Santa Clara University School of Law

Santa Clara High Technology Law Journal

Volume 29 | Issue 3 Article 4

4-23-2013

Open Source Software Compliance: The Devil is Not So Black As He is Painted

Maxim V. Tsotsorin

COMMENT

OPEN SOURCE SOFTWARE COMPLIANCE: THE DEVIL IS NOT SO BLACK AS HE IS PAINTED

Maxim V. Tsotsorin†

Abstract

Many commercial enterprises effectively utilize open source code when developing various software products—virtually every software developer uses open source in his or her work. But along with economic benefits and production efficiency come significant legal risks, exacerbated by the wide availability of OSS components. While some licenses are permissive and demand very little, others require any work based on, or even containing only parts of an open source code, to be distributed only as OSS.

Most commercial enterprises and software developers recognize potential business and legal risks and implement some sort of compliance mechanism as a best practice. But what should the enterprise do if its software developer either intentionally or inadvertently incorporates open source code? Can one remedy such a situation? What are the chances that the licensor will actually enforce the license requirements? And if the company decides to comply, what does compliance then entail?

This Comment, in an attempt to answer these questions, concludes that the risks associated with OSS, although not minimal, are generally known and an effective toolset to prevent intermixing of open source code with closed code is available. If the violation nevertheless occurs, there are steps a business could take to either remedy the violation or comply with the licensing requirements.

† J.D Candidate, Santa Clara University School of Law, May 2013; B.Juris., Kaliningrad State University, Kaliningrad, Russia. I would like to thank Professor Anna Han, Santa Clara University School of Law, for her guidance in initial crafting of this Comment, and open source guru Heather J. Meeker for the extremely helpful comments and suggestions. Thank you to the entire editorial board of the Computer and High Technology Law Journal for their hard work and thoughtful edits. I would also like to thank my family for their unconditional support in all my endeavors.

TABLE OF CONTENTS

I. INTRODUCTION

It is no longer the predominant view that open source software (OSS)[1] and proprietary software are mutually exclusive.[2] Many commercial enterprises effectively utilize open source code[3] when developing various software products. This "mixed-source" software model reduces development costs and times, thus improving the return on investment and overall productivity in developing a product. Incorporating virtually free and available-for-all code into proprietary software avoids the unnecessary work of "reinventing the wheel" because developers do not have to develop the code from scratch, which may be both costly and time consuming.[4] While some studies place the percentage of software developers who regularly use open source code in their work at around ninety percent,[5] it is probably safe to assume that the actual number is much closer to a hundred percent:

---

1. For the purposes of this discussion, "open source software" (OSS) also includes "free software," which is similar, but not identical, to OSS. The differences between "free software" and "open source software" are described in Richard Stallman, *Why Open Source Misses the Point of Free Software*, GNU OPERATING SYS., http://www.gnu.org/philosophy/open-source-misses-the-point.html (last updated Mar. 31, 2013). *See also* Christian H. Nadan, *Open Source Licensing: Virus or Virtue?*, 10 TEX. INTELL. PROP. L.J. 349, 353-55 (2002). "Proprietary software" is software that is subject to licenses that typically do not grant access to the software's source code. *See* Jim Markwith, *The Coexistence of Open Source and Proprietary Software*, 954 PLI/Pat 227, *in* OPEN SOURCE SOFTWARE 2008: BENEFITS, RISKS AND CHALLENGES FOR SOFTWARE USERS, DEVELOPERS AND INVESTORS 227, 236 n.8 (2008). In this Comment, the author uses the term "OSS" to create a relatively bright—although an arbitrary—line that separates free and open source software from proprietary ("closed source") software, as well as any applicable licenses.

2. *See generally* T. Robert Rehm, Jr., *Navigating the Open Source Minefield: What's a Business to Do?*, 10 WAKE FOREST INTELL. PROP. L.J. 289, 313-21 (2010).

3. Source code is written by computer programmers to direct computers to perform specific tasks. OSS source code is available (i.e., "open") so that anyone can change it. Proprietary software is typically distributed without its source code being made available (and therefore it is "closed"). *See generally id.*; *see also* LAWRENCE ROSEN, OPEN SOURCE LICENSING: SOFTWARE FREEDOM AND INTELLECTUAL PROPERTY LAW 1-3 (2005) [hereinafter ROSEN ON OPEN SOURCE], *available at* http://www.rosenlaw.com/oslbook.htm.

4. *See* Kirk D. Rowe, Comment, *Why Pay for What's Free? Minimizing the Patent Threat to Free and Open Source Software*, 7 J. MARSHALL REV. INTELL. PROP. L 595, 607 (2008) ("Without the need to reinvent the wheel, programmers typically patch hundreds or even thousands of pre-existing programs and algorithms together in such a way as to produce a novel result."); Michael Kozubek, *Code Conduct: Open-Source Software License Provisions Can Jeopardize IP Assets*, INSIDE COUNS., Dec. 2012, at 32, 32, *available at* http://www.insidecounseldigital.com/insidecounsel/201212?pg=36#pg32 ("[C]ompare[] software developers' use of OSS to 'lawyers who never would start a contract draft from scratch and always borrow parts and pieces from other places. Why reinvent the wheel?'" (quoting Matt Jacobs, Corporate Counsel of Black Duck Software)).

5. *See, e.g.*, *Developer Survey Results Announced by Outercurve Foundation*, PR NEWSWIRE (Dec. 14, 2011), http://www.prnewswire.com/news-releases/developer-survey-results-announced-by-outercurve-foundation-135574343.html.

virtually every software developer uses open source in his or her work.[6]

But along with economic benefits and production efficiency come significant legal risks, exacerbated by the wide availability of OSS components.[7] Open source software source code is usually made available under a generally applicable copyright-based license to use, modify and distribute the software.[8] In return, the licensee usually must comply with certain requirements,[9] such as providing required copyright notices and making the source code available to others.[10] While some licenses are permissive and demand very little, others require any work based on, or even containing only parts of an open source code, to be distributed only as OSS.[11] This variation of copyright[12]—with its departure from the traditional right to restrict

---

6. *See* E-mail from Heather J. Meeker, Chair, IP/IT Licensing and Transactions Group, Greenberg Traurig, LLP, to author (Dec. 17, 2012, 12:57 PM) (on file with author).

7. *See OpenLogic Scan Shows Open Source License Violations for iPhone and Android: More Than 70%+ of Mobile Applications Containing Open Source Fail to Comply*, OPENLOGIC (Mar. 8, 2011), http://www.openlogic.com/news/bid/154650/OpenLogic-Scan-Shows-Open-Source-License-Violations-for-iPhone-and-Android-More-Than-70-Of-Mobile-Applications-Containing-Open-Source-Fail-to-Comply. The study found that seventy-one percent of scanned Android and iPhone apps containing OSS components failed to comply with the following key obligations: (1) to provide or to offer to provide the source code under General Public License (GPL)/Lesser General Public License (LGPL), (2) to provide a copy of the license under GPL/LGPL and Apache licenses, (3) to provide notices and attributions under Apache license. *Id.*

> Out of the 635 apps scanned, OpenLogic identified 52 applications that use the Apache license and 16 that use the GPL/LGPL license.
>
> OpenLogic found that among the applications that use the Apache or GPL/LGPL licenses, the compliance rate was only 29%. Android compliance was 27% and iPhone/iOS compliance was 32%. Overall compliance of Android applications using the GPL/LGPL was 0%.

Id.

8. *See* Greg R. Vetter, *"Infectious" Open Source Software: Spreading Incentives or Promoting Resistance?*, 36 RUTGERS L.J. 53, 57 (2004).

9. *See id.*

10. *Id.* at 84 n.71.

11. *See generally* Richard E. Fontana, *Open Source License Enforcement and Compliance*, 989 PLI/Pat 77, *in* OPEN SOURCE AND FREE SOFTWARE 2009: BENEFITS, RISKS AND CHALLENGES IN TODAY'S ECONOMIC ENVIRONMENT 77, 81-91 (2009). Different types of OSS licenses and their characteristics are discussed in greater detail *infra* Part II.

12. This Comment only discusses copyright protection of OSS because it is the primary enforcement mechanism. *See* David McGowan, *Legal Aspects of Free and Open Source Software*, *in* PERSPECTIVES ON FREE AND OPEN SOURCE SOFTWARE 361, 362-63 (Joseph Feller et al. eds., 2005) (describing a copyright infringement action as a "powerful enforcement tool" for open source licensors). Although software is generally patentable, *see* Diamond v. Diehr, 450 U.S. 175 (1981), *In re* Alappat, 33 F.3d 1526 (Fed. Cir. 1994), patents are rarely used by open source software developers, both for ideological and practical reasons. *See* David S. Evans & Anne Layne-Farrar, *Software Patents and Open Source: The Battle over Intellectual Property*

use of copyrighted material[13]—has been dubbed "copyleft"[14] by the Free Software Foundation (FSF)[15] to emphasize the unique nature of some OSS licenses that do not restrict use of copyrighted material, but rather promote it.[16]

The requirement to make the source code publically available when proprietary software contains copyleft-covered source code prompted some to name such licenses "viral" or "infectious."[17] For a software enterprise developing proprietary programs, such intermixing essentially presents the enterprise with a Hobson's choice: either to comply with the OSS licensing requirements and possibly lose valuable rights to intellectual property—rights in the parts of the software that include proprietary code—or possibly face a copyright infringement lawsuit.[18] Therefore, it is extremely important to know which license covers OSS or specific source code being used

Rights, 9 VA. J.L. & TECH. no. 10, at 11-27 (2004) (summarizing OSS community arguments against patents). Trade secret protection is unavailable for OSS code by definition—source code that is open and freely available to the public is not secret; related development procedures and techniques, however, may be protected. *See* VAN LINDBERG, INTELLECTUAL PROPERTY AND OPEN SOURCE: A PRACTICAL GUIDE TO PROTECTING CODE 130-31 (2008). In addition, breach of contract theory is a viable remedy, but is limited in comparison with copyright protection. *See, e.g.*, ROSEN ON OPEN SOURCE, *supra* note 3, at 278-82; Rehm, *supra* note 2, at 291 n.5.

13. *See* U.S. Copyright Act, 17 U.S.C. § 106 (2011); *see also* BLACK'S LAW DICTIONARY 386 (9th ed. 2009) (defining "copyright" as a "property right in an original work of authorship . . . fixed in any tangible medium of expression, giving the holder the exclusive right to reproduce, adapt, distribute, perform, and display the work").

14. "Copyleft" is slang for a "software license that allows users to modify or incorporate open-source code into larger programs on the condition that the software containing the source code is publicly distributed without restrictions." BLACK'S LAW DICTIONARY, *supra* note 13, at 386. This category of licenses is also called "reciprocal." *See* ROSEN ON OPEN SOURCE, *supra* note 3, at 106.

15. *See About the FSF*, FREE SOFTWARE FOUND., www.fsf.org/about/ (last visited Apr. 5, 2013). The FSF defines copyleft as a "general method for making a program free software and requiring all modified and extended versions of the program to be free software as well." It continues: "[I]nstead of putting . . . software in the public domain, we "copyleft" it. Copyleft says that anyone who redistributes the software, with or without changes, must pass along the freedom to further copy and change it. Copyleft guarantees that every user has freedom." *Licenses*, GNU OPERATING SYS., http://www.gnu.org/licenses (last updated Feb. 28, 2013).

16. *See Licenses*, *supra* note 15 ("Proprietary software developers use copyright to take away the users' freedom; we use copyright to guarantee their freedom. That's why we reverse the name, changing 'copyright' into 'copyleft.'").

17. *See* Vetter, *supra* note 8, at 58; *see also* Nadan, *supra* note 1, at 359-60.

18. *See* 17 U.S.C. § 501 (2011). An instance where "a patent licensee combines the licensed code with [copyleft-covered] code [resulting in] releasing someone else's patented software to the public" may essentially negate a patent. *See* Evans & Layne-Farrar, *supra* note 12, at 10. These situations are not specifically addressed in this Comment, but the suggestions offered by the discussion *infra* are directly applicable and may be used in order to avoid and mitigate such risks of patent loss.

in the development of proprietary software, and its legal implication.[19]

Most commercial enterprises and software developers recognize the business and legal risks—and associated costs—that the use of OSS code entails.[20] Consequently, many implement some sort of compliance mechanism as best practice.[21] But what should the enterprise do if its software developer either intentionally or inadvertently[22] incorporates open source code covered by one of the most "viral" licenses—the GNU[23] General Public License (GPL)?[24]

---

19. *See generally* Markwith, *supra* note 1, at 231; *see also* ROSEN ON OPEN SOURCE, *supra* note 3, at 29 (stating that the need to track applicable license is justified by the fact that the "new authors are subject to the licenses of previous authors who preceded them, and each of those contributions may have different license restrictions on its use."). Additional difficulties may arise when—due to the absence of control over contributions to certain OSS projects—it is impossible to determine origin of the code and trace back the "chain of title." *See* Jim Markwith, *Open Source Software: Intellectual Property, Due Diligence, Litigation, and Industry Trends*, 55 PRACTICAL LAW. 31, 33-34 (2009). Another facet of this "backtracking" problem is the efforts to identify and send out appropriate notices and to identify and provide relevant source code. In addition to delays in product development and diversion of resources away from engineering, the actual costs of "backtracking" are usually more than either the lawyers' fees to settle disputes, or the amount of damages demanded by a typical open source plaintiff. *See* E-mail from Heather J. Meeker, *supra* note 6.

20. In addition to software development, significant OSS risks also arise in the context of mergers and acquisitions. *See, e.g.*, Gordon Caplan & Maurice Lefkort, *Caveat Emptor: The Threat to Value from Target Company Use of Open Source Software*, 12 M & A LAW. 9 (2008).

