OPEN SOURCE SOFTWARE LICENSING

Steve H. Lee*

[*Pre-Publication Version (as of 4/28/1999):* NOTE – This paper is still in DRAFT form. Therefore, there may be some corrections and adjustments that need to be made to various parts of this paper. This paper ***does not*** – in any way – constitute legal advice.]

TABLE OF CONTENTS [TENTATIVE]

---

* JD, Harvard Law School, Class of 2000. The author would like to thank Lawrence Lessig, the author's faculty advisor. The author is extremely grateful to Bruce Perens – the author of the 'Open Source Definition' and the 'Debian Free Software Guidelines' – for his helpful responses to my questions. A special thanks goes to Pamela Samuelson, of Boalt Hall, for her helpful comments regarding what was formerly known as UCC Article 2B. The author would also like to thank Alex Macgillivary [and others?] for helpful comments on earlier drafts of this paper. The author also thanks Christopher O.B. Wright, Esq., of Cooley Godward LLP [and others?] for helpful comments and suggestions along the way. Finally, I would like to thank William P. Alford for lending me a copy of his book, *To Steal a Book is an Elegant Offense*, *see* WILLIAM P. ALFORD, TO STEAL A BOOK IS AN ELEGANT OFFENSE: INTELLECTUAL PROPERTY LAW IN CHINESE CIVILIZATION (1995). Since I kept the book in my possession for such a long period of time, I thank him for not thinking that I took the title of his book literally.

Comments and questions regarding this paper should be directed to: <grokopen@email.com>.

I. INTRODUCTION

"[T]he key formula for the coming age is this: Open, good. Closed, bad." – Peter Schwartz & Peter Leyden[1]

British mathematician and 'artificial intelligence' ('AI') pioneer Alan Turing argued that, when we could no longer tell the difference between the response a computer might give and a response a human might give to the same set of questions, a computer can be judged to have human-like intelligence.[2] Perhaps, in some not too distant future, it may become possible that a human could not tell the difference between a computer's response and a human's response to the same scenario. However, with the current state-of-the-art, there is a significant distinction between how a computer processes information and how a human being processes information.

Computers respond to 'ones' and 'zeros.' In other words, computers respond to binary instructions.[3] These binary instructions are sometimes referred to as 'binary code' or 'object code.'[4]

Human beings, on the other hand, find it almost impossible to read and make sense out of the large number of binary code that underlies contemporary computer software.[5] Humans "therefore ... prefer to write programs in an understandable computer language ('source code')

---

[1] Peter Schwartz & Peter Leyden, *The Long Boom*, WIRED, July 1997, at 115, 173. This line was quoted in Mark A. Lemley & David McGowan, *Legal Implications of Network Economic Effects*, 86 CAL. L. REV. 479, 523 (1998).

[2] *See* Kenneth M. Ford & Patrick J. Hayes, *On Computational Wings: Rethinking the Goals of Artificial Intelligence*, SCIENTIFIC AMERICAN PRESENTS: EXPLORING INTELLIGENCE, Winter 1998, at 78, 79-80; *see also* Arthur W. Burks, *Turing Machine*, *in* THE CAMBRIDGE DICTIONARY OF PHILOSOPHY 815, 816 (Robert Audi ed., 1995).

[3] *See* Marci A. Hamilton & Ted Sabety, *Computer Science Concepts in Copyright Cases: The Path to a Coherent Law*, 10 HARV. J.L. & TECH. 239, 266 (1997).

[4] *See id.*

[5] *See id.*; Josh McHugh, *For the Love of Hacking*, FORBES, Aug. 10, 1998, at 94, 96 (stating that software in its binary form is "unreadable even by advanced programmers").

and then use a compiler program to translate that computer language into binary-coded instructions ('object code')."[6] Therefore, a computer programmer needs access to the source code of a program in order to make any fundamental modifications to that program.[7]

The vast majority of commercially available software is what we can call 'closed source'[8] software. In other words, "[u]nder the proprietary [i.e., closed source] software model, most software developers withhold their source code from users."[9] Since most business models of software development and sales rest on the notion of strong intellectual property protection[10] – and since "[s]ource code is the *core* of any software product"[11] – most major producers of

---

[6] Hamilton & Sabety, *supra* note 3, at 266.

[7] *See* PHILIP E. MARGOLIS, *Source Code*, *in* RANDOM HOUSE WEBSTER'S COMPUTER & INTERNET DICTIONARY 521 (1999) [hereinafter Computer & Internet Dictionary] (defining 'source code'). The Computer & Internet Dictionary states the following in relation to its definition of source code:

> Source code is the *only* format that is readable by humans. When you purchase programs, you usually receive them in their machine language [binary code] format. This means that you can execute them directly, *but you cannot read or modify them*.

Id. (emphasis added). *See also* Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240, 1246-49 (3d Cir. 1983), *cert. denied*, 464 U.S. 1033 (1984) (discussing the distinction between source code and object code).

[8] For the purposes of this paper, 'closed source' will mean the exact opposite of 'open source.' In other words, 'closed source' software means that the source code is not freely distributed and that certain copyright protections – in particular the 'derivative works' right, *see infra* Part IV.A.3 – may be enforced by the software producer, *see, e.g.*, Mark Leibovich, *The Spreading Grass-Roots Threat to Microsoft: Eric Raymond Crusades for 'Open-Source' Software*, WASH. POST, Dec. 3, 1998, at A1 (reporting on Eric Raymond, a key figure in the open source movement; Eric Raymond uses 'closed source' in the way that it is used in this paper).

Another way of putting this is that 'closed' is synonymous with 'proprietary' as defined in the Computer & Internet Dictionary: "[*P*]*roprietary* is the opposite of *open*. ... [It] implies that the company has not divulged specifications ...," Computer & Internet Dictionary, *supra* note 7, at 452 (emphasis in original).

[9] Ira V. Heffan, Note, *Copyleft: Licensing Collaborative Works in the Digital Age*, 49 STAN. L. REV. 1487, 1490 (1997) (alterations added); *see also* McHugh, *supra* note 5, at 95-96 (describing the significance of keeping source code proprietary – i.e., not releasing the source code to the public).

[10] *See, e.g.*, RICHARD A. POSNER, ECONOMIC ANALYSIS OF LAW § 3.3, at 43 (Aspen 5th ed. 1998) (illustrating one of the main justifications for intellectual property protection – to give creators of intellectual property an incentive to produce).

[11] MICHAEL A. CUSUMANO & DAVID B. YOFFIE, COMPETING ON INTERNET TIME: LESSONS FROM NETSCAPE AND ITS BATTLE WITH MICROSOFT 139 (1998) (emphasis added).

commercial software tend to hide their source code from the public.[12]

Recently, however, 'open source' software – software that has its underlying source code freely available to evaluate, copy, and/or modify[13] – has been garnering considerable attention in both the computer/Internet industry and the 'mainstream' media.[14] The most celebrated example of open source software is Linux – an open source version of the Unix operating system that can be run on most PC's.[15] Besides coverage by the trade media, mainstream news outlets like *Forbes*, the *Economist*, the *New York Times*, the *Wall Street Journal*, and the *Washington Post*, have reported on the growing popularity of Linux.[16]

---

[12] *See, e.g.*, McHugh, *supra* note 5, at 95-96, 98 (describing the tendency of major computer industry companies – like Microsoft – to keep their respective source code proprietary and out of the hands of the public).

[13] *See* McHugh, *supra* note 5, at 96 (describing what open source software is and how it defers from the usual 'closed source' software); Alex Lash, *Source Code for the Masses*, CNET NEWS.COM, (Feb. 2, 1998) <http://www.news.com/SpecialFeatures/0,5,18652,00.html> (describing what open source software is) [hereinafter Lash, *Source Code for the Masses*]. A more detailed definition will be given in Part II.A., *infra*.

[14] *See* Amy Harmon, *A Surge in Popularity of Software that Unlocks the Code*, N.Y. TIMES ON THE WEB, (Jan. 4, 1999) <http://www.nytimes.com/library/financial/010499outlook-tech-web.html> (reporting on the surge in popularity of open source software); Charles C. Mann, *Programs to the People*, TECHNOLOGY REVIEW, Jan.–Feb. 1999, at 36 (describing advances in the open source movement); John Markoff, *Sentiment Growing for Freeware*, N.Y. TIMES ON THE WEB, (Apr. 13, 1998) <http://www.nytimes.com/library/tech/98/04/biztech/articles/13freeware.html> (reporting on the growing movement to freely share underlying source code for software); McHugh, *supra* note 5 (reporting on the increasing influence of open source software in the information technology industry); *see also* Lash, *Source Code for the Masses*, *supra* note 13.

[15] *See* Mann, *supra* note 14, at 39-41 (providing a brief history of the development and the increasing acceptance of Linux). A good definition of Linux is provided in the "What is Linux?" panel in the table of contents found in every issue of the LINUX JOURNAL, *see, e.g.*, LINUX JOURNAL, Jan. 1999, at 4. It should be noted – for reasons too complicated to be explained in this paper – that Linux is essentially a clone of the Unix operating system and not a pure version of it. However, for all intents and purposes, Linux can be seen as a full-fledged version of Unix, *see* ERIC S. RAYMOND, *Linux*, *in* THE NEW HACKER'S DICTIONARY 281-82 (1996) [hereinafter Raymond, TNHD] (giving a hacker's definition of Linux).

[16] For examples of stories in trade publications, s*ee, e.g.*, Anita Karvé, *Linux Goes Mainstream*, NETWORK MAGAZINE, Dec. 1998, at 36 (reporting on the growing popularity of Linux); Stephen Shankland, *Linux Market Share Leaps by 212%*, CNET NEWS.COM, (Dec. 16, 1998) <http://www.news.com/News/Item/0,4,30027,00.html>; *see also* Lash, *Source Code for the Masses*, *supra* note 13; Mann, *supra* note 14. For the FORBES story, *see* McHugh, *supra* note 5. For the ECONOMIST story, *see Revenge of the Hackers*, ECONOMIST, July 11, 1998, at 63. For the NEW YORK TIMES story, *see* Katie Hafner, *Mac, Windows And Now, Linux*, N.Y. TIMES, Oct. 8, 1998, at E1. For the WALL STREET JOURNAL story, Lee Gomes, *Linux's Appeal Compels Big Firms to Respond*, WALL ST. J., Oct 22, 1998, at B4. For the WASHINGTON POST story, Elizabeth Corcoran, *LINUX: UNIX Power For Peanuts: Linus Torvalds and His Free Operating System*, WASH. POST, May 22, 1995, at F15.

The growing interest in Linux and other open source software programs has even drawn the attention of Microsoft. In its so-called 'Halloween' memos, Microsoft analyzed the potential effects that open source software might have on Microsoft's monopoly in the software market.[17]

Besides Linux, the Apache Web Server[18] is another example of open source software that has gained a great deal of attention recently.[19] The not-for-profit Apache Group[20] – which develops the Apache Web Server – announced a deal to license its open source web server software to IBM under Apache's open source software license.[21] In this deal – rather than receiving monetary compensation (which the Apache Group had little or no interest in) in exchange for licensing its software to IBM – the Apache Group will receive cooperation from

---

[17] *See* Memorandum from Vinod Valloppillil, Open Source Software (Aug. 11, 1998) <http://www.opensource.org/halloween1.html> [hereinafter Halloween I] (Microsoft's confidential memorandum that was leaked to the open source software community assessing the threat that open source software – particularly Linux – posed to Microsoft's strategy); Memorandum from Vinod Valloppillil & Josh Cohen, Linux OS Competitive Analysis (Aug. 11, 1998) <http://www.opensource.org/halloween2.html> [hereinafter Halloween II] (Microsoft's confidential memorandum that was leaked to the open source software community assessing the threat that Linux posed to Microsoft's strategy); Microsoft, *Microsoft Responds to Internal Memo Regarding the Open Source Model and Linux* (Nov. 5, 1998) <http://www.microsoft.com/ntserver/highlights/editorletter.asp> (Microsoft conceding that the Halloween memos were genuine and appear to be confidential Microsoft documents). For news coverage of the 'Halloween' memos (note: these memos are commonly referred to as the 'Halloween' memos), *see* Stephen Shankland, *Microsoft Spins "Halloween" Memos*, CNET NEWS.COM, (Nov. 6, 1998) <http://www.news.com/News/Item/0,4,28490,00.html>; Lee Gomes, *Microsoft Acknowledges Growing Threat of Free Software for Popular Functions*, WALL ST. J., Nov. 3, 1998, at B6.

[18] The Apache Web Server can loosely be defined as a piece of software that is used to allow a computer (called a 'server') to deliver World Wide Web pages to a computer that requests the page (called a 'client'), *see* Computer & Internet Dictionary, *supra* note 7, at 21 (defining 'Apache Web Server') & 606 (defining 'Web Server').

[19] *See, e.g.*, Larry Seltzer, *Key Open Source Projects*, PC MAGAZINE, Mar. 23, 1999, at 172 [hereinafter Seltzer, *Key Open Source Projects*]; McHugh, *supra* note 5, at 95-96.

[20] The 'Apache Group' is ***not*** a formal organization by any stretch of the imagination. The Apache Group is a lose-knit group of computer enthusiasts [hereinafter hackers] that volunteer their time to develop the Apache HTTP/Web Server – which is free of charge and, since it is open source, free of many of the restrictions associated with copyright, *see* McHugh, *supra* note 5, at 95-96, 100; Apache Group, *About the Apache HTTP Server Project* (visited Aug. 5, 1998) <http://www.apache.org/ABOUT_APACHE.html> (explaining what the Apache Project is and effectively acknowledging that its web server software is open source software).

[21] *See* McHugh, *supra* note 5, at 95-96; *see also* Apache Group, *IBM joins the Apache Project, Plans to bundle and support the Apache HTTP Server* (June 22, 1998) <http://www.apache.org/press/22Jun98.html> [hereinafter Apache/IBM deal].

IBM on further developing the Apache Web Server's code and in abiding by Apache's open source philosophy and licensing agreement.[22] In other words, IBM would have to share any relevant modifications to the Apache code, not only with the Apache Group, but – since Apache's source code is available to anyone who wants it – with the rest of the world.[23]

IBM is not the only commercial software company to (partially) embrace the open source software model. Netscape has released the source code for its Communicator web browser under its own open source software license.[24] Again – like Linux or Apache, and unlike Microsoft's Windows 98 or Internet Explorer – Netscape Communicator's source code can be downloaded, modified, and re-distributed (under certain conditions) by virtually anyone on the Internet.[25]

Besides Apache and the Netscape Communicator, there are other examples of open source software that play important roles on the Internet. In fact, according to *Forbes*, "two of the most fundamental pieces of the Internet are open source software":[26] Both BIND[27] – which allows Web users to type in domain names (e.g., www.law.harvard.edu) rather than hard-to-remember

---

[22] *See* McHugh, *supra* note 5, at 95-96.

[23] *See id.*

[24] *See* McHugh, *supra* note 5, at 96 (describing Netscape's acceptance of the open source software development model and its release of the Communicator source code); Netscape, *Netscape Communicator Open Source Code Whitepaper* (visited Aug. 12, 1998) <http://home.netscape.com/browsers/future/whitepaper.html> [hereinafter Netscape Whitepaper] (providing a primer on what 'open source software' and 'open source software licensing' is and why Netscape is now pursuing such a strategy); *see also* CUSUMANO & YOFFIE, *supra* note 11, at 36 (1998) (describing Netscape's decision to give away "Netscape's crown jewel" – the source code for Communicator – as "stunning").

Note: Netscape's merger with AOL will not end the open source effort, *see* Paul Festa, *Concerns About AOL Eased at Mozilla*, CNET NEWS.COM, (Dec. 2, 1998) <http://www.news.com/News/Item/0,4,29463,00.html> (reporting that AOL will leave Mozilla.org – Netscape's open source software development arm – intact).

[25] *See* CUSUMANO & YOFFIE, *supra* note 11, at 139.

[26] McHugh, *supra* note 5, at 99.

