A history of FOSS law and licensing

author:[Engelfriet,Arnoud]

Surprisingly, programmable computers are older than copyright law: Charles Babbage's analytical engine of 1837 predates the Berne Convention for the Protection of Literary and Artistic Works (1886). However, unlike the Convention, Babbage's mechanical contraption designed to perform mathematical calculations never became a real-world success. Several special-purpose machines, such as the Hollerith tabulating machine, did sell on the market in the late 19th and early 20th century. The first computers in the modern sense only became available in the 1950s, thanks to ground-breaking research by John von Neumann and Alan Turing in particular.

Companies such as IBM that provided these computers as part of what we today would call "solutions": businesses were provided with specially-programmed computers to support certain business activities. Customers would pay for continued support and new programs as part of the deal, but no one would consider buying or selling computer programs by themselves.[1] Users would get together to share their in-house software developments with others, eventually leading to the building of pools of software. This was actually encouraged by most vendors, as it stimulated the sale of hardware that could use this software. One might thus say that the idea of free and open source is as old as the computing business itself.[2]

Age of the shrink-wrap

New developments in computing power over the next decade caused a fundamental shift away from this model. A landmark event occurred in 1969 when IBM unbundled its hardware and software activities, effectively giving birth to the software industry as such.[3] A further stimulus for this market came from the personal computer, a low-cost alternative to the "big iron" mainframe computers that were mainly in use at the time. In the early 1970s, several do-it-yourself kits became available that allowed hobbyists to build their own personal computers, and in 1977 the Commodore PET and the Apple II appeared on the market as the first ready-to-use computers.[4] With hardware and software products separated, software as such was ready to develop as a market. But many

---

[1]Martin Campbell-Kelly, "Development and Structure of the International Software Industry, 1950-1990", Business and Economic History 24/2 (1995): 73-110. Also see Richard M. Stallman, "The GNU Operating System and the Free Software Movement", in: Open Sources: Voices from the Open Source Revolution, O'Reilly 1999.

[2]Steven Levy, Hackers: Heroes of the Computer Revolution, Doubleday 1984. Also see Peter Salus, The Daemon, The Gnu, And The Penguin, Reed Media Services 2008.

[3]Burton Grad, "A Personal Recollection: IBM's Unbundling of Software and Services", IEEE Annals of the History of Computing, vol. 24, no. 1, pp. 64-71, Jan.-Mar. 2002 and Thomas W. Hamilton, "IBM's unbundling decision: Consequences for users and the industry", Programming Sciences Corporation, 1969.

[4]Paul E. Ceruzzi, A History of Modern Computing, MIT Press 1998.

software vendors were concerned about the viability of this market, as it was unclear which legal protection they would receive.

Debate among lawmakers and lobbyists led to various proposals, ranging from copyright and patent law to a *sui generis* right specifically designed to protect software. After much debate, a consensus emerged that copyright would be the most appropriate legal means.[5] The USA was the first to adopt copyright protection for software.[6] The European Economic Community followed in 1991 with its Software Directive,[7] which was largely based on the US law but with more liberal provisions on reverse engineering. With copyright protection firmly in place, software vendors were able to sell their software "as a book" (to quote from the Borland software licenses) by charging royalties for each copy sold.[8] This created a large market for software, sometimes called the "Age of the Shrink-wrap" because of the way that software was made available: in large, shrink-wrapped boxes.

Software distributors did something peculiar in this market: while software was offered "as a book", they did not actually sell the software. They included long legal documents that solemnly declared that the software was merely licensed and that the user had no rights other than as provided in this license. The legal validity of these licenses has been hotly disputed but today seems to be grudgingly accepted in the general case.[9] Two areas of particular attention in these licenses are the restriction of statutory rights, such as the right to reverse engineer the software, and the disclaimer of all warranties and liabilities.

The right to reverse engineer is a peculiar one. As a general rule, buyers of a product are expected and permitted to examine what they buy and to discover how it works. Copyright (or patent) law may restrict the building and selling

---

[5] "Legal Protection for Computer Programs: a Survey and Analysis of National legislation and Case Law" by Michael S. Keplinger (document UNESCO/WIPO/GE/CCS/2). Also see the Model provisions on the protection of computer software, prepared by the International Bureau of the World Intellectual Property Organization, 1978.

[6] Computer Software Copyright Act of 1980, following the Final Report of the National Commission on New Technological Uses of Copyrighted Works (CONTU) of 31 July 1978. Also see Mark A. Lemley et al., Software And Internet Law, Aspen Law & Business 2000, pp. 34-35.