21. *See generally* Brian Fan, Andrew Aitken & John Koenig, *Open Source Intellectual Property and Licensing Compliance: A Survey and Analysis of Industry Best Practices*, OLLIANCE GRP. 8 (2004), http://olliancegroup.com/opensource/Olliance%20-%20IP%20and%20Licensing%20Best%20Practices.pdf [hereinafter OLLIANCE GROUP SURVEY]. For example, Microsoft routinely performs "technical due diligence" for every transaction involving code acquisition, which includes identification of licenses that apply to acquired code, analysis of the code's quality and security, and determination of whether it is being used in a manner consistent with the terms of applicable licenses. *See* Markwith, *supra* note 19, at 33. *But see* Kozubek, *supra* note 4, at 33 ("Many companies do not realize that their proprietary software can include OSS and be covered under a [copyleft] license." (quoting James Kunick, Chair of IP and Technology Practice, Much Shelist, P.C.) (internal quotation marks omitted)).

22. Although it is not true that "every programmer who downloads the code should have known it was [subject to an OSS license], or that all open source software includes the [license] notice in a conspicuous place," many experienced programmers "would be hard-pressed to claim that they did not suspect that the code was subject to [OSS license], or that their failure to look for the [license] notice was reasonable." *See* Nadan, *supra* note 1, at 366.

23. GNU stands for "GNU's Not Unix" and is an operating system developed by the free software movement GNU Project. *See Overview of the GNU System*, GNU OPERATING SYS., http://www.gnu.org/gnu/gnu-history.html (last updated Mar. 10, 2013). The GNU GPL was originally written by Richard Stallman for use by the GNU Project. *See generally* SAM WILLIAMS, FREE AS IN FREEDOM: RICHARD STALLMAN'S CRUSADE FOR FREE SOFTWARE 149-68 (2002). The FSF, founded by Richard Stallman in 1985, is the current sponsor of the Project and also publishes the GNU family of licenses. *See About the FSF*, *supra* note 15; *see also* Brett Smith, *Free Software Licensing Resources*, FREE SOFTWARE FOUND. (Nov. 6, 2006), http://www.fsf.org/licensing/education.

Can one remedy such a situation and prevent losing valuable intellectual property rights? What are the chances that the licensor will actually enforce the license requirements? And if the company decides to comply, what does compliance then entail?

In addressing these questions and concerns, Part II details categories of OSS licenses and surveys their enforceability. Part III then addresses mechanisms, tools, and practices designed to prevent and mitigate the risks of using open source code. Part IV addresses complying with the OSS license requirements. This discussion will show that: (1) the risks associated with OSS, although not minimal, are generally known; (2) an effective toolset to prevent intermixing of open source code with closed code exists; and (3) even if the intermixing occurs, the likelihood of judicial enforcement is unclear or at least not particularly significant, especially if the license violation is subsequently remedied.

II. OSS LICENSES: AN OVERVIEW

A. Categories of OSS Licenses

A broad variety of OSS licenses exists[25] that fall into numerous, distinct categories.[26] In determining the effect of an OSS license on proprietary software and its possible "constraints on the ability of downstream[27] recipients to license out derivative works under terms more restrictive than the original upstream license,"[28] OSS licenses

24. *GNU General Public License*, GNU OPERATING SYS., http://www.gnu.org/copyleft/gpl.html (last updated Feb. 28, 2013) [hereinafter GPLv.3].

25. *See, e.g.*, Tom Callaway, *Licensing: Main*, FEDORA PROJECT, https://fedoraproject.org/wiki/Licensing (last updated Mar. 4, 2013) (a list of over 300 licenses (both "good" and "bad") maintained by the project); *Licenses by Name*, OPEN SOURCE INITIATIVE, http://www.opensource.org/licenses/alphabetical (last visited Apr. 5, 2013) (listing sixty nine most popular open source licenses approved by the Open Source Initiative (OSI)); *Various Licenses and Comments About Them*, GNU OPERATING SYS., http://www.gnu.org/licenses/license-list.html (last updated Mar. 10, 2013) (listing over hundred licenses, both "free" and "non-free").

26. *See generally* ROSEN ON OPEN SOURCE, *supra* note 3, at 69-71.

27. As used in this Comment, the "upstream work" is a program or set of programs that an open source project develops and licenses on certain terms. A "downstream work" is the work that is created by using all or parts of the "upstream work," hence it must comply with the applicable terms of the license covering the upstream work. *See generally Definition of "Downstream" and "Upstream,"* STACK OVERFLOW, http://stackoverflow.com/questions/2739376/definition-of-downstream-and-upstream (last visited Apr. 5, 2013). In licensing context, the basic idea is that the owner of upstream work who makes it available to others is the licensor, and the "downstream" user is the licensee; this general structure can be complicated by various other factors.

28. Fontana, *supra* note 11, at 83.

can be divided into three broad categories: strong copyleft (also called "reciprocal" or "hereditary"[29]), weak copyleft, and non-copyleft licenses.[30]

1. Strong Copyleft Licenses

The best example of a strong copyleft license is the GNU GPL and its variants.[31] The FSF updated the GPL to version 3 in 2007,[32] but the majority of open source developers still use version 2.[33] Because GPL is extremely restrictive and poses significant compliance risks if not properly and carefully followed, this family of licenses creates the most concern for proprietary software developers and other commercial enterprises.[34]

Under GPLv.2, any distributed or published software that "in whole or in part contains or is derived from [upstream work] or any part thereof, [is] to be licensed *as a whole* at no charge to all third

29. Copyleft licenses are often called reciprocal because they create a "license bargain" that can be stated as follows: "You may have this free software on condition that any derivative works that you create from it and distribute must be licensed to all under the same license." ROSEN ON OPEN SOURCE, *supra* note 3, at 103. This requirement to license all derivative works as a whole under the terms of the same license has also been described as "hereditary." *See* David Turner, *The LGPL and Java*, GNU OPERATING SYS., http://www.gnu.org/licenses/lgpl-java.html (last updated Feb. 28, 2013); *see also* Luis Villa, *The License Term Smorgasbord: Copyleft, Share-Alike, Reciprocal, Viral, or Hereditary?*, LUIS VILLA (Feb. 3, 2012), http://tieguy.org/blog/2012/02/02/the-license-term-smorgasbord-copyleft-share-alike-reciprocal-viral-or-hereditary/.

30. *See* Fontana, *supra* note 11, at 83.

31. *See generally Licenses*, *supra* note 15. Besides GPL, the GNU Affero General Public License is also popular. *See GNU Affero General Public License*, GNU OPERATING SYS., http://www.gnu.org/licenses/agpl.html (last visited Jan. 19, 2013). Another well-known copyleft license is Reciprocal Public License. *Reciprocal Public License 1.5 (RPL-1.5)*, OPEN SOURCE INITIATIVE, http://opensource.org/licenses/rpl-1.5 (last visited Apr. 5, 2013) [hereinafter RPLv.1.5]. RPL is considered to be less permissive than GPL because of additional restrictions on internal use. *See* RPLv.1.5 § 6.1; *see also infra* Part IV.A.3. *But see* Ian Skerrett, *Two Open Source Ideologies That Are Just Wrong*, IAN SKERRETT (June 2, 2011, 2:32 PM), http://ianskerrett.wordpress.com/2011/06/02/two-open-source-ideologies-that-are-just-wrong/ ("I would suggest strict copyleft license [sic] like the AGPL and GPL are a dying breed. At a recent open source confernce [sic], a well know [sic] open source lawyer claimed GPL v3 has been a failed experiment, with little adoption."). For a discussion of OSS evolution and recent trends see Richard Fontana, *The Decline of the GPL and What to Do about It*, Presentation at Linux Foundation Collaboration Summit, Apr. 3-5, 2012, *available at* https://events.linuxfoundation.org/images/stories/pdf/lfcs2012_fontana.pdf.

32. *See* GPLv.3, *supra* note 24.

33. *See GNU General Public License, Version 2*, GNU OPERATING SYS., http://www.gnu.org/licenses/gpl-2.0.html (last updated Mar. 16, 2013) [hereinafter GPLv.2].

34. *See* ROSEN ON OPEN SOURCE, *supra* note 3, at 2 ("Proprietary software vendors love the software freedom provided by [non-copyleft licenses], but some of them hate and fear the software freedom guaranteed by the GPL.").

parties under the terms of [GPLv.2]."[35] Similarly, GPLv.3 requires that "a work based on [the upstream work], or the modifications to produce it from [upstream work], in the form of source code" must be licensed as "*the entire work, as a whole*, under [GPLv.3] to anyone who comes into possession of a copy. This License will therefore apply . . . to the whole of the work, and all its parts, regardless of how they are packaged."[36] Furthermore, the GPL prohibits "impos[ing] any further restrictions on the exercise of the rights granted or affirmed under [the GPL]."[37] Finally, it requires that any work distributed in the "non-source form"[38] must have complete source code available to downstream users.[39]

The plain language of the GPL and the ideology behind it imposes expectations upon the licensor, which reinforce the "strength" of the copyleft.[40] The goal of the license is "to limit the ability of licensees to distribute proprietary enhancements of free works, thereby preserving a commons of free software even as that software evolves through downstream modification."[41] The licensors also typically expect that "copyleft scope ordinarily extend[s] to 'the whole work', rather than just some constituent part of it," because

35. GPLv.2, *supra* note 33, § 2(b) (emphasis added).

36. GPLv.3, *supra* note 24, § 5(c) (emphasis added). According to version 3 of the GPL, a work is "based on" an earlier work when it "cop[ies] from or adapt[s] all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy." *Id.* § 0. It has also been suggested that GPL applies only to the "derivative" works, but not to the "collective" works (that are simply collections of independent software programs). *See, e.g.*, ROSEN ON OPEN SOURCE, *supra* note 3, at 119-21. The most important question is the scope of the phrase "based on" for works that use source code covered by GPLv.2. Many companies, software developers, and lawyers are confused—and rightly so—about where the border truly lies and what separates a "based on" work from an independent and nonderivative work. For an in-depth analysis of the issue see HEATHER J. MEEKER, THE OPEN SOURCE ALTERNATIVE: UNDERSTANDING RISKS AND LEVERAGING OPPORTUNITIES 183-221 (2008) [hereinafter MEEKER ON OPEN SOURCE].

37. GPLv.2, *supra* note 33, § 6; GPLv.3, *supra* note 24, § 10.

38. "Non-source" code typically refers to object code or an executable. Object code is a "machine language representation of programming source code." *Definition of: Object Code*, PCMAG.COM ENCYCLOPEDIA, http://www.pcmag.com/encyclopedia/term/48210/object-code (last visited Apr. 5, 2013). Executable work is "software in a form that can be run in the computer" and "typically refers to machine language, which is the set of native instructions the computer carries out in hardware." *Definition of: Executable Code*, PCMAG.COM ENCYCLOPEDIA, http://www.pcmag.com/encyclopedia/term/42842/executable-code (last visited Apr. 5, 2013).

39. *See* GPLv.2, *supra* note 33, § 3 ("[C]omplete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable."); *see also* GPLv.3, *supra* note 24, § 6.

40. *See generally* ROSEN ON OPEN SOURCE, *supra* note 3, at 107-09.

41. Fontana, *supra* note 11, at 86.

otherwise any modification or enhancement could "easily be structured as a separate part of that work falling outside of copyleft."[42]

2. Weak Copyleft Licenses

Weak copyleft licenses—also called "file-level copyleft"[43]—were created in part as a response to mounting criticism of the GPL's restrictive nature.[44] These restrictions were preventing the development of proprietary applications for some GPL-licensed libraries.[45] The first such license was FSF's GNU Library General Public License (LGPL),[46] followed by others, most notably the Mozilla Public License, version 1.1 (MPL)[47] and Eclipse Public License, version 1.0 (EPL).[48] Weak copyleft licenses do not possess the unconditional "viral" characteristics of the GPL that require any works derived from a copyleft work to be themselves be licensed under copyleft when distributed. Instead, they generally permit

42. *Id.*

43. Weak copyleft requirement typically attaches only to the distribution of modified original files (so it applies to a specific file or files, not the program as a whole); any add-ons and new features that do not modify existing files are free from copyleft obligations. Thus, a weak copyleft allows inclusion of separate files than can be licensed on different terms. *See, e.g.*, *MPL 2.0 FAQ*, MOZILLA, http://www.mozilla.org/MPL/2.0/FAQ.html (last visited Apr. 5, 2013) ("The [Mozilla Public License] is a simple copyleft license. The MPL's 'file-level' copyleft is designed to encourage contributors to share modifications they make to your code, while still allowing them to combine your code with code under other licenses (open or proprietary) with minimal restrictions.").

44. *See* Fontana, *supra* note 11, at 87.

45. A library is generally "a collection of software routines that programmers incorporate into their applications" and that are "linked into the program when it is compiled." *See Definition of: Library*, PCMAG.COM ENCYCLOPEDIA, http://www.pcmag.com/encyclopedia/term/46063/library (last visited Apr. 5, 2013).

46. *GNU Library General Public License, Version 2.0*, GNU OPERATING SYS., http://www.gnu.org/licenses/old-licenses/lgpl-2.0.html (last updated Mar. 16, 2013). In 1999, version 2.1 was named the Lesser General Public License. *See GNU Lesser General Public License, Version 2.1*, GNU OPERATING SYS., http://www.gnu.org/licenses/lgpl-2.1.html (last updated Mar. 16, 2013) [hereinafter LGPLv.2.1]. In 2007, version 3 of LGPL was released. *See GNU Lesser General Public License*, GNU OPERATING SYS., http://www.gnu.org/licenses/lgpl.html (last updated Feb. 28, 2013) [hereinafter LGPLv.3].