[27] Acronym for Berkeley Internet Name Domain, *see* Computer & Internet Dictionary, *supra* note 7, at 54 (defining BIND).

numerical IP addresses (140.247.200.68)[28] – and Sendmail – "which routes about 80% of the E-mail that courses over the Internet"[29] – are open source.[30]

In addition to Apache, BIND, and Sendmail, Perl (short for 'Practical Extraction and Report Language') is another example of open source software that serves an important role on the Internet.[31] Perl – often called "the duct tape of the Web"[32] – is a programming language that serves a crucial role in the vast majority of interactive Web sites by processing the data entered in by visitors to the site and displaying the results.[33]

At this point one may wonder why anyone would want to give away such valuable intellectual property. Why didn't these open source developers simply take a Linux, a Perl, a BIND, etc., and simply keep the source code proprietary and out of the view of potential competitors? After all, the act of making source code publicly available – as the open source software development community has done – has garnered descriptions ranging from being the "equivalent [to] revealing the recipe for Coca-Cola"[34] all the way to giving away a "potential gold mine."[35]

---

[28] *See* McHugh, *supra* note 5, at 99. For a more thorough definition, *see* Computer & Internet Dictionary, *supra* note 7, at 54 (defining BIND), 168 (defining DNS).

[29] McHugh, *supra* note 5, at 99.

[30] *Id.*

[31] *See id.*

[32] *See* Seltzer, *Key Open Source Projects*, *supra* note 19, at 172, 176.

[33] *See* McHugh, *supra* note 5, at 99. For a more technically correct definition, *see* Computer & Internet Dictionary, *supra* note 7, at 423 (defining Perl); *see also id.* at 85 (defining CGI). According to THE NEW HACKER'S DICTIONARY, Perl is the "language[ ] of choice" among Unix system administrators ("sysadmins"), Raymond, TNHD, *supra* note 15, at 353.

[34] CUSUMANO & YOFFIE, *supra* note 11, at 139.

[35] McHugh, *supra* note 5, at 96.

A partial answer to these sorts of questions is that open source programmers consider themselves to be in a community with a set of social norms that demand that information, such as source code, be shared with both the community and with the rest of the world.[36]

But what happens when programs created within the open source community move into a realm largely dominated by the closed source model of software development? How can open source software developers openly share their intellectual property in a world that jealously guards intellectual property without sacrificing the philosophy that drove them to freely share their code? For example, how can someone who wants to make source code freely available prevent others from 'closing' open source code by claiming proprietary rights over all or part of that code?

Law seems to be one possible route by which the kind of structure and order necessary to carry out the vision of open source development can be established and maintained.[37] In fact, open source developers utilize law as a means of 'regulating' how their code is handled in 'cyberspace.' Open source developers use intellectual property licensing – through various types of 'open source licenses' – in order to maintain the integrity of the open source project.[38]

Considering the fact that open source developers *give away* intellectual property, the ironic choice of utilizing intellectual property licensing to achieve and preserve the goals of the open source movement, logically, leads us to the critical question: Are open source licenses legally enforceable?[39] If these licenses are not enforceable, then those outside of the open source

---

[36] *See infra* Part V.A.; *see also* McHugh, *supra* note 5, at 96-99.

[37] Law is one way in which behavior is cyberspace can be regulated, *see* Lawrence Lessig, *The Constitution of Code: Limitations on Choice-Based Critiques of Cyberspace Regulation*, 5 COMMLAW CONSPECTUS 181, 183 (1997).

[38] *See* Larry Seltzer, *License to Drive Software Development*, PC MAGAZINE, Mar. 23, 1999, at 171 [hereinafter Seltzer, *License*].

[39] The question of legal enforceability is not simply a philosophical abstraction, *see* Dennis Berman, *Linux*

movement will probably succeed in 'closing' open code. In other words, if 'licensees' cannot be prevented from claiming intellectual property rights over what use to be freely available source code, the open source movement itself might be in jeopardy.[40]

This paper will argue that open source software licenses, in general, are legally enforceable. This paper will also conclude that the terms associated with *specific* types of open source licenses – regardless of their level of strictness – are enforceable as well.

In addition to the legal analysis, this paper will also explore the possibility that the cultural norms associated with the open source community can serve as a viable extra-legal mechanism of control necessary to preserve the integrity of open source movement.

Finally, I will address why people may want to use the open source model of development – via the legal device of an open source license. This paper will argue that the open source development model and open source licensing can be a desirable and viable means of producing and disseminating computer software as well as other types of creative works.

II. BACKGROUND: DEFINITIONS AND A BRIEF HISTORY OF THE OPEN SOURCE MOVEMENT

A. Definitions: What is Open Source Software? What is an Open Source Software License?

A simple definition of 'open source' software – sometimes referred to as 'free software'[41]

May be Running on Some Spindly Legal Legs, BUSINESS WEEK ONLINE, (Apr. 27, 1999) <http://www.businessweek.com/bwdaily/dnflash/apr1999/nf90427b.htm> [hereinafter Berman, *Linux Legal*] (reporting that some people question the legal enforceability of open source licenses – like the GNU GPL – which undergirds open source software programs like Linux).

[40] The proprietization of open source programs and protocols might be Microsoft's strategy in dealing with the success of the open source movement, *see* Halloween I, *supra* note 17 (apparently advocating a strategy of denying open source software producers "entry into the market" by establishing intellectual property rights over what has been fairly open and non-proprietary protocols on the Internet).

[41] In this context, the word 'free' does not necessarily refer to the price of the software (although open source software is often free in the sense of having a nominal price of zero). 'Free,' in this context, is more akin to 'freedom'. In other words, the 'free' in 'free software' means liberty or lack of constraint, *see* Free Software Foundation, *What is Free Software? – GNU Project – Free Software Foundation (FSF)* (visited Aug. 5, 1998) <http://www.gnu.org/philosophy/free-sw/html> [hereinafter FSF, *What is Free Software?*] (providing links to pages that define what the FSF/GNU Project calls 'free software' – which is synonymous with 'open source software').

– is that the software's source code is available to the public.[42] However, simply making the source code available does not necessarily mean that a program meets the definition of open source software.[43] A fuller definition of open source software is given below.[44]

In order for a program to be classified as open source software, "[t]he distribution terms [i.e., the license] of an open-source program must comply with the .... criteria" listed below.[45] Please note that the following list of defining factors is not intended to be exhaustive. The criteria listed below were designed to: 1.) represent the *consensus* view on just what open source software is, and 2.) to focus in on those defining factors that will be important and relevant to the analysis while leaving out factors that are less relevant to the analysis:[46]

- **Source Code**[47]

---

[42] *See, e.g.*, McHugh, *supra* note 5, at 96.

[43] *See The Open Source Definition (Version 1.0)* at Preamble, *reprinted in* Bruce Perens, *The Open Source Definition*, *in* OPEN SOURCES: VOICES FROM THE OPEN SOURCE REVOLUTION 171, 176 (Chris DiBona et. al. eds., 1999) [hereinafter Perens, *The Open Source Definition*] ("Open source doesn't just mean access to the source code.").

[44] Most of the defining conditions and terms are adapted from the official *Open Source Definition, Version 1.0*, *see id*.; *see also* Open Source Initiative & Bruce Perens, *The Open Source Definition (Version 1.0)* (no longer at the OSI website; on file with the author and also reprinted in Perens, *The Open Source Definition*, *supra* note 43, at 176-80) [hereinafter *OSD (v.1.0)*]. The definition(s) relied on in this paper will also be derived from the definitions adopted by others in the open source community. It should be noted, however, that most of the definitions given in the following citations – with a few exceptions that will be noted in Part III. of this paper – conform to the definition given in the *OSD (v.1.0)*, *see* Debian GNU/Linux Developers, *A Social Contract* (visited Jan. 7, 1999) <http://www.debian.org/social_contract> [hereinafter Debian Social Contract] (defining open source software); FSF, *What is Free Software?*, *supra* note 41; Netscape Whitepaper, *supra* note __ .

Note: The Open Source Initiative's version of the *OSD* has been revised, and – as of the time of writing – is slightly different than Bruce Perens' original version 1.0 for OSI. This paper will use Perens' version 1.0 since it is the closest to the Debian Group's and the FSF's definition of free/open source software. For OSI's latest version of the *OSD*, *see* Open Source Initiative & Bruce Perens, *The Open Source Definition* (visited Apr. 24, 1999) <http://www.opensource.org/osd.html> [hereinafter OSI, *Definition*].

[45] *OSD (v.1.0)*, *supra* note 44, at Preamble.

[46] The definition that this paper gives conforms to the core defining terms of open source software as laid out by the author of the *Open Source Definition*: the right to make copies, the right to freely distribute those copies, open access to source code, and the freedom to modify the source code, *see* Perens, *The Open Source Definition*, *supra* note 43, at 172.

[47] *See OSD (v.1.0)*, *supra* note 44, ¶ 2.

"The program must include source code, and must allow distribution in source code as well as [in] compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of downloading the source code, without charge, via the Internet. .... Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed."[48]

· **Integrity of The Author's Source Code**[49]

It is strongly recommended that "*all authors not restrict any files, source or binary, from being modified.*"[50] "The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software."[51]

· **Derivative Works**[52]

"The [open source software] license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original [open source] software."[53]

---

[48] *Id.*

[49] *Compare* Debian Social Contract, *supra* note 44 (see 'The Debian Free Software Guidelines,' ¶ 4), *with OSD (v.1.0)*, *supra* note 44 , ¶ 4. The Debian guidelines seem less tolerant of restricting modifications to the original code. I will use language conforming to the Debian viewpoint since that is closer to what the Free Software Foundation might find acceptable.

[50] Debian Social Contract, *supra* note 44 (see 'The Debian Free Software Guidelines,' ¶ 4) (emphasis in original).

[51] *OSD (v.1.0)*, *supra* note 44, ¶ 4.

[52] *See, e.g., id.* ¶ 3.

[53] *Id. See also* Heffan, *supra* note 9, at 1489 (arguing that authors of non-proprietary/collaborative works should use copyright licenses in order to prevent other less well-meaning people from establishing "proprietary rights in a derivative work" – thereby negating the goals of the original author).

· **'Free' Redistribution**[54]

> "The license may not restrict any party from ... giving away [or selling][55] the [open source] software ...."[56]

'Closed source' – or 'proprietary' – software is defined as software that does not meet the criteria stated above.[57] For example, with closed/proprietary software, a typical user is not allowed access to the source code, is not allowed to modify the source code, and is usually prohibited from redistributing the software.

An 'open source software license' can be defined as a license that attempts to confer the kind of rights, privileges, and obligations associated with the definition of open source software. The 'Preamble' of one type of open source software license captures the spirit and the intent behind most open source licenses:

> The licenses for most software are designed to take away your freedom to share and change it. By contrast, [this open source software license] is intended to *guarantee your freedom to share and change free* [i.e., open source] *software – to make sure the software is free for all its users.*[58]

As we shall see in Part II.B., *infra*, virtually every type of open source software has some variant of an open source software license associated with it. **In fact, in order for a program to be defined as 'open source,' the program must be distributed under some sort of open source**

---

[54] *See, e.g., OSD (v.1.0), supra* note 44, ¶ 1. 'Free' in this case means 'freedom from constraints.'

[55] *See infra* note 139 & accompanying text (discussing why 'free redistribution' does not mean someone could not sell the open source software).

[56] *OSD (v.1.0), supra* note 44, ¶ 1.

[57] *See supra* note 8 & accompanying text.

[58] Free Software Foundation, *GNU General Public License* (visited Aug. 4, 1998) <http://www.gnu.org/copyleft/gpl.html> (emphasis added) [hereinafter FSF, *GNU GPL*].

license.[59] In other words, the use of open source licensing is an essential element in making an open source program 'open source.'[60] Therefore, the question of whether or not open source licenses are legally enforceable is a *critical* factor in answering the question of whether or not the open source development model can succeed.

A more detailed examination of open source software licenses will be included in Part III. of this paper.

1. A Note on Terminology and Usage: 'Free' vs. 'Open'

It should be noted that some factions within the open source movement prefer the use of the word 'free' or 'free software' over 'open source.'[61] The vast majority of the open source community, however, seem to prefer the terms 'open' and 'open source.'[62] Those outside of the open source movement – such as the media and novice computer users – also tend to use the term 'open source.'[63]

To reiterate,[64] 'open source' (or 'open software,' etc.) and 'free software' mean

---

[59] *See OSD (v.1.0)*, *supra* note 44, at Preamble("Open source doesn't just mean access to the source code. The distribution terms of an open-source program must comply with the [criteria listed on that Web page.]"); Perens, *The Open Source Definition*, *supra* note 43, at 180 (open source programs are "clearly copyrighted and covered by a license"); Seltzer, *License*, *supra* note 38, at 171.

[60] *See OSD (v.1.0)*, *supra* note 44, at Preamble.

[61] *See, e.g.*, Free Software Foundation, *Categories of Free and Non-Free Software* (visited Aug. 8, 1998) <http://www.gnu.org/philosophy/categories/html> [hereinafter FSF, *Free and Non-Free Software*] (stating that "[t]he term 'open source' is used by some people to mean more or less the same thing as free software. We prefer the term 'free software' ...").

[62] *See, e.g.*, Open Source Initiative, *OpenSource.org* (visited Jan. 6, 1999) <http://www.opensource.org>; OPEN SOURCES: VOICES FROM THE OPEN SOURCE REVOLUTION (Chris DiBona et al. eds, 1999) [hereinafter OPEN SOURCES].

[63] *See, e.g.*, *supra* note 16.

[64] *See supra* note 41 and accompanying text.

essentially the same thing;[65] therefore, this paper will usually use the more popular term – open source software.

The reader should also note that, when the word 'free' is used, it does not necessarily mean the same thing as 'zero price.' 'Free' – in relation to open source software – means 'freedom' from constraints.[66]

B. A Brief History of the Open Source Movement

This section will present a bare-bones history of the open source software movement. Rather than an all-encompassing retelling of the evolution of the open source movement, this section will, instead, seek to provide enough background information to aid in the analysis of open source software licensing.

1. Beginnings: GNU and Perl – 1983-1990

In 1984, a former programmer at MIT's AI Lab, Richard M. Stallman, founded the 'Free Software Foundation' [hereinafter FSF].[67] The main technical goal of the FSF was to create an open source Unix-like operating system called 'GNU.'[68] Although Stallman was never able to finish the 'kernel' (the "central module")[69] of the GNU operating system,[70] he and other

---

[65] *See* Debian GNU/Linux Developers, *What Does Free Mean? or What Do You Mean by Open Software?* (visited Jan. 7, 1999) <http://www.debian.org/intro/free> [hereinafter Debian, *Free/Open Software*] (stating that 'free software' and 'open software' mean "the same thing since 'Open refers to the availability of the source code'); *cf.* FSF, *Free and Non-Free Software*, *supra* note 61 (stating that the two terms "more or less" mean the same thing).

[66] *See supra* note 41.

[67] *See* Mann, *supra* note 14, at 38; McHugh, *supra* note 5, at 98. For more on the story of Richard M. Stallman and the Free Software Foundation, *see* Andrew Leonard, *The Saint of Free Software: Maverick Richard Stallman Keeps the Faith – and Gives Bill Gates the Finger*, SALON, (Aug. 1998) <http://www.salonmagazine.com/21st/feature/1998/08/cov_31feature.html> [hereinafter Leonard, *Saint*]. For an excellent background on the adventures Richard M. Stallman prior to starting the FSF, *see* STEVEN LEVY, HACKERS 415-27 (1984) (describing Richard M. Stallman as 'the Last True Hacker').

[68] *See* Mann, *supra* note 14, at 38. GNU is a recursive acronym for 'GNU's Not Unix,' *see* Raymond, TNHD, *supra* note 15, at 220 (defining 'GNU'), 381-82 (defining 'recursive acronym').

[69] Computer & Internet Dictionary, *supra* note 7, at 302 (defining 'kernel').

programmers associated with the FSF were able to produce extremely useful and popular pieces of open source software like the GNU EMACS[71] and the GNU C compiler.[72]

Perhaps the most important goal of the GNU Project, however, was to promote Stallman's philosophy regarding software development.[73] Stallman believes that "information is community property and all software source [code] should be shared."[74] In Stallman's mind, proprietary software, with its source code closed off from the public, prevented the sort of cooperation and communitarian spirit necessary – in Stallman's opinion – for advancing software development.[75]

The FSF's solution to the 'dilemma' of closed source / proprietary software was a concept that Stallman called 'copyleft':[76] Keep source code freely available so that everyone could change and/or redistribute the software.[77]

In order to bring the concept of 'copyleft' to fruition, Stallman and the FSF created the first example of an open source software license: the GNU General Public License [hereinafter

---

[70] *See* Mann, *supra* note 14, at 39.