[7] Council Directive 91/250/EEC of 14 May 1991 on the legal protection of computer programs.

[8] Thom Holwerda, "Borland in the 1980s: 'Treat Software Just Like a Book'", OSNews 15 oktober 2009, http://www.osnews.com/story/22342/Borland_in_the_1980s_ Treat_Software_Just_Like_a_Book_.

[9] See e.g. H. Ward Classen, A practical guide to software licensing for licensees and licensors, American Bar Association, Section of Business Law 2007, p. 149-152, Lawrence D. Graham, Legal battles that shaped the computer industry, Greenwood Publishing Group 1999, p. 119-125, Clive Gringras, "The Validity of Shrink-Wrap Licences", International Journal of Law and Information. Technology 4(2), p. 77-111, R.H. Stern, "Shrink-wrap license restrictions-preempted?", IEEE Micro 17(1), Jan/Feb 1997, p. 75-78 and David L. Hayes, "The enforce-ability of shrink-wrap license agreements on-line and off-line", Fenwick & West, March 1997. The most famous case was ProCD, Inc. v. Zeidenberg, 86 F.3d 1447 (7th Cir. 1996), in which the US appeals court held that a shrink-wrap license was valid and enforceable as a contract, provided they do not otherwise violate generally accepted principles of contract law.

of cloned products, but the investigation itself is not restricted. For software, these licenses would put this expectation on its head. No one was permitted to discover how the licensed software works — not even if a compatibility issue or a serious bug arose. This gave the licensor a very comfortable position, as he could now charge maintenance fees for any and all work to be done on the software. Legislators did try to introduce at least some right to reverse engineer software for compatibility purposes, but these clauses are generally regarded as weak and risky to invoke.[10]

Free software and the GPL

As soon as software by itself started to become a viable market, access to software in its human-readable source code form was becoming more and more difficult. Software companies regarded their source code a treasured business secret (in today's terms: their intellectual property) and severely restricted its access and distribution to third parties. For famous MIT hacker Richard M. Stallman, this "hoarding" of software was simply unacceptable. Software should be freely available to all, and its programmers should not be able to set legal or other restrictions on other people's use of the code.[11] In 1983, Stallman quit MIT and founded what would become the Free Software Foundation (FSF), an organisation dedicated to creating and stimulating software that would be freely available to all.

To help achieve this goal Stallman drafted the *GNU General Public License* or GPL, a software license that allows anyone to freely use, distribute and adapt the licensed software at no charge. The only restriction he added was that any adapted or extended version of the software could only be distributed under the GPL as well. This ensured the continued availability of the source code to anyone who came in contact with the software. With this license, he turned copyright on its head — an early name for the model is *copyleft*.

Fifteen years later, Stallman's work was shown to be a resounding success. Almost single-handedly he had written most of the software needed to program and use a general-purpose computer. Only one significant part was missing: the *kernel*, the key part of a computer operating system that controls all software and interfaces with the underlying hardware. Operating systems are expensive and difficult to develop, but in 1991 a Finnish second-year computer science student announced he was going to create one as "just a hobby, won't be big and professional". His creation *Linux* (mixing his first name Linus and Unix according to hacker tradition) would become one of the most valuable pieces of

---

[10]In Europe, Directive 91/250/EEC provides a right to decompile if that is "indispensable to obtain the information necessary to achieve interoperability" but sets limitations so severe it is hard to come up with a real-world case that evidently would prevail in court. Also see Blaney Harper and Vaishali Udupa, "Drafting Electronic Software Licenses to Prevent Reverse Engineering", Jones Day February 2004.

[11]Richard M. Stallman, "The GNU Operating System and the Free Software Movement", in: Open Sources: Voices from the Open Source Revolution, O"Reilly 1999. Also see Sam Williams, Free as in Freedom: Richard Stallman's Crusade for Free Software, O'Reilly 2002.

software ever written. Torvalds chose to use the GPL, calling it "one of the very best design decisions" he ever made.[12]

Legal entanglements around BSD

If not for a legal battle around free software elsewhere, Linux might not have been. The Berkeley Software Distribution (BSD) was a collection of extensions, patches and add-ons to the Bell Labs Unix operating system created at Berkeley University. Based on the principles of free and open source software, the BSD had already been available since about 1978. Over the years, BSD morphed into a complete operating system, culminating in the so-called Networking Release 2 in June 1991. All code was available under the BSD license, allowing anyone to use it provided credit was given to the original authors.

