Open Source Software

By Sitsofe Wheeler
20th November 1999

Table of Contents

1. Copyright

Online versions of this document should be available from either:
http://www.sucs.swan.ac.uk/~sits/articles/19991120.html or
http://geocities.yahoo.com/articles/19991120.html

2. Introduction

Although often used to describe the availability of a program's code, open source is really a method to engineer software. Its recent popularity is due to the success of the operating system Linux, but open source software was being developed as far back as 1976 in the form of the text editor Emacs.

This essay aims to explain the basic concepts behind open source software and how they could be applied today. In section 3.1, I state the criteria that software must follow in order to be open source and go on to describe the differences between open and closed source in section 3.1. Although open source is advocates are keen to stress its advantages, I discuss some of its pitfalls in section 7.

More commercial developers are looking at the methods employed by open source development and trying to learn from them. Now that the benefits have been proved, even the largest of the closed source software companies, Microsoft[1], has looked at the ideas raised by open source software. Another such company is Netscape and in section 6 I look at its Mozilla project.

[1] "The Halloween Documents."
http://www.opensource.org/halloween/

3. What does Open Source mean?

3.1 A definition

For software to be officially open source, it must provide/comply with the following criteria (definition taken from the open source web site):

1. Free Redistribution.
2. Source Code.
3. Derived Works.
4. Integrity of The Author's Source Code.
5. No Discrimination Against Persons or Groups.
6. No Discrimination Against Fields of Endeavour.
7. Distribution of License.
8. License Must Not Be Specific to a Product.
9. License Must Not Contaminate Other Software."

[1]

As the site says, open source is more than access to source code. It provides a means for others to take work that has been done and expand it as they see fit.

Examples of Licences which conform to open source include the GNU Public License (GPL), the BSD license used with Berkeley Unix derivatives, the X Consortium license for the X Window System, and the Mozilla Public License.

It is interesting to note points 5 and 6. Their inclusion might mean that a terrorist organisation could take an open source encryption program and use it to communicate securely amongst each other. Even if you knew about this, under this licence you would be powerless to stop it.

4. Order vs. Anarchy?

4.1 The Cathedral and the Bazaar

In his paper "The Cathedral and the Bazaar", open source advocate Eric Raymond launches an open source project of his own to see if Linux' methods can be applied to other programs.

In this paper he compares two approaches of engineering software. The first is the cathedral:

> "…(operating systems and really large tools like Emacs) needed to be built like cathedrals, carefully crafted by individual wizards or small bands of mages working in splendid isolation, with no beta to be released before its time."
>
> [2]

In short, there is a centralised body governing the overall construction of the software, where debugging and coding go on behind closed doors.

The alternative to this is the bazaar:

> "…release early and often, delegate everything you can, be open to the point of promiscuity… a great babbling bazaar of differing agendas and approaches… out of which a coherent and stable system could seemingly emerge only by a succession of miracles."
>
> [2]

No governing bodies here, just anarchy and frequent releases. So how could such disorganisation lead to "coherent and stable" software?

A developer creates his software but also allows access to the source code so that his work can be scrutinised. If they wish, the program's users can make changes to the source and submit changes of their own to the author. Thus the users are elevated to the status of co-developers.

Open source works so well because you garner more support than a closed project could ever afford. With so many people using and developing the software, there is a greater chance that bugs will be spotted and someone will have the correct "toolkit" to deal with the problem simply. Linus Torvalds puts it thus: "Given enough eyeballs, all bugs are shallow." – one person finds a problem, another person understands and fixes it.

Having more users also increases the number of concept suggestions. It is possible that someone may find a concept that you were too focussed to see. This concept may lead to a large evolutionary step in the program.

5. Open Source Lessons

Could open source help companies to create better products by allowing users to have greater feedback in the development of their products? Could open source teach the guidelines required for large-scale collaborations?

According to Eric Raymond, the answer would be a resounding "Yes". Supporters of Linux often talk about the world domination of their software as inevitable. This could be arrogance, but it is hard to disagree with the results – open source is often better (and cheaper) than its commercial counterparts.

Whether it succeeds or not, open source software has made huge economic contributions to the Internet. More often than not, it is the glue binding pieces of the net together (as in the case of Sendmail).

Many open source projects begin life as a programmer "scratching an itch". This itch may be specific to the programmer but when released to the world as a whole and modified slightly, it can be useful to many more people than originally imagined.

This evolution from an individual's scratch to the solution of a community's problem can push the technology in new directions. Tim Berners-Lee created the World-Wide-Web to help high-energy physicists share their work. Today, you can go to Africam web site (http://www.africam.co.za/) and see live pictures from an African game park.

From a social point of view, open source could be considered ethically more sound. Should access to useful programs be limited by wealth? With computers being used more in everyday life, access to software is becoming increasingly important. For example, the Web is proving to be more pivotal to doing business as time goes on. With the free server software Apache, a small company has as much chance to set up their own site and carry out e-commerce as a large one.

6. Mozilla - an open source trial

6.1 The Mozilla model

On the 31st of March 1998, Netscape released the first version of its open source browser Mozilla (http://www.mozilla.org/). Mozilla differs from other open source software in that it started life as a commercial product, Netscape Communicator. Netscape still uses in-house teams to build its browser, but also chooses the highest quality contributions from the open source community. In doing so, it hopes to get the best of both worlds – the quality assurance and documentation produced by traditional methods and the innovations and fast debugging of the open source community.