[71] EMACS is a programmable text editor described by Eric Raymond as the "ne plus ultra of hacker editors," Raymond, TNHD, *supra* note 15, at 172 (defining EMACS).

[72] *See* Mann, *supra* note 14, at 38-39; Raymond, TNHD, *supra* note 15, at 220.

[73] *See generally* Heffan, *supra* note 9, at 1504-9 (providing an excellent summary of Stallman's work and philosophy).

[74] Raymond, TNHD, *supra* note 15, at 220. *See also* Leonard, *Saint*, *supra* note 67; Mann, *supra* note 14, at 38.

[75] *See* Heffan, *supra* note 9, at 1505; Leonard, *Saint*, *supra* note 67; Mann, *supra* note 14, at 38-39; McHugh, *supra* note 5, at 98-99.

[76] *See* Heffan, *supra* note 9, at 1504-9; *see generally* Free Software Foundation, *What is Copyleft?* (visited Aug. 4, 1998) <http://www.gnu.org/copyleft/copyleft.html> [hereinafter FSF, *Copyleft*] (defining 'copyleft'); Raymond, TNHD, *supra* note 15, at 126 (defining 'copyleft').

[77] *See* FSF, *Copyleft*, *supra* note 76.

GNU GPL].[78] The process of 'copylefting' using the GNU GPL is similar to normal copyright licensing: (1) the FSF copyrights the software, and (2) the FSF offers the software under the terms of the license.[79] The key difference is that at step (2), the license allows the licensee to act in ways that, under the typical closed source license, would violate the licensor's exclusive rights under copyright law. Namely, the GNU GPL gives the licensee the permission to copy, distribute, and modify the program so long as certain conditions are met.[80]

If it seems like Stallman and the FSF are trying to weaken the intellectual property rights that they would normally be entitled to via the GNU GPL, then that assessment is correct.[81] As strange as it may seem in a world where creators of intellectual property desire ever stronger intellectual property protection in order to have an incentive to produce,[82] Stallman actually wants to *weaken* intellectual property rights.[83] In Stallman's opinion, those who produce closed source software are 'software hoarders' and, thus, "[c]opyleft uses the tools of the software hoarders [i.e., copyright and copyright licensing] against them."[84]

It should come as no surprise that the concept of copylefting is controversial even within

---

[78] *See* FSF, *GNU GPL*, *supra* note 58; Heffan, *supra* note 9, 1507-9.

[79] *See* FSF, *GNU GPL*, *supra* note 58 , at Preamble; *see also infra* Parts III. & IV. for details on the mechanics of software licensing.

[80] *See infra* Part III.C. It should be noted, for now, that the rights to copy, distribute, and modify, are three of the five *exclusive* rights that copyright confers on copyright holders, *see* 17 U.S.C. §106(1)-(6).

[81] *See* Heffan, *supra* note 9, 1504-9; FSF, *Copyleft*, *supra* note 76; FSF, *GNU GPL*, *supra* note 58.

[82] *See* McHugh, *supra* note 5, at 99; POSNER, *supra* note 10, § 3.3, at 43.

[83] *See, e.g.*, Richard Stallman, *Reevaluating Copyright: The Public Must Prevail*, 75 OR. L. REV. 291, 291-97 (1996).

[84] Mann, *supra* note 14, at 38; *see* FSF, *Copyleft*, *supra* note 76; Heffan, *supra* note 9, at 1504-9.

the open source software community.[85] Other figures in the open source movement disagree – often strongly – with Stallman's utopic vision of a world without the intellectual property protections that he believes serve to 'hinder' the 'liberation' of software code.[86]

Regardless of Stallman's controversial views, the present-day open source movement, along with the various open source licenses, are based on both the GNU GPL and Stallman's 'free software' views.[87] As will be seen shortly, the GNU GPL is especially important in this context because of the importance of Linux to the open source movement. The GNU GPL – and how the philosophy of 'copylefting' distinguishes it from other open source licenses – will be discussed in Part III.C. of this paper.

Throughout the remainder of the 1980's, only limited progress was made in the open source – or, as Stallman likes to call it, the 'free software' – movement.[88] For the purposes of this paper, the only notable development in open source software in the 1980's came in 1987. In 1987, Larry Wall invented Perl, "a Unix-based programming language he created to scan, manipulate, and print text files."[89] Initially, Wall posted Perl on Usenet under the GNU GPL.[90] Later on,

---

[85] *See* Raymond, TNHD, *supra* note 15, at 220 (stating that Stallman's campaign against intellectual property protection for software – along with his charge of 'software hoarding' – is controversial among hackers). For more on the controversy, *see infra* Part III.D.

[86] *See* Leibovich, *supra* note 8 (discussing the disagreements between Eric Raymond, a major figure in the open source community, and Stallman); Leonard, *Saint*, *supra* note 67 (reporting on the conflicts that Stallman has had with others in the open source movement); Perens, *The Open Source Definition*, *supra* note 43, at 174 (stating that Stallman and Raymond are often cast in adversarial positions – even though they both champion open source).

[87] *See, e.g.*, Leonard, *Saint*, *supra* note 67; Perens, *The Open Source Definition*, *supra* note 43, at 172-73 (the current open source movement, and the associated open source licensing schemes, derive from Stallman's ideas).

[88] *See* Alex Lash, *Evolution of a Net Community*, CNET NEWS.COM, (Feb. 2, 1998) <http://www.news.com/SpecialFeatures/0,5,18619,00.html> [hereinafter Lash, *Evolution*] (providing a timeline of the open source movement); Mann, *supra* note 14, at 39.

[89] Lash, *Evolution*, *supra* note 88; *see also supra* note 33 and accompanying text; McHugh, *supra* note 5, at 99.

[90] *See* Lash, *Evolution*, *supra* note 88.

however, Wall decided that the GPL's terms were "too restrictive"[91] and created his own open source software license called the 'Artistic License.'[92] The differences between the GPL and the Artistic License will be discussed in Part III.C. of this paper.

2. *The Emergence of an Open Source Community: Linux, FreeBSD, Apache, and the Internet Revolution – 1990-1997*

By the late 1980's, the FSF's attempts to create a Unix-like operating system was effectively derailed by an injury to Richard Stallman's hands.[93] After years of typing code, Stallman's hands gave out and his dream of creating the GNU kernel – at least for the time being – seemed out of reach.

In 1991, however, the dream of a Unix-like open source operating system was realized by a 21-year old Finnish undergraduate student named Linus Torvalds.[94] With the goal of getting a Unix-like operating system running on, what was then, a typical PC (which lacked the power and memory necessary to have a full-fledged version of Unix running on it), Linus wound "up with something like a Unix kernel."[95] Linus' friends decided to dub the new operating system 'Linux' (pronounced 'Linn-ucks') in honor of its creator.[96]

---

[91] *Id.*; s*ee infra* Part III.C. (more on the differences between various forms of open source licenses). As will be discussed in Part III.D., Larry Wall is among those in the open source community that disagrees with Stallman on certain issues. For an example of Wall's disagreement with Stallman, *see* Leonard, *Saint*, *supra* note 67.

[92] *See id.* For examples of the Artistic License, *see* The Perl Institute, *The "Artistic License"* (visited Jan. 8, 1999) <http://language.perl.com/misc/Artistic.html> [hereinafter Perl, *Artistic License*]; Open Source Initiative, *Artistic License* (visited Dec. 22, 1998) <http://www.opensource.org/artistic-license.html>.

[93] *See* Mann, *supra* note 14, at 39.

[94] *See* Mann, *supra* note 14, at 39; McHugh, *supra* note 5, at 96. For the idea that Linux is the heir apparent of the hoped for GNU kernel, *see* Mann, *supra* note 14, at 39; Raymond, TNHD, *supra* note 15, at 281-82 (defining Linux). For a brief biography of Linus Torvalds, *see* Doug Levy, *Linux Creator: Next Bill Gates?*, USA TODAY, (Jan. 7, 1999) <http://www.usatoday.com/life/cyber/tech/cte102.htm>.

[95] Mann, *supra* note 14, at 39.

[96] *See id.*; McHugh, *supra* note 5, at 96.

In creating Linux, Linus Torvalds relied heavily on the toolset and the necessary subsidiary programs created by the GNU Project.[97] In order to honor the contributions of the GNU Project, Linus 'copylefted' Linux (i.e., released Linux under the GNU GPL).[98]

Linux was not the only open source operating system to be developed in the 1990's. In 1993, another open source version of Unix, FreeBSD,[99] was distributed through the Internet and on CD-ROM.[100] FreeBSD was released on another type of open source license called the 'BSD License.'[101]

At about the same time that the open source software development model was beginning to pick up steam,[102] the Internet – emerging from the confines of academe and research facilities – took on an ever-increasing role in society.[103] By 1997, the number of people, worldwide,

---

[97] *See* Mann, *supra* note 14, at 39; McHugh, *supra* note 5, at 98 ("Several pieces of GNU software are vital to the operation of Linux."); Raymond, TNHD, *supra* note 15, at 282. It should be noted that Richard Stallman likes to refer to Linux as 'GNU/Linux,' *see* Leonard, *Saint*, *supra* note 67. However, most people – including those in the open source community – prefer to simply call it 'Linux.' This paper will use 'Linux' rather than 'GNU/Linux.'

[98] *See* Lash, *Evolution*, *supra* note 88; Linux Documentation Project, *Chapter 17: The GNU General Public License* (visited Jan. 8, 1999) <http://metalab.unc.edu/LDP/LDP/tlk/appendices/gpl.html> [hereinafter LDP, *Linux GPL*] (stating that Linux is licensed under the GNU GPL – i.e., 'copyleft'); *see also* Mann, *supra* note 14, at 39.

[99] BSD stands for 'Berkeley Software Distribution.' *See* Raymond, TNHD, *supra* note 15, at 92 (defining BSD). Note: BSD – unlike Linux – is a variant of Unix rather than a Unix clone.

[100] *See* Lash, *Evolution*, *supra* note 88; *see generally* Computer & Internet Dictionary, *supra* note 7, at 228 (defining FreeBSD); FreeBSD, Inc., *FreeBSD* (last visited Jan. 12, 1999) <http://www.freebsd.org>.

[101] *See* Lash, *Evolution*, *supra* note 88; *see also* FreeBSD, Inc., *The FreeBSD Copyright* (last visited Jan. 12, 1999) <http://www.freebsd.org/copyright/freebsd-license.html> [hereinafter *FreeBSD License*]; FreeBSD, Inc., *The 4.4BSD Copyright* (last visited Jan. 12, 1999) <http://www.freebsd.org/copyright/license.html> [hereinafter *4.4BSD License*]; Open Source Initiative, *The BSD License* (last visited Jan. 8, 1999) <http://www.opensource.org/bsd-license.html>.

When we discuss the BSD License, this paper will usually refer to the sample BSD-style license provided by the Debian Group due to its adaptability to software developed outside of the ambit of UC Berkeley and the FreeBSD Group, *see* Debian GNU/Linux Developers, *Sample BSD Style License* (visited Jan. 12, 1999) <http://www.debian.org/misc/bsd.license> [hereinafter *Sample BSD-style License*].

[102] *Cf.* Lash, *Evolution*, *supra* note 88 (providing a timeline of the progress the open source movement has made in recent years).

[103] *See generally* STEVEN SEGALLER, NERDS 2.0.1: A BRIEF HISTORY OF THE INTERNET (1998).

connected on the Internet grew to about 100 million people, and, by 1998, "it [took] only 100 days for the Internet's volume of traffic to double."[104] As noted in the Introduction, many of the core components and protocols of the Internet – like Perl, BIND, and Sendmail – were non-proprietary and open source.[105] The Internet also became the most frequently used means for the distribution and redistribution of open source software.

The World Wide Web is one of the best illustrations of how the growth of the open source movement coincided with the growth of the Internet. Consider the following fact: The largest area of growth on the Internet is the World Wide Web.[106] The web server software that is running most of the web sites on the World Wide Web is the open source Apache Web Server.[107]

The 1990's have seen – along with an increasing number of open source software and its increasingly important role in the expansion of the Internet[108] – the rise of an open source software development community.[109] Tied together by the Internet and an interest in 'liberating code,' the open source community has adopted a Stallman-esque collaborative model of software

---

[104] *Id.*

[105] *See supra* Part I.; *cf.* SEGALLER, *supra* note 103, at 290 (arguing that the World Wide Web succeeded because it was a "collaborative" project rather than being a commercial product).

[106] *See* CARL SHAPIRO & HAL R. VARIAN, INFORMATION RULES: A STRATEGIC GUIDE TO THE NETWORKED ECONOMY 6 (1999) (stating that "more than 60 percent of Internet traffic is to Web sites").

[107] *See* Netcraft, *The Netcraft Web Server Survey* (last visited Jan. 10, 1999) <http://www.netcraft.com/Survey/> [hereinafter *Netcraft Web Server Survey*] (the Apache Web Server – with 53.78% server share – has almost two times the server share of the nearest competitor – Microsoft, with 23.68% server share). The Apache Web Server is licensed under a slightly modified version of the BSD license, *compare* Apache Group, *Apache License* (last visited Jan. 8, 1999) <http://www.apache.org/LICENSE> [hereinafter *Apache License*], *with Sample BSD-style License*, *supra* note 101 (the Apache License is essentially the same as the BSD license with a few additions and modifications).

[108] *See, e.g.*, Halloween I, *supra* note 17 (analyzing the increasing number of open source projects and the importance of open source software to the growth of the Internet). There will be more on the interrelationship between open source software and the Internet in Part VI., *infra*.

[109] *See generally* Mann, *supra* note 14, at 36-42; McHugh, *supra* note 5, at 94-100.

development – where programmers create software in virtual communities by sharing the code, the ideas, and the time necessary to make it all work.[110] Needless to say, this kind of communitarian model of software development runs counter to the closed source model of software development used by the vast majority of software companies.

3. *Open Source Hits the Big Time: The Increasing Attention Paid to the Open Source Movement, and the Commercialization of Open Source Software – 1998 to the Present*

As stated in the Introduction, the open source movement – especially Linux – has garnered considerable attention recently.[111] Influenced by the growing open source community – and especially by what many consider to be the open source community's equivalent of the 'Declaration of Independence,' Eric Raymond's *The Cathedral and the Bazaar*[112] – Netscape decided to release the source code of its Communicator browser to the public.[113] As almost every open source software developer has had to do – and especially because of the fact that it is a for-

---

[110] *See* McHugh, *supra* note 5, at 94-100; Eric S. Raymond, *The Cathedral and the Bazaar* (visited Aug. 12, 1998) <http://sagan.earthspace.net/esr/writings/cathedral-bazaar/> [hereinafter Raymond, *The Cathedral and the Bazaar*] (arguing in favor of a collaborative model of software development epitomized by the open source movement). For an example of the enthusiasm of the open source community, *see* Steven Levy, *Code Warriors*, NEWSWEEK, Jan. 18, 1999, at 60 [hereinafter Levy, *Code Warriors*] (reporting on the zeal of the Philadelphia Linux Users Group as Eric Raymond preaches the virtues of open source).

[111] *See supra* Part I.

[112] *See* Raymond, *The Cathedral and the Bazaar*, *supra* note 112. On the importance and the influence of Raymond's piece, *see, e.g.*, Leibovich, *supra* note 8; Levy, *Code Warriors*, *supra* note 110, at 60; McHugh, *supra* note 5, at 99; Doc Searls, *Betting on Darwin*, LINUX JOURNAL, Aug. 1998, at 12 (interviewing Marc Andreessen – co-founder of Netscape – and Tom Paquin – in charge of Mozilla.org [Mozilla.org is the division of AOL/Netscape responsible for the development of open source software projects, *see* CUSUMANO & YOFFIE, *supra* note 11, at 36, 39, 139] – about Netscape's decision to release the source code of its software; its decision was heavily influenced by Raymond's article).

[113] *See* Netscape Whitepaper, *supra* note 24; Leibovich, *supra* note 8 (reporting on Eric Raymond's influential role in both the open source community and in Netscape's decision to go open code); Searls, *supra* note 112, at 12-25 (interviewing Marc Andreessen and Tom Paquin of Netscape) (stating that Netscape was influenced by *The Cathedral and the Bazaar*).

