# The Open Source Definition

by Bruce Perens

(selection)

http://perens.com/OSD.html

Michal Sroka
MTF STU
2007 / 2008

**The Open Source Definition**

*Bruce Perens*
*Treasurer, Open Source Initiative*
*This is the last draft that I submitted to the editor of O'Reilly's "Open Sources". The text is GPL-ed, and I am the copyright holder.*

The typical computer user owns lots of software that he bought years ago, and no longer uses today. He may have upgraded his computer or changed brands, and then the program wouldn't work any longer. The software might have become obsolete. The program may simply not do what he needs. He may have bought two or more computers, and doesn't want to pay for a second copy of the software. Whatever the reason, the software that he paid for years ago isn't up to the task today. Does that really need to happen?

What if you had the right to get a free upgrade whenever your software needed it? What if, when you switched from a Mac to a PC, you could switch software versions for free? What if, when the software doesn't work or isn't powerful enough, you can have it improved or even fix it yourself? What if the software was still maintained even if the company that produced it went out of business? What if you can use your software on your office workstation, and your home desktop computer, and your portable laptop, instead of just one computer? You'd probably still be using the software you paid for years ago. These are some of the rights that *Open Source* gives you.

The *Open Source Definition* is a bill of rights for the computer user. It defines certain rights that a software license must grant you to be certified as *Open Source*. Those who don't make their programs Open Source are finding it difficult to compete with those who do, as users gain a new appreciation of rights they always should have had. Programs like the *Linux* operating system and Netscape's web browser have become extremely popular, displacing other software with more restrictive licenses. Companies that use Open Source software have the advantage of its very rapid development, often by several collaborating companies, and much of it contributed by individuals who simply need an improvement to serve their own needs.

The volunteers who made products like *Linux* possible are only there, and the companies are only able to cooperate, because of the rights that come with Open Source. The average computer programmer would feel stupid if he put lots of work into a program, only to have the owner of the program sell his improvement without giving anything back. Those same programmers feel comfortable contributing to Open Source because they are assured of these rights:

- The right to make copies of the program, and distribute those copies.
- The right to have access to the software's source code, a necessary preliminary before you can change it.
- The right to make improvements to the program.

These rights are important to the software contributor because they keep all contributors at the same level relative to each other. Everyone who wants to is allowed to sell an Open Source program, so prices will be low and development to reach new markets will be rapid. Anyone who invests the time to build knowledge in an Open Source program can support it, and this provides users with the option of providing their own support, or the economy of a number of competing support providers. Any programmer can tailor an Open Source program to specific markets in order to reach new customers. People who do these things aren't compelled to pay royalties or license fees.

The reason for the success of this somewhat communist-sounding strategy, while the failure of communism itself is visible around the world, is that the economics of information are fundamentaly different from those of other products. There is very little cost associated with copying a piece of information like a computer program. The electricity involved costs less than a penny, and the use of the

equipment not much more. In comparison, you can't copy a loaf of bread without a pound of flour.

### *History*

The concept of *free software* is an old one. When computers first reached universities, they were research tools. Software was freely passed around, and programmers were paid for the act of programming, not for the programs themselves. Only later on, when computers reached the business world, did programmers begin to support themselves by restricting the rights to their software and charging fees for each copy. *Free Software* as a *political* idea has been popularized by Richard Stallman since 1984, when he formed the *Free Software Foundation* and its *GNU Project.* Stallman's premise is that people should have more freedom, and should appreciate their freedom. He designed a set of rights that he felt all users should have, an codified them in the *GNU General Public License* or *GPL.* Stallman punningly cristened his license *the Copyleft* because it leaves the right to copy in place. Stallman himself developed seminal works of free software such as the *GNU C Compiler*, and *GNU Emacs*, an editor so alluring to some that it is spoken of as it it were a religion. His work inspired many others to contribute free software under the GPL. Although it is not promoted with the same libertarian fervor, the Open Source Definition includes many of Stallman's ideas, and can be considered a derivative of his work.

The Open Source Definition started life as a policy document of the *Debian GNU/Linux Distribution.* Debian, an early Linux system and one still popular today, was built entirely of free software. However, since there were other licenses than the Copyleft that purported to be free, Debian had some problem defining what was free, and they had never made their free software policy clear to the rest of the world. I was the leader of the Debian project, at that time, and I addressed these problems by proposing a *Debian Social Contract* and a *Debian Free Software Guidelines* in July, 1997. Many Debian developers had criticisms and improvements that I incorporated into the documents. The *Social Contract* documented Debian's intent to compose their system entirely of free software, and the *Free Software Guidelines* made it possible to classify software into *free* and *non-free* easily, by comparing the software license to the guidelines.