47. *Mozilla Public License Version 1.1*, MOZILLA, http://www.mozilla.org/MPL/1.1/ (last visited Apr. 5, 2013) [hereinafter MPLv.1.1]. The most current version is version 2.0, announced by Mozilla last year. *See Updating the MPL: Announcing Version 2.0 of the Mozilla Public License*, MOZILLA.ORG (Jan. 3, 2012, 9:45 AM), https://mpl.mozilla.org/2012/01/03/announcing-mpl-2-0/. Version 2.0 was "designed to encourage contributors to share modifications they make to MPL-licensed code, while still allowing users to create projects that combine MPL-licensed code with code under other licenses (either open or proprietary)." *Id.*

48. *Eclipse Public License—v. 1.0*, ECLIPSE, http://eclipse.org/legal/epl-v10.html (last visited Apr. 5, 2013) [hereinafter EPLv.1.0].

distribution of the executable programs under proprietary terms while requiring distribution of the binary code[49] with the corresponding source code only for the weak copyleft-covered portions.[50]

In contrast to the GPL's strict requirements, weak copyleft licenses typically allow for the incorporation of OSS code into proprietary software. For example, the MPL does not apply to "the whole work" but covers only licensee's "modifications" defined as "[a]ny addition to or deletion from the contents of a file containing Original Code or previous Modifications" and "[a]ny new file that contains any part of the Original Code or previous Modifications."[51] It also specifically provides for the right to "create a Larger Work by combining Covered Code with other code not governed by the terms of this License and distribute the Larger Work as a single product,"[52] possibly in whole or in part under a license other than the MPL.[53]

To determine whether a downstream work is covered by weak copyleft or not, one scrutinizes the extent of changes the programmer made to the original code. Modifications to the original source code would likely be covered by the license, while unchanged parts incorporated into a larger work probably would not. The EPL requirements, for example, are similar to the MPL: EPL applies to "contributions," defined as both the initial code and any changes or additions made by downstream users.[54] EPL, however, does not apply to "additions to the Program which: (i) are separate modules of software distributed in conjunction with the Program under their own license agreement, and (ii) are not derivative works of the Program."[55]

The LGPL, which aims to relax the GPL requirements,[56] also provides for some exceptions to the strong copyleft license.

---

49. Binary code is a coding system in which data and instructions are represented by a series of two symbols—0s and 1s. *See generally Definition of: Binary*, PCMAG.COM ENCYCLOPEDIA, http://www.pcmag.com/encyclopedia/term/38618/binary (last visited Apr. 5, 2013).

50. *See, e.g.*, MPLv.1.1, *supra* note 47, § 3.6; EPLv.1.0, *supra* note 48, § 3; LGPLv.3, *supra* note 46, §§ 4-6.

51. MPLv.1.1, *supra* note 47, § 1.9.

52. *Id.* § 3.7.

53. *See* Fontana, *supra* note 11, at 88.

54. *See* EPLv.1.0, *supra* note 48, § 1.

55. *Id.* Although some licenses do not define "derivative work," by its reference to the U.S. copyright law it typically means "a work based upon one or more preexisting works . . . which, as a whole, [is] an original work of authorship." *See* 17 U.S.C. § 101 (2011).

56. *See Why You Shouldn't Use the Lesser GPL for Your Next Library*, GNU OPERATING SYS., http://www.gnu.org/philosophy/why-not-lgpl.html (last updated Feb. 28, 2013) ("[U]sing the Lesser GPL permits use of the library in proprietary programs.").

Specifically, it states:

> A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library." Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of [LGPL].[57]

A modified copy of a library, or any portion of a library that has modifications, creates a "work based on the Library" and is covered by the LGPL as a whole.[58] This means that distribution of the modified version (or a portion that is modified or a derivative of it) must be accompanied with "the corresponding machine-readable source code."[59]

The requirement for distribution of the "work that uses the Library," however, is markedly different: LGPL provides for an exception to the requirement to provide source code for works that use "a suitable shared library mechanism for linking."[60] Such mechanism

> uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and . . . will operate properly with modified version of the library . . . as long as the modified version is interface-compatible with the version that the work was made with.[61]

Thus, if the software program is simply linked to the shared library without incorporating any portion of it—as opposed to the work being an executable linked with the library—such program may be distributed "under terms of your choice," provided that all requirements are satisfied.[62]

3. Non-Copyleft Licenses

Non-copyleft—or permissive[63]—licenses are the least restrictive,

---

57. LGPLv.2.1, *supra* note 46, § 5. A software link is a "call to another program or subroutine." *See Definition of: Link*, PCMAG.COM ENCYCLOPEDIA, http://www.pcmag.com/encyclopedia/term/46141/link (last visited Mar. 15, 2013). *See also infra* note 132.

58. LGPLv.2.1, *supra* note 46, § 2.

59. *Id.* § 4.

60. *Id.* § 6(b).

61. *Id.*

62. *Id.* § 6.

63. Non-copyleft licenses "allow unfettered use of the open source code, including the crucial aspect of embedding the open source code into proprietary applications." LINDBERG, *supra* note 12, at 177. *See also infra* Part IV.B.1.

and typically pose minimal risks to developers of proprietary software and other downstream users.[64] The purpose of non-copyleft licenses is to make OSS available to anyone, with minimal restrictions, thus making the code nearly equivalent to a public domain work.[65] Such licenses maximize the downstream utilization of the code, while shielding—by way of warranty and liability disclaimers—the upstream developers from litigation and reputational harm. Code licensed under non-copyleft licenses may be freely, and commonly is, incorporated into software that may be then licensed under any other terms, either proprietary or open source. Distribution of permissive-licensed source code and any work based on or derived from it does not require the vendor to provide associated source code to downstream users.

B. Enforcement of OSS Licenses

1. Community Enforcement

Since its inception in the early 1980s, the OSS movement—involving both free software developers and downstream users—has assumed that OSS licenses were harmonized with applicable law and were generally enforceable.[66] This assumption "created a kind of legal confidence among [OSS] licensors that reinforced existing licensing models and shaped the behavior of parties in licensing disputes."[67]

GPL enforcement began soon after that license was formalized in 1989.[68] During the 1990s, the Free Software Foundation (FSF), the licensor of many GNU programs, actively enforced the GPL on

64. The most common non-copyleft licenses are: the Apache license, the three-clause BSD license, and the MIT License. *See Apache License, Version 2.0*, APACHE SOFTWARE FOUND., http://www.apache.org/licenses/LICENSE-2.0.html (last visited Apr. 6, 2013) [hereinafter ApLv.2.0]); *The BSD 3-Clause License*, OPEN SOURCE INITIATIVE, http://www.opensource.org/licenses/BSD-3-Clause (last visited Apr. 6, 2013) [hereinafter BSD]); *The MIT License (MIT)*, OPEN SOURCE INITIATIVE, http://www.opensource.org/licenses/mit-license.php (last visited Apr. 6, 2013) [hereinafter MIT]).

65. *See* ROSEN ON OPEN SOURCE, *supra* note 3, at 107 ("Any licensee under [a non-copyleft] open source license can take that free software, create derivative works from it, and then distribute those derivative works under a proprietary license."). A list of licenses (currently at forty) that place no restrictions on use, distribution, modification, derivation, combination, and application of software is maintained by the Copyfree Initiative. *See Copyfree Licenses*, COPYFREE.ORG, http://copyfree.org/licenses/ (last visited Apr. 6, 2013).

66. *See* Fontana, *supra* note 11, at 91.

67. *Id.* at 92.

68. *See* Bradley M. Kuhn, Aaron Williamson & Karen M. Sandler, *A Practical Guide to GPL Compliance*, SOFTWARE FREEDOM LAW CTR., at 1 (Aug. 26, 2008), http://www.softwarefreedom.org/resources/2008/compliance-guide.pdf.

behalf of its community of OSS developers.[69] The enforcement "was generally a private process; the FSF contacted violators confidentially and helped them to comply with the license."[70] In the early 2000s, negative publicity surrounding alleged violations supplemented such community enforcement efforts and increased the pressure on commercial enterprises to comply.[71] In 2003, the FSF went one step further and established a formal GPL Compliance Lab, which increased the volume of enforcement and encouraged amicable settlements with violators.[72] Thus, the OSS community's dispute resolution practices were self-sufficient and did not depend on any involvement from the conventional legal system until the mid-2000s, when the "multilateral introduction [of] a more professional approach . . . finally spilled over into the court system."[73]

The Software Freedom Conservancy and its Executive Director, Bradley M. Kuhn, undertook a collective effort approach to community enforcement.[74] The Conservancy, a non-profit that provides a broad range of services to free software projects,[75] has unified many of its member projects "to ensure compliance with their Free Software licenses . . . [and] that the rights embodied in Free Software licenses are fully upheld for all developers, users, and the general public."[76] In addition to seven member projects, seven

69. *See id.*

70. *Id.*; *see also* Bradley M. Kuhn, *Some Thoughts on Conservancy's GPL Enforcement*, SOFTWARE FREEDOM CONSERVANCY (Feb. 1, 2012), http://sfconservancy.org/blog/2012/feb/01/gpl-enforcement/ ("Every enforcement action opens as a conversation, asking the violator to meet a few simple requests so that their permission to engage in copyright-governed activity can be restored, and they can go about their new business as a fine, upstanding, compliant Free Software redistributor."). Currently, the Linux Foundation offers the Open Compliance Directory, which allows open source developers to contact compliance officers at companies using Linux. *Open Compliance Directory and Rapid Alert System*, THE LINUX FOUND., http://www.linuxfoundation.org/programs/legal/compliance/directory (last visited Apr. 6, 2013). This system facilitates communication and coordination and assists in addressing developers' concerns in a timely fashion. *Id.*

71. Kuhn, *supra* note 68.

72. *See id.* at 1-2. In 2004, Harald Welte, creator of gpl-violations.org, took similar approach in Europe. *See generally About the Gpl-Violations.org Project*, GPL-VIOLATIONS.ORG, http://gpl-violations.org/about.html (last visited Apr. 6, 2013). *See also FSFE Legal—The Freedom Task Force*, FREE SOFTWARE FOUND. EUR., http://fsfe.org/activities/ftf/ftf.en.html (last visited Apr. 6, 2013).

73. *See* Fontana, *supra* note 11, at 92.

74. *Conservancy Projects Launch Coordinated Free Software Compliance Efforts*, SOFTWARE FREEDOM CONSERVANCY (May 29, 2012), http://sfconservancy.org/news/2012/may/29/compliance/.

75. The Conservancy is well-known for being a "GPL enforcement agent for various BusyBox copyright holders." BUSYBOX, http://www.busybox.net/ (last visited Apr. 6, 2013).

76. *Conservancy Projects Launch*, *supra* note 74.

individual copyright holders in the Linux kernel, who have contributed to Linux under the GPLv.2, have also engaged the Conservancy in compliance efforts.[77] The Conservancy's enforcement approach is to encourage license compliance by all software users[78] and to work with distributors of free software "in a friendly spirit of cooperation and participation."[79] To achieve this goal, the "Conservancy's Free Software compliance work always centers around assisting companies to become productive and cooperative participants in Free Software development."[80] Continuing this philosophy, the Conservancy's Executive Director Bradley Kuhn specifically expressed his intent to avoid litigation because lawsuits are time-consuming, expensive, unpredictable, and complicated.[81]

2. Judicial Enforcement

OSS licenses first appeared on court dockets in the early 2000s in the context of business disputes, but most of the corresponding court rulings were not directly relevant to the enforcement of copyleft licenses.[82] During the *SCO v. IBM*[83] litigation the SCO Group made a

77. *Id.*; Bradley M. Kuhn, *Conservancy's Coordinated Compliance Effort*, BRADLEY M. KUHN'S BLOG (May 29, 2012), http://ebb.org/bkuhn/blog/2012/05/29/compliance.html.

78. One of the Conservancy's focus areas is the requirement to provide build and installation instructions in addition to the source code (i.e., complete source code with scripts used to control compilation and installation of the executable), so the software users can actually compile and install the software. *See* Kuhn, *supra* note 70.

79. *Conservancy Projects Launch*, *supra* note 74.

80. *Id.* (quoting Bradley M. Kuhn, Executive Director of Software Freedom Conservancy) (internal quotation marks omitted).

81. Bradley M. Kuhn, *Conservancy's Compliance Project (episode 0x2A)*, at 11:00, FREE AS IN FREEDOM (May 29, 2012) (podcast); *see also* Kuhn, *supra* note 70.

> I do find litigation particularly annoying, time-consuming, and litigation also makes GPL compliance take longer than it should. That's why litigation has always been a last resort, and that 99.999% of GPL enforcement matters get resolved without a lawsuit. Lawsuits are only an option, in my view, when a violation is egregious, *and* multiple attempts to begin a friendly conversation with the violator are consistently ignored.

Id.

82. *See, e.g.*, Computer Assocs. Int'l v. Quest Software, Inc., 333 F. Supp. 2d 688 (N.D. Ill. 2004) (granting preliminary injunction based on likelihood of finding literal copying of source code containing OSS); Progress Software Corp. v. MySQL AB, 195 F. Supp. 2d 328 (D. Mass. 2002) (denying preliminary injunction against software under GPL license); Order of Dismissal, Drew Techs., Inc., v. Soc'y of Auto. Eng'rs, Inc., No. 2:03-cv-74535-NGE (E.D. Mich. June 20, 2005) (the parties agreed to dismiss the case); MontaVista Software, Inc. v. Lineo, Inc., No. 2:02 CV-0309J (D. Utah filed July 23, 2002) (plaintiff accused defendant of modifying, copying, and redistributing its GPL-licensed software without proper copyright notices; the parties settled in the third quarter of 2003). *See also* LINDBERG, *supra* note 12, at 225 ("[T]he cases have not continued to the point where the court issued a final ruling, but the GPL enforcers settled on favorable terms.").

number of arguments targeting enforceability of the GPL, among them alleging that the GPL was unconstitutional.[84] The court, however, has yet to rule on the GPL's constitutionality: the court has stayed the litigation due to bankruptcy filing by the SCO Group.[85] Even if this case does go back to trial, the controversy will most likely be limited to contract claims; the issue of copyright infringement was effectively resolved by a jury verdict, which confirmed that Novell Inc. owned the code the SCO Group claimed as its own.[86]

Probably due to the highly effective informal enforcement mechanism,[87] judicial enforcement of copyleft licenses only recently gained significant traction in the United States.[88] In 2007, the OSS community filed its first copyleft enforcement lawsuit.[89] The Software Freedom Law Center (SFLC) separately sued multiple defendants on

---

83. *See* Second Amended Complaint, SCO Grp. Inc., v. Int'l Bus. Machs. Corp., No. 03-CV-0294, 2004 WL 3482623 (D. Utah Feb. 27, 2004); Amended Complaint, SCO Grp., Inc. v. Int'l Bus. Machs. Corp., No. 03-CV-0294, 2003 WL 24136857 (D. Utah July 22, 2003); Complaint, Caldera Sys., Inc. v. Int'l Bus. Machs. Corp., No. 030905199 (D. Utah Mar. 6, 2003). *See generally* Andrew LaFontaine, Note, *Adventures in Software Licensing:* SCO v. IBM *and the Future of the Open Source Model*, 4 J. ON TELECOMM. & HIGH TECH. L. 449 (2006). For a detailed account of this extremely complex and lengthy litigation see *SCO Overview—Links*, GROKLAW, http://www.groklaw.net/staticpages/index.php?page=20061212211835541 (last updated Mar. 18, 2011).