For an insiders' view of Netscape's decision to join the open source movement, *see generally* Jim Hamerly et al., *Freeing the Source: The Story of Mozilla*, *in* OPEN SOURCES, *supra* note 62, 197 (1999) [hereinafter Hamerly, *Story of Mozilla*].

profit company that makes commercial software – Netscape had to make a decision to either: a.) adopt (and, perhaps, slightly modify) an existing open source license, or b.) develop its own open source license.[114] Netscape decided to create its own open source licenses by adopting elements from existing open source licenses, as well as creating terms that were appropriate given Netscape's position as a commercial software producer.[115] Netscape's open source licenses will be discussed in Part III.C. of this paper.

In addition to Netscape's decision to go open source, the current period of the open source movement has seen an increasing level of commercialization.[116] Some software companies have either begun to 'sell' open source software by charging for value added services[117] or have utilized open source software as components of software products that may or may not be open source.[118] The emergence of commercialized open source software – which may lead to open source software becoming 'closed' or 'quasi-open'[119] – highlights the importance of how open source licenses are structured and whether or not they are enforceable.

---

[114] *See* Mozilla.org, *Netscape Public License FAQ* (last visited Aug. 12, 1998) <http://www.mozilla.org/NPL/FAQ.html> [hereinafter *Netscape License FAQ*] (see, particularly, ¶3); *see also* Netscape Whitepaper, *supra* note 24.

[115] *See Netscape License FAQ*, *supra* note 114; *see also infra* Part III.C.4.

[116] *See, e.g.*, Alex Lash, *Making Money with Free Software*, CNET NEWS.COM, (Feb. 2, 1998) <http://www.news.com/SpecialFeatures/0,5,18617,00.html> [hereinafter Lash, *Making Money*]; Nikki Goth Itoi, *Freeware: Sold*, RED HERRING, Feb. 1999, at 46 [hereinafter Itoi, *Freeware*]; Larry Seltzer, *Open-Source Means Business*, PC MAGAZINE, Mar. 23, 1999, at 176 [hereinafter Seltzer, *Business*].

[117] *See* Lash, *Making Money*, *supra* note 116; McHugh, *supra* note 5, at 99-100.

[118] *See* Lash, *Making Money*, *supra* note 116; Perens, *The Open Source Definition*, *supra* note 43, at 175-76, 186 (discussing the difficulty that the makers of KDE – a Graphical User Interface ('GUI') for Linux – had when they tried to mix open source software with a component that was under a closed source software license).

[119] Some commercial developers that utilize open source software in their products have alleged that they have at most a 'moral' – but not a legal – obligation to abide by the terms of open source licenses, *see* Lash, *Making Money*, *supra* note 116; Lash, *Source Code for the Masses*, *supra* note 13; *see also* Perens, *The Open Source Definition*, *supra* note 43, at 175-76, 186 (the KDE situation).

III. OPEN SOURCE LICENSING

A. Some Preliminary Issues

Before we can define the parameters of an open source license – as well as describe the differences between various types of open source licenses – we need to take a brief look at the mechanics of copyright[120] licensing for software.

The process of open source software licensing is similar, in some respects to the process of closed source software licensing.[121] First, the creator copyrights the open source software. After copyrighting the software, the creator offers the open source software to licensees under the terms of the license. As the reader may have noticed, the process of licensing open source software is identical to the FSF's process of 'copylefting' as described in Part II.B., *supra*.[122]

As was discussed in relation to copylefting – when we go beyond the superficial similarities – we find that there are some striking differences between open source licenses and closed source licenses. The differences between open and closed source licenses will be described in Parts III.B. and IV.

We should note for now, however, several points that are important for the rest of this paper:

1. The Copyrightability of Software

---

[120] Since every open source license that we are dealing with is a copyright license, this paper will not deal with other intellectual property licenses – such as licenses for patents, trademarks, etc.

[121] For background information on copyright licensing (in general), *see generally* JAY DRATLER, JR., LICENSING OF INTELLECTUAL PROPERTY §§ 1.01-1.05 (1997) (providing an overview of intellectual property licensing); 1 ROGER M. MILGRIM, MILGRIM ON LICENSING §§ 5.00-5.80 (1998) [hereinafter MILGRIM ON LICENSING] (discussing copyright licensing).

For background on issues specific to computer software licensing, *see generally* William H. Neukom & Robert W. Gomulkiewicz, *Licensing Rights to Computer Software*, *in* TECHNOLOGY LICENSING AND LITIGATION 1993, 354 PLI/PAT 775 (1993) (Microsoft's attorneys describing licensing in the computer software industry).

[122] *See* discussion *supra* Part II.B.1; *see also* FSF, *GNU GPL*, *supra* note 58, at Preamble.

The availability of copyright protection for computer software is a well-established principle in copyright law.[123] Computer software can be protected through copyright in both its source code and object (binary) code forms.[124] Generally speaking, copyright for computer software applies regardless of whether the software is fixed on a magnetic or optical storage media (usually 'disks'), on a paper print out, or on a semiconductor ('chips').[125]

Despite the general tendency to allow copyright protection for software, it should be noted that the courts have not always been willing to protect every element of a given piece of software via copyright law.[126] For the purposes of this paper, however, the reader should assume that the open source software in question – in both source and object code form – is copyrightable in its entirety, and as to every component, unless otherwise noted.[127]

2. 'Licensing' as Opposed to 'Selling'

---

[123] *See* 17 U.S.C. §§ 101, 117 (§ 117 essentially implies that computer programs are covered by the exclusive rights of copyright holders stated in § 106); Whelan Associates, Inc. v. Jaslow Dental Library, Inc. 797 F.2d 1222, 1234 (3d Cir. 1986), *cert. denied*, 479 U.S. 1031 (1987) (holding that computer software can receive copyright protection as 'literary work'); Hamilton & Sabety, *supra* note 3, at 240 (stating that "the Copyright Act unambiguously protects computer programs ...").

[124] *See Apple Computer*, 714 F.2d at 1246-49.

[125] *See* 17 U.S.C. § 101; *Apple Computer*, 714 F.2d at 1249 (expression in Read Only Memory (ROM) meets the "statutory requirement of 'fixation'"). This principle has been taken to its logical extremes. For example, courts have held that if software code is held by a computer's Random Access Memory (RAM) for a few minutes, it is 'fixed' for the purposes of copyright law, *see* MAI Systems Corp. v. Peak Computer, Inc., 991 F.2d 511, 518 (9th Cir. 1993), *cert. denied*, 510 U.S. 1033 (1994); Advanced Computer Services v. MAI Systems Corp., 845 F.Supp. 356, 363 (E.D. Va. 1994); *but see* 17 U.S.C. § 117 (if the temporary fixation on computer memory (either on disks or on chips) is an "essential step" in the utilization of the program, or it is for an archival copy, then the fixation does not amount to a copyright infringement); Digital Millennium Copyright Act, Pub. L. No. 105-304, §§ 301-302, __ Stat. __ (1998) (legislatively overruling *MAI v. Peak* for the purposes of maintaining or repairing a computer).

[126] *See, e.g.*, Lotus Development Corp. v. Borland International, Inc., 49 F.3d 807, 819 (1st Cir. 1995), *aff'd by an equally divided court*, 116 S. Ct. 804 (1996) (the 1st Circuit holding that Lotus' menu command structure was not entitled to copyright protection since it was a "method of operation"); Computer Associates International, Inc. v. Altai, Inc., 982 F.2d 693, 703-12 (2d Cir. 1992) (holding that a program consisted of a number of different elements and that noncopyrightable elements of the program should be excised before a court can consider the overall copyrightability of the program in question).

[127] This assumption is made so as to avoid unnecessary complications. From this point forward, this paper will focus on the enforceability of the license rather than the copyrightability of the underlying software.

Typically, software is 'licensed' rather than 'sold.'[128] Most closed source software producers have several reasons for licensing their software: First, they want to negate the 'Doctrine of First Sale.'[129] Second, software producers want to define the terms of their warranty.[130] Third, software producers, may want to use the license to establish and extend their intellectual property rights.[131] Finally, producers may want to use the license to add in "Other Terms and Conditions" – including 'boilerplate' provisions such as choice of venue and law, limited liability, and terms that attempt to limit the rights that the licensee may ordinarily have under intellectual property law.[132]

Open source software producers, on the other hand, have different motivations for licensing their software. The goals of open source software licensing will be explored in the next subsection of this paper.

3. A Note on Terminology

There are several different types of open source licenses. When this paper uses the term 'open source software license' – or some derivative of that phrase – this paper will refer to open source software licenses in its most generic form. In other words, 'open source software license' will be used to refer to open source licenses in general and/or it will refer to an idealized model of an open source license that is intended to represent the core elements shared by all the different

---

[128] *See* Neukom & Gomulkiewicz, *supra* note 121, at 777-78; Perens, *The Open Source Definition*, *supra* note 43, at 181.

[129] *See* Neukom & Gomulkiewicz, *supra* note 121, at 778. Under the 'First Sale' Doctrine, "once a copy of a copyrighted work has been sold, the copyright holder's rights in that particular copy are exhausted, and that copy must be freely resold, leased or loaned," *id.*

[130] *See id.*

[131] *See id.*

[132] *Id.* These restrictions will be discussed later on in this paper, *see* discussion *infra* Part IV.

types of open source licenses.

When terms like 'GNU GPL,' 'BSD-style License,' 'Apache License,' etc., are used in this paper, those terms will be used to refer to the specific type of open source software license that the phrase connotes.

B. Defining the Parameters of Open Source Licensing

As stated in Part II.A., the simplest definition of an open source software license is that it is a license that attempts to confer the kind of rights, privileges, and obligations associated with the goals of the open source software movement. For the purposes of this paper, a license will have to meet the following conditions in order to qualify as an open source software license:[133]

(1) The license must keep the 'source code'[134] 'open' and available.[135]

(2) a.) The license must maintain the integrity of the author's source code.[136]

---

[133] Most of the defining terms listed here are adapted from the sources cited in Part II.A., *see supra* notes __ and accompanying text. **Note:** This paper's definition of an open source license will depart, slightly, from the definition of an open source license as stated in the *OSD*. There are two reasons for not totally embracing the definition given by that authoritative source:

(1) In order to keep the analysis tractable, some of the details of open source licensing need to be left out in order to focus in on the core issues that present unique problems for open source licensing.

(2) Some of the defining conditions: a.) may be controversial within the open source community, and b.) may be logically contradictory. For example, ¶ 9 of the *OSD* states that a license cannot "insist that all other programs distributed" along with the open source licensed software be open source, *see OSD (v.1.0)*, *supra* note 44. Yet ¶ 10 states that the [GNU] GPL is an example of a license that conforms "to the Open Source Definition" – despite the fact that the GNU GPL might be construed so as to insist that **all** software distributed along with GNU GPL licensed software be open source, *see* discussion *infra* Part III.C. & D.; *see also* Perens, *The Open Source Definition*, *supra* note 43, at 182, 185 (the GNU GPL would not allow a 'mix' between a copylefted/open source program and a closed source program; *but see id.*, at 179-80 (the author of the Open Source Definition, apparently, sees no logical contradiction). However, ¶ 9 might not cause a problem for the GNU GPL since it may only prohibit 'integration' at the source code level rather than 'aggregation' on the same medium of distribution, *see infra* Part III.C.

[134] The phrase 'source code,' as used here, refers to the 'original source code base' and ***not necessarily*** to works derived from that original source code base. An open source license – to qualify as such – need not require that modifications to the original source code base stay open, *see* discussion *infra* Part III.C. An open source license only needs to make sure that source code can be used, modified, and distributed "without corrupting the ***original*** projects," Seltzer, *License*, *supra* note 38, at 171 (emphasis added).

[135] Corresponds to Point 1 of Part II.A. (definition of open source software), *supra*.

b.) Also, with a few exceptions, the license should acknowledge the authorship of the code.[137]

(3) "The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original [open source] software."[138]

(4) The license must allow 'free redistribution' of the open source software. This can mean that a licensee can make copies of the software and give it away. However, since 'free' in this context means 'freedom from constraint' rather than 'zero price,' the license should allow the licensee to *sell* the software.[139]

(5) a.) "The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties."[140]

b.) "The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the

---

[136] Corresponds to Point 2 of Part II.A., *supra*.

[137] The rationale behind this condition is the following: "People take great pride in their work and do not want someone else to come along and remove their name from it or claim they wrote it." Debian, *Free/Open Software*, *supra* note 65. [However, the 'modified BSD' license may carve out an exception to giving credit where credit is due. *See* discussion *infra* Part III.D.]

[138] *OSD (v.1.0)*, *supra* note 44, ¶ 4. Corresponds to Point 3 of Part II.A., *supra*. Notice that open source licenses do not have to require licensees to put their derivative works under the same license as the original program. Open source licenses are merely required to *allow* provisions that would put derivative works under the same license as the original work. For the reasoning behind this latitude, *see* discussion *infra* Part III.C.

[139] *See* Debian, *Free/Open Software*, *supra* note 65.

[140] *OSD (v.1.0)*, *supra* note 44, ¶ 7.

terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software."[141]

? **In other words, the terms and conditions of the open source software is distributed and/or redistributed along with the software. The terms of the license are effectively attached to the software and cannot be severed from the code.**

The goals behind open source software licensing are very different from those underlying closed source software licensing. Whereas closed source licenses are designed to guard and even extend the licensor's intellectual property protections, opens source licenses are designed to essentially weaken the licensor's intellectual property rights.[142] Besides protecting the highly valuable source code of a program, a closed source license tends to restrict the redistribution of both the source code and the binary code.[143] Open source licenses, on the other hand, place little or no restrictions on the redistribution of open source software.

C. Specific Examples of Commonly Used Open Source Licenses: GNU GPL, Artistic, BSD-style, Netscape PL, and Mozilla PL

By in large, the open source community is unified in the goal of advancing open source software development.[144] In furtherance of this goal, the open source movement has adopted

---

[141] *Id.* ¶ 8.

[142] *See* discussion *infra* Part IV.; *see also* Debian, *Free/Open Software*, *supra* note 65, (comparing the use of copyright licensing by closed source and open source developers); FSF, *GNU GPL*, *supra* note 58, at Preamble (arguing for the use of open source licenses in order to provide 'freedom' for programmers and software users).

[143] *See* ALDUS CORP., ALDUS LICENSE AGREEMENT (1991), *reprinted in* Neukom & Gomulkiewicz, *supra* note 121, at 791 [hereinafter Aldus License] (this closed source license only allows one copy of the software to be used on a single computer – with an allowance for the need to make one archival copy).

[144] *See generally* OPEN SOURCES, *supra* note 62 (the short pieces contained in this anthology represent a broad

software licensing as a way of sustaining the open source development model.

Rather than adopting a unified open source licensing scheme, however, elements within the open source movement have created and utilized various types of open source licenses. The diversification of licensing schemes is due to the differing philosophies within the open source community on how best to advance the open source movement[145] and the need to have particularized licensing provisions to deal with issues that surround a particular piece of open source software.[146]

Since all open source licenses have to meet the criteria stated in Part III.B. of this paper – which includes keeping the source code open, allowing derivative works, and allowing the (relatively) unfettered redistribution of software – the key difference between the various types of open source licenses are what mechanism, if any, will be utilized to enforce the 'openness' of open source software. Some licensing schemes utilize licensing provisions in the attempt to maintain a strict wall of separation between open and closed source software. Other licensing schemes – while demanding that the original source code be open – allows the licensee to have discretion over the question of to what degree open source software should be 'open' (or 'free') and 'untainted' by closed source software.

The question of how to maintain the 'openness' of open source software via licensing and

---

spectrum of views within the open source movement).

[145] *See* Perens, *The Open Source Definition*, *supra* note 43, at 182 (explaining why some in the open source community refuse to use the GNU GPL because of ideological differences with GNU's founder, Richard Stallman); *cf.* Eric S. Raymond, *ESR on O'Reilly Summit*, SLASHDOT, (Mar. 11, 1999) <http://slashdot.org/features/99/03/11/1352249.shtml> (Raymond downplays clashes within the open source community over licensing but acknowledges that open source licenses are "everybody's favorite subject for doctrinal warfare"). Further discussions about the conflicting philosophies within the open source community will be dealt with in Part III.D., *infra*.

