated table of contents and an equation editor. The version tested was 0.8, part of the KDE 2.0.1 release and suffered some stability problems; the Word document importer froze on documents that Abiword handled perfectly. Stability is much improved on earlier versions. Kword is viable, but until the importer is stable it is not usefully interoperable.

[50] OpenOffice is derived from Star Office which Sun purchased and then Open Sourced.

A.3 Microsoft Excel Alternatives.

A.3.1.1 OpenCalc, Gnumeric and Kspread are all capable spread sheets.

A.3.1.2 OpenCalc is the OpenOffice spreadsheet component. It can import Excel spreadsheets. We tried several examples, and a few showed some minor formatting issues, but the data and formulae were correctly imported. Graphing is provided by the Chart component - this was entirely satisfactory. A BASIC interpreter is built in - this is a similar idea to the use of Visual Basic in Excel. Speadsheet loading was noticeably slow, but no other performance problems were noticed. OpenCalc is both viable and interoperable. The user interface is slightly different to Excel, so some retraining would be needed.

A.3.1.3 Gnumeric is stable, fully featured and has a very good Excel importer. The current release (0.67) does not have graphing enabled, nor does it support all the Excel text formatting features. It does implement about 95% of the Excel functions. Gnumeric also has some very powerful features, which are unusual, like a goal seeking optimiser, and flicker free scrolling, even on large sheets. Graphing is available in unstable development versions; we expect it will be in releases within three months. When graphing is fully supported, Gnumeric will be both viable and interoperable. The user interface and formula language are slightly different to Excel so some cross-training would be needed. There is even prototype support for embedded Visual Basic.

A.3.1.4 Kspread is more complete but less stable than Gnumeric. Object embedding is supported, as is graphing, but Excel compatibility is much more limited. The function library is not as complete as Gnumeric. Kspread is viable, but not yet so close to being interoperable. The user interface is perhaps slightly less like Excel's than Gnumeric; again some cross-training would be needed. We expect it will be at least six months before Kspread can round trip spreadsheets with Excel.

A.4 Alternatives to PowerPoint

A.4.1.1 Impress is the presentation component of OpenOffice. The user interface is very similar to PowerPoint and it can import and export PowerPoint files. We successfully loaded, modified and saved several PowerPoint presentations. The only slight niggle is that some bullet symbols were replaced by question marks; in other respects Impress is both viable and interoperable.

A.4.1.2 Kpresenter is the KDE presentation programme. It has a full set of features, and is quite easy to use, so is a viable choice. Until recently, very little had been published about the PowerPoint file format, and Kpresenter does not have a converter to read these files, so there is no interoperability. A converter from PowerPoint to HTML[51] has recently become available, so we expect that it is only a matter of time before the Kpresenter can import PowerPoint presentations.

A.4.1.3 A Gnome project to develop a presentation application has been started. It is called Achtung, and is at a very early stage of development. We doubt there will be anything usable, even by enthusiasts, within six months.

A.5 Alternatives to Internet Explorer

A.5.1.1 There are several good Open Source Browsers; Mozilla, Galeon and Konqueror all work well. The commercial Netscape browser is closely based on an Open Source browser called Mozilla. Mozilla supports most modern standards; it is somewhat heavy weight. Galeon uses the rendering core of Mozilla in a faster, lightweight Gnome shell; it is very responsive but not yet as stable as Mozilla. Konqueror is also a very viable choice, and is integrated into the KDE file manager. The user interfaces are simple and intuitive, and the differences do not matter greatly. The only barrier we can see is the use of web pages that depend on non-standard features of IE, which may not offer full functionality on other browsers. This is not a problem of Open Source *per se,* but rather of lock-in to a particular set of Microsoft features.