84. The plaintiff dropped the unconstitutionality argument in 2004. *See SCO Drops Its Claim That the GPL Is Unconstitutional*, GROKLAW (Apr. 29, 2004, 11:59 AM), http://www.groklaw.net/article.php?story=20040428235932742; *see generally* Jason B. Wacha, *Taking the Case: Is the GPL Enforceable?*, 21 SANTA CLARA COMPUTER & HIGH TECH. L.J. 451, 459-62 (2005) (a rebuttal of GPL unconstitutionality claim).

85. *See* Notice of Filing for Bankruptcy, SCO Grp., Inc. v. Int'l Bus. Machs., No. 2:03-CV-0294 (D. Utah Sept. 14, 2007). In February 2012, the parties modified the stay by stipulation to allow some of their claims go to trial. Order Granting Stipulation and Order Modifying the Automatic Stay, *In re* TSG Grp., Inc., No. 07-11337 (Bankr. D. Del. Feb. 17, 2012). In June 2012, the SCO Group filed a motion requesting that the court allow the case to go to trial. The SCO Group, Inc.'s Request to Submit for Decision, SCO Grp., Inc. v. Int'l Bus. Machs. Corp., No. 2:03-CV-002294 (D. Utah June 14, 2012).

86. SCO Grp., Inc. v. Novell, Inc., 439 F. App'x 688 (10th Cir. 2011); SCO Grp., Inc. v. Novell, Inc., 721 F. Supp. 2d 1050 (D. Utah 2010).

87. It is reasonable to assume that a prototypical defendant would prefer to settle as early as possible, since the costs of voluntary rectifying a violation would presumably be much less than litigating it. *But see supra* note 19 ("backtracking" in order to comply with license requirements also can be very costly).

88. Enforcement of the GPL has been very successful in Europe in recent years. *See, e.g.*, Till Jaeger, *Enforcement of the GNU GPL in Germany and Europe*, 1 J. INTELL. PROP., INFO. TECH. & E-COMMERCE L. 34 (2010), *available at* http://www.jipitec.eu/issues/jipitec-1-1-2010/2419/dippadm1268746871.43.pdf; Martin von Willebrand, *Case Law Report: A Look at* EDU 4 v. AFPA, *Also Known As the "Paris GPL Case"*, 1 INT'L FREE & OPEN SOURCE SOFTWARE L. REV. 123 (2009), *available at* http://www.ifosslr.org/ifosslr/article/view/17/42.

89. *See* Andersen v. Monsoon Multimedia, Inc., No. 07-CV-08205, 2007 WL 2777698 (S.D.N.Y. Sept. 19, 2007).

behalf of two copyright holders of the BusyBox software.[90] The SFLC asserted essentially the same claims in December 2009 against fourteen consumer electronics companies.[91] In 2008, the SFLC brought a copyright infringement action on behalf of the FSF against Cisco Systems, relating to networking products sold by Cisco's subsidiary, Linksys.[92] Artifex Software is among the first private companies to judicially enforce the GPL, which it did in 2008 and 2009.[93] All of these GPL enforcement lawsuits quickly settled, generally resulting in a payment to the plaintiffs and a commitment to comply with the GPL requirements.[94]

At present, Red Hat is pursuing a GPL enforcement action against Twin Peaks Software.[95] In response to patent infringement claims by Twin Peaks Software, Inc., Red Hat filed a counterclaim, alleging that:

> Twin Peaks copied substantial portions of open source code into [its] products, including source code originally authored by Red Hat. . . .
>
> By selling or providing [its products] under proprietary license agreements and not making any of their code available to the public, Twin Peaks has failed to comply with the explicit

90. *See* Andersen v. High-Gain Antennas, L.L.C., No. 07-CV-10456, 2007 WL 6353333 (S.D.N.Y. Nov. 19, 2007); *see also* Andersen v. Verizon Commc'ns, Inc., No. 07 CV 11070, 2007 WL 6353336 (S.D.N.Y. Dec. 6, 2007); Andersen v. Super Micro Computer, Inc., No. 108-CV-05269, 2008 WL 2755743 (S.D.N.Y. June 9, 2008); Andersen v. Extreme Networks, Inc., No. 08-CV-06426, 2008 WL 4486847 (S.D.N.Y. July 17, 2008).

91. *See* Complaint, Software Freedom Conservancy, Inc. v. Best Buy Co., No. 09-CIV 10155 (S.D.N.Y. Dec. 14, 2009). One of the defendants—Westinghouse—stopped defending the lawsuit and entered bankruptcy, which eventually lead to the default judgment against it. *See* Software Freedom Conservancy, Inc. v. Best Buy Co., No. 09 CIV 10155(SAS), 2010 WL 2985320 (S.D.N.Y. July 27, 2010). The judge found the infringement to be valid and awarded treble statutory damages of $90,000. *Id.* at *3. In addition to the damage award, the court entered a permanent injunction prohibiting distribution of infringing products and mandating infringing products' forfeiture to the plaintiff; the court also invited the plaintiff to submit a reimbursement request for attorneys' fees. *Id.* at *2-4.

92. *See* Free Software Found., Inc. v. Cisco Sys., Inc., No. 1:08-CV-10764, 2008 WL 8449470 (S.D.N.Y. Dec. 11, 2008).

93. *See*, *e.g.*, Complaint for Copyright Infringement, Artifex Software Inc., v. Diebold, Inc., No. 308-CV-04837, 2008 WL 5457246 (N.D. Cal. Oct. 22, 2008) (stipulated dismissal filed on June 25, 2009); *see also* Complaint for Copyright Infringement, Artifex Software Inc., v. Palm Inc., No. CV 09 5679 RS, 2009 WL 4813582, (N.D. Cal. Dec. 2, 2009) (pending).

94. *See* Fontana, *supra* note 11, at 93. Virtually every such settlement is made confidential, so specific details and the payout amounts are typically unknown.

95. *See* Defendants/Counterclaim-Plaintiffs Red Hat, Inc.'s and Gluster, Inc.'s First Amended Answer and Counterclaims to Plaintiff Twin Peaks Software Inc.'s First Amended Complaint for Patent Infringement, Twin Peaks Software Inc. v. Red Hat, Inc., No. 5:12-CV-00911, 2012 WL 5403098 (N.D. Cal. Sept. 13, 2012).

> conditions of the GPL[v.2]. . . .
>
> By reproducing, copying, and distributing Red Hat's original source code in [Twin Peaks's products], without approval and authorization by Red Hat and only subject to its own proprietary license agreement, Twin Peaks is infringing and has infringed Red Hat's exclusive copyrights, and likewise is inducing and has induced its customers to infringe.[96]

While one can only speculate about the outcome of this case, it will certainly be watched closely by open source community. If this case directly addresses the ambiguity and uncertainty stemming from lack of judicial interpretation of the GPL, it could become a seminal ruling.

Despite the few court decisions involving OSS licenses[97] the legal community generally agrees that OSS licenses are enforceable.[98] Notably, in *Wallace v. Free Software Foundation*[99] the court acknowledged—in the context of antitrust violation claims—that the GPL is

> a software licensing agreement through which the GNU/Linux operating system may be licensed and distributed to individual users so long as those users "cause any work that [they] distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License."[100]

In *Jacobsen v. Katzer*[101] the court recently examined the original version of another OSS license, the Artistic License.[102] The court held

---

96. *Id.* at 12-13. The original countercomplaint filed by Red Hat did not have GPL allegations. *See* Defendants Red Hat, Inc.'s and Gluster, Inc.'s Answer and Counterclaims to Plaintiff Twin Peaks Software Inc.'s First Amended Complaint for Patent Infringement, Twin Peaks Software Inc. v. Red Hat, Inc., No. 5:12-CV-00911, 2012 WL 5403091 (N.D. Cal. Aug. 2, 2012).

97. Most court decisions mention OSS licenses in contexts other than copyright infringement claims. *See, e.g.*, Kelly v. Sky Angel, U.S., LLC, 1:09-CV-59, 2010 WL 2776580 (E.D. Tenn. July 14, 2010) (wrongful termination claim in retaliation for plaintiff's reporting of alleged copyright violations by the defendant); MedioStream, Inc. v. Microsoft Corp., 749 F. Supp. 2d 507 (E.D. Tex. 2010) (breach of GPL in the context of patent infringement action).

98. *See generally* Wacha, *supra* note 84 (a general discussion of arguments against enforceability of the GPL and a very persuasive and convincing rebuttal); James Gatto, *Doubts Wane over GPL Enforceability*, 166 MANAGING INTELL. PROP. 33 (2007).

99. Wallace v. Free Software Found., Inc., 1:05-CV-0618, 2006 WL 2038644 (S.D. Ind. Mar. 20, 2006).

100. *Id.* at *3 (citing relevant section of GPLv.3) (alteration in original).

101. *See* Jacobsen v. Katzer, 535 F.3d 1373 (Fed. Cir. 2008).

102. *The "Artistic License"*, PERL CORE DEV., http://dev.perl.org/licenses/artistic.html (last visited Apr. 6, 2013). The OSS community is split about the Artistic License: it has been

that the terms of the Artistic License incorporated enforceable copyright conditions.[103] The Federal Circuit rejected the defendant's argument that providing software to the public at no charge could not warrant copyright protection, citing numerous economic benefits of open source licensing that "range far beyond traditional license royalties."[104] Courts will likely follow the considerations and rationale of *Jacobsen* in analyzing other OSS licenses.[105]

3. Quasi-Judicial (Administrative) Enforcement

In addition to community and judicial enforcement of OSS licenses, the United States International Trade Commission (ITC) may issue an order directing Customs and Border Protection (CBP) to exclude infringing goods from importation into the United States (subject to certain restrictions and limitations).[106] The CBP has the authority to enforce copyright, so long as the copyright is registered with the Library of Congress and recorded with the CBP, and may therefore exclude the infringing goods on its own.[107]

To date, this method of enforcement based on a GPL violation has happened only once.[108] A Linux kernel developer posted a blog entry threatening a U.S. Customs case after he was unable to obtain the source code for a tablet device from the maker of the tablet, even

---

approved by OSI, *see Artistic Licenses*, OPEN SOURCE INITIATIVE, http://www.opensource.org/licenses/artistic-license.php (last visited Apr. 6, 2013), but was criticized by FSF for its vagueness, *see Various Licenses and Comments About Them*, *supra* note 25.

103. *See Jacobsen*, 535 F.3d at 1382.

104. *See id.* at 1379.

105. A recent decision arguably narrows the holding of *Jacobsen*. *See* MDY Indus., LLC v. Blizzard Entm't, Inc., 629 F.3d 928 (9th Cir. 2010), *as amended on denial of reh'g*, No. 09-15932, 2011 WL 538748 (9th Cir. Feb. 17, 2011). The ruling requires a "nexus" between the exclusive right and the copyright condition for a license to be enforceable. *See generally* Sean Hogle, *Open Source Licensing and the Viability of the Free Software Movement*, 3 LANDSLIDE no. 6, July–Aug. 2011, at 8, *available at* http://www.americanbar.org/content/dam/aba/publications/landslide/landslide_august_2011/hogle_landslide_julyaug_2011.authcheckdam.pdf.

106. Tariff Act of 1930, 19 U.S.C. § 1337 (a)(1)(B)(i) (2006).

107. *See* Copyright Act of 1976, 17 U.S.C. § 602 (2011); *see also* 19 C.F.R. § 133.42 (2012).

108. *See* Heather Meeker, *US Customs Case to Be Filed Based on GPL Violations*, COPYLEFT CURRENTS (Sept. 13, 2010, 9:53 PM), http://www.heathermeeker.com/news/2010/9/13/us-customs-case-to-be-filed-based-on-gpl-violation.html; *see also* Matthew Garrett, *Things*, MATTHEW GARRETT'S JOURNAL (Sept. 9, 2010, 9:58 AM), http://mjg59.livejournal.com/126865.html; *Matthew Garrett Files Case with U.S. Customs Against Fusion Garage*, LWN.NET (Sept. 10, 2010), http://lwn.net/Articles/404450.

though the source code was governed by the GPL.[109] Although there were no subsequent posts detailing any further developments in this case,[110] it nonetheless, "shows another approach to litigation in the open source sphere—using a tactic already popular for patent and other intellectual property claims."[111]

III. COPYLEFT PREVENTION

There is a wide spectrum of potential targets for OSS license enforcement.[112] On one side there are licensees that operate exclusively within the OSS community and who violate license requirements inadvertently while (most of the time) making a good faith effort to comply with them. The diametrically opposite side of this spectrum would include licensees that operate outside of OSS community and are (most of the time), unaware that their proprietary software contains open source code subject to an OSS license. For the first group, enforcement risks are typically lesser, since the violations are generally insignificant and may be easily remedied by amicable efforts to comply. Most enforcement focuses on the second group; commercial enterprises, often seen as "enemies,"[113] frequently make material violations of OSS license requirements just because they are unaware either that they are supposed to comply with them, or what the actual requirements are.[114]

Although risk of the copyleft enforcement or litigation is relatively small and may not be the only reason to justify open source compliance, considerations of business ethics, respect for property rights, and desire to maintain good community relations and invest in low-cost development tools provide necessary incentives for sound compliance mechanisms.[115]

Such mechanisms could be divided into two categories: (1) internal and (2) external. While internal compliance programs are generally aimed at the employees of a company and at internal policies, the external mechanisms cover third-party conduct—primarily that of software vendors.