[146] *See Netscape License FAQ*, *supra* note 114 (explaining why Netscape/Mozilla, as a commercial software producer that may have needs that are different from universities or hobbyists, decided to create its own open source licenses rather than rely on available licenses commonly used by open source developers).

the choice between a license that has stricter provisions and a license that has more lenient provisions, will depend in large part on the ideological take adopted by a particular faction within the open source community. The ideological differences and the affects such differences have on the choice of open source licensing schemes will be analyzed in Part III.D, *infra*. In the meantime, however, the following examples of some commonly used open source licenses illustrate how the differing needs and visions of various factions within the open source community are reflected in the various types of open source licenses.

***

A note to the reader: This paper will not analyze every single open source license that currently exists. The reasons for this analytical choice are as follows: 1.) Limiting the number of licenses analyzed keeps the analysis tractable; 2.) The licenses analyzed in this paper are the most commonly used licenses among open source developers; and 3.) Conceptually speaking, the analysis applied to the licenses covered in this paper are applicable, with minor alterations, to licenses not covered in this paper.[147]

1. The GNU GPL

The progenitor of all open source licenses,[148] the GNU General Public License is one of the most commonly used licenses in the open source community. While it is the licensing scheme of choice for many open source software projects, its biggest claim to fame is the fact that Linux is licensed under the GNU GPL.

One could easily imagine from the rhetoric and philosophy of Richard M. Stallman, the

[147] In addition, it should also be noted that licensing schemes that are at best 'quasi-open' but do not meet the commonly accepted standards of 'open source' – such as Sun's 'Community Source Licensing' – will not be analyzed in this paper. [Although I imagine that much of the analysis here will be applicable to community source licensing as well.] For more on Sun's 'Community Source Licensing,' *see* Richard P. Gabriel & William N. Joy, *Sun Community Source Licensing Principles*, (last visited Mar. 11, 1999) <http://www.sun.com/981208/scsl/principles.html>.

[148] *See supra* Part II.B. for the background behind the GNU GPL.

founder of the GNU Project/Free Software Foundation, with his belief that closed source software producers are "software hoarders"[149] and his opposition to intellectual property rights over software,[150] that the GNU GPL would include licensing provisions that would tend to take a harder stand on maintaining the 'purity' of open source software. In fact, the GNU GPL is the strictest of all of the commonly used open source licenses when it comes to utilizing the mechanism of licensing provisions to ensure the 'openness' of 'copylefted' software.

There are two principle methods by which an open source licensing scheme can keep a high degree of separation between the closed source and the open source development models. The first way in which an open source license can maintain a high degree of 'purity' is by denying a licensee the discretion to take the licensee's modifications to the open source code base and turn that modification into a closed source/proprietary piece of software. In other words, a relatively strict license goes beyond the baseline open source license because a baseline license, as defined in Part III.B., would only attempt to ensure that the original source code base remains open. A stricter open source license would go beyond our hypothetical baseline license by attempting to ensure that any **derivative works** from that original source code base also remain open.[151]

The second way in which an open source license can maintain the 'openness' of open source software is through the use of licensing provisions to prevent the 'mixing' of open source software with closed source software. It should be noted that what we mean by 'mixing' is that an extremely strict open source license would prevent the integration of open source code with closed source code to create a mixed, part-open/part-closed piece of software. A provision

---

[149] *See* FSF, *Copyleft*, *supra* note 76; *see also* supra note 84 and accompanying text.

[150] *See generally* Richard Stallman, *Reevaluating Copyright: The Public Must Prevail*, 75 OR. L. REV. 291 (1996) (Stallman is critical of intellectual property protections for digital intellectual 'property').

[151] *See generally* Perens, *The Open Source Definition*, *supra* note 43, at 180-85. *See also Netscape License*

against mixing, normally, will **not** mean that a piece of open source software cannot be delivered on the same medium with a piece of closed source software, nor will it mean that open source software cannot be used along with closed source software on a user's computer.[152]

The GNU GPL is the strictest of the open source licenses because it has licensing provisions to prohibit both the 'closing' of modifications to a copylefted (i.e., licensed under the GNU GPL) open source program and the mixing of open and closed source codes.[153] § 2(b) of the GNU GPL attempts to prevent the licensee from having the ability to keep any work that has been derived from a copylefted program private and, therefore, away from the open source community:

> You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the [copylefted] Program or any part thereof, to be licensed as a whole at no charge to all third parties *under the terms of this License*.[154]

In other words, in order to prevent someone who has benefitted from the sharing of source code from 'closing' off – to both the open source community and the public-at-large – works derived from that open source code, the GNU GPL "prevents a programmer from establishing copyright or patent rights in the [copylefted] software."[155]

FAQ, *supra* note 114 (providing an excellent explanation of the rationale behind many open source licensing schemes).

[152] *See* Perens, *The Open Source Definition*, *supra* note 43, at 180-83.

[153] *See id.* at 181-82, 185; Halloween I, *supra* note 17; *Netscape License FAQ*, *supra* note 114, ¶¶ 6, 10; *see also* Debian GNU/Linux Developers, *Comparison of Software Licenses*, (visited Jan., 12, 1999) <http://www.debian.org/intro/license_disc>.

[154] FSF, *GNU GPL*, *supra* note 58, at § 2(b) (emphasis added). *See also supra* note 153.

[155] Heffan, *supra* note 9, at 1508. *See also id.* at 1507; FSF, *Copyleft*, *supra* note 76 (stating that "it is illegal to distribute [an] improved version [derived from copylefted code] except as free software ...").

The other way in which the GNU GPL attempts to maintain a strict barrier between open code and closed code is by prohibiting the integration of a copylefted open source program with a closed/proprietary source program at the source code level. According to the language of the GNU GPL, "[The] General Public License does not permit incorporating your program [that was derived from or contains, in whole or in part, a copylefted program] into proprietary programs."[156] While there are ways of getting around this particular prohibition,[157] the advocates of 'copylefting' – such as Richard Stallman – strongly encourage open source developers to accept this limiting condition along with the prohibition against proprietizing modifications.[158]

a. Is the GNU GPL 'Viral'?[159]

From a practical standpoint, the strictness of the GNU GPL has led to speculation that the GNU GPL may actually 'infect' closed source software and negate any associated intellectual property rights attached to those pieces of software. The GPL's provisions to prevent both the proprietization of derivative works and the integration of open source code and closed source code – along with the provision for the terms of the license to automatically apply each time the

[156] FSF, *GNU GPL*, *supra* note 58, at 'How to Apply These Terms ...," ¶ 14. This particular prohibition stems from the language of § 2(b) which covers works that "in whole or in part *contains* ... the Program or any part thereof ..." *Id.* § 2(b). *See also* Halloween I, *supra* note 17; Perens, *The Open Source Definition*, *supra* note 43, at 182 (stating that the "GPL doesn't allow the incorporation of a GPL-ed program into a proprietary program").

[157] Foreseeing the problems that might arise when a subroutine library is involved (where it might be necessary to link proprietary software to that library) – the FSF created the GNU Library General Public License (the 'LGPL'). This paper will not analyze the LGPL since the analysis offered in the rest of this paper can be easily modified and applied to this less stringent, *vis a vis* the GPL, licensing scheme. For more information on the LGPL, *see* Perens, *The Open Source Definition*, *supra* note 43, at 182-83.

[158] *See* Richard Stallman, *Why You Shouldn't Use the Library GPL for Your Next Library*, SLASHDOT, (Feb. 1, 1999) <http://slashdot.org/articles/99/02/01/1730200.shtml> [hereinafter Stallman, *Library*] (encouraging open source developers who use the LGPL to use the GPL in order to support the vitality of the open source development community).

[159] The word 'viral' was used in the Economist to describe the GNU GPL. *See Hackers Rule*, ECONOMIST, Feb. 20, 1999, at 63 [hereinafter Economist, *Hackers Rule*].

copylefted program is redistributed[160] and the provision that "if a user cannot distribute derivative software covered by the GNU GPL without a patent [or a copyright, either of which may violate the terms of the GNU GPL], then the user may not distribute the software at all"[161] – lend credence to the suspicions of some that introducing a program that has been licensed under the GNU GPL into an environment that contains closed source software may lead to a virus-like usurpation of the intellectual property protections over the closed source software present in that environment.[162] In other words, there is a fear that having copylefted open source programs along with closed source programs may mean that the intellectual property rights attached to the closed source software may be trumped by the GNU GPL, a license that weakens intellectual property rights.

The answer to the question of whether or not the GNU GPL is 'viral' is both a yes and a no. Yes, the GNU GPL's provisions apply automatically as a copylefted program is redistributed, and yes its provisions apply when derivative works are created as well as when the copylefted program is integrated with other programs.[163] So, in some sense, the GNU GPL is 'viral' because its affects can be 'spread' through the redistribution of a copylefted program.[164]

---

[160] *See* FSF, *GNU GPL*, *supra* note 58, § 6.

[161] Heffan, *supra* note 9, at 1508 (citing § 7 of the GNU GPL).

[162] It should be noted that this is essentially based on rumor and, therefore, is hard to substantiate. However, there is some circumstantial evidence that tends to show that these concerns do exist. For example, some businesses allegedly refuse to allow the use of any software licensed under the GNU GPL (i.e., copylefted software), *see also* Anonymous, *Rumours*, SLASHDOT, (Feb. 10, 1999) <http://slashdot.org/articles/99/02/10/2143243.shtml> (**Note: This article in Slashdot is only a rumor. It has not been verified. The author does not vouch for the accuracy of what was stated in this article.**). Analysis of the GNU GPL, arguably, hints at this sort of problem, *see* Perens, *The Open Source Definition*, *supra* note 43, at 181-82; *Netscape License FAQ*, *supra* note 114, ¶ 10(1)-(4) (explaining that concerns stemming from the presence of closed source code that had associated intellectual property restrictions, that Netscape either did not want to or could not interfere with, prevented Netscape from utilizing the GNU GPL).

[163] *See* FSF, *GNU GPL*, *supra* note 58, § 6.

[164] *See* Berman, *Linux Legal*, *supra* note 39 (legal scholars, like Mark Lemley, recognize the idea that the terms of the GNU GPL spread "virus-like" down the "license chain" binding "the recipient and anything the recipient

However, the fears that the GNU GPL might infect and negate intellectual property protections for non-copylefted, non-open source programs are probably overblown. The GNU GPL only attempts to hinder the integration of copylefted programs with closed source programs *at the source code level*.[165] The GNU GPL prohibits the mixing of copylefted source code with proprietary source code in order to create a kind of 'hybrid' program. The GNU GPL does not attempt to prohibit "the mere aggregation of another work not based on the [copylefted] Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium ..."[166] In other words, it is fine under the GNU GPL to have both copylefted and non-copylefted programs residing on the same disk, CD-ROM, Ethernet cable, etc. However, problems will arise when copylefted source code is combined with non-copylefted closed source code so that a copylefted program "actually incorporates part [or the whole] of another [closed source] program into itself,"[167] or when the reverse, a closed source program incorporates a copylefted program, is true.

Although having a program licensed under the GPL will not infect and usurp the existing intellectual property rights of a closed source program that may be used along with the copylefted program, that does not mean, however, that there might not be other problems. The GNU GPL can cause problems when someone wants to use or modify copylefted software but either is not fully committed to using *only* open source software (or free software in GNU-speak) or

---

passes on").

[165] Technically, the incorporation problem also includes scenarios where programs are linked to libraries.

[166] FSF, *GNU GPL*, *supra* note 58, § 2. *See also* Perens, *The Open Source Definition*, *supra* note 43, at 180. Bruce Perens uses the 'derivation' versus 'aggregation' distinction to explain this concept. However, I believe that it is somewhat more conceptually coherent to distinguish between the integration at the source code level versus simply having the different categories of software on the same disk, hard drive, CD-ROM, cable, etc.

[167] Perens, *The Open Source Definition*, *supra* note 43, at 180.

unintentionally integrates a copylefted piece of software with a proprietary piece of software. Because the GNU GPL is 'viral' in the sense that it demands that the copylefted program and any derivative work remain open source and remain under the GPL – and that every recipient of a copylefted program comply with the GPL – anyone along the chain of the distribution of the program is affected by the strict provisions of the GPL.

It is worth noting, here, that there have already been at least two instance where the incorporation of closed source programs with copylefted programs have led to licensing controversies within the open source community: KDE[168] – a popular Graphical User Interface [GUI] for Linux – and Mosix (for Linux)[169] – a cluster computing support module for Linux that contains "modified copies of Linux kernel files."[170] As will be discussed in Part III.D., the relative strictness of its terms have caused some in the open source development community to adopt other licensing schemes rather than the GNU GPL.[171]

2. The Artistic License

The Artistic License was originally created by Larry Wall to cover his open source Perl programming language,[172] but has been used to license other open source software as well.[173] The

[168] *See id.* at 175-76, 182, 186 (KDE – licensed under the GPL – depended on a closed source graphical library called Qt from Troll Tech; this situation led to problems for third parties that redistributed KDE along with Qt because these third parties may have been inadvertently violating the GPL).

[169] The leader of the GNOME project (another GUI for Linux), Miguel de Icaza (for more on GNOME, *see* Mann, *supra* note 14, at 36-38), and others in the open source community claim that Mosix is violating the GNU GPL by not releasing the source code to a module that is a modification of the Linux kernel. *See* Miguel de Icaza & 'S,' *GPL Violation of the Linux kernel?*, SLASHDOT, (Feb. 27, 1999) <http://www.slashdot.org/articles/99/02/27/076204.shtml> [hereinafter Slashdot, *GPL Violation*] (note: I will refer to 'Slashdot' as the 'author' of this piece throughout the rest of this paper since there were many others who had a hand in writing that page besides Miguel de Icaza and 'S'). *See also* Mosix, *GPL* (published on the Mosix website, but has since been removed) (on file with the author) (Mosix responding that any violation of the GNU GPL was inadvertent and that they will seek to comply with the terms of the license).

[170] Mosix, *GPL*, *supra* note 169.

[171] *See, e.g.*, *Netscape License FAQ*, *supra* note 114, ¶ 10.

[172] For background on the Artistic License, *see supra* Part II.B.1. (discussing Perl).

Artistic License is – intentionally – more lenient in its licensing terms than the GNU GPL.[174]

Unlike the GNU GPL, the Artistic License will normally not prohibit a licensee from claiming intellectual property rights over modifications to the licensed program (i.e., the original source code base).[175] Therefore, with the exception of a few potentially limiting conditions,[176] the Artistic License places little or no barriers to 'closing' the source code of the works derived from the original open source code base.[177] In addition, again differing from the GNU GPL, the Artistic License does not place limits on the 'mixing' of open and closed source programs – either in integrated or in aggregated forms.[178]

In short, the Artistic License does not place as many limitations on the licensee as the GNU GPL. Rather than attempting to erect or sustain a high wall of separation between open and closed source programs, the Artistic License merely allows an open source developer, as a copyright holder, to "maintain[] some semblance of artistic control over the development of the

---

[173] *See* Perens, *The Open Source Definition*, *supra* note 43, at 183.

[174] Larry Wall had originally copylefted Perl, but decided that the GPL's terms were 'too restrictive.' *See supra* Part II.B.1.; *see also* discussion *infra* Part III.D.

[175] *See* Perl, *Artistic License*, *supra* note 92, ¶¶ 2, 3, 6, 7; *see also* Lash, *Evolution*, *supra* note 88 (stating that both the Perl (Artistic) and the BSD License, unlike the GNU GPL, "[do] not require developers to submit changes to the source code back to the community"); Perens, *The Open Source Definition*, *supra* note 43, at 183-84.

[176] There are some conditions that need to be met in order to avoid the few limitations that the Artistic License could place on derivative works. *See* Perl, *Artistic License*, *supra* note 92, ¶¶ 3(a)-(b), 6, 7. However, the 'safe harbors' that are provided under the Artistic License are, practically speaking, easily met.

[177] But, like every open source license (*see* Part III.B., §1), the Artistic License attempts to keep the original source code base open and freely available by stating that any modifications to the original "Standard Version' must meet one of four conditions. *See id.* ¶ 3 (a)-(d).