Debian's guidelines were lauded in the free software community, especially among Linux developers, who were working their own free software revolution at the time in developing the first practical free operating system. When Netscape decided to make their web browser free software, they contacted Eric Raymond. Raymond is the Margaret Meade of free software: he has written several anthropological articles explaining the free software phenomenon and the culture that has grown around it, works that are the first of their kind and have shown a spotlight on this formerly little-known phenomenon. Netscape management was impressed with Raymond's essay *The Cathedral and the Bazaar,* a chronicle of a successful free software development using unpaid volunteer contributors, and asked him to consult, under a non-disclosure agreement, while they developed a license for their free software. Raymond insisted that Netscape's license comply with Debian's guidelines for it to be taken seriously as free software.

Raymond and I had met occassionally at the *Hacker's Conference,* a by-invitation-only gathering of creative and unconventional programmers. We had corresponded on various subjects via e-mail. He contacted me in February of 1997 with the idea for *Open Source.* Raymond was concerned that conservative business people were put off by Stallman's freedom pitch, which was, in contrast, very popular among the more liberal programmers. He felt this was stifling the development of Linux in the business world while it flourished in research. He met with business people in the fledgeling Linux industry, and together they concieved of a program to market the free software concept to people who wore ties. Larry Augustin of VA Research and Sam Ockman (who later left VA to form Penguin Computing) were involved, and others who aren't known to me.

Some months before *Open Source,* I had concieved of the idea of *Open Hardware,* a similar concept but for hardware devices and their interfaces rather than software programs. *Open Hardware* has not been as

successful as *Open Source* to date, but it is still operating and you can find information on it at http://www.openhardware.org/ .

Raymond felt that the *Debian Free Software Guidelines* were the right document to define *Open Source,* but that they needed a more general name and the removal of Debian-specific references. I edited the *Guidelines* to form *The Open Source Definition.* I had formed a corporation for Debian called *Software in the Public Interest*, and I offered to register a trademark for *Open Source* so that we could couple its use to the definition. Raymond agreed, and I registered a certification mark, a special form of trademark meant to be applied to other people's products, on the term. About a month after I registered the mark, it became clear that Software in the Public Interest might not be the best home of the Open Source mark, and I transferred ownership of the mark to Raymond. Raymond and I have since formed the *Open Source Initiative,* an organization exclusively for managing the Open Source campaign and its certification mark. At this writing, the Open Source Initiative is governed by a 6-person board chosen from well-known free software contributors, and seeks to expand its board to about 10 people.

At the time of its conception there was much criticism for the *Open Source* campaign, even among the *Linux* contingient who were already bought-in to the free software concept. Many pointed to the existing use of the term *Open Source* in the political intelligence industry. Others felt the term *Open* was already over-used. Many simply preferred the established name *Free Software.* I contended that the over-use of *Open* could never be as bad as the dual meaning of *Free* in the English language - either *liberty* or *price,* with price being the most oft-used meaning in the commercial world of computers and software. Richard Stallman later took exception to the campaign's lack of an emphasis on freedom, and the fact that as *Open Source* became more popular, his role in the genesis of free software, and that of his *Free Software Foundation* were being ignored - he complained of being "written out of history". This situation was made worse by a tendency for people in the industry to compare Raymond and Stallman as if they were proponents of competing philosophies rather than people who were using different methods to market the same concept. I probably exacerbated the situation by pitting Stallman and Raymond against each other in debates at *Linux Expo* and *Open Source Expo.* It became so popular to type-cast the two as advesaries that an email debate, never intended for publication, appeared the online journal *Salon.* At that point, I asked Raymond to tone down a dialogue that it had never been his intent to enter.

When the Open Source Definition was written, there were already a large number of products that fit the definition. The problem was programs that did not meet the definition, yet were seductive to users.

### *KDE, Qt, and Troll Tech*

The case of KDE, Qt, and Troll Tech is relevant to this chapter because the KDE group and Troll Tech tried to place a *non-Open-Source* product in the infrastructure of Linux, and met with unexpected resistance. Public outcry and the threat of a fully Open Source replacement for their product eventually convinced Troll to switch to a fully Open Source license. It's an interesting example of the community's enthusiastic acceptance of the Open Source Definition that Troll Tech *had* to make its license comply, if their product was to succeed.