---

109. *See* Garrett, *supra* note 108.

110. *See id.* (comments).

111. Heather J. Meeker, *Open Source and the Age of Enforcement*, 4 HASTINGS SCI. & TECH. L.J. 267, 280 (2012), *available at* http://hstlj.org/wp-content/uploads/2012/09/MeekerV4I2.pdf.

112. *See* Fontana, *supra* note 11, at 96.

113. *See* Stallman, *supra* note 1 ("[Our] enemy is proprietary (nonfree) software.").

114. *See* Fontana, *supra* note 11, at 96.

115. *See id.* at 97-98.

A. Internal Preventive Mechanisms

To effectively manage the risks associated with using OSS, a software company must take a minimum of three steps: (1) perform a comprehensive audit, with the involvement of the software developers, in order to understand how and where the company's source code uses open source software;[116] (2) establish an internal-facing written policy to govern the process for adopting and licensing any open source software; (3) alert employees to the OSS policy and educate them as to its importance, followed by periodic audits and reminders of the policy.[117]

1. Policies and Training

A recent *Open Source Software Development Survey* conducted by Sonatype revealed alarming numbers.[118] Only forty-nine percent of 2,500 software developers, software architects, and information technology (IT) managers across all industries, company sizes, and geographic regions said they have an open source policy in place.[119] The same percentage of respondents said they have no effective licensing in place, and only thirty-two percent maintain detailed records of the software components used in production application.[120] A 2011 Gartner survey revealed substantially the same dire situation.[121]

It is suggested that every company that develops software or manufactures products that use embedded software have a comprehensive policy covering OSS.[122] Such a policy "should strike a

116. A very basic but practical approach to OSS audit is described in Rehm, *supra* note 2, at 318. A very cost-effective "'quick and dirty' self-assessment" could also be performed by "searching [a company's] own source code for the word 'copyright' or the '©' symbol." *See* Adam Kubelka & Matthew Fawcett, *No Free Beer—Practice Tips for Open Source Licensing*, 22 SANTA CLARA COMPUTER & HIGH TECH. L.J. 797, 811 (2006).

117. *See* Eric Lobenfeld, *IP: Open Source Software is Licensed—It's Not "Free,"* INSIDE COUNS. (Dec. 13, 2011), http://www.insidecounsel.com/2011/12/13/ip-open-source-software-is-licensedits-not-free.

118. *See* Katherine Noyes, *Open Source is Driving Business App Development, Survey Finds*, PCWORLD (Apr. 24, 2012, 10:09 AM), http://www.pcworld.com/businesscenter/article/254296/open_source_is_driving_business_app_development_survey_finds.html.

119. *Id.*

120. *Id.*

121. Kozubek, *supra* note 4, at 32.

122. *Cf.* MEEKER ON OPEN SOURCE, *supra* note 36, at 119 ("It is not crystal clear yet whether having a written corporate open source policy is a best practice . . . . While policies have certain settled legal effects in other areas of law . . . , the effect of policies in the open source arena is untested.").

balance between the many benefits of using open source, while implementing a structured program to manage the risks and consequences of violating open source licenses."[123] A basic corporate open source policy should generally include:

> (1) corporate open source philosophy;
>
> (2) black lists, white lists, and grey lists—describing which licenses are acceptable, not acceptable, or require clearance on a case-by-case basis;
>
> (3) notices to be included with company products;
>
> (4) patent considerations—explaining how using open source code fits into company's overall patent strategy and policy;
>
> (5) source code check-in and storage considerations—a major part of a policy establishing procedures of acquiring and using source code;
>
> (6) procurement guidelines—focusing on interactions among engineering, legal, and purchasing departments in acquiring third-party products and tools; and
>
> (7) reversioning—explaining company's position on employee's contribution to open source projects.[124]

To ensure compliance with these policies, it is also necessary to conduct periodic trainings for software-related staff.[125] Wide availability of the open source code increases the possibility of its importation into a proprietary product without legal oversight. Providing software developers with information regarding allowed or prohibited uses of upstream open source code would successfully minimize any such risks. Software developers certainly have

---

123. OLLIANCE GROUP SURVEY, *supra* note 21, at 8.

124. *See* MEEKER ON OPEN SOURCE, *supra* note 36, at 120-22. The book also contains a very helpful sample of an open source corporate policy. *Id.* at 123-34.

125. In this context, the company itself, through its officers and managing personnel, must comply with the established policies to avoid any risks that may arise from its own non-compliance. *See. e.g.*, Kelly v. Sky Angel, U.S., LLC, 1:09-CV-59, 2010 WL 2776580 (E.D. Tenn. July 14, 2010) (the company did not follow up on employee's concerns about possible GPL violations, which led the employee to voice his concerns by posting on the Internet and notifying all Linux copyright holders and developers of alleged copyright violations). While sometimes management pressure to comply with open source policies may be met with resistance by software developers, there are two possible solutions for this problem. First, compliance with open source policies may be integrated with an effort to participate responsibly in the open source movement (rather than just to avoid liability). *See* MEEKER ON OPEN SOURCE, *supra* note 36, at 74. Second, legal department may hold educational sessions that allow engineering leadership to express their views on open source; "[i]n such sessions, engineers and lawyers usually find much common ground." *Id.*

considerable freedom to use all available tools necessary to create their work, but this freedom should be exercised in a reasonable and responsible manner.[126]

The best OSS practices may prescribe certain conduct for a certain category or categories of OSS licenses. For example, a company could permit use of any source code licensed under non-copyleft without restrictions, but require that the development team keeps track of the type of license used and other necessary information should a legal issue arise.[127] Alternatively, a company could proscribe completely the use of software code covered by a specific license to avoid any exposure to possible risks;[128] the company could require pre-approved by the legal department only in exceptional circumstances.[129] A company's open source policy may provide additional steps of various complexities in the event that any other licensing issues arise.[130] The policy should, at a minimum, contain a provision for mandatory clearance by the legal department of any instance of use of open source code when the type of license is uncommon or undetermined.[131]

---

126. *See* Fontana, *supra* note 11, at 98. Some companies have their engineers use special forms to request inclusion of open source code in a product. *See* MEEKER ON OPEN SOURCE, *supra* note 36, at 78-81 (an example of such form).

127. Many companies establish Open Source Review Boards or Program Offices that typically include representatives from management, legal, and engineering departments and administer OSS policies and procedures. *See* OLLIANCE GROUP SURVEY, *supra* note 21, at 8.

128. *See id. See also* MEEKER ON OPEN SOURCE, *supra* note 36, at 75.

> Organizations can be idiosyncratic when they decide which licenses are unacceptable for use. . . . The bottom line is that every organization has its own calculus for this classification, based more on the characteristics of the organization than those of the licenses, the key factors being the extensiveness and value of the company's [intellectual property] portfolio and its tolerance for risk.

Id.

129. MEEKER ON OPEN SOURCE, *supra* note 36, at 78 ("Lawyers should be available to review difficult cases, but the objective is to create policies that avoid legal review except in edge cases.").

130. *See, e.g.*, *id.* at 74-75. For this purpose, most companies typically classify open source licenses into the following three categories: (1) always approved (usually including many permissive licenses), (2) never approved, (3) requiring legal review (usually including GPL and LGPL licenses). *Id.*

131. DejaCode Enterprise License Library—an online repository of proprietary and open source software licenses—is a "useful tool to get a reality check on the licenses you don't see every day." Heather Meeker, *DejaCode—Everything You Wanted to Know about Open Source Licenses*, COPYLEFT CURRENTS (July 6, 2012, 7:51 PM), http://www.heathermeeker.com/news/2012/7/6/dejacode-everything-you-wanted-to-know-about-open-source-lic.html. The company is also about to launch Component Catalog, which will list "public open source and proprietary software components with detailed metadata for each component, including origin, license, technology, and functionality." *See* DEJACODE,

Lawyers also need specific training in OSS to strengthen their ability to assist in compliance efforts. Although the legal department may occasionally use technical assistance from developers to understand the complex structure of a program or product and the interdependencies of its parts,[132] general knowledge of OSS license types and requirements, programming practices in OSS community, and how to extract licensing information from source code[133] is required for every lawyer dealing with software licensing issues.

2. Recordkeeping, Due Diligence, and Reporting

In addition to implementing a sound OSS policy and training personnel, an important part of a company's preventive practices should include good recordkeeping, which will result in effective due diligence and timely reporting of potential violations.[134] Another important consideration is avoiding duplicate work for any future projects—a centralized approach to recordkeeping will streamline the compliance process and minimize its costs.[135]

Due diligence in the context of software development ensures that the software is not combined "in ways that will violate the licensing terms that apply to each [component]."[136] To avoid violations, the developer must make sure that the "'inbound' rights—

---

http://www.dejacode.com/ (last visited Apr. 6, 2013).

132. It is extremely important to know which license covers a particular work, not only in context of "derivative" or "modified" works, but also in the context of "based on" works and software linking, when a software component makes a call to another program. The GPLv.2, for example, provides that "output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does." GPLv.2, *supra* note 33, § 0. Therefore, depending on the level of interactions between different parts of a software program, software libraries, and other components, the GPL may or may not apply. *See id.* § 2 ("If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works."); *see also* Brian Fitzgerald et al., *Legal Issues Relating to Free and Open Source Software*, 12 J.L. & INF. SCI. 159, 183-84 (2001). *But cf.* ROSEN ON OPEN SOURCE, *supra* note 3, at 115-18 (arguing that "linking" makes no difference in applying GPL) *and* LGPLv.2.1, *supra* note 46, Preamble, ¶9 (stating that the specific purpose of the LGPL is to allow "linking" of the proprietary software to the open source libraries). *See generally* LINDBERG, *supra* note 12, at 226-38; Malcolm Bain, *Software Interactions and the GNU General Public License*, 2 INT'L FREE & OPEN SOURCE SOFTWARE L. REV. 165 (2010), *available at* http://www.ifosslr.org/ifosslr/article/view/44/74.

133. For step-by-step instructions on how to extract license information see Fontana, *supra* note 11, at 100-02. For a practical approach see MEEKER ON OPEN SOURCE, *supra* note 36, at 76-77.

134. *See generally* MEEKER ON OPEN SOURCE, *supra* note 36, at 71-82.

135. *See id.* at 82.

136. *Id.* at 54.

the rights granted from others to [the developer]—are as broad as [the developer's] 'outbound' rights—the rights [he or she has] exercised, or granted to others."[137] The basic approach to due diligence is implementation of two steps: information gathering and legal analysis.[138]

Basic recordkeeping is vital in gathering information regarding inbound source code and licenses that cover it. This "provenance method consists in recording and following an audit trail, based on internal records, of what code is in the code base; determining what licenses cover each element . . . ; and discovering how the code is used in the code base . . . ."[139] While seemingly fragile and unreliable, this approach may be more accurate than it seems when "[m]uch open source code is used with little or no modification."[140] If the code is integrated into a larger product with unchanged file names, it is relatively easy to discover applicable licenses through basic online research.[141] Of course, heavily modified source code that is fully integrated into a software module would require code scanning, because recordkeeping would be of little help.[142]

Some copyleft licenses explicitly require distributing "complete" and "corresponding" source code;[143] thus, the use of a revision control system as part of "provenance checking" is crucial.[144] A revision control system, also called a version control or source control system, is a database that tracks and stores changes made to a collaborative project.[145] It is usually a part of a fully automated configuration management system, which includes source code and all related documentation, detects all components used to build executable programs, and is able to recreate each build and earlier environments in order to maintain previous versions of a product.[146] A revision

137. *Id.* For a more detailed review of due diligence process and very helpful illustrations of process components see *id.* at 54-57.

138. *Id.* at 71.

139. *Id.* at 72.

140. *Id.* at 73.

141. *Id.*

142. *Id.*

143. *See, e.g.*, GPLv.2, *supra* note 33, § 3.

144. *See* SOFTWARE FREEDOM LAW CTR., *Managing Copyright Information within a Free Software Project*, at 1-2 (Sept. 17, 2012), *available at* http://softwarefreedom.org/resources/2012/ManagingCopyrightInformation.pdf.

145. *See Definition of: Version Control*, PCMAG.COM ENCYCLOPEDIA, http://www.pcmag.com/encyclopedia/term/53750/version-control (last visited Apr. 6, 2013).

146. *See Definition of: Configuration Management*, PCMAG.COM ENCYCLOPEDIA, http://www.pcmag.com/encyclopedia/term/40233/configuration-management (last visited Apr.

control system also aids in establishing and monitoring of a "clean software baseline," which includes an inventory of all open source software approved for use.

Code scanning tools, which identify program components that may include open source code and flag potential violations, are also invaluable in performing periodic internal due diligence checks.[147] Such tools typically range from very simple (e.g., those that scan only copyright notices) to very complex (e.g., those that perform comparison between the code being scanned and an independent database of licenses known to cover that code).[148]

Regardless of which method—recordkeeping or code scanning—one uses, all flagged items should be investigated, and full reports on the potential issues should be forwarded to the legal department for clearance.[149] In addition to facilitating the process of measuring the extent and gravity of possible violation, as well as determining the value of intellectual property at risk, timely and comprehensive reporting procedures ensure efficient and speedy compliance and implementation of appropriate remedial measures.

Specific procedures detailing the recordkeeping, conducting due diligence, and reporting would depend on many different factors and may involve a collaboration of legal, operations, IT, engineering, and other departments. There are many useful practical guides on how to structure and implement monitoring and preventive mechanisms.[150]

3. Open Source Insurance

In recent years, some companies have begun offering a range of

---

6, 2013).