[178] *See* Perl, *Artistic License*, *supra* note 92, ¶¶ 5, 8 (Note: The only limitation is that a combination of closed source and open source is not such that a user may reasonably think that the combination is actually the same as the original open source program). It should be noted that ¶ 8 of the Artistic License actually encourages incorporation of closed source and open source programs by requiring that any combination of the two is done so that the open source program "is embedded" into a commercial distribution. *See also id.* ¶ 6 (allows closed source scripts and libraries to be associated with the open source software without it necessarily falling under the Artistic License or the copyright of the open source software).

[program], while giving the users of the [program] the right to use and distribute the [program] in a more-or-less customary fashion, plus the right to make reasonable modifications."[179] The Artistic License, contrary to the GNU GPL, allows the licensees to have a great deal of discretion. In fact, so much discretion is allowed that a licensee could keep the benefits derived from the original open source code base – i.e., the modifications – private and away from the open source development community.

It is worth noting that the use of the Artistic License is waning within the open source community because the Artistic License, legally speaking, is worded in such a way as to create some major logical inconsistencies with the aims of open source development.[180] Someone who is considering developing open source software should, therefore, consider using the GNU GPL, a BSD-style license, or other licenses that are worded more carefully from a technical and legal perspective so as to stay within the ambit of open source development.

3. BSD-style Licenses

The BSD license was, originally, utilized as an open source licensing scheme for a variant of BSD Unix known as 'FreeBSD.'[181] As the open source community has grown and developed, the BSD license, often with slight modifications, have been utilized by other open source developers – including the Apache Group.[182] For most open source developers, the BSD-style[183]

---

[179] *Id.* at Preamble.

[180] *See* Perens, *The Open Source Definition*, *supra* note 43, at 184.

[181] Please note that this paper will only deal with the variant of BSD known as 'FreeBSD.' However, that should not suggest that other versions of BSD – such as 'NetBSD,' 'OpenBSD,' etc. – do not qualify as open source. They normally will fall under the definition of 'open source' so long as they are licensed under some sort of a BSD-style license. For background information on BSD and FreeBSD, *see* Part II.B.2. For a more detailed description of FreeBSD (and BSD in general), *see* Cameron Laird & Kathryn Soraiz, *The Story of FreeBSD*, LINUXWORLD, (last visited Jan. 14, 1999) <http://www.linuxworld.com/linuxworld/lw-1998-12/lw-12-freebsd.html>.

[182] *See* Perens, *The Open Source Definition*, *supra* note 43, at 183; *see also supra* note 107. BIND and Sendmail are also licensed under BSD-style licenses. *See* Seltzer, *License*, *supra* note 38, at 171.

license is the most popular alternative to the GNU GPL.

The BSD-style license, unlike the GNU GPL, allows the licensee to take any modifications to the original open source code base private and allows the licensee to mix closed source software with open source software without the GNU GPL's limits on integration.[184] In fact, there are very few limitations placed on a licensee beyond giving credit to the original copyright holder(s) and to those who contributed to the source code of a BSD-style licensed program.[185] The following analysis by Netscape sums up the wide latitude that is given to a licensee under the BSD-style license:

> The BSD license is ... [an open source] license .... [that] is very non-restrictive in its terms, *basically allowing anyone to do anything with code covered by the license*, but requiring a reference to the copyright holder in accompanying documentation – *essentially requiring only credit where credit is due*. This makes the license acceptable to commercial developers, *but opens others to the possibility that their work may be incorporated into products that may be proprietary to someone else*.[186]

Those who use the BSD-style Licensing scheme tend to be open source developers who would like to have as little constraints as possible between the open source community and the closed source world.[187] Because the BSD-style License is a license that "let[s] you do anything

---

[183] For the reasons why this paper will often use the term 'BSD-style' License rather than 'BSD' License, *see supra* note 101.

[184] *See* Perens, *The Open Source Definition*, *supra* note 43, at 183, 185; Seltzer, *License*, *supra* note 38, at 171; *see also* Halloween I, *supra* note 17; *Netscape License FAQ*, *supra* note 114, ¶¶ 5, 12.

[185] *See Sample BSD-style License*, *supra* note 101, ¶¶ 1-4.

[186] *Netscape License FAQ*, *supra* note 114, ¶ 5 (emphasis added).

[187] *See* Laird & Soraiz, *supra* note 181 ("[The] 'BSD-style' license ... leads some to fear that [FreeBSD] might

with the software licensed under [it],"[188] it is well suited to those who want to have a greater degree of flexibility, *vis a vis* the GNU GPL – especially in a commercial context.[189] It is important to note, however, that the BSD-style licenses – while, arguably, the most lenient of the open source licenses – is still an open source license and, therefore, programs licensed under a BSD-style license still meet the definition of an open source program.[190]

Despite the relative paucity of limitations under the BSD-style license, the BSD-style licenses do carry conditions that attempt to 'give credit where credit is due.'[191] Under most BSD-style licenses, each redistribution of the licensed open source software must carry along with it the terms of the license[192] – which includes the copyright notice[193] and a condition that "[t]he name of the author may not be used to endorse or promote products derived from this software without specific prior written permission."[194] The BSD-style license also contains a condition that "all advertising materials mentioning features or use of this software" display an acknowledgment of the originator of the licensed program[195] – for FreeBSD, the University of California,[196] and for

---

go commercial one day, leaving its users no open-source recourse. .... [However,] several organizations resist [using the GNU GPL] ... because of the constraints it imposes on commercialization of their own work.").

[188] Perens, *The Open Source Definition*, *supra* note 43, at 183.

[189] *See* Laird & Soraiz, *supra* note 181; Larry Seltzer, *Apache Rules the Web*, PC MAGAZINE, Mar. 23, 1999, at 178 [hereinafter Seltzer, *Apache*] (noting that the BSD License gives users of Apache a great deal of flexibility in a commercial context); Seltzer, *License*, *supra* note 38, at 171.

[190] *See* Perens, *The Open Source Definition*, *supra* note 43, at 183; *see also supra* Parts II.A. & III.B.

[191] *See* James C. Luh, *FreeBSD Offers a Sound Open Source Alternative*, INTERNET WORLD, (Apr. 12, 1999) <http://www.internetworld.com/print/1999/04/12/webdev/19990412–freebsd.html> (BSD license – according to the CEO of FreeBSD, Inc., Jordan Hubbard – imposes on the licensee the duty to give credit where credit is due).

[192] *See Sample BSD-style License*, *supra* note 101, ¶¶ 1-2.

[193] *See id.*

[194] *Id.* ¶ 4.

[195] *Id.*; *see* Perens, *The Open Source Definition*, *supra* note 43, at 183.

Apache, the Apache Group.[197] However, it is worth noting, that the 'advertising clause' has been criticized on many fronts[198] and some open source developers use a 'modified BSD-style license' that has the advertising clause removed.[199]

a. The Apache License

The Apache License is essentially a BSD-style license (including the advertisement clause) plus some additional conditions.[200] On its face, the Apache License seems to do little more than to make a few alterations necessary to meet some of the specific needs that the Apache Group may have had while maintaining the general theme of a BSD-style license. However, in practice, the Apache License may actually give incentives for a licensee to have a greater commitment to the open source development model than would be required by an ordinary BSD-style license.

The Apache Group "takes the BSD-style open source model and extends it by allowing check-ins to the core codebase by external parties."[201] While there is nothing to discourage contributions to the core source code base in a regular BSD-style license – and, like the regular BSD-style license but unlike the GNU GPL, there is no condition to force a licensee to contribute improvements back to the community – the Apache License, arguably, has provisions that can serve as incentives for a programmer/licensee to send derivative works back to the Apache

---

[196] *See 4.4BSD License*, *supra* note 101.

[197] *See Apache License*, *supra* note 107.

[198] *See, e.g.*, Free Software Foundation, *The BSD License Problem* (last visited Feb. 1, 1999) <http://www.gnu.org/philosophy/bsd.html>; Perens, *The Open Source Definition*, *supra* note 43, at 183.

[199] *See* Debian GNU/Linux Developers, *Sample Modified BSD-style License* (last visited Jan. 12, 1999) <http://www.debian.org/misc/modified.bsd.license> [hereinafter *Modified BSD-style License*]. It is worth noting that a BSD-style license with the advertising clause removed is essentially the same as another open source license not discussed in this paper – the 'X license,' *see* Perens, *The Open Source Definition*, *supra* note 43, at 183.

[200] *See Apache License*, *supra* note 107. For background information on Apache, *see supra* Part II.B.2.

[201] Halloween I, *supra* note 17; *see* McHugh, *supra* note 5, at 95, 100; *see generally* Apache Group, *About*

development community.

First, the Apache License makes it clear that the licensed program was made possible by contributions of code back into the open source development community: "This software consists of voluntary contributions made by many individuals on behalf of the Apache Group ..."[202] Through this statement, the Apache Group is using its license to put potential contributors on notice that useful modifications to the Apache Web Server code are welcomed.

Second, the prohibitions against using the names "Apache Server" and "Apache Group" to promote any derivative works without the prior written permission of the Apache Group,[203] may appear to be, at most, an attempt to protect the authors' artistic integrity and authorship over the Apache Web Server rather than an attempt at preventing someone from privatizing modifications to the Apache source code without contributing back to the open source community. However, the inability for a would be – in Richard Stallman-esque language – 'software hoarder' to use the highly valuable name 'Apache' for any of his or her derivative works, would create some incentives for the licensee to at least consider cooperating with the Apache Group.

Finally, clause 6 – which states that "[r]edistributions of *any* form *whatsoever* must retain the following acknowledgment: 'This product includes software developed by the Apache Group for use in the Apache HTTP server project (http://www.apache.org/)"[204] – means that, any licensee that contemplates 'closing' his or her modifications to the open Apache code and wishes to redistribute the closed source modifications, must acknowledge that the closed source redistribution is derived from the work of the Apache Group. Why? Because the language of

---

the Apache HTTP Server Project, *supra* note 20.

[202] *Apache License*, *supra* note 107, at Coda.

[203] *See id.* ¶ 4.

clause 6 – which applies regardless of what form the redistribution takes – arguably includes derivative works, the licensee in our hypothetical scenario would be obliged (assuming open source licenses are legally enforceable) to acknowledge the Apache Group.

Admittedly, none of the conditions of the Apache License can be seen as a strong barrier to proprietizing modifications to open code. However, the terms of the Apache License may have created enough of an incentive for IBM, a company that is well aware of the reaches and the limits of intellectual property law, to agree to contribute improvements to the core Apache source code in exchange for using the Apache Web Server in one of its products.[205]

***

Even taking into account Apache's refinements to the BSD-style license, the terms of the BSD-style licenses – including the Apache License – fall well short of the Free Software Foundation's attempts, through the GNU GPL, to erect and maintain strict walls of separation between the closed source and open source development models. Assuming, for now, that open source licenses of any kind are legally enforceable, no one can seriously argue that the BSD-style licenses – even the Apache License – place substantial legal barriers to any would be 'software hoarder.'[206] The BSD-style licenses, therefore, raises the question of how the open source development model can be sustained under the BSD's lenient conditions. This question will be dealt with in Part V. of this paper.

4. The Netscape Public License and the Mozilla Public License

When Netscape made their monumental decision to publicly release the source code of

---

[204] *Id.* ¶ 6. (emphasis added).

[205] *See supra* notes 21-22 and accompanying text.

[206] *See* Seltzer, *Apache*, *supra* note 189, at 178; Seltzer, *License*, *supra* note 38, at 171.

their web browser,[207] Netscape created the Netscape Public License[208] [hereinafter NPL] and the Mozilla Public License[209] [hereinafter MozPL] in order to balance the needs of a commercial software producer with the call by the open source community to adopt a licensing approach that stays true to the definition of open source software.[210] Both the NPL and the MozPL are of interest here because they represent the first attempts by a for-profit software producer to create and apply a 'commercial open source license' to some of their programs.

The NPL and the MozPL "require that any and all changes to code covered by the [NPL or the MozPL] must be made publicly available."[211] In other words, like the GNU GPL, the NPL and the MozPL adopts stricter terms when dealing with derivative works in order to encourage developers to "'give back' to the common code base."[212] Netscape – through its open source

---

[207] For background information, *see supra* Parts I. & II.B.3.; Perens, *The Open Source Definition*, *supra* note 43, at 184; Hamerly et al., *Story of Mozilla*, *supra* note 113, at 197-206.

[208] *See* Mozilla.org, *Netscape Public License, Version 1.0* (last visited Mar. 16, 1999) <http://www.mozilla.org/NPL/NPL-1.0.html> [hereinafter *Netscape PL 1.0*]. Recently, Netscape and Mozilla.org – Netscape's open source projects development division – have released a draft of a revised version of the Netscape Public License – version 1.0M, *see* Mozilla.org, *Netscape Public License, Version 1.0M draft* (last visited Mar. 16, 1999) <http://www.mozilla.org/NPL/NPL-1.0M.html>. As of the time of writing, Mozilla.org is taking comments on the revised NPL. For more information on the draft of the modified NPL, *see* Mozilla.org, *NPL Version 1.0M FAQ* (last visited Mar. 16, 1999) <http://www.mozilla.org/NPL/NPL-1.0M-FAQ.html>. Regardless of whether or not the modifications are formally adopted, the basic analysis of this paper will still apply.

[209] *See* Mozilla.org, *Mozilla Public License, Version 1.0* (last visited Mar. 16, 1999) <http://www.mozilla.org/NPL/MPL-1.0.html> [hereinafter *Mozilla PL 1.0*]. Note: if the changes to the NPL go through, *see supra* note 208, then these changes will also apply to the MozPL, *see* Mozilla.org, *NPL Version 1.0M FAQ*, *supra* note 208, ¶ 2. **Note: The changes have taken place, *see infra* note 226.** Again, regardless of whether or not the modifications are formally adopted, the basic analysis of this paper will still apply.

[210] *See Netscape Licensing FAQ*, *supra* note 114; Perens, *The Open Source Definition*, *supra* note 43, at 184.

[211] *Netscape License FAQ*, *supra* note 114, ¶ 9; *see Netscape PL 1.0*, *supra* note 208, § 3.2; *Mozilla PL 1.0*, *supra* note 209, § 3.2.

[212] *Netscape License FAQ*, *supra* note 114, ¶ 3; *see id.* ¶¶ 6, 9. It should be noted that Bruce Perens, the author of the Open Source Definition, ***somewhat*** inaccurately stated that the NPL and the MozPL allow a licensee "to take modifications private," Perens, *The Open Source Definition*, *supra* note 43, at 184; *see also id.* at 185. Both Bruce Perens and this author agree that it is relatively easy for someone to have closed-source add-ons to licensed code by simply adding a function call to the licensed code, *see* E-mails from Bruce Perens, Author of the Open Source Definition and the Debian Free Software Guidelines (Apr. 27, 1999) [hereinafter Perens, E-mails] (on file with the author), *and* E-mails from Steve Lee <grokopen@email.com> to Bruce Perens (Apr. 27, 1999) [hereinafter Lee, E-mails to Perens].

development arm, Mozilla.org[213] – decided to reject adopting terms similar to the more lenient terms of the BSD-style License – which would allow modifications to the source code base to become 'closed'[214] – because such terms would "not go far enough to ensure that developers will return their modifications to the Communicator source code to the community."[215]

Unlike the GNU GPL, however, neither the NPL nor the MozPL prohibit the integration of closed and open source codes: "[The licenses] allow [the licensee] to combine covered code with other code to create a larger work without requiring that other code be covered by the [NPL or the MozPL]."[216] In other words, so long as the requirements of the license(s) are met for the original "Covered Code,"[217] the licensee may "create a Larger Work by combining Covered Code with other code not governed by ... [either the NPL or the MozPL] and distribute the Larger Work as a single product."[218] Netscape did not utilize a GNU GPL-like limitation in this case, because of both commercial considerations – including respecting third party intellectual property rights – as well as the desire to encourage development of its opened source code, especially by commercial developers that may want to retain some interest in their work product.[219]

However, both of us agree that – legally speaking – the NPL and the MozPL would force a licensee to keep modifications open **if** those modifications were derived from the licensed code, *see* Perens, E-mails, *supra*, *and* Lee, E-mails to Perens, *supra*.

[213] *See supra* note 112 and accompanying text.

[214] *See* discussion *supra* Part III.C.3.; *Netscape License FAQ*, *supra* note 114, ¶ 5.

[215] *Netscape License FAQ*, *supra* note 114, ¶ 12.