Not all open source projects are immediately successful, however. When Netscape decided to turn its browser open source, it hoped to gather the sort of support that Linux had mustered to push it forward. This did not initially happen and Netscape spent a frustrating year losing ground to Microsoft. Open source is not a magic bullet for the ails of failing software. Rather, it is an approach when properly implemented will lead to good reliable software. As with all methods, the rules have to be followed correctly to achieve the desired results.

When developing open source, it is important that users have a program that they can use and play with otherwise they will quickly lose interest. This has only been possible recently with Mozilla owing to various Netscape licensing problems. As such, there has been disillusionment and development stalled until these licences were circumvented.

There is great interest in how well Mozilla does – if it fails it could set the open source movement back many years. It is the first commercial product to embrace open source concepts and as such, it may prove useful as a model for this sort of development.

6.2 The need for open standards

Some believe that the damage has been done. Since Microsoft now has a browser monopoly it will become harder for Mozilla to gain market share. Open source advocates argue that software as important as an Internet browser must never be controlled by any one company. The browser could be used to control standards and eventually corrupt the open standards the Web is built on.

7. Problems with open source

7.1 Preconditions of open source

For open source to work, you have to follow certain preconditions. Your users must have software that they can easily run and modify (as Netscape found out to Mozilla's cost). It is also difficult to start an open source project from scratch – projects tend to go through a process of evolution rather than revolution. Finally, the co-ordinator must have the appropriate skills. The ability to motivate and communicate is crucial, as is the eye for spotting a good design.

7.2 Accessibility

One of the problems that the open source developers are trying to address is accessibility. Often, open source is developed by hackers for hackers. As such new users often have a hard time learning how to use the software. Steps are being taken to try and ease this problem (especially with Linux) but complexity is still rife.

7.3 Forking

A serious problem is forking of the source. This is where two (or more) sides cannot agree on a single solution so the source code actually splits into two (the fork). This can lead to two different versions of the same piece of software which are not compatible with each other (BSD Unix being a case in point). For a commercial developer, this would be disastrous.

7.4 Accountability

Accountability is an issue that is currently being grappled with. If open source medical software were to be developed, who would accept liability for bugs in the software? Would it be the student who wrote the code or the hospital for using it? In theory, the peer review and cheap cost offered by open source would be ideal for the medical world, but could open source software pay the cost of being sued? How many programmers would be interested enough to create and test the code? Is the fast release and testing cycle of open source suited to such situations?

7.5 Trojan horses

A common argument is that the insertion of a Trojan horse (a piece of code that masquerades under the guise of a legitimate program) is possible. However, this could equally be applied to closed software and is arguably more of a problem when fewer people can see the underlying source.

7.6 Money

"If open source so brilliant, why haven't all programmers given up their jobs and embraced it whole heartedly?"

As one person I talked to put it, programmers need money too. In western society capitalism is King and goods cost money to buy. Until such a time as that changes, programmers will have to take paying jobs like the rest of the population. If the only incentive to produce programs is for the greater good, far fewer programs will be developed.

8. Conclusion

I doubt that open source holds the key to the creation of all programs. The funding provided by companies has been a key factor in the success of the computer industry and I feel that, if this is withdrawn, much of the impetus gained over the past few years would be lost.

The benefits that open source provides should not be overlooked, however. Companies need to explore how its benefits can be absorbed into their own products by providing some sort of extra functionality. Support companies could charge for the added value whilst still maintaining links with an open source base. Alternatively, as in the case of Mozilla, you could invite the open source community to provide the extras whilst keeping control of the main source yourself. I think that this is especially important for manufactures of hardware peripherals, where sharing the driver would result in benefits for all.

If the Mozilla or Red Hat models can be shown to work over the next few years, it will provide a way to harness the best of both worlds. The support and shrink-wrapped packages of the closed source model could be combined with the benefits and rapid development of the open source community. Those programmers who were submitting the most useful code could be employed full time and thus help guide the direction of the software whilst receiving a reward for their work.

9. References

[1] Perens, Bruce. "The Open Source Definition." (Version 1.7)
The Open Source Initiative
http://www.opensource.org/osd.html
Visited: 17 November 1999

[2] Raymond, Eric S. "The Cathedral and the Bazaar."
Chapter 1. The Cathedral and the Bazaar
08 August 1999
http://www.tuxedo.org/~esr/writings/cathedral-bazaar/
Visited: 17 November 1999

10. Bibliography

Leonard, Andrew. "Life or Death Software."
Salon
08 May 1999
http://www.salon.com/tech/feature/1999/08/05/anesthesia/index.html
Visited: 17 November 1999

O'Reilly, Tim. "Lessons From Open- Source Software Development."
Communications of the ACM
April 1999, Vol. 44, No. 4 p33-37
http://www.acm.org/pubs/articles/journals/cacm/1999-42-4/p32-o_reilly/p32-o_reilly.pdf
Visited: 17 November 1999

Raymond, Eric S. "The Cathedral and the Bazaar."
August 1999
http://www.tuxedo.org/~esr/writings/cathedral-bazaar/
Visited: 17 November 1999

"Computer programming. Hackers rule."
The Economist
20 February 1999
http://www.economist.com/
Visited: 17 November 1999

"Netscape Communicator Open Source Code White Paper"
Netscape
http://home.netscape.com/browsers/future/whitepaper.html
Visited: 17 November 1999

"XEmacs Internals Manual - A History of Emacs."
http://w4.lns.cornell.edu/public/COMP/info/xemacs/internals/internals_1.html
Visited: 17 November 1999