KDE was the first attempt at a free graphical desktop for Linux. The KDE applications were themselves under the GPL, but they depended on a proprietary graphical library called *Qt*, from *Troll Tech. Qt's* license terms prohibited modification or use with any display software other than the senescent *X Window System.* Other use required a $1500 developer's license. Troll Tech provided versions of *Qt* for Microsoft Windows and the Macintosh, and this was its main revenue source. The pseudo-free license for *X* systems was meant to leverage the contributions of Linux developers into demos, examples, and accessories for their pricey Windows and Mac products. Although the problems with the *Qt* license were clear, the prospect of a graphical desktop for Linux was so attractive that many users were willing to overlook its non-Open-Source nature. Open Source proponents found KDE objectionable because they percieved that the KDE developers were trying blur the definition of what was free software to include partially-free

items like *Qt.* The KDE developers contended that their programs were Open Source, even though there were no runnable versions of the programs that did not require a non-Open-Source library. I, and others, asserted that KDE applications were only Open Source fragments of non-Open-Source programs, and that an Open Source version of *Qt* would be necessary before KDE could be referred to as Open Source. The KDE developers attempted to partially address the problem of *Qt's* license by negociating a *KDE Free Qt Foundation* agreement with Troll Tech, in which Troll and KDE would jointly control releases of the free version of *Qt,* and Troll Tech would release *Qt* under an Open-Source-complaint license if the company was ever purchased or went out of business. Another group initiated the *GNOME* project, a fully Open Source competitor of KDE that aimed to provide more features and sophistication, and a separate group initiated a *Harmony* project to produce a fully Open Source clone of *Qt* that would support KDE. As GNOME was being demonstrated to accolades and Harmony was about to become useful, Troll Tech realized *Qt* would not be successful in the Linux market without a chance in license. Troll Tech released a fully Open Source license for *Qt*, defusing the conflict and removing the motivation for the Harmony project. The GNOME project continues, and now aims to best KDE in terms of functionality and sophistication rather than its license.

Before they released their new Open Source license, Troll Tech provided me with a copy for auditing, with the request that it be kept confidential until they could announce it. In my enthusiasm to make peace with the KDE group and an embarassing feat of self-deception, I pre-announced their license eight hours early on a KDE mailing list. That email was almost immediately picked up by *Slashdot* and other online news magazines, to my chagrin.

Troll Tech's new license is notable in that it takes advantage of a loophole in the Open Source Definition that allows patch files to be treated differently from other software. I would like to address this loophole in a future revision of the Open Source Definition, but the new text should not place Qt outside of Open Source.

At this writing, proponents of Open Source are increasing exponentially. The recent Open Source contributions of IBM and Ericsson have been in the headlines. Two Linux distributions, *Yggdrasil* and *Debian,* are distributing complete Linux system distributions, including many applications, that are entirely Open Source, and several others including *Red Hat* are very close. With the completion of the *GNOME* system, an Open Source GUI desktop OS capable of competing with *Microsoft NT* will have been realized.

### *Analysis of the Open Source Definition*

In this section, I'll present the entire text of the Open Source Definition, with commentary. You can find the canonical version of the Open Source Definition at http://www.opensource.org/osd.html .

Pedants have pointed out minor ambiguities in the Open Source Definition. I've held off revising it as it's little more than a year old and I'd like people to consider it *stable.* The future will bring slight language changes, but only the most minor of changes in the *intent* of the doucment.

The Open Source Definition

(Version 1.0)

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

*Note that the Open Source Definition is not itself a software license. It is a specification of what is permissible in a software license for that software to be referred to as Open Source. The Open Source Definition was not intended to be a legal document. The inclusion of the Open Source Definition in*

*software licenses, such as a proposed license of the Linux Documentation Project, has tempted me to write a more rigorous version that would be appropriate for that use.*

*To be Open Source, all of the terms below must be applied* ***together, and in all cases.*** *For example, they must be applied to derived versions of a program as well as the original program. It's not sufficient to apply some and not others, and it's not sufficient for the terms to only apply some of the time. After working through some particularly naive' interpretations of the Open Source Definition, I feel tempted to add* ***this means you!***

1. Free Redistribution

The license may 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 may not require a royalty or other fee for such sale.

*This means that you can make any number of copies of the software, and sell or give them away, and you don't have to pay anyone for that privilege.*

*The "aggregate software distribution containing programs from several different sources" was intended to fit a loophole in the* ***Artistic License****, a rather sloppy license in my opinion, originally designed for Perl. Today, almost all programs that use the Artistic License are also available under the GPL. That provision is thus no longer necessary, and may be removed from a future version of the Open Source Definition.*

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 downloading the source code, without charge, via the Internet. 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.

