The Case for Government Promotion of Open Source Software

A NetAction White Paper

By Mitch Stoltz
mitch@netaction.org

Abstract

An alternative method of software development, called open source software, creates robust, secure software through a process of widespread peer review. This paper explains the open source concept and attempts to show how government can use open source as a vehicle for promoting economic development and as a policy tool which could assist the Justice Department in its antitrust action against Microsoft.

About the Author

Mitch Stoltz is currently finishing his senior year as a Computer Science and Public Policy Analysis major at Pomona College, outside of Los Angeles. His interests are computer networking, encryption, consumer privacy issues and equitable access to technology. His next project is a senior thesis on Open Source as a social movement.

NetAction
601 Van Ness Ave., #631
San Francisco, CA 94102

Phone: (415) 775-8674
Fax: (415) 673-3813
Web: http://www.netaction.org

The Case for Government Promotion of Open Source Software

Introduction

Computers and the Internet have changed the way we work, study, and interact, yet there are many things about computers and software which we find dissatisfying. Proprietary software is increasingly expensive and memory-hungry. Bugs, security flaws, and other errors appear in even the most trusted programs. Microsoft's monopoly control of the operating system market stifles innovation. Many computer systems are not equipped to handle the upcoming turn of the century, creating a multi billion-dollar problem and dire predictions of a global electronics breakdown.

An alternative method of software development exists, called open source software, which offers a very low cost solution to all of these problems. Open source is not a technology, but rather a different way of thinking about and organizing the software development process. Whereas traditional proprietary software development (which created most of the programs we use daily) adheres to the principle of strict protection of intellectual property found in the publishing industry, open source software (OSS) development is more of a collaborative process that has evolved along with the Internet.

Open source software is growing its market share in a few key areas because of its natural strengths of reliability, security, and low cost. However, open source has advantages on a broader level as well: it eliminates economic waste caused by the duplication of work, and it presents a challenge to harmful monopoly power in the software industry, such as the anticompetitive practices which are under scrutiny now in the Justice Department's antitrust case against Microsoft. It also provides a cost-effective solution to the Year 2000 problem. For these reasons, increased use of open source software serves more than private economic gain -- it serves a public good as well.

This paper will describe open source software, including a brief history of the idea, discuss its inherent strengths as both a private and a public good, explain why the government should be involved in promoting open source software development, and offer some recommendations for government action.

What is Open Source?

The most basic definition of open source software is software for which the source code is distributed along with the executable program, and which includes a license allowing anyone to modify and redistribute the software. Source code is the actual instructions which programmers write to create a piece of software, the "recipe" for the program. Once a program has been "compiled" into a form which can be installed and run on a computer, its source code is irretrievable. It is practically impossible to make changes to a program without having a copy of its source code. If a program's license includes the right to modify the program, this right is meaningless unless the source code is readily available.

Actual licenses for OSS vary between different companies and development projects, but they have certain characteristics in common. The Open Source Initiative, a group of developers who disseminate information on the benefits of open source,[1] has posted on its web site a "meta-definition" of basic conditions which they feel should be included in an OSS license.[2]

These include:

- Allowing free redistribution of the software without royalties or other fees to the author.
- Requiring that source code be distributed with the software or otherwise made available for no more than the cost of distribution.
- Allowing anyone to modify the software or derive other software from it, and to redistribute the modified software under the same license terms.

Any software which is distributed under a license which conforms to these requirements is open source software, according to the Open Source Initiative.[3]

Proprietary software, which makes up the majority of the software we use on a daily basis, is distributed under much different conditions. A proprietary license prohibits modification, copying, or redistribution without the company's permission. It ensures that only one entity -- the company or individual that created the software -- has the right to make changes or even see the software's internal structure.

In addition to its legal definition, another distinction between OSS and proprietary software is the way in which it is developed. Proprietary software is created by a relatively small group of developers within a particular company, often working under deadline pressure. They complete a program and then try to remove as many flaws (software errors or "bugs," and security "holes") as possible before the software goes to market. Any flaws which remain after shipping time become the consumer’s problem, leading to lost work and frustration. Purchasers of proprietary software become involuntary testers. What's more, if users find a flaw, even if they know how to solve it, the software license prohibits them from making the fix themselves.