147. A number of companies offer source code scanners, as well as a variety of other compliance and audit tools and related services. Some of such companies are: Antelink (ANTELINK, http://www.antelink.com (last visited Apr. 6, 2013)); Black Duck (BLACK DUCK, http://www.blackducksoftware.com (last visited Apr. 6, 2013)); OpenLogic (OPENLOGIC, http://www.openlogic.com (last visited Apr. 6, 2013)); Palamida (PALAMIDA, http://www.palamida.com (last visited Apr. 6, 2013)); and Protecode (PROTECODE, http://www.protecode.com (last visited Apr. 6, 2013)). A number of compliance tools are also available from the Linux Foundation. *See Compliance Tools*, THE LINUX FOUND., http://www.linuxfoundation.org/programs/legal/compliance/tools (last visited Apr. 6, 2012).

148. MEEKER ON OPEN SOURCE, *supra* note 36, at 72.

149. For example, Microsoft, with numerous distinct business units, has developed a structure where open source compliance is managed by expert legal staff in each unit, and an open source issue elevates to the corporate department only if it cannot be managed internally within that unit. *See* OLLIANCE GROUP SURVEY, *supra* note 21, at 9.

150. *See, e.g.*, Ibrahim Haddad, *Open Source Compliance: Getting Started Guide*, OPEN SOURCE MAG. (Jan. 10, 2010, 2:00 PM), http://opensource.sys-con.com/node/1181405; Kuhn et al., *supra* note 68.

indemnification and insurance products, although they seem not to be gaining popularity. The most notable example of an insurance solution focusing specifically on OSS is the product offered by Open Source Risk Management (OSRM).[151]

In 2004, OSRM announced its plans to protect users of OSS from potential licensor's copyright infringement claims. OSRM offered an insurance-like protection and claims to provide "the industry's first and only vendor-neutral open-source indemnification . . . for around 3 percent a year of the maximum desired coverage."[152] A year later, OSRM, together with Kiln PLC of Lloyd's of London as an underwriter, and Miller Insurance Services Limited, a Lloyd's of London broker, introduced the Open Source Compliance Insurance, which became "the world's first insurance policy to cover the specialized risks faced by enterprises that include or rely upon elements of . . . open source software in their commercial products or internal IT infrastructure."[153] It provided up to $10 million for the following:

> • Loss of profits resulting from a legal settlement preventing the use or sale of the insured's product(s) resulting from the requirement to distribute certain code or products, in compliance with an Open Source software license;
>
> • The impaired valuation of an acquisition agreement or adjusted sale price thereof, resulting from the requirement to distribute code or products exchanging Open Source software in compliance with an open [sic] Source software license;
>
> • Costs to repair or replace code so that it complies.[154]

Despite some skepticism and uncertainty about its utility,[155] the OSRM insurance policy addresses some potential open source risks.

---

151. *See Open Source Compliance Representation and Warranty Insurance*, OSRM (Jan. 2007), http://www.osriskmanagement.com/downloads/OSRM_PROTECT_Jan07.pdf.

152. *See OSRM Launches Insurance-Like Protection, Opens OSLD Center*, AG-IP-NEWS (Apr. 20, 2004), http://www.ag-ip-news.com/news.aspx?id=18813.

153. *See First-Ever Open Source Compliance Insurance Now Available Through Partnership Between London-Based Lloyd's Underwriter Kiln, Lloyd's Broker Miller and Open Source Risk Management*, PR NEWSWIRE (Oct. 31, 2005), http://www.prnewswire.com/news-releases/first-ever-open-source-compliance-insurance-now-available-through-partnership-between-london-based-lloyds-underwriter-kiln-lloyds-broker-miller-and-open-source-risk-management-55667247.html.

154. *Open Source Compliance Representation and Warranty Insurance*, *supra* note 151.

155. *See, e.g.*, James Grimmelmann, *Great Ideas Dept.: Open Source Insurance*, LAWMEME (Mar. 17, 2005, 6:37 PM), http://lawmeme.research.yale.edu/modules.php?Newsid=1379.

Although it is certainly "not cheap"[156] and is "not a comprehensive policy and does not mitigate all IP and other legal risks,"[157] it may be a useful tool in combination with other components of overall compliance strategy.[158] Currently, however, there is very little—if any at all[159]—activity in the open source insurance industry, as most companies either rely on contractual indemnities[160] or bear risks on their own.[161]

B. External Preventive Mechanisms

1. Due Diligence and Full Disclosure

A company must scrutinize any software procured from third parties, including commercial vendors.[162] The corresponding source code should undergo the due diligence process, ideally by both the vendor and the acquiring company, regardless of whether it is proprietary or open source. Even proprietary software formally obtained from a respectable vendor may include noncompliant open source code, of which the vendor may or may not be aware of.[163] Thus, involvement of purchasing personnel or contract administrators—even if limited only to information-sharing—can be very helpful; although open source code generally requires no fee payment, "it is sensible to handle its acquisition through the same channels that the organization generally uses"[164] for obtaining third party software.

A company must implement due diligence—focusing on

---

156. James G. Gatto, Client Alert, *Pioneering Open Source Compliance Insurance Product*, PILLSBURY WINTHROP SHAW PITTMAN LLP, Nov. 2, 2005, at 1, *available at* http://www.pillsburylaw.com/siteFiles/Publications/4F77DDBB5EE1EAC97CB9A4745AE615B3.pdf.

157. *Id.*

158. *See id.*

159. OSRM no longer offers its products and apparently does not transact business in its home state of North Carolina. *See* Application for Certificate of Withdrawal of Open Source Risk Management, Inc., No. C201235400853 (N.C. Dep't of the Sec'y of State Dec. 31, 2012), *available at* http://www.secretary.state.nc.us/corporations/Filings.aspx?PItemId=6545239.

160. *See, e.g.*, *Open Source Assurance*, RED HAT, http://www.redhat.com/rhel/details/assurance/ (last visited Apr. 6, 2013).

161. *See* E-mail from Heather J. Meeker, *supra* note 6.

162. This includes commercial transactions, mergers and acquisitions, and other relevant contexts. *See generally* MEEKER ON OPEN SOURCE, *supra* note 36, at 237-44.

163. For example, Microsoft acquired over 500 million lines of software code in 2008. Every acquisition contained from ten to ninety percent of third party code, both OSS and proprietary. Of the third party code, approximately one third was covered by a reciprocal license. *See* Markwith, *supra* note 19, at 33.

164. MEEKER ON OPEN SOURCE, *supra* note 36, at 81.

determining relevant licensing information—as early as possible, especially if the inbound code will be embedded in a product.[165] This will allow for ample time to resolve compliance problems, if any arise during the product development cycle. Untimely due diligence and inconsistent compliance efforts increase the possibility of a license violation surfacing only at the late stages of product development—if at all—and may lead to a stalled production cycle, inability to release a product on the market on time, or an obligation to pull a product off the shelves. All this may translate into higher development costs, production and product shipment delays, and ineffective and untimely remediation of compliance problems.[166]

The best recordkeeping, due diligence, and reporting practices used for software developed in-house are equally applicable to the software procured from third party vendors or through mergers and acquisitions. Because any distributor of a product that includes OSS is responsible for full compliance,[167] it is necessary to know not only whether the third party software includes open source components, but also how it is used and whether it satisfies the license obligations.[168]

Thorough due diligence is extremely important because the third party software is usually not controlled by the same compliance policies and monitoring tools used in-house. Therefore, as a practical matter, knowledge of upstream OSS compliance procedures, both vendors' and their suppliers' and as farther upstream as possible, is extremely helpful in making initial determination of the applicable level of scrutiny. Comparable or even more stringent compliance tools used by a third party may not lessen the thoroughness of due diligence, but the absence of effective procedures should warrant full review and comprehensive audit of the incoming software.

It is also a good practice to facilitate the third party's full disclosure of all licenses and any relevant documentation relating to its software, as well as identifying all upstream developers or distributors.

Recently, there have been attempts to standardize sharing of

165. A recent history of copyleft enforcement shows that the majority of lawsuits involved distribution of OSS-licensed object code in embedded Linux-based products without complying with the license requirements regarding the availability of the source code or copyright notices. *See, e.g.*, cases cited *supra* notes 89-92.

166. *See* Fontana, *supra* note 11, at 99.

167. For a discussion regarding what is and is not considered a "distribution," see MEEKER ON OPEN SOURCE, *supra* note 36, at 233-36.

168. *See, e.g.*, Haddad, *supra* note 150.

license components and copyrights associated with a software program. The Software Package Data Exchange (SPDX) specification, hosted by Linux Foundation,[169] "reduces redundant work by providing a common format for companies and communities to share important data about software licenses and copyrights, thereby streamlining and improving compliance."[170] An SPDX file is included with every software project and contains specific data including version number and applicable licenses.[171] In May 2012, the Linux Foundation announced the availability of Free and Open Source Software Bar Code Tracker,[172] which is designed to simplify the way OSS components are tracked and reported.[173] It uses an auto-generated, custom QR code that contains comprehensive information about the product.[174] Although "the effort is ambitious, and may aptly be compared to herding cats or solving the meaning of life," this project, if successful and universally accepted, may solve "the biggest challenges in open source today."[175]

2. Warranty and Indemnification

Software obtained from or through a third party may include open source code which could be licensed under any OSS license, including the GPL. Hence, it is extremely important to take such concerns into account when negotiating software-related agreements. For example, as discussed earlier, the third party should fully disclose whether the provided software contains any open source code. "[A]greements for bringing software into an organization whose software utilization strategy" either does not accept or limits use of OSS "should include representations and warranties by the software

169. THE LINUX FOUND., http://www.linuxfoundation.org/ (last visited Apr. 7, 2013).

170. *See About SPDX*, SOFTWARE PACKAGE DATA EXCHANGE, http://www.spdx.org/ (last visited Apr. 7, 2013). *See generally* Phil Odence & Kate Stewart, *A Common Software Package Data Exchange Format: 1.0 Release Update and Discussion*, THE LINUX FOUND, http://www.linuxfoundation.org/sites/main/files/publications/lf_foss_compliance_spdx.pdf (last visited Apr. 7, 2013).

171. *See* Julie Bort, *Linux Foundation Releases Specification to Ease Licensing Headaches*, NETWORKWORLD (Aug. 18, 2011, 10:38 AM), http://www.networkworld.com/news/2011/081811-linux-foundation-spdx-249857.html.

172. *See Compliance Tools*, *supra* note 147.

173. Darryl K. Taft, *The Linux Foundation Launches Open-Source Compliance Tool*, EWEEK.COM (May 30, 2012), http://www.eweek.com/c/a/Linux-and-Open-Source/The-Linux-Foundation-Launches-Open-Source-Compliance-Tool-305410.

174. *Id.*

175. Heather J. Meeker, *SPDX and the Meaning of Life*, NEW MATTER, Summer 2012, at 30.

provider to that effect."[176] As an added safeguard, a recipient of software who wants to be absolutely sure that the incoming software either contains no, or only certain types of, OSS, may demand an open source audit of the software along the lines described above, and certification by the software provider that it meets the software recipient's expectations in this regard.[177]

Indemnification against any alleged violations of third party copyright, patent or trade secret rights is a common provision in a software license, and helps significantly reduce risks associated with procurement of software from a third party.[178] It should spell out the rights to control the defense, the payment of legal fees and expenses, and the payment of all adjudicated claims. In recent years, many large proprietary software companies have offered broad protection against infringement claims against their products.[179] Although indemnification by itself cannot completely negate having non-compliant OSS code in a product received from a third party, it does nonetheless provide a powerful incentive for software vendors to exclude such instances; after all, the potential exposure could result in a sizeable monetary loss for a vendor.

IV. REMEDYING VIOLATIONS AND COMPLYING WITH OSS LICENSES

A company implementing the preventive measures described above should be in good position to ensure that the intermixing of code is avoided; there is, however, no guarantee it will not happen. Once and as soon as the issue is identified and properly reported, it is the legal department's job to determine the type of license, its terms, and the specific code the license covers. These determinations, in conjunction with other factors, will guide how to handle the situation and what steps to take either to remedy the situation (if at all possible), or to comply with copyleft requirements.

176. *See* Rehm, *supra* note 2, at 320 (footnote omitted).

177. *See id.*

178. *See* Greg R. Vetter, *Exit and Voice in Free and Open Source Software Licensing: Moderating the Rein Over Software Users*, 85 OR. L. REV. 183, 208-09 (2006).

179. *See, e.g.*, Ronald J. Mann, *Commercializing Open Source Software: Do Property Rights Still Matter?*, 20 HARV. J.L. & TECH. 1, 42 n.174 (2006). *See also* MEEKER ON OPEN SOURCE, *supra* note 36, at 242 ("There is no clear industry custom here; some licensors bear risk for open source components and some do not.").

A. Remedial Efforts and Other Considerations

1. Likelihood of Enforcement

There are generally two broad categories of licensees: those who operate within the OSS community, and those who operate outside of it.[180] There are also two distinct ways of enforcing compliance with OSS licenses: community enforcement and judicial enforcement, with the latter rarely applied to "insiders."[181] A related consideration is the kind of claimant who might pursue an enforcement action: an advocacy organization, an individual author, or a private enterprise.[182]

Most commercial enterprises concerned about open source risks have historically been the "outsiders," and have harbored antagonism towards OSS.[183] In recent years, however, attitudes have changed.[184] Some companies embraced developing OSS products in parallel with proprietary software;[185] others use "dual-licensing,"[186] which has proved to be a viable business model.[187] Yet others have made significant contributions to the OSS community, in the form of patented or copyrighted intellectual property—including closed source code[188]—or by providing financial and other support for

---

180. *See supra* Part III.