*Source code is a necessary preliminary for the repair or modification of a program. The intent here is for source code to be distributed with the initial work, and all derived works.*

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.

*Software has little use if you can't maintain it (fix bugs, port to new systems, make improvements), and modification is necessary for maintainance. The intent here is for modification of any sort to be allowed. It must be* ***allowed*** *for a modified work to be distributed under the same license terms as the original work. However, it is not* ***required*** *that any producer of a derived work* ***must*** *use the same license terms, only that the option to do so be open to them. Various licenses speak differently on this subject - the BSD license allows you to take modificiations private, while the GPL does not.*

*A concern among some software authors is that this provision could allow unscrupulous people to modify their software in ways that would embarass the original author. They fear someone deliberately making the software perform incorrectly in a way that would make it look as if the author was a poor programmer. Others are concerned that software could be modified for criminal use, by the addition of* ***trojan horse*** *functions or localy-banned technologies such as cryptography. All of these actions, however, are covered by* ***criminal law.*** *A common misunderstanding about software licenses is that they must specify* ***everything****, including things like "don't use this software to commit a crime". However, no license has*

*any valid existence outside of the body of civil and criminal law. Considering a license as something apart from the body of applicable law is as silly as considering an English-language document as being apart from the dictionary, in which case none of the words would have any defined meaning.*

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.

*Some authors were afraid that others would distribute source code with modifications that would be percieved as the work of the original author, and would reflect poorly on that author. This gives them a way to enforce a separation between modifications and their own work without prohibiting modifications. Some consider it un-esthetic that modifications might have to be distributed in a separate "patch" file from the source code, even though Linux distributions like Debian and Red Hat use this procedure for all of the modifications they make to the programs they distribute. There are programs that automaticaly merge patches into the main source, and one can have these programs run automaticaly when extracting a source package. Thus, this provision should cause little or no hardship.*

*Note also that this provision says that in the case of patch files, the modification takes place* ***at build-time****. This loophole is employed in the Qt Public License to mandate a different, though less restrictive, license for the patch files, in contradiction of section 3 of the Open Source Definition. There is a proposal to clean up this loophole in the definition while keeping Qt within Open Source.*

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.

*This means that Netscape, for example, can insist that only they can name a version of the program* ***Netscape Navigator(tm)*** *while all free versions of the program must be called* ***Mozilla*** *or something else.*

5. No Discrimination Against Persons or Groups.

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

*A license provided by the Regents of the University of California, Berkeley, prohibited an electronic design program from being used by the police of South Africa. While this was a laudable sentiment in the time of apartheid, it makes little sense today. Some people are still stuck with software that they acquired under that license, and their derived versions must carry the same restriction. Open Source licenses may not contain such provisions, no matter how laudable their intent.*

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.

*Your software must be equally usable in an abortion clinic, or by an anti-abortion organization. These political arguments belong on the floor of congress, not in software licenses. Sone people find this lack of discrimination extremely offensive!*

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.

*The license must be automatic, no signature required. Unfortunately, there has not been a good court test in the U.S. of the power of a no-signature-required license when it is passed from a second party to a third. However, this argument considers the license in the body of* contract law*, while some argue that it should be considered as* copyright law*, where there is more precedent for no-signature licenses. A good court test will no doubt happen in the next few years, given the popularity of this sort of license and the booming nature of Open Source.*

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.

*This means you can't restrict a product that is identified as Open Source to only be free if you use it with a particular brand of Linux distribution, etc. It must remain free if you separate it from the software distribution it came with.*

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.

*A version of GhostScript (a PostScript rendering program) requires that the media on which it is distributed contain* only *free software programs. This isn't permissible for Open Source licenses. Fortunately, the GhostScript author distributes another (somewhat older) version of the program with a true Open Source license.*

*Note that there is a difference between* ***derivation*** *and* ***aggregation.*** *Derivation is when a program actually incorporates part of another program into itself. Aggregation is when you include two programs on the same CD-ROM. This section of the Open Source Definition is concerned with aggregation,* ***not*** *derivation. Section 4 is concerned with derivation.*

10. Example Licenses.

The GNU GPL, BSD, X Consortium, and Artistic licenses are examples of licenses that we consider conformant to the Open Source Definition. So is the MPL.

*This would get us in trouble if any of these licenses are ever changed to be non-Open-Source - we'd have to issue a revision of the Open Source Definition immediately. It really belongs in explanatory text, not in the Open Source Definition itself.*