A.6 Alternatives to Outlook

A.6.1.1 Outlook combines an e-mail client with a calendar, contact management and document management. This combination of functionality can easily be provided by several separate Open Source tools, but we known of only one project to produce a direct competitor; Evolution is part of the Gnome desktop. It is still beta software, but the last release (version 0.14) is very usable. Not all features are complete, but the overall impression is solid. We expect that it will be complete and released within six months. Note that Evolution uses Internet protocols, and does not use the proprietary Microsoft Exchange protocol. However Exchange servers can be set to support POP3 and IMAP, which Evolution does use, so it could be integrated into an existing network where Windows servers host the mail.

A.6.1.2 Basic e-mail functionality can be provided by any one of a number of clients. Kmail is very good and widely used in our group. Balsa is a simple but adequate mail client for Gnome, and there are many others.

A.6.1.3 Calendar and contact management functions are provided by so called PIM (Personal Information Management) applications. Gnome has Gnome-Cal and KDE has Ical. Both are entirely adequate, and include features to sync with a Palm or similar device.

A.6.1.4 OpenOffice does not include e-mail or PIM components.

A.6.1.5 The use of multiple applications to do the disparate jobs that are bundled into Outlook is likely to be unpopular with users who are used to Outlook. We therefore feel that in some cases, Evolution will be the only choice (ie, as an OSS alternative to MS Outlook).

[51] The converter is bundled with an Excel converter. The software is at http://www.xlhtml.org

A.7 Roll-out, Training and Support.

A.7.1.1 We have seen that the Open Source desktop applications are developing rapidly but are not yet fully mature. There will be a further delay once they are mature before large scale roll-out is practical. Few companies have the expertise to undertake large scale roll-out of Linux desktop products, and it will take some time before installers and support staff can be trained. Redhat, SUSE and Mandrake offer some end-user training packages and call centre support, but we doubt that they yet have the resources to support a large-scale deployment.

A.7.1.2 We expect that the developing world is a likely first area of significant deployment for Open Source desktops. A software license that costs say £500 is not a great barrier for most UK companies; it is worth paying to save a few days (or even hours) of employee time. In the developing world, this is not true, and free desktop software looks much more attractive. If Open Source desktops are deployed widely in the developing world, this will make the crucial difference to their viability; it is only by exposure to large numbers of unsympathetic customers that the applications will reach full maturity.

A.7.1.3 There are reports[52] of Open Source desktops being used in organisations like Police Authorities and councils. A closed community may have a reduced need to interoperate with the standard Microsoft applications, and the number of desks offers potential savings in up-front costs. Early adopters are also likely to be organisations that possess the in-house expertise to manage what is not yet a mainstream IT configuration. The experiences of these early adopters will form important evidence to see if there is a Total Cost of Ownership (TCO) advantage in using an Open Source desktop in these scenarios. We recommend that the status of Open Source desktops should be re-considered in a year, as the software will be significantly more mature, and there should be results from these early adopters.

[52] See Computing Aug 2 2001, pg 3 which reports that Central Scottish Police and an unnamed local council have adopted Sun's Open Source Star Office suite. A comparable American initiative in the City of Largo, Florida, is reported at http://www.newsforge.com/article.pl?sid=01/08/10/1441239

B Appendix B: OSS Licenses

B.1.1.1 There is a widely agreed definition of what constitutes an Open Source license. It can be found at http://www.opensource.org/docs/definition_plain.html. The definition is reproduced below:

B.2 The Open Source Definition

Version 1.8

Introduction

Open source doesn't just mean access to the source code. The distribution terms of open-source software must comply with the following criteria:

1. Free Redistribution

The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

2. Source Code

The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost–preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.

3. Derived Works

The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

4. Integrity of The Author's Source Code

The license may restrict source-code from being distributed in modified form *only* if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

5. No Discrimination Against Persons or Groups

The license must not discriminate against any person or group of persons.

6. No Discrimination Against Fields of Endeavor

The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. Distribution of License

The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8. License Must Not Be Specific to a Product

The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

9. License Must Not Contaminate Other Software

The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.