[216] *Id.* ¶ 9; *see Netscape PL 1.0*, *supra* note 208, § 3.7; *Mozilla PL 1.0*, *supra* note 209, § 3.7.

[217] "Covered Code," in the Netscape licenses, "means the Original Code or Modifications or the combination of the Original Code and Modifications, in each case including portions thereof," *Netscape PL 1.0*, *supra* note 208, § 1.3, *and Mozilla PL 1.0*, *supra* note 209, § 1.3.

[218] *Netscape PL 1.0*, *supra* note 208, § 3.7, *and Mozilla PL 1.0*, *supra* note 209, § 3.7; *see generally* Hamerly et al., *Story of Mozilla*, *supra* note 113, at 200-203.

[219] *See Netscape License FAQ*, *supra* note 114, ¶ 10; *see also id.* ¶¶ 3, 6, 9.

To summarize, through its two licenses, Netscape is trying to strike a balance between the two 'extremes' in open source licensing as represented by the GNU GPL and the BSD-style licenses: On one hand, like the GNU GPL, the Netscape licenses require that derivative works remain open source and be contributed back into the community in order to ensure everyone concerned will reap the benefits of open source development; On the other hand, like the BSD-style licenses, the Netscape licenses allow developers – whether commercial or non-commercial – to be able to combine closed and open source codes in order to allow developers to participate in open source development without having to keep out closed source software that may be helpful to the development effort.[220]

If the GNU GPL reflects a rejection of the closed source world by its creator, Richard Stallman, and some of the adherents of copylefting, and if the BSD-style licenses reflect an optimistic belief that strict legal conditions are not necessary to the advancement of open source development, then both the NPL and the MozPL reflect a compromise born out of commercial considerations. Reflecting their commercial origins, the Netscape licenses do not represent an absolute rejection of the closed source model of development. However, the NPL and the MozPL seem to reject the optimistic idea that non-legal norms alone can be relied on to sustain open source development.[221]

These sorts of issues will be dealt with further in Parts III.D. & IV.[] of this paper, *infra*.

a. The Distinction Between the NPL and the MozPL

Simply stated, the NPL applies when someone (besides Netscape or a party exempted by Netscape) uses the original NPL licensed source code base and/or any modifications to that code

[220] *See id.* ¶¶ 3, 4, 9, 10.

[221] For more on non-legal norms as an enforcement mechanism for open source licenses, *see infra* Part V.

base, in whole or in part.[222] The MozPL will apply when the licensee does not use the original source code base nor any modifications that base.[223] To illustrate, if someone is modifying the original source code base, then the NPL will apply.[224] If someone is only adding a new file that does not contain any of the original source code base or modifications to that base, the MozPL can be applied – even if this new file is called or referenced by a NPL licensed program.[225]

It is important to note that – if, for example, modifications are being made to Netscape/Mozilla released source code – the NPL ***must*** apply.[226] However, while Netscape highly recommends the use of the MozPL, the decision of what license to apply is up to the contributor of the new code.[227]

Another way in which the NPL differs from the MozPL is due to the fact that the NPL "contains special privileges that apply to Netscape and nobody else."[228] Amendment IV. of the NPL, allows Netscape to 'close' up the source code that it had originally released – along with

---

[222] *See Netscape License FAQ*, *supra* note 114, ¶¶ 4, 13.; *Netscape PL 1.0*, *supra* note 208, at Amendments.

[223] *See Netscape License FAQ*, *supra* note 114, ¶ 4.

[224] *See id.* ¶ 13.

[225] *See id.* ¶ 4.

[226] *See id.* ¶ 13; *Netscape PL 1.0*, *supra* note 208, at Amendments, § I.

[227] *See Netscape License FAQ*, *supra* note 114, ¶¶ 14, 15. It should be noted that the while almost any license that is legally compatible with the NPL can be chosen – up to and including a license that keeps the new code 'closed' – this option only applies "for new code and not a modification[] to existing [NPL covered] code," *id.* ¶ 14.

Aside: Netscape/Mozilla – with regard to the unmodified versions of the NPL and the MozPL – considers the GNU GPL to be legally incompatible with the NPL, because (in the opinion of Netscape) the GNU "GPL is by design incompatible with all other licenses, since it prohibits the addition of any restrictions or further permissions to its boundaries," Hamerly, *Story of Mozilla*, *supra* note __, at 202; *see Netscape License FAQ*, *supra* note 114, ¶ 14. However, the proposed modifications to both the NPL and the MozPL – which would allow code to be released under multiple licenses, including the GPL – may eliminate any trace of legal incompatibility between the GNU GPL and the NPL/MozPL, *see* Mozilla.org, *NPL Version 1.0M FAQ*, *supra* note __, ¶¶ 11-13. The new NPL and MozPL – Version 1.1 – has just been released at the time of writing, *see* Mozilla.org, *Mozilla Public License – Version 1.1*, (last visited Apr. 27, 1999) <http://www.mozilla.org/NPL/NPL-1_1Final.html>.

[228] Perens, *The Open Source Definition*, *supra* note 43, at 184; *see Netscape License FAQ*, *supra* note 114, ¶¶ 13, 20; *Netscape PL 1.0*, *supra* note 208, at Amendments, §§ III. (retaining trademark rights), IV., V.

modifications to that source code base, even if they are made by non-Netscape contributors – if it is necessary due to contractual obligations related to third party code that has been licensed to Netscape.[229] Amendment V., among other things, allows Netscape to use its opened source code in other additional products "without such additional products becoming subject to the terms of [the NPL]."[230]

These conditions – born out of commercial necessity – might keep the NPL, but not the MozPL, outside of the realm of open source licensing.[231] However, Netscape argues that the NPL meets the relevant guidelines that would qualify it as an open source license.[232] Additionally,

---

[229] *See Netscape PL 1.0*, *supra* note 208, at Amendments, § IV.

[230] *Id.* at Amendments, § V.2.

[231] *See OSD (v.1.0)*, *supra* note 44, ¶ 10 (the MozPL – referred to as the 'MPL' – is included as an example of an open source license, but the NPL is not); *cf. Netscape License FAQ*, *supra* note 114, ¶ 4 (acknowledging that the NPL's grant of special privileges to Netscape made the NPL controversial); Hamerly et al., *Story of Mozilla*, *supra* note 113, at 201-202 (describing the controversy surrounding the NPL's provisions).

[232] *See* Mozilla.org, *Netscape Public License Homepage* (last visited Mar. 17, 1999)

members of the open source community – while perhaps not entirely happy with the NPL – seem to accept the 'openness' of the NPL.[233]

5. A Quick Comparison of the Different Licenses

The following table summarizes the key differences (and similarities) between the major open source licenses analyzed in this section:[234]

Open Source Licenses	Works ***derived*** from the original open source code base must ***also*** remain 'open'/freely available. I.e., modifications cannot become 'closed source'/privatized.	Licensed open source software ***cannot*** be integrated or combined with closed source software (at a fundamental level beyond merely residing on the same medium of

<http://www.mozilla.org/NPL/> (stating that it meets the Open Source Definition and the Debian Free Software Guidelines).

[233] *See* Richard Stallman, *Apple's Non-Free Source License*, LINUX TODAY, (Mar. 20, 1999) <http://linuxtoday.com/stories/4263.html> (Stallman calls the NPL a "free software license" in order to contrast it with Apple's attempts at open source licensing, which he criticizes); *see also* Perens, *The Open Source Definition*, *supra* note 43, at 184 (Bruce Perens and Eric S. Raymond, both prominent members of the open source community, participated in the development of both the NPL and the MozPL). The MozPL (which is clearly open source) was – in part – a response by the Mozilla team to appease the open source community, *see* Hamerly, *Story of Mozilla*, *supra* note __, at 201-202.

[234] The table here is similar to the tables found in Bruce Perens' OPEN SOURCES piece (however, the table in this paper uses a more accurate assessment of the Netscape licenses, *see supra* note 212) and in the first Halloween Memo, *see* Halloween I, *supra* note 17; Perens, *The Open Source Definition*, *supra* note 43, at 185.

		distribution)
GNU GPL	Yes	Yes
Artistic License	No (under most circumstances)	No (under most circumstances)
BSD-style licenses (including Apache)	No	No
NPL	Yes	No
MozPL	Yes	No

It should also be noted that all of the open source licenses examined here have a common feature beyond meeting the baseline requirements necessary for an open source license:[235] "[T]hey each disclaim all warranties."[236] The justification for including the disclaimer of warranties is the following: "If [open source software] authors lose the right to disclaim all warranties and find themselves sued ... they'll stop contributing [open source] software to the world."[237] This is one of the few things that open source licenses have in common with closed source licenses.[238]

D. Battle of the Licenses?

While all of these licenses maintain the openness of source code and the ability to freely redistribute the licensed software, the open source licenses analyzed here are substantially

[235] *See supra* Part III.B.

[236] Perens, *The Open Source Definition*, *supra* note 43, at 181; *see* FSF, *GNU GPL*, *supra* note 58, §§ 11,12; Perl, *Artistic License*, *supra* note 92, ¶ 10; *Sample BSD-style License*, *supra* note 101, at Coda; *Netscape PL 1.0*, *supra* note 208, §§ 7, 9, 12; *Mozilla PL 1.0*, *supra* note 209, §§ 7, 9, 12.

[237] Perens, *The Open Source Definition*, *supra* note 43, at 181. Another justification is that most open source developers often give away their work at no cost, therefore, this sort of condition can be seen as being reasonable – especially when taking into account the likelihood that the developer does not have deep enough pockets to pay for damages or to pay for liability insurance, *see id.*

[238] *See, e.g.*, Neukom & Gomulkiewicz, *supra* note 121, at 778. Although it should be noted that most closed source software licenses do not disclaim all potential liability, but simply seek to limit the warranties and the potential liabilities associated with their product, *see id.*; Aldus License, *supra* note 143, at 792.

different from each other. These differences often reflect the ideological tensions within the open source community on how best to further the goals of the open source movement.[239]

On one end of the spectrum are the passionate and idealistic defenders of the philosophy of 'free software' – the idea that intellectual property protections are barriers to a utopian vision of communitarian software development and that only the 'freeing' (or the 'opening') of source code can lead to freedom for both software users and developers.[240] This faction of the open source movement is made up of Richard Stallman – the leader of the Free Software Foundation – and Bruce Perens[241] – the author of the 'Open Source Definition' and a co-founder of the Open Source Initiative.

On the other end of the spectrum are those who take a more pragmatic and utilitarian approach to the open source movement. Whereas the 'free software' advocates want to promote open source software as being part of a larger quasi-political philosophy, open source advocates that do not subscribe to the Stallman-esque vision of digital utopia put more emphasis on the

---

[239] *See, e.g.*, Jonas Oberg, *On the Subject of RMS*, SLASHDOT, (Mar. 29, 1999) <http://slashdot.org/features/99/03/28/2119239.shtml> [hereinafter Oberg, *RMS*] (chief webmaster of the GNU Project stating that the differences between the licenses are based on philosophical disagreements); *See generally* Rebecca L. Eisenberg, *Internal Rift Among Linux Advocates*, S.F. EXAMINER, Mar. 7, 1999 [Web: http://eXaminer.com/990307/0307eisenberg.shtml] [hereinafter Eisenberg, *Rift*]; Thomas Scoville, *Whence the Source: Untangling the Open Source/Free Software Debate*, O'REILLY OPEN SOURCE, (last visited Mar. 8, 1999) <http://opensource.oreilly.com/news/scoville_0399.html> [hereinafter Scoville, *Open/Free Debate*]; Stephen Shankland, *"Open Source" Infighting Grows*, CNET NEWS.COM, (Feb. 19, 1999) <http://www.news.com/News/Item/Textonly/0,25,32644,00.html> [hereinafter Shankland, *Infighting*].

[240] *See* discussion *supra* Part II.B.1.; Eisenberg, *Rift*, *supra* note 239 [ ] (GNU "is a philosophy"; The FSF is a "political philosophy about how society can best cultivate and nurture innovation and progress.").

[241] *See, e.g.*, Bruce Perens, *It's Time to Talk about Free Software Again*, SLASHDOT, (Feb. 18, 1999) <http://slashdot.org/articles/99/02/18/0927202.shtml> [hereinafter Perens, *Free Software*]. Bruce Perens quit the Open Source Initiative – which he co-founded with Eric S. Raymond – because of his belief that the Open Source Initiative was insufficiently committed to the 'free software' philosophy of Richard Stallman, *see id.*; Andrew Leonard, *Name that Movement: Open Source or Free Software?*, SALON, (Feb. 17, 1999) <http://www.salonmagazine.com/21st/log/1999/02/17log.html> [hereinafter Leonard, *Open or Free*]; Shankland, *Infighting*, *supra* note 239.

qualitative virtues (e.g., Linux not crashing as often as Windows) of open source software.[242] This faction of the open source community includes Eric S. Raymond[243] – a co-founder of the Open Source Initiative, Tim O'Reilly[244] – a publisher of popular books on open source software, and Larry Wall[245] – the inventor of Perl.

The more utilitarian wing of the open source movement disagree with their communitarian brethren in other ways as well. Compared to the 'free software' advocates,[246] the utilitarian faction tends to be more welcoming of the commercialization of the open source movement.[247] More importantly – while the 'free software' advocates tend to despise intellectual property rights[248] – the utilitarian faction is often indifferent to the question of whether intellectual property

---

[242] *See, e.g.*, Andrew Leonard, *Let My Software Go!*, SALON, (Apr. 14, 1998) <http://www.salonmagazine.com/21st/feature/1998/04/cov_14feature.html> [hereinafter Leonard, *ESR*] (Eric S. Raymond – who has often disagreed with both Stallman and Perens – emphasizes the quality of open source software from an engineering and users' perspectives); Eric S. Raymond, *The Evangelists*, RED HERRING, Feb. 1999, at 48, 48-49 [hereinafter Raymond, *Evangelists*] (Eric S. Raymond arguing that higher quality and lower prices associated with open source software is what makes open source development superior to closed source development).

[243] *See supra* note 242; Scoville, *Open/Free Debate*, *supra* note 239 (comparing Raymond's "utilitarian" approach with Stallman's more dogmatic approach to open source advocacy).

[244] *See* Tim O'Reilly, *Hardware, Software, and Infoware*, *in* OPEN SOURCES, *supra* note 62, 189, 189-96 [hereinafter O'Reilly, *Infoware*] (reflecting a more utilitarian approach – which includes being open to the commercialization of open source software); Tim O'Reilly, *Ask Tim [Re: Free vs. Open]*, O'REILLY (last visited Mar. 23, 1999) <http://www.oreilly.com/ask_tim/> [hereinafter O'Reilly, *Free vs. Open*] (explaining his disagreements with Stallman and the free software movement).

[245] *See* Lash, *Source Code for the Masses*, *supra* note 13 (stating that Larry Wall is hostile to some of the ideology of the Free Software Foundation).

[246] *See, e.g.*, *id.* ("Stallman abhors the 'proprietary jungle' fostered by typical business practices.").

[247] *See* Eisenberg, *Rift*, *supra* note 239 (Raymond, and his faction, adopted the term 'open source,' in part, to prevent software companies from being "put off" by the 'free software' label); Raymond, *Evangelists*, *supra* note __, at 48-49 (touting the virtues of open source software to the business community); Eric S. Raymond, *ESR on O'Reilly Summit*, SLASHDOT, (Mar. 11, 1999) <http://slashdot.org/features/99/03/11/1352249.shtml> (downplaying any dissension between business people and the open source community); *see also* Itoi, *Freeware*, *supra*, note 116 at 46, 55-56 (stating that Tim O'Reilly's publishing business – O'Reilly & Assoc. – profits from selling books that cover open source software).

[248] *See* discussion *supra* Part II.B.1.; *see also* Stallman, *Reevaluating Copyright*, *supra* note 83.

laws are justified for digital intellectual property.[249]

The choice that an open source developer has to make between the various open source licenses is often affected by the ideological debates within the open source community.[250] For example, the GNU GPL will often be utilized by open source developers who are more sympathetic to 'free software' advocates like Stallman and Perens.[251] The proponents of the GNU GPL tend to want to keep the line between 'open source' and 'closed source' software clear and unambiguous.[252] Therefore, the proponents of copylefting find the tougher conditions of the GNU GPL to be closer in spirit to the goals of the 'free software' movement than the more lenient terms of the other open source licenses.[253]

The more utilitarian and commercially-friendly wing of the open source movement, however, is more open to the use of open source licenses – like the BSD-style licenses – that are

---

[249] *See* Leonard, *ESR*, *supra* note 242 (Raymond stating that his advocacy of the open source movement is "180 degrees removed from any ideology about whether intellectual property rights are good or not," and goes on to state that he does not really care about that question); O'Reilly, *Free vs. Open*, *supra* note 244 (stating that he has no problems with those who want to use licenses that retain intellectual property protection over source code).