181. *See* ROSEN ON OPEN SOURCE, *supra* note 3, at 282 ("[T]he open source community is not particularly litigious. Licensors give away so many copyright and patent rights that there's very little left of value worth suing over."). Regardless of infringer's status in the OSS community, however, a number of substantive and procedural considerations must be addressed to bring a successful suit. *See generally id.* at 269-83.

182. *See generally* Meeker, *supra* note 111, at 287-90.

183. *See, e.g.*, Nadan, *supra* note 1, at 371-73 (discussing Microsoft's "war on Linux").

184. Despite continuing integration of OSS and proprietary software, there still a significant tension between the two. *See, e.g.*, Florian Mueller, *Red Hat Feeds the Patent Trolls and Fools the FOSS Community*, FOSS PATENTS (Mar. 14, 2011, 5:25 PM), http://www.fosspatents.com/2011/03/red-hat-feeds-patent-trolls-and-fools.html.

185. *See, e.g.*, *Free and Open Source Software*, ORACLE, https://oss.oracle.com (last visited Apr. 7, 2013).

186. *See infra* Part IV.A.4.

187. *See, e.g.*, *MySQL, Sleepycat, and Trolltech Say They Prove Strength of Dual-License Model*, LXER LINUX NEWS (Mar. 16, 2004, 3:59 AM), http://lxer.com/module/newswire/view/7172 ("Sleepycat Software, Trolltech AS and MySQL AB today jointly announced that 2003 software license revenues for the companies increased an average of 65 percent over the previous year, largely due to their use of the dual-license business model. . . . Under this model, vendors offer their products under both an open source license and a commercial license.").

188. *See, e.g.*, *IBM Pledges 500 U.S. Patents to Open Source in Support of Innovation and Open Standards*, IBM (Jan. 11, 2005), http://www-03.ibm.com/press/us/en/pressrelease/7473.wss; *Sun Grants More Than 1,600 Patents to Open Source Community*, COVER PAGES (Jan. 25, 2005), http://xml.coverpages.org/SunPatents1600.html. More recently, Google pledged the free use of

certain OSS projects.[189] In this context, OSS licensors are least likely to bring an enforcement action against an active and valuable supporter of OSS community or its particular upstream project.[190] It is also possible that in some instances a commercial enterprise having especially friendly and mutually beneficial relationship with a particular OSS licensor may obtain special permission to use code beyond what was provided by the licensing terms.[191]

2. Good Faith Efforts to Comply

Good faith efforts to comply, willingness to seek amicable resolution, and an open and forthcoming attitude may not themselves remedy the violations, but will set a positive tone in the remedial efforts. The GPLv.3, while prescribing automatic termination of rights upon occurrence of a violation, explicitly provides for curative opportunities.[192] A license from a particular copyright holder is reinstated provisionally if the violator discovers the wrongdoing and "cease[s] all violation[s];" it is restored permanently if the copyright holder fails to act to terminate the license within 60 days after the cessation.[193] Furthermore, the rights under the license may also be permanently restored if a violator corrects the first instance of wrongdoing within 30 days after the copyright holder provides notice of the violation.[194] Taking advantage of cure opportunities may result in automatic restoration of rights and escaping litigation. GPLv.3 is thus less likely to be judicially enforced than GPLv.2, which does not provide for curative opportunities.[195]

It is possible that partial compliance may also suffice and be allowed by a particular licensor. The outcome will depend on the scope and scale of the non-compliance, relationships between the

certain of its patents to open source community on the terms of "non-assertion": Google would not assert its patents unless sued first. *See Google Issues Open Source Patent Pledge: We Won't Sue First*, VentureBeat (Mar. 28, 2013, 9:50 AM), http://venturebeat.com/2013/03/28/google-issues-open-source-patent-pledge-we-wont-sue-first/. *See also Patents in the Service of Open Source*, GOOGLE, http://www.google.com/patents/opnpledge/pledge/ (last visited Apr. 7, 2013).

189. *See, e.g.*, IBM, IBM IS COMMITTED TO LINUX AND OPEN SOURCE (2010), *available at* http://public.dhe.ibm.com/common/ssi/ecm/en/lxb03001usen/LXB03001USEN.PDF.

190. *See* Fontana, *supra* note 11, at 107-08. *See also* OLLIANCE GROUP SURVEY, *supra* note 21, at 13-14. Obviously, this is something no one can or should count on, and the copyright holder's decision to file a lawsuit or pursue alternative enforcement efforts will depend on variety of factors, not only this sole consideration.

191. *See* Fontana, *supra* note 11, at 108.

192. *See* GPLv.3, *supra* note 24, § 8.

193. *See id.*

194. *See id.*

195. *See* Fontana, *supra* note 11, at 107.

licensor and the violator, licensor's inclination to pursue both community and judicial enforcements, and many other factors. Combining these considerations with the fact that most lawsuits filed in the recent years have settled,[196] it may be safe to say that if a violator has significant monetary or other interests in the potentially infringing product, a convincing offer that includes partial compliance may be accepted by the licensor. Partial compliance, for example, may include a stipulated agreement excusing the alleged infringer's compliance with the particular violation, but require a commitment to full compliance in the future—possibly also bolstered by a liquidation damages clause.

3. Rewriting, Contributing, or Internal Use

If the development team or the legal department caught intermixed code before the end product was finished, developers could possibly rewrite the infringing OSS or extract the code.[197] Although this route will likely involve significant cost increases, it is nevertheless possible for projects in their early stages.

Developers can rewrite the code at any time before distribution of the finished product. A company will want to analyze the cost effectiveness of replacing each instance of open source code, but removing and rewriting such code in proprietary software takes the product out of the scope of OSS licenses, and is thus generally advisable. Monitoring and preventive mechanisms described in Part III are extremely important tools for identifying the occurrence of intermixing, and may reduce the costs and labor significantly if implemented in the beginning of the project.

When the intermixing has occurred at an early stage of the project and rewriting the code is either too costly or undesirable for other reasons, a company may choose to abandon the developed parts of the program and contribute its work to OSS developers.[198] In this case, the components that have been developed may be released as open source, taking into consideration possible exposure and loss of valuable copyrighted material, trade secrets, etc.[199] Subject to

196. *See supra* section II.B.

197. *See* Nadan, *supra* note 1, at 360 n.49.

198. There have been arguments that, since the U.S. Copyright Act of 1976, 17 U.S.C. § 103(a) (2011), provides that the infringer does not own the infringing derivatives and does not specify what happens to them, such derivatives may possibly "fall into the public domain, free of the claims of the underlying [copyleft] owner" and become free to be taken private by anyone (except the infringer). *See* Nadan, *supra* note 1, at 371.

199. *See generally* MEEKER ON OPEN SOURCE, *supra* note 36, at 135-51.

clearance by the legal department, such release, however, may become a valuable contribution to OSS community and may help building good relationships with upstream OSS developers.[200]

If the incorporated open source code cannot be rewritten and the software contains valuable intellectual property, thus preventing its release to public, internal use may be the only option (e.g., for testing purposes). As opposed to the downstream *distribution*, the *use* of OSS generally does not involve compliance with any "intricate conditions of [OSS] licenses, or warranties, or patent defenses, or other esoteric legal issues"[201] because the code never leaves the confines of the company. All open source software and any copies of that software "can be freely *used* by anyone [within the company], anywhere [within the company], for any purpose whatsoever [short of distribution]" and without concern about the specific license terms.[202]

This general rule, however, applies unconditionally only to unmodified OSS works. In the context of modified and derivative works, internal use may or may not be permitted depending on the specific terms of the license. A GPL, for example, permits internal use of a product that incorporates both the proprietary code and the GPL-covered code.[203] Some licenses, however, specifically prohibit any use—including internal—of the modified software.[204] Of course, most software development enterprises are not in business of producing software products for their own use, so this option may not be available for them.

---

200. *See, e.g.*, *Licenses*, *supra* note 15 ("When we explain to the employer that it is illegal to distribute the improved version except as free software, the employer usually decides to release it as free software rather than throw it away.").

201. *See* ROSEN ON OPEN SOURCE, *supra* note 3, at 49.

202. *See id.* (emphasis added).

203. *See, e.g.*, GPLv.3, *supra* note 24, § 0 (GPL's definition of the term "propagate" does not include "modifying a private copy"); *Frequently Asked Questions about Version 2 of the GNU GPL*, GNU OPERATING SYS., http://www.gnu.org/licenses/old-licenses/gpl-2.0-faq.html#GPLRequireSourcePostedPublic (last visited Jan. 8, 2013).

> The GPL does not require you to release your modified version. You are free to make modifications and use them privately, without ever releasing them. This applies to organizations (including companies), too; an organization can make a modified version and use it internally without ever releasing it outside the organization.

Id. See also LINDBERG, *supra* note 12, at 225 ("[A] company using an embedded Linux system to control its manufacturing operations can compile proprietary source code into the Linux kernel and use the resulting binary without permission and without violating the GPL.").

204. For example, RPL requires that all "changes, bug fixes, extensions, etc. must be made available to the open source community at large when you Deploy in any form—either internally or to an outside party." RPLv.1.5, *supra* note 31, Preamble. The only uses that are permitted are for research and personal non-commercial purposes. See *id.* §§ 1.2, 1.11, 1.13.

4. Purchasing a Commercial License

Sometimes an open source code licensed, for example, under the GPL, may also be licensed on commercial terms by an entity engaged in "dual licensing."[205] Under this model, the copyright owner of the original code releases source code under an open source license (e.g., with the goal to gain market share or to receive "peer-reviewed" improvements) and object code under a proprietary license (typically, to gain revenue).[206] In this situation—but only if there are no upstream developers and the licensing entity is the sole copyright owner—it is highly advisable to purchase a commercial license for the code that is or will be incorporated in the proprietary software.

Moreover, regardless of the business model used by the licensor, it is always advisable to approach the copyright holder of open source code in order to negotiate licensing of the code on proprietary terms.[207] The source code developer is free to do so because "he who writes the code gets to choose his license."[208]

B. Compliance

Each OSS license has a specific set of requirements, but the most common are (1) a notice requirement, and (2) a source code requirement. Although some copyleft licenses may contain clauses raising other legal issues, such as patent license grants,[209] such clauses do not require taking affirmative acts and, therefore, create no compliance concerns.

Compliance with OSS licenses typically includes satisfying both

---

205. *See generally* Vetter, *supra* note 178, at 224-26.

206. *See* MEEKER ON OPEN SOURCE, *supra* note 36, at 143-46; Vetter, *supra* note 178, at 224.

207. In a recent complaint for copyright infringement, one company acknowledged that purchasing a commercial license would effectively allow the defendant to avoid the lawsuit. *See* Complaint for Copyright Infringement at 4, Artifex Software Inc., v. Palm Inc., No. CV-09-5679-RS, 2009 WL 4813582 (N.D.Cal. Dec. 2, 2009) ("Consistent with its history and tradition, Artifex offers MuPDF to the public, free of charge, under the [GPL] for non-commercial use. But if a licensee wishes to use MuPDF in a way that does not comply with the GPL, Artifex requires the licensee to purchase a commercial license.").

208. *See* GLYN MOODY, REBEL CODE: INSIDE LINUX AND THE OPEN SOURCE REVOLUTION 266 (Basic Books 2002) (2001) (quoting Linus Torvalds, original developer of Linux kernel). *See also* MEEKER ON OPEN SOURCE, *supra* note 36, at 63 ("[A]uthors are the ultimate arbiters of what licensing terms will apply to their code.").

209. *See*, *e.g.*, EPLv.1.0, *supra* note 48, § 2; GPLv.3, *supra* note 24, § 11; MPLv.1.1, *supra* note 47, § 2. "Patent license" in the open source context is an express agreement or commitment not to enforce a patent, such as an express permission to practice a patent or covenant not to sue for patent infringement; to "grant" such a patent license means to make such a covenant not to enforce a patent against the licensee. *See* GPLv.3, *supra* note 24, § 11.

notice and source code requirements. The process of preserving copyright notices and providing source code is generally pretty straightforward, although ambiguous language may make compliance efforts quite onerous.[210]

1. Notice Requirement

In general, the notice requirement is easier to comply with and poses no or very minimal risks.[211] It may be, however, close to impossible—or at least very time-consuming—to comply with every notice provision for code in a highly complex product.[212] Another difficulty a company may face is the method of delivery: a consumer may not like dozens of pages containing copyright licenses for every part of software, and posting such notices online would effectively put the product information and its components on public display.[213] An alternative approach would be to only list copyright notices for the former, and to set up limited customer-only web portal for the latter.[214]

As an example of a notice requirement by a non-copyleft license, the Berkeley Software Distribution (BSD) license requires that "[r]edistributions of source code must retain [a] copyright notice, [a] list of conditions and [a] disclaimer;"[215] redistribution in binary form must follow the same requirements.[216] A licensee does not have to do much more than simply copying the text of the license to a location where it may be accessible by a downstream user; in practice, however, it could be very time consuming.

Notice requirements for copyleft licenses are similar. The GPL license, for example, requires that verbatim copies of the source code

210. *See, e.g.*, ROSEN ON OPEN SOURCE, *supra* note 3, at 98-101 (characterizing the Artistic License as written like a "philosophical statement" that "a lawyer would have difficulty explaining and that a judge would probably not be able to understand"). Many licenses are written by software developers themselves, who "cringe when a lawyer attempts to write high-quality software [but] feel no qualms about writing their own open source licenses." *Id.* at 98. *See also* MEEKER ON OPEN SOURCE, *supra* note 36, at 48 ("The tendency of developers unfamiliar with licensing practice to write their own licenses has resulted in some legal conundrums."). While license terms written in plain English are easy to read and comprehend, the everyday language used may be insufficient to define precise legal terms and result in inconsistent interpretation.

211. *See* MEEKER ON OPEN SOURCE, *supra* note 36, at 83 ("Notice requirements are not complicated to interpret and are rarely the subject of litigation claims or disputes.").

212. *See id.* at 83-85.

213. *See id.* at 86-87.