Open source software, in contrast, is often developed by loosely organized communities of programming enthusiasts, collaborating via the Internet. Anyone with an interest and some requisite degree of ability is welcome to contribute sections of the program or to look for errors in existing sections. Because no one is excluded from the development process, potentially hundreds of people can contribute to a project, providing a diverse group of talents and techniques. If a particular company has a financial interest in the success of an open-source project (through strengthening its brand name, increasing demand for related products, or through sales of technical support), they will often hire programmers to work on the project. Other contributors (and for some projects, all contributors) may be individuals working in their spare time, out of interest rather than for compensation.

Open source enthusiast Eric Raymond describes a successful OSS development project in his essay "The Cathedral and the Bazaar."[4] Good OSS projects, he says, reuse as much code from other projects as possible to avoid duplicated work. They rely heavily on feedback and suggestions from users of the software, operating under the principle of "release early, release often, and listen to your customers."[5] This intense peer review process, shared among a potentially large group of developers and testers, finds and eliminates errors in software faster than any proprietary effort could. In Raymond's words, "Given a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix obvious to someone. Or, less formally, 'Given enough eyeballs, all bugs are shallow.'"[6]

Open source software is closely connected to open communications standards (or protocols), such as the Internet standards, which allow many types of computer equipment to communicate over networks. Both are developed through widespread informal collaboration. Any

communications protocols used in open source software are inherently open, since an implementation of the protocol is revealed in the open source code. The existence of an open source implementation of a communications protocol ensures the openness of the protocol. Conversely, proprietary software allows for proprietary protocols, since details of the protocol can be hidden in the secret source code. The Apache project, makers of an open source web server which serves over half of all World Wide Web pages, explains this relationship well:

> To the extent that the protocols of the World Wide Web remain "unowned" by a single company, the Web will remain a level playing field for companies large and small. Thus, "ownership" of the protocol must be prevented, and the existence of a robust reference implementation of the protocol [in the form of working software], available absolutely for free to all companies, is a tremendously good thing.[7]

Though anyone can contribute to them, open source projects often have some form of central authority which collects and combines the changes which others make. These authorities, whether an individual, a corporation, or a nonprofit, work to maintain compatibility and conformance to standards in balance with improvements from the community of developers. This keeps open source software compatible with the Internet and other important standards, and prevents "fracturing," the divergence of several concurrent versions of the same software to the point of incompatibility. An example of this sort of informal coordination can be seen in the Linux operating system: though a large group of developers contribute to new versions, Linus Torvalds, Linux's original creator, has final say over what code is included. Similarly, the Apache web server is overseen by the Apache Group, a self-selected group of programmers.

Open source software is economically viable and presents numerous opportunities for profit. This seems counterintuitive, since the availability of source code allows the user to obtain a piece of software at no cost. But many companies have already realized substantial profit from their OSS efforts. Red Hat Software has become a fast-growing and profitable company selling boxed versions of the Linux operating system, complete with manuals, technical support via telephone, and a simple installer program. Red Hat customers, some 7.5 million by the company's estimate, chose to pay about $50 for the added value which the company provides to Linux users -- even though the company itself gives Linux away for free. As the company says:

> "Most of the software is available -- at no charge -- to anyone with the time and inclination to download it. But not everyone has that much time...A company's distribution has its own feature set, and some are geared towards specific types of computer systems. Like many other such groups, our approach is to bundle all the necessary bits and pieces into a cohesive distribution."[8]

Both individuals and businesses are willing to pay for this convenience. Additionally, the software itself retains some economic value even if it is given away for free. The Open Source Initiative web site explains this way: "If having a program written is a net economic gain for a customer over not having it written, a programmer will get paid whether or not the program is going to be free after it's done."[9]

An eloquent testimonial to the effectiveness of OSS comes improbably from one of its opponents, the Microsoft Corporation. An internal Microsoft memo now known as the "Halloween Document," which was leaked to Eric Raymond and subsequently posted on the Internet, says the following about OSS:

> "The ability of the OSS process to collect and harness the collective IQ of thousands of individuals across the Internet is simply amazing. More

importantly, OSS evangelization scales with the size of the Internet much faster than our own evangelization efforts appear to scale."[10]

History of the Idea

Although the term "open source software" was coined rather recently, the idea has existed for many years. In the 1960's, when computers were cumbersome and esoteric, all software was essentially "open source." As all computer users at the time were effectively also software developers, whose work required them to make changes to the software they used on a regular basis, the computers made by manufacturers such as IBM were generally shipped with source code included. In the close-knit community of computer scientists, programs and ideas were shared freely. Beginning in the 1970's, when many types of businesses (and later, individuals) began using computers for more diverse tasks, software became proprietary in the interest of profit. With only a small fraction of computer users actually writing software, it became profitable to limit source code access and modification rights to within a single company.