B.2.1.1 Although the supporters of Open Source software often stress that it is free only in the sense of “free speech”, it is important to note that Clause 1 above explicitly states that the license shall not require a royalty or other fee for such a sale. This means that although it is permissable to charge for derived works, the underlying OSS remains freely available (in every sense of the word) to the IT community.

B.2.1.2 The Open Source definition also makes it clear that the originators of OSS cannot stop another group from cloning the software and producing a separate project development line that creates a variant of the OSS. This process is known in the community as “forking” a project. A number of high profile OSS projects have created competitive forks – often with significant animosity between the different projects. A benefit of OSS is that when the original developers, or the original investors, lose interest in a project there is nothing to stop another group picking up and running with software (indeed this situation is quite common and happened recently to the Python programming language).

B.2.1.3 There are two major variants in OSS licenses. There is the GNU General Public License (GPL – http://www.gnu.org/copyleft/gpl.txt) and there are other licenses, which tend to be referred to as Berkeley-style licenses. The differences between these licenses appear subtle, but are in fact extremely significant. It is not unusual for the arguments as to the benefits and weaknesses in these two styles of licenses to get very heated

indeed. The importance and emotion can be best understood by understanding why the GPL was produced (the Berkeley style license was the original OSS license):

B.2.1.4 The University of California at Berkeley produced a large amount of software that made up a significant part of the leading UNIX operating system in the late 1970s. The Berkeley license allowed proprietary system vendors, such as SUN, IBM and HP to separately fork closed source variants of Berkeley's software. Each vendor's UNIX became subtly incompatible so that programs written for one closed variant of UNIX would not run unchanged on other closed variants of UNIX. This did a lot to destroy the role of UNIX as an interoperability standard. The GPL was created with a clause that outlaws the creation of closed forks. It does this by requiring that any derived work must be distributed with the source text under GPL. For this reason the GPL is sometimes called a viral license, because it infects forks with the GPL's license conditions. This means that GPL is highly suited to OSS that is intended to act as an interoperability standard, such as Linux. It also undermines some commercial exploitations, as it limits what can be charged for a derived work because the deriving organisation is forced to freely publish the source text of the derived work.

B.2.1.5 The original Berkeley license allows greater freedom for commercial exploitation, but at the danger that multiple, incompatible closed derivatives will emerge.

B.2.1.6 An additional feature is that it is easy for a GPL project to incorporate Berkeley-licensed code, but a Berkeley-licensed OSS project has to be careful not be infected by GPL code.

B.2.1.7 GPL proved to be unsuitable for some software libraries that were an integral part of some popular software development tools developed under GPL. This led to the situation that if you compiled a program using one of these tools, your program would contain some of the libraries, so your program had to be published under GPL. A lesser GPL (LGPL) was constructed which stated that linked libraries did not infect the programs to which they linked.

B.2.1.8 There are a host of different Berkeley style licenses, which have subtle differences (such as the precise definition of derived works), but there are no differences that the authors feel are significant enough to discuss.

This page is intentionally blank

Report documentation page

1. Originator's report number:			
2. Originator's Name and Location:	Dr Nic Peeling		
3. MOD Contract number and period covered:			
4. MOD Sponsor's Name and Location:			
5. Report Classification and Caveats in use:	6. Date written: October 2001	Pagination: xiii + 37	References:
7a. Report Title:	Analysis of the Impact of Open Source Software		
7b. Translation / Conference details (if translation give foreign title / if part of conference then give conference particulars):			
7c. Title classification:			
8. Authors:	Dr Nic Peeling and Dr Julian Satchell		
9. Descriptors / Key words:	**OPEN SOURCE SOFTWARE**		
10a. Abstract. (An abstract should aim to give an informative and concise summary of the report in up to 300 words).			
10b. Abstract classification:	FORM MEETS DRIC 1000 ISSUE 5		

This page is intentionally blank