214. *Id.*

215. BSD, *supra* note 64, cl. 1.

216. *See id.* cl. 2.

be conveyed with an "appropriate copyright notice" published on each copy "conspicuously and appropriately," that a copy of the license itself, and that all other applicable notices be kept intact.[217] Conveying a modified version of work under GPL, however, adds a requirement to include "prominent notices stating that you modified [the code], and giving a relevant date," and modifies a requirement of keeping the notices intact.[218] To make compliance easier, the FSF provides the full text of required notices, instructs the programmer to attach recommended notices "to the start of each source file to most effectively convey the exclusion of warranty" and also include "at least the 'copyright' line and a pointer to where the full notice is found;" it also advises to "add information on how to contact you by electronic and paper mail."[219] In practice, it is highly advisable to provide source code upfront, if possible: in addition to saving time with using "baked-in" notices that are already incorporated in the source code, it also deflects source code requests.[220]

An example of a significant notice requirement, which exceeds what is typically mandated, is a Reciprocal Public License's requirement that

> You must cause any Modifications that You create or to which You contribute to be documented in the Source Code, clearly describing the additions, changes or deletions You made. You must include a prominent statement that the Modifications are derived, directly or indirectly, from the Licensed Software and include the names of the Licensor and any Contributor to the Licensed Software in (i) the Source Code and (ii) in any notice displayed by the Licensed Software You distribute or in related documentation in which You describe the origin or ownership of the Licensed Software. You may not modify or delete any pre-existing copyright notices, change notices or License text in the Licensed Software without written permission of the respective Licensor or Contributor.[221]

Thus, although some OSS licenses demand a significant effort on the part of OSS users in order to comply with applicable notice

---

217. GPLv.2, *supra* note 33, § 1; GPLv.3, *supra* note 24, § 4.

218. GPLv.3, *supra* note 24, § 5(a)-(b). Some other OSS licenses also require that any modifications of the upstream source files must be identified as such. *See, e.g.*, ApLv.2.0, *supra* note 64, § 4(2); GPLv.2, *supra* note 33, § 2(a).

219. *See GNU General Public License*, GNU OPERATING SYS., http://www.gnu.org/copyleft/gpl.html (last updated Feb. 28, 2013) (below the heading "How to Apply These Terms to Your New Programs").

220. E-mail from Heather J. Meeker, *supra* note 6.

221. RPLv.1.5, *supra* note 31, § 6.2.

requirement, most of them do not. Notice requirements for both copyleft and non-copyleft licenses are similar. Although compliance could be burdensome and time consuming in some cases (e.g., with a complex product or distribution of binaries), it rarely creates significant difficulties.

2. Source Code Requirement

Non-copyleft licenses do not require making the source code available upon distribution of the software, merely permitting it. As a practical matter, however, it may be beneficial at least for companies working within the OSS community to make or offer to make the portions of the proprietary program containing the open source code available for downstream users.[222]

a. Strong Copyleft

The source code requirement of the GPL may be generally satisfied in two ways: (1) by distributing the source code with the product, or (2) by providing an offer to make the source code available upon request.[223]

Conveyance of the source code alongside the executable program is typically preferred because it satisfies the license requirements at the time of distribution. The source code may be embedded in the same product if it contains embedded software,[224] be included on the same or accompanying physical "medium customarily used for software interchange,"[225] or be downloaded from an Internet location, if executable software is made available through network distribution.[226]

The offer to provide source code is only available when distribution of an executable program is in a physical product or on a physical distribution medium.[227] Under GPLv.2, the offer must be valid for at least three years,[228] while GPLv.3 requires the offer be

222. *See* ROSEN ON OPEN SOURCE, *supra* note 3, at 80.

223. *See* GPLv.2, *supra* note 33, § 3; GPLv.3, *supra* note 24, § 6.

224. Embedded software is a set of instructions that permanently reside within a machine's or device's memory, as opposed to regular software programs that are stored on a disk and must be loaded for execution. *See Definition of: Embedded Software*, PCMAG.COM ENCYCLOPEDIA, http://www.pcmag.com/encyclopedia/term/42552/embedded-software (last visited Apr. 7, 2013).

225. GPLv.2, *supra* note 33, §3; GPLv.3, *supra* note 24, § 6.

226. GPLv.2, *supra* note 33, §3; GPLv.3, *supra* note 24, § 6.

227. *See* GPLv.3, *supra* note 24, § 6(b). *But see* GPLv.2, *supra* note 33, § 3(b), last para.

228. GPLv.2, *supra* note 33, § 3(b).

valid for "as long as you offer spare parts or customer support for that product model."[229] The GPLv.3 allows the offer to make the source code available solely from a network download location, while GPLv.2 requires that the offer convey the code by a physical medium.[230] While the offer option may be beneficial for use with an embedded product with limited storage capacity or with a product that is not accompanied by a storage medium, the downside is that it significantly increases the duration of compliance obligations: at least three years, and possibly even longer under GPLv.3.[231] The written offer may postpone the need to produce the source code until a downstream licensee requests it; this essentially reduces the general availability of the code, but increases the risk of non-compliance and does not satisfy the GPL obligations of downstream work.

Products with embedded software require extra attention because the executable programs will invariably contain a mixture of both copyleft-licensed and proprietary components.[232] It is therefore crucial to identify and distinguish the components, and identify which ones require compliance with the source requirement.

GPLv.3 also adds an extra caveat to compliance: it requires installation information to be supplied with a locked-down consumer device, where the software in the device is modifiable by the upstream party.[233] The required information must enable a skilled developer to install functioning, modified versions of the licensed components.[234] It has been suggested that the companies that may be subjected to comply with this requirement simply avoid using GPLv.3-licensed code altogether.[235]

b. Weak Copyleft

Weak copyleft generally allows the code covered by a copyleft license to be used as a library or linked-to code, covered by proprietary licenses.[236] Nevertheless, the source code licensed under weak copyleft must be compliant with the license requirements.

For example, LGPLv.3 allows for licensing a program that uses the work covered by LGPL (a library) under proprietary terms, as

229. GPLv.3, *supra* note 24, § 6(b).
230. GPLv.2, *supra* note 33, § 3(b); GPLv.3, *supra* note 24, § 6(b).
231. *See* Kuhn et al., *supra* note 68, at 5.
232. *See id.* at 9.
233. *See* GPLv.3, *supra* note 24, § 6.
234. *See id.*
235. *See* Fontana, *supra* note 11, at 106.
236. *See supra* Part II.A.2.

long as one licenses such work under LGPL terms.[237] The modifications of the Library itself must be licensed under the LGPL (or the GPL), however.[238] LGPLv.2.1 also requires that the entire executable must "permit modification of the work for the customer's own use and reverse engineering for debugging such modifications."[239] LGPLv.3 contains a similar, although a little narrower, requirement: the proprietary terms "taken together" must "not restrict modification of the portions of the Library contained in the Combined Work and reverse engineering for debugging such modifications."[240] The language of LGPLv.3 suggests that the reverse engineering may be prohibited for the proprietary components of the executable and that "a practice of non-enforcement of broader restrictions on modification and reverse engineering may be good enough."[241] In addition, similarly to GPLv.3, LGPLv.3 prescribes installation information be made available to downstream users "to the extent that [it] is necessary to install and execute a modified version of the Combined Work produced by recombining or relinking the Application with a modified version of the Linked Version."[242]

A separate issue specific to LGPL is the ten-line limitation on small macros and inline functions.[243] Macros and inline functions are prewritten and self-contained subroutines that are called for throughout a program and perform a specific, predefined task.[244] The LGPL creates an exception for such subroutines of "ten lines [of code] or less in length" and does not restrict their use "regardless of whether [they are] legally derivative work."[245] Thus, macros and

237. LGPLv.3, *supra* note 46 § 4.

238. *Id. at* § 2.

239. LGPLv.2.1, *supra* note 46, § 6.

240. LGPLv.3, *supra* note 46, § 4. *See also* MEEKER ON OPEN SOURCE, *supra* note 36, at 220-21 ("[W]hen licensing in a mixed-rights environment, . . . place an exception in the proprietary license . . . clarifying that any terms of the proprietary license that would conflict with an open source license covering included code will be governed by the open source license rather than the proprietary license.").

241. *See* Fontana, *supra* note 11, at 105.

242. LGPLv.3, *supra* note 46, § 4(e).

243. *See* LGPLv.2.1, *supra* note 46, § 5; LGPLv.3, *supra* note 46, § 3. *See also* MEEKER ON OPEN SOURCE, *supra* note 36, at 218-20; Ben Gertzfield, *The Ten-Line Weakness of the LGPL and the Effects on GTK+/GLib* (July 08, 1999, 2:36 PM), http://lists.debian.org/debian-devel/1999/07/msg00605.html.

244. *See Definition of: Function*, PCMAG.COM ENCYCLOPEDIA, http://www.pcmag.com/encyclopedia/term/43578/function (last visited Apr. 7, 2013); *Definition of: Macro*, PCMAG.COM ENCYCLOPEDIA, http://www.pcmag.com/encyclopedia/term/46458/macro (last visited Apr. 7, 2013).

245. *See* LGPLv.2.1, *supra* note 46 § 5.

inline functions that *are* longer than ten lines of code[246] would be covered by LGPL and must comply with the requirements of section 6.[247] Version 3 of the LGPL has a slightly different requirement: if the ten-line limit is exceeded, one needs only to "[g]ive prominent notice with each copy of the object code that the Library is used in it and that the Library and its use are covered by this License [and a]ccompany the object code with a copy of the GNU GPL and [the LGPL]."[248]

While a ten-line limit was pretty reasonable back in 1999, when the text of the license was written and the memory was a scarce resource, it is inconsistent with contemporary programming practices.[249] Because memory has become much cheaper to manufacture, "there is not so much pressure to save space [but] more pressure to speed up processing."[250] In addition, "[m]any engineers have commented that a 10-line limitation is . . . impossible to police, given the workings of modern development environments"[251] in which modern compilers[252] make automated decisions to optimize the efficiency of the code.

The provisions of other weak copyleft licenses are less detailed and less strict than LGPL's, but it is generally assumed that they do not permit written offers as an option for complying with the source code requirement.[253] For example, the MPL prescribes that the source code may either accompany the executable or be made available "via an accepted Electronic Distribution Mechanism . . . for at least twelve (12) months after the date it initially became available, or at least six (6) months after a subsequent version of that particular Modification has been made available."[254] The EPL requires that the instructions on

246. The LGPL does not provide any good reason why it arbitrarily uses "ten lines" as the threshold. It also does not specify whether the length of a line of code would have any bearing on applicability of this exception. Although typically a line of code is 80 characters or less long, a line can contain a whole program, and be thousands of characters long; most developers follow standard coding guidelines, however, and insert carriage returns in places that make the code both readable and readily understandable.

247. *See* LGPLv.2.1, *supra* note 46 § 5.

248. LGPLv.3, *supra* note 46 § 3.

249. MEEKER ON OPEN SOURCE, *supra* note 36, at 218-20, 264.

250. *Id.* at 219.

251. *Id.* at 264.

252. A compiler is a program that translates a source code, written in a high-level programming language (such as C or C++), into machine language that is readable by the computer. *See Definition of: Compiler*, PCMAG.COM ENCYCLOPEDIA, http://www.pcmag.com/encyclopedia/term/40105/compiler (last visited Mar. 15, 2013).

253. *See* Fontana, *supra* note 11, at 105.

254. MPLv.1.1, *supra* note 47, § 3.2. *See also id.* § 3.6.

how to obtain source code "in a reasonable manner on or through a medium customarily used for software exchange" should be included with a product.[255]

VI. CONCLUSION

Open source software has become ubiquitous. Businesses large and small capitalize on its broad availability and utility. OSS code and software components provide valuable and cost-effective solutions for a resourceful enterprise. Virtually all software developers currently use OSS code in their work, so every business enterprise needs to understand what code it is using and how. With proper acquisition controls, companies may not only safely use open source software, but also allow their employees to work with open source code in proprietary setting.

Among the wide variety of open source licenses the "copyleft" licenses—the General Public License being the most well-known—pose the most risk to proprietary software. Such licenses prescribe that any work "based on" the copyleft-licensed source code must also be licensed as copyleft. A company incorporating source code covered under a copyleft license into a proprietary product that is later distributed might be violating the terms of the license agreement and thus infringe on the licensor's copyright. Although judicial enforcement of OSS licenses has been sporadic at best, informal enforcement by members of OSS community have been proven to work, forcing infringers to settle before charges are brought.

Both the software developers and the lawyers are largely aware of the risks, but managing these risks can be an extremely daunting task for an enterprise of any size. Therefore, companies must have policies and procedures in place to police use of OSS. Identifying a potential violation, determining the applicable license terms, and complying with the license requirements should be central to a company's open source compliance efforts. A combination of both internal (covering in-house conduct) and external (covering third party conduct) mechanisms creates a robust and effective compliance program. The most important components are a comprehensive open source policy, periodic trainings of personnel, efficient recordkeeping and timely reporting of potential problems, due diligence, and requiring broad indemnity coverage from third-party vendors.

If a violation nonetheless occurs, there are ways to remedy the

255. EPLv.1.0, *supra* note 48, § 3.

situation. Although judicial enforcement may be available, open source licenses tend to be enforced informally and most disputes usually settle on terms favorable to each party involved. Since most open source licenses are violated when an infringing product is distributed, a company typically has a choice of rewriting the code, contributing it to an open source community under the applicable license, or possibly keeping it for internal use only. Often, however, it is also possible to negotiate favorable license terms with a copyright holder, unless the copyright holder already offers commercial licenses under dual licensing model. The developed product also may be released under applicable open source license, including copyleft licensed that require release of the source code—although terms and provisions of every specific license will inevitably vary and must be carefully analyzed.

Thus, although the risks associated with OSS are not minimal, they are generally known and are manageable. A company may implement a number of control mechanisms to effectively filter incoming OSS components and to monitor their use in a proprietary setting. If a violation nevertheless occurs, there are steps a business may take to either remedy the violation or comply with the licensing requirements.