The one area where OSS has thrived in the period of proprietary software development is in the creation of the Internet. The programs on which the Internet depends are for the most part open source software. The Apache web server, the Sendmail mail-forwarding program, and BIND, which manages Internet addresses, are all open source, and each dominates its market.

Many developers who had done work in the earlier period of software development felt that a sense of collaboration and cooperation had been lost, and that software quality would suffer as a result. One of these was Richard Stallman, who in 1983 founded the Free Software Foundation (FSF)[11] to promote collaborative, open software development. Today the FSF remains a key player in the open source initiative.

Another major event in the re-emergence of OSS occurred in 1991 when Linus Torvalds, a college student in Finland, created an open source version of the Unix operating system. This operating system, called Linux, is now a mature product, claiming over 7.5% of the 3.5 million installed server operating systems (compared to Microsoft Windows NT's 36%).[12] It is among the top five operating systems in use worldwide. Many people consider Linux to be faster and more error-free than Windows NT or other proprietary operating systems. A large group of developers from almost every continent, as well as several commercial companies,[13] maintain and update Linux. Development of Linux occurs very fast. During periods of intense development work, new versions are sometimes released as often as once a day. When new features are added, they are scrutinized by the development community, which finds and corrects all manner of errors. New versions of Linux often evolve to shipping quality in a matter of weeks, rather than the months or even years of testing which are common in proprietary software development.

In March of 1998, Netscape Communications, makers of the most popular Internet browser, shocked the software community by announcing that they would release the source code to their browser and begin to accept changes and improvements from the Internet community. In essence, the Netscape browser was changed from proprietary to open source. This decision arguably made Netscape the first well-known, mass marketable piece of software to embrace the open source model. Though Netscape will soon cease to exist as an independent company, having been acquired by America Online, the browser code it released belongs permanently to the open source community (through the independent entity mozilla.org) and can never be returned to its proprietary state.

Private and Public Advantages

Open source software has several distinct advantages over proprietary software. As described above, the widespread peer review process involved in open source development creates software which is more error-free and resource-efficient than proprietary software. In addition, OSS is a must for security-critical applications. As computer security expert Bruce Schneier points out, true security is never achieved by attempting to conceal any security defects that a program may have, but rather by allowing anyone interested to seek out these flaws and eliminate them.[14] Open source software makes this possible. Many government agencies will not use a piece of software in a security-critical application unless the agency itself can examine the source code for flaws; in the case of proprietary software, this often means difficult and costly negotiations allowing the agency access to the source code. If open source software is available to fill such a need, source code is available at no extra cost to the government, and in many cases the software is already more secure.

These advantages to the individual customer which are provided by open source software obviously benefit government users as well. Low cost, reliability, security, and the ability to modify software to suit specific needs are all important priorities to government purchasing authorities. However, these benefits do not make a case for specific government action to promote OSS. In Raymond's words,

> "The open-source culture will triumph not because cooperation is morally right or software "hoarding" is morally wrong, ...but simply because the closed-source world cannot win an evolutionary arms race with open-source communities that can put orders of magnitude more skilled time into a problem."[15]

However, the widespread use of OSS would benefit the U.S. economy as a whole in several ways, and it is for these reasons that government policy which facilitates and promotes open source software development would serve a public good, and is therefore a justifiable and beneficial government endeavor.

The first public benefit of OSS is that it eliminates the economic loss which results from duplicated work. The vast majority of all code (a standard estimate is 75%)[16] written for a specific task by a single company, government agency, or military branch, and is never used for any other purpose. Many problems in computer engineering show up in multiple fields and applications. If a private company creating software for scientific research, for example, must spend its cash and programmer time to create a specific tool from scratch when a military research facility has already written software which performs the same function, economic waste occurs which hurts U.S. productivity as a whole. If source code developed for a specific government application is made publicly available, corporations can spend their resources to improve this software, add value, and find new markets for it, rather than recreating it from scratch. The reverse is true as well: government and military agencies could use source code developed by corporations at no cost, allowing huge savings in government procurement and R&D expenditures.