AT&T, owners of the Unix copyrights at the time, felt that this release infringed on its copyrights and filed a lawsuit against the university.[13] In a counterclaim, Berkeley alleged that AT&T had used BSD code without adhering to its license, as credits and copyright notices had been removed.[14] The suit dragged on until a settlement was obtained in 1994. By that time however the development of BSD had slowed down significantly, allowing Linux to gain developer attention. Torvalds has observed he would not have created Linux if the BSD operating system had been available to him at the time.[15]

The rise of open source

With the Linux kernel available, the free software movement finally was in a position to make significant growth and maybe overtake the "proprietary" world. However, it only had limited success. Over the years, the FSF had developed a rather confrontational attitude towards companies that did not subscribe to its ideas, which made companies hesitant to adopt free software.[16] A related complication was that the GPL was often misunderstood by lawyers, creating a false belief that using any piece of GPL software in one's product would require the "freeing" of the entire software stack.

In 1998 Netscape Communications announced that it would release the source code of its Web browser. This spurred a group of prominent free software developers to promote free software principles under the newly-coined term "open source": collaboration between programmers world-wide who jointly improve

---

[12]Robert Young, "Interview with Linus, the Author of Linux", Linux Journal, 1 March 1994.

[13]Greg Lehey, The complete FreeBSD: documentation from the source, O'Reilly 2003, pp. 8-9.

[14]Marshall Kirk McKusick, "Twenty Years of Berkeley Unix", in: Open Sources: Voices from the Open Source Revolution, O'Reilly 1999.

[15]Mike Linksvayer, "The Choice of a GNU Generation — An Interview With Linus Torvalds", Meta magazine 1993. http://gondwanaland.com/meta/history/interview.html.

[16]Bruce Perens, "The Open Source Definition", in: Open Sources: Voices from the Open Source Revolution, O'Reilly 1999.

software in a way that no individual company could achieve.[17] The Open Source Initiative (OSI) was subsequently founded as a public benefit corporation, which amongst other things offers a certification program for open source software licenses.[18] Their trademark application on the term "open source" was refused. Certified licenses now wear the label "OSI Certified Open Source".

The choice for the term "open source" has not been well received by the free software community. The main point of criticism is that "free software" focuses on freedoms for all users, while "open source" waters down the focus to just community-driven development.[19] In recent years the term *free and open source software* or FOSS has arisen as a neutral alternative, a practice which this law book also adheres to. A lesser-used variant is *free/libre/open source software* (FLOSS), which uses "libre" to clarify the meaning of free as in "freedom" rather than as in "at no charge".

Legal validity

There are hundreds of FOSS licenses, although they can be grouped roughly into three categories. The first category is often referred to as the *academic licenses*, which basically require that credit be given and the authors are shielded from warranty claims. The second category provides *copyleft* — the requirement that source code be made available to users, including derivative works. The third category is somewhat of a compromise: *weak copyleft* requires that source code of the software itself (and modifications to it) are shared but larger works may be kept proprietary. Most FOSS today is licensed under the GPL, the canonical copyleft license.[20] The BSD license is the most popular academic license. Netscape used the weak-copyleft Mozilla Public License when releasing its browser in 1998.

Much has been written about the legal validity of FOSS licenses in various jurisdictions.[21] Legal debate focuses mostly on the copyleft provisions: what is the scope of a "derivative work" and how far can a license extend the obligation of relicensing under a copyleft license? Other legal areas of contention are the limitations of liability and the interpretation of rights and obligations under international law. However, the number of lawsuits worldwide challenging these licenses can be counted on the fingers of two hands. The aforementioned lawsuit around BSD may well be regarded as the first, at least in spirit.

---

[17]Eric S. Raymond, "Goodbye, 'free software'; hello, 'open source'", 16 June 2007 http://www.catb.org/~esr/open-source.html. Also see Eric S. Raymond, The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary, O'Reilly 2001.

[18]History of the OSI, http://www.opensource.org/history.

[19]Richard M. Stallman, Why "Open Source" misses the point of Free Software, GNU.org 24 September 2007, http://www.gnu.org/philosophy/open-source-misses-the-point.html.

[20]Top 20 Most Commonly Used Licenses in Open Source Projects, Black Duck Open Source Resource Center, April 2011.

[21]Lawrence Rosen, Open Source Licensing, Prentice Hall PTR, 2005, Andrew M. St. Laurent, Understanding open source and free software licensing, O'Reilly 2004 and of course the very book you are reading now.