Another area which the government has already identified as a public interest is working to solve the Year 2000 (Y2K) problem. This refers to the errors which may occur when many computer systems' clocks reach January 1, 2000. Since many important systems store years by their last two digits only, these systems will read 2000 as 1900. This could cause many critical computer systems at banks, hospitals, and in government, to stop working, with the potential for catastrophic failure. President Clinton and Congress have taken an interest in working to prevent such failures, including the passage of legislation which allocates funds to the solution of the problem and encourages private companies to begin working towards a solution as well.

If more computer systems utilized open source software, a solution to the Y2K problem would be much simpler and less expensive. This is because the Y2K problem is uniquely suited to solution by an open source effort -- the problem is extremely widespread but each individual solution is simple. For many programs, solving the problem requires a programmer to find each reference to a date and each calculation performed on dates, and expand it to allow for four-digit year references. The difficulty is finding all references to the date in very large programs. Doing so requires a massively cooperative effort to see that no reference is overlooked. If software is proprietary, the number of people with access to source code, and therefore the number of people available to find and correct all date references, is severely limited. With open source software, on the other hand, an almost unlimited group of programmers can share this work, allowing for a much more effective solution. In addition, access to source code allows a government agency or company to verify for itself that Y2K problems have been solved, without having to trust the manufacturer's claims.[17] Because the solution to the Y2K problem is easy in the context of open source development, almost all commercial quality open source software on the market today, such as the Linux operating system, is already Y2K-ready.

Perhaps the most compelling reason why the promotion of open source software serves a public good is that OSS is inherently anti-monopolistic, and may serve as an effective antidote for the monopolistic tendencies which some economists believe exist in the software industry. The market for computer operating systems and other key applications is currently dominated by the Microsoft Corporation. In its ongoing antitrust lawsuit against Microsoft, the Justice Department has claimed that Microsoft is using illegal methods to maintain and extend its monopoly in a way which hinders competition and stifles innovation[18].

Economist Brian Arthur theorized that the phenomenon of network externalities (also called "increasing returns") creates monopolies in high-tech industries and allows inferior products to maintain market dominance at the expense of consumers. In software, according to his theory, "there is no presumption...that superior technology wins."[19] Network externalities means that the value of a product increases with the number of people using it. This is often true in software, since the more people using a particular operating system, the more incentive developers have to write applications for that operating system and not others, which in turn reinforces the market dominance of the operating system. A small initial advantage can lead to a virtually unbreakable monopoly.[20] This phenomenon can also occur in other product relationships besides the operating system-application relationship, such as software which interacts over a network using a particular protocol.

Monopoly control through network externalities depends on keeping the underlying structure of software, and the details of how it interacts with other software, a secret. Because only Microsoft has access to and control over the software interface through which application programs interact with the Windows operating system, it would be nearly impossible for any other company to design an operating system which could run programs designed for Windows. Microsoft's exclusive control of the Windows programming interface is what has allowed it to exclude other operating system competitors, such as IBM's OS/2. The open source operating system Linux, on the other hand, does not allow this sort of monopolistic exclusion. Because the source code of Linux is open and freely available, anyone can distribute Linux or write another operating system which can run Linux application programs. Thus, application developers would have no need to favor a particular operating system manufacturer, and the cycle of monopoly "lock-in" would be broken. Even if Linux captured a majority of the market for operating systems, no single company would be able to erect barriers to competition. In fact, there are several companies and organizations developing professional-quality Linux systems, and the vast majority of software written for any of these systems will run on any other.

Microsoft is well aware of how OSS eliminates monopoly power. In the "Halloween Document," the internal Microsoft memo previously mentioned, Microsoft technician Vinod Valloppillil observes that:

> "OSS poses a direct, short-term revenue and platform threat to Microsoft... Additionally, the intrinsic parallelism and free idea exchange in OSS has benefits that are not replicable with our current licensing model and therefore present a long term developer mindshare threat."[21]

Valloppillil also acknowledges that OSS prevents monopoly control because it guarantees the availability of open protocols for software interaction. "Linux can win," he says, "as long as services/protocols are commodities,"[22] referring to the open communications standards which Linux uses.

Government, through the Justice Department and other agencies, is already committing resources to correct the apparent problems caused by monopoly power in the software industry. A program of OSS promotion and encouragement would be a cost-effective contribution to this effort.