In 2002, it looked like the GPL would have its day in court when the GPL-licensed MySQL database became subject of a lawsuit between owner MySQL AB and the US software company Progress.[22] Progress had allegedly created a derivative work of MySQL by adding support for its own database format. However, after MySQL obtained a preliminary injunction, the parties settled before the question could be addressed by the court.

The Software Freedom Law Center has filed several lawsuits against misappropriation of GPL code. Several cases have been filed focusing on Busybox, starting with *BusyBox vs Monsoon* in 2007.[23] However, all cases so far have been settled out of court. Few others have brought cases. Two notable examples are *SCO vs. IBM* and *Wallace vs. FSF*. The less said about those legal disasters the better.[24]

An important milestone was the 2008 JMRI lawsuit, focusing on model railroad train software licensed under the little-used "Artistic License". In a much-welcomed ruling, the US Federal appeal court ruled that the principle of open source licensing was a valuable goal that copyright law ought to support.[25] In addition, the court held that the source code sharing obligations were limitations of the copyright license and not mere covenants to the license. This distinction is important, as violating a license limitation allows the full force of copyright law to be used against the licensee. A mere covenant has to be addressed as a contract breach.[26]

In Germany, Linux kernel hacker Harald Welte did manage to achieve several legal successes against companies using his GPL-licensed netfilter software. Notably, Welte obtained injunctions against D-Link, Fortinet and SMC (Skype).[27] Subsequently, German courts had little trouble accepting the GPL as a legally

---

[22]Progress Software Corp. v. MySQL AB, Civil Action No. 01-11031 PBS, filed on June 15, 2001.

[23]"On Behalf of BusyBox Developers, SFLC Files First Ever U.S. GPL Violation Lawsuit", Software Freedom Law Center 20 September 2007. Other cases brought involved Xterasys, High-Gain Antennas and Verizon Communications. Compare Eben Moglen, "Enforcing the GNU GPL", GNU.org 10 September 2001.

[24]SCO vs. IBM was filed in 2003 as a $1 billion lawsuit alleging that IBM had somehow harmed SCO by contributing to Linux. Particular allegations involved copyright claims on number codes for Linux interfaces and the unconstitutionality of the GPL. Wallace vs. FSF (467 F.3d 1104, 7th Cir. 2006) alleged the GPL was a form of illegal price fixing. His pro se lawsuit was thrown out after the plaintiff failed to state an actionable claim despite three chances to do so.

[25]Jacobsen v. Katzer, 535 F.3d 1373 (Fed. Cir. 2008).

[26]This touches upon the legal question whether a FOSS license is to be regarded as a contract, as a bare license, as a declaration of promissory estoppel or another legal construct. See e.g. Pamela Jones, "The GPL is a license, not a contract", LWN.net 3 December 2003, http://lwn.net/Articles/61292/, Eben Moglen, "Enforcing the GNU GPL", GNU.org 10 September 2001, Lawrence Rosen, Open Source Licensing, Prentice Hall PTR, 2005 , page 53-54 and Axel Metzger and Till Jaeger, "Open Source Software and German Copyright Law", IIC Vol. 32, 2001, p. 52.

[27]J. Höppner, "The GPL prevails: An analysis of the first-ever Court decision on the validity and effectivity of the GPL", (2004) 1:4 SCRIPT-ed 628.

binding license agreement and enjoining those who did not adhere to its terms.[28] Welte founded the *GPL-violations.org* project in January 2004 to raise public awareness of the infringing use of free software.[29] The project has reported numerous successful settlements and a 100% success rate in enforcing its licenses.

Towards the future

FOSS shows no sign of slowing down. Linux is widely in use in embedded environments (mobile phones, televisions, cars, robots) and the open source Firefox web browser is the most popular alternative to Microsoft's Internet Explorer. Most of the infrastructure of the Internet runs on open source software.

Licensing-wise, FOSS is here to stay. There is consensus that the principles of FOSS are legally sound. FOSS licensing is also being recognized as valuable by policymakers. Lawsuits in the coming years may address some of the open issues, such as the scope of derivative works and liability for FOSS developers. More is to be expected from voluntary compliance and a growing maturity in how the legal community approaches FOSS licensing principles. We have come a long way already.

---

[28]Most recently, the Regional Court of Hamburg found FANTEC GmbH guilty of violating the GPL in their media player. Notably, Fantec's reliance on its suppliers was deemed insufficient. LG Hamburg 14.06.2013, AZ 308 O 10/13, http://www.ifross.org/sites/default/files/130618%20Urteil%20Fantec.pdf

[29]About the gpl-violations.org project, http://gpl-violations.org/about.html#history