Recommendations for Government Action

There are several inexpensive ways in which government could help the open source effort:

- Being the world's largest consumer of computer software, the U.S. Government has the ability to promote the widespread use and continued development of open source software through its purchasing policies. Not only would many government agencies benefit from the added reliability and security which OSS products provide, but the increased demand for these products would encourage more corporations and independent programmers to embrace OSS methods. This trend has already begun on a small scale: the U.S. Postal Service, for example, uses a highly modified version of Linux to read addresses on envelopes electronically. Many other agencies use Linux for network administration tasks, as it is considerably more affordable then the competing Windows NT software. A risk-free way to assess the benefits of OSS to particular government agencies would be for Congress to initiate a study by the General Accounting Office. The conclusions of such a study could serve as a road map for future software procurement. The study could address the following questions:

    1. Does open source software deliver more reliability and security relative to its cost than proprietary software?

    2. Which government agencies could benefit from a transition to open source software, such as the Linux operating system?

    3. Would it be feasible for these agencies to begin a transition to the use of open source software?

- Another possible government action involves the vast pool of software created for internal tasks within the government and the military. Collecting nonclassified source code in a series of repositories for the purpose of allowing public access would benefit both government and the public. Companies and individuals will have access to the expertise of government and military software engineers, obviating the need to solve software problems which have already been solved. Additionally, if some individual or group takes an interest

in improving some piece of software in use in a government agency, the agency will reap the benefit, at no cost to taxpayers.

Many agencies would no doubt object to the perceived security risk involved in disclosing government source code. As mentioned above, computer security experts consider this argument fallacious. A determined attacker can find security flaws in software with or without the source code, so concealing the source is actually more of a hindrance to those who could seek out and correct security flaws than to those who would exploit them. To put it simply, concealing source code leads to a false sense of security. Opening source code to the public, though it may create short-term apprehensions, will result in more secure software in the long run.

- These actions can be addressed in terms of the ongoing effort to eliminate the Year 2000 Problem from government systems. Using nearly any recently developed OSS software assures Y2K readiness, and opening the source code of internally developed software allows for easier modification of that software to solve the problem. These issues should be brought to the attention of the Council on Year 2000 Conversion recently established by President Clinton.

- Finally, as discussed above, open source and open standards go hand in hand. Simple, open communications protocols and standards of compatibility facilitate OSS development, as they form a fundamental building block of any OSS project. "OSS projects have been able to gain a foothold in many server applications because of the wide utility of highly commoditized, simple protocols," writes Microsoft's Valloppillil. Significantly, the strategy proposed in that document for competing against OSS is "extending these protocols and developing new protocols,"[23] which implies replacing open standards with proprietary ones. It is precisely this sort of predatory practice which the government should oppose, both on antitrust grounds and specifically to prevent Microsoft from using its control of protocols to interfere with OSS development. The government should more vigorously lend its support to the open standards developed by industry, such as the Internet Engineering Task Force's standard set.

Conclusions

Open source software has natural strengths which benefit individual users. Above and beyond this, the widespread use of open source software holds advantages for the U.S. economy as a whole, and it is for this reason that a careful program of government promotion and encouragement would be justified and beneficial.

End Notes

---

[1] See: <http://www.opensource.org>.

[2] The full text of these guidelines are found at <http://www.opensource.org/osd.html>.

[3] An example of such a license is the Free Software Foundation's GNU General Public License, which can be found at <http://www.opensource.org/gpl-license.html>.

[4] See: <http://www.tuxedo.org/~esr/writings/cathedral-bazaar/cathedral-bazaar.html>.

[5] Ibid.

[6] Ibid.

[7] See: <http://apache.org/ABOUT_APACHE.html>.

[8] See: <http://www.redhat.com/linux_what.phtml>.

[9] See: <http://www.opensource.org/open-jobs.html>.

[10] See: <http://www.opensource.org/halloween1.html>.

[11] See: <http://www.fsf.org>.

[12] See: <http://www.news.com/News/Item/0,4,23811,00.html?st.ne.fd.gif.c>.

[13] One such company is Red Hat Inc., <http://www.redhat.com>.

[14] See Bruce Schneier's Applied Cryptography, second edition, Wiley, 1996.

[15] "The Cathedral and the Bazaar," chapter 10 (see note 4).

[16] See: <http://www.opensource.org/open-jobs.html>.

[17] See: < http://www.opensource.org/y2k>.

[18] See the NetAction White Paper, "From Microsoft Word to Microsoft World: How Microsoft Is Building A Global Monopoly," by Nathan Newman, <http://www.netaction.org/msoft/world/>.

[19] Cassidy, John, "The Force of an Idea," The New Yorker, January 12, 1998, p. 32.

[20] Ibid.

[21]"Halloween 1" (see note 10).

[22] Ibid.

[23] Ibid.