[250] *See* Laird & Soraiz, *supra* note 181 ("Along with ... increasingly subtle technical distinctions, several organizations decide between Linux [which is copylefted] and FreeBSD [which is under the BSD license] based on their different licensing models."); *Netscape License FAQ*, *supra* note 114; Oberg, *RMS*, *supra* note 239 (the chief webmaster of the GNU Project stating that he does not like the BSD or the Artistic licenses because of "philosophical issues"); O'Reilly, *Free vs. Open*, *supra* note 244; Perens, *The Open Source Definition*, *supra* note 43, at 182.

[251] *See* Seltzer, *License*, *supra* note 38, at 171 ("The goal of ... [the GNU GPL] is to make the program and the work built on it perpetually 'free.'"); Leander Kahney, *Linux's Forgotten Man*, WIRED NEWS, (Mar. 5, 1999) <http://www.wired.com/news/technology/story/18291.html> ("[The GNU GPL] is the foundation of the free-software movement."); Debian GNU/Linux Developers, *Comparison of Software Licenses*, (last visited Jan. 12, 1999) <http://www.debian.org/intro/license_disc> [hereinafter Debian, *Comparison*] ("Most people who release software using the GPL do not consider [its] restrictions bad, because it allows others to use and improve the software while it prevents (for all practical purposes) others from making money off of their hard work without [the copyright holder's] permission.").

[252] *See, e.g.*, Perens, *The Open Source Definition*, *supra* note 43, at 175-76, 186 (warning of the dangers of "partially-free" programs and arguing that "almost-free-enough" programs could kill off some categories of open source development).

[253] *See, e.g.*, Seltzer, *License*, *supra* note 38, at 171 ("Purists object to the BSD license ..."); Oberg, *RMS*, *supra* note 239 ("I dislike ... [the BSD and the Artistic] licenses because I do not agree with the philosophical issues ... [behind those licenses].").

less stringent *vis a vis* the GNU GPL.[254] The non-'free software' faction of the open source community tends to believe that an open source developer should have the discretion to put his software under any license that the developer feels is appropriate, even if that licensing scheme blurs the line between open and closed code.[255] In fact, some non-'free software' open source developers and advocates object to the use of the GNU GPL because of the ideology it embodies and because of its restrictions against proprietizing derivative code – a condition that tends to hinder the commercialization of copylefted software.[256]

Recently, established commercial software producers – notably Netscape – have adopted the open source development model for some of their products. Oddly enough, in one respect, the established commercial software producers – when creating their open source licensing schemes – agree with the 'free software' advocates: Like the GNU GPL, the commercial open source licenses, like the NPL and the MozPL, prevent a licensee from building proprietary derivatives off of the licensed source code base.[257]

Why would established commercial software producers agree with software communitarians like Richard Stallman when it comes to keeping derivative code 'open'? There are two main reasons for this ironic meeting of the minds: 1.) Both the GNU GPL and the

---

[254] *See, e.g.*, O'Reilly, *Free v. Open*, *supra* note 244.

[255] *See id.*

[256] *See* Laird & Soraiz, *supra* note 181 ("[S]everal organizations resist Copylefted software because of the constraints it imposes on commercialization of their own work."); Luh, *supra* note 191 ("'By using the GPL, [the users of Linux] inherited an agenda, which was Richard Stallman's agenda," [Jordan] Hubbard [the CEO of FreeBSD, Inc.] said ... 'That's a pretty strong agenda, unfortunately, and for a lot of businesses, it's just not acceptable.'"); Perens, *The Open Source Definition*, *supra* note 43, at 182 ("The political rhetoric in the GPL puts some people off. Some of them have chosen a less appropriate license for their software simply because they eschew Richard Stallman's ideas and don't want to see them repeated in their own software packages.").

[257] *See supra* Part III.C.

commercial open source licenses attempt to legally prevent the 'forking' of code.[258] In other words, both the free software purists and the commercial open source software producers are concerned that more lenient licenses, like the BSD-style licenses, "can let a program become fragmented as developers spin off their own proprietary versions."[259] 2.) 'Free software' advocates – for ideological reasons – and commercial software producers – for business reasons[260] – argue that it is important to return modifications to the open source community in order to maintain the viability of open source development.[261]

Those who use BSD-style licenses, as well as those in the open source community that take a more utilitarian approach to open source development, tend to respond by arguing that reliance on extralegal forces – such as the cultural norms of the open source community[262] – are sufficient to prevent code forking and to ensure that enough derivative code goes back into the

[258] For how the restrictions on proprietizing derivative code – as found in both the GNU GPL and the NPL/MozPL – prevents the 'forking' (or the fragmenting of software into different and, at times, incompatible versions), *see* Halloween I, *supra* note 17, at 'Software Licensing Taxonomy' (crediting the GPL's requirement that all derivative works must be copylefted with preventing the 'forking' of code).

[259] Seltzer, *License*, *supra* note 38, at 171; *see* Halloween I, *supra* note 17, at 'Code Forking' (placing the blame for the forking of BSD Unix to the BSD-style license's lack of GPL-style restrictions on derivative code).

[260] This is true if one takes the view that Netscape and other commercial software producers are utilizing the open source development model as a way of harnessing community creativity in order to boost product development efforts, *see* CUSUMANO & YOFFIE, *supra* note 11, at 138-40, 321; *see also* discussion *infra* Part VI.

[261] *See Netscape License FAQ*, *supra* note 114, ¶¶ 6, 12; Free Software Foundation, *Categories of Free and Non-Free Software*, (last visited Aug. 5, 1998) <http://www.gnu.org/philosophy/categories.html> [hereinafter FSF, *Free and Non-Free*], at 'Non-copylefted free software' (arguing that licenses that do not require that derivative works be open as well has allowed some open source development efforts to become at least partially closed source); *cf.* Stallman, *Library*, *supra* note 158 (explaining how the GPL's strict conditions caused at least one software development effort to become open rather than closed and how this sort of scenario gives the open source community "a real boost").

[262] This issue will be discussed in Part V, *infra*. For background on the cultural norms of the open source community, *see generally* Raymond, *The Cathedral and the Bazaar*, *supra* note 112; Peter Wayner, *Glory Among the Geeks*, SALON, (Jan. 28, 1999) <http://www.salonmagazine.com/21st/feature/1999/01/28feature.html> [hereinafter Wayner, *Glory*] (both *The Cathedral and the Bazaar* and Wayner's article point out that – within the open source programming community – contributing code back into recognized open source development projects, like Linux, is considered to be a tremendous honor).

community (i.e., stays 'open') to sustain open source development efforts.[263]

Therefore, the question of which open source licensing scheme is better boils down to one of 'means to an end.' In other words, the central issue is finding the best means to achieve the ultimate goal of sustaining and advancing the open source development model. No one in the open source development community seriously disagree with each other about what the end game is: Promoting open source development as a viable alternative to the more typical closed source development model.[264] However, there are serious disagreements about what set of ideologies should drive, and what methods should be used, in achieving the shared goals of both the utilitarian and the communitarian open source advocates.

IV. Are Open Source Licenses Legally Enforceable?

Although the question of legal enforceability has never been tested through litigation,[265] this paper will argue that existing and proposed principles of copyright and contract law support the idea that open source licenses, in general, are legally enforceable. This paper will also argue that the terms and the conditions that are specific to each of the various types of licenses, regardless of whether or not the terms and conditions are strict or lenient, are enforceable as well.

A. Preliminary Copyright Issues

1. Open Source Licensing and its Effects on the Exclusive Rights Under Copyright Law

[263] *See* Laird & Soraiz, *supra* note 181 (stating that – despite going 'commercial' – there has, historically, been support for maintaining FreeBSD as a viable open source operating system among programmers; e.g., programmers still contribute to FreeBSD even though they are not obligated to do so); Seltzer, *License*, *supra* note 38, at 171 (although developers can create their own proprietary versions with some slight modifications to the code base, "[i]n practice, many developers opt to give their work back to the community anyway.").

[264] *See, e.g.*, O'Reilly, *Free vs. Open*, *supra* note 244 (Tim O'Reilly believes that he has a lot more in common with Richard Stallman then it may appear to outsiders); Scoville, *Open/Free Debate*, *supra* note 239 (both Stallman and Eric Raymond would agree that "distributing a computer programs's source code ... is a good thing.").

[265] *Cf.* Heffan, *supra* note 9, at 1509 (pointing out that the legal enforceability of the GNU GPL has never been litigated). As of the date of writing, a WESTLAW search using the search terms "OPEN SOURCE SOFTWARE LICENSE," "OPEN SOURCE LICENSE," and "GENERAL PUBLIC LICENSE," in the ALLCASES database (which contains all cases, state and federal, after 1944) yielded no results.

Copyright holders have five exclusive rights under copyright law: 1.) the right to reproduce, 2.) the right to prepare derivative works based upon the original copyrighted work, 3.) the right to distribute, 4.) the right to publicly perform, and 5.) the right to publicly display.[266] All open source licenses – even the more lenient BSD and Artistic Licenses – surrender most if not all of these otherwise exclusive rights to the licensees. All open source licenses force the licensor to give up the derivative works right to licensees by allowing licensees to create modifications and derivative works off of the licensed source code base.[267] Open source licensors are also required to allow virtually unfettered redistribution of the open source software by the licensees.[268] It seems reasonable to assume that the reproduction right, the performance right, and the display right – to the extent that the performance and the display rights are relevant to open source software – are also diluted by the requirements of open source licensing.

It is worth noting that on both a practical level – since the exclusive rights of the copyright holder are being diluted or given away – and on a philosophical level – since these licenses force copyright holders to open their source code, which is "the *core* of any software product"[269] – open source licenses serve to weaken intellectual property rights. This feature of open source licenses is in effect regardless of whether or not the licensor approves or disapproves of intellectual property rights for software. In other words, regardless of whether the open source developer falls under the 'utilitarian' camp or the 'free software' camp, those who license their software under an open source license, in affect, weaken whatever intellectual property rights they

[266] *See* 17 U.S.C. § 106 (1)-(6).

[267] *See supra* Part III.B.§3.

[268] *See supra* Part III.B.§4.

[269] CUSUMANO & YOFFIE, *supra* note 11, at 139.

may have had over their work product.

The typical closed source software license, on the other hand, attempts to ensure that almost all of the exclusive intellectual property rights granted under copyright law remain with the licensor.[270] As noted earlier in this paper, closed source licenses usually do not allow licensees to create derivative works (unless it is necessary for archival purposes) and severely restricts the ability of licensees to redistribute the licensed software.[271]

2. Should Open Source Developers Copyright Their Software?

It is important to note that open source software – since it is always distributed under some sort of copyright license – is *not* the same as public domain software.[272] Public domain software has virtually no limitations associated with it.[273] For example, if software is in the public domain, then that means that the software is not protected under copyright law[274] and can be copyrighted and then licensed by someone other than the original creator – so long as the work in the public domain is *slightly* modified from the original (i.e., a derivative work is created).[275]

At this point in the paper, one could ask a rather obvious question: Why should open source developers copyright their software in the first place? If open source developers want to use licenses in order to allow their source code to be publicly available and easily redistributed –

---

[270] *See generally* Neukom & Gomulkiewicz, *supra* note 121, at 775-92; *supra* Part III.A. & B.

[271] *See, e.g.*, Aldus License, *supra* note 143, at 791 (prohibiting the creation of derivative works (unless for archival purposes) and severely restricting the redistribution of the licensed software).

[272] *See* Perens, *The Open Source Definition*, *supra* note 43, at 180.

[273] *See id.*

[274] *See* BLACK'S LAW DICTIONARY 1229 (6th ed. 1990) (defining 'public domain'); 1 MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT § 2.03[G] (1998) [hereinafter NIMMER ON COPYRIGHT] (U.S. copyright law does not protect works in the public domain).

[275] *See* 1 NIMMER ON COPYRIGHT §§ 3.01 & 3.07[C].

and open source licenses effectively weaken intellectual property protections – then why not allow open source software to simply enter the 'public domain'?

Open source developers chose not to release their work as public domain because of the total absence of any restrictions on the end user. While open source licenses may be considerably less restrictive than closed source licenses, even open source licenses have restrictions.[276] For 'free software' purists, there is an understanding that the use of copyright law – "the tool[] of the software hoarders"[277] – is a necessary 'evil' in order to place restrictions against someone from establishing proprietary rights over work that was intended to be non-proprietary.[278] Even licenses that are arguably the closest to the public domain in terms of the lack of restrictions – such as the BSD-style licenses – need some restrictions in order to ensure that credit is given to the authors of the open source program.

Legally speaking, the restrictions typically placed on open source software – including the all important restriction that the original source code base remain open – can only be ensured by copyrighting the software and then distributing it under some type of open source license. If open source software lacked any sort of intellectual property protections – as it would if it was in the public domain – then there would be little or no *legal* way to prevent the 'closing' of source code.

This situation highlights a key irony of open source software licensing: The open source movement depends on a strong and enforceable system of intellectual property protection – including copyright law – in order to weaken intellectual property rights.[279] We will see this irony

[276] *See* Debian, *Free/Open Software*, *supra* note 65 (stating that open source software is "not totally free of constraints" *vis a vis* the public domain).

[277] Mann, *supra* note 14, at 38.

[278] *See* discussion *supra* Part II.B.1.; Heffan, *supra* note 9, at 1489, 1511, 1513.

[279] Ira V. Heffan suggests this irony in his analysis of the GNU GPL, *see* Heffan, *supra* note 9, at 1505-8 (pointing out that Richard Stallman – "opposes the application of intellectual property and contractual rules to software

again when we examine the contractual element of open source licensing.

3. Derivative Works and the Distribution Right Under Copyright Law

The loosening up of the otherwise exclusive derivative works right is an important part of open source licensing. As noted previously, every open source license allows the licensee to create derivative works off of the original source code base.

The derivative works right[280] – sometimes called the 'adaptation right'[281] – can best be defined (albeit somewhat tautologically) in the following manner: A derivative work must have been derived from the pre-existing work to such an extent that "it would [have been] ... considered an infringing work if the material that it has derived from ... had been taken without the consent of a copyright [holder] of such pre-existing work"[282] or if the prior work had not entered the public domain.[283] It is important to note that the derivative works right will not come into play unless the derivative has been "*substantially* copied from a prior work."[284] In other words, if the derivative work "consists merely of ideas and not of the expression of ideas, then although the work may have been in part been derived from prior works, it is not a derivative work [for copyright purposes]."[285] This sort of requirement is necessary in order to prevent the untenable position – as illustrated by Mr. Justice Joseph Story's observations in *Emerson v.*

---

on ethical grounds" – but, "[i]ronically, then, in order to share its software with the public while preventing others from establishing their own proprietary rights in derivative versions, the GNU Project needed to *establish strong proprietary rights in its works*" [emphasis added]).

[280] *See* 17 U.S.C. § 106(2).

[281] *See* 2 NIMMER ON COPYRIGHT § 8.09.

[282] 1 NIMMER ON COPYRIGHT § 3.01.

[283] *See id.*

[284] *Id.* (emphasis in original).

[285] *Id.* (footnote omitted).

Davis[286] – that almost every extant work of art, science, or literature can be called a 'derivative work' since 'new' ideas tend to be built upon an existing body of knowledge.[287] If the pre-existing work in question is "protected by copyright, then its unauthorized incorporation in a derivative or collective work will constitute an act of copyright infringement."[288]

The derivative works issue also comes into play with open source licenses – like the GNU GPL – that place restrictions on the integration of closed source and open source software. Since the GNU GPL seeks to restrict the integration of proprietary and open source programs at the source code level, it can be seen as placing a restriction on the creation of derivative works. This is because a derivative work, for the purposes of copyright law, involves "recasting or transformation, *i.e.*, changes in the pre-existing material, whether or not it is juxtaposed in an arrangement with other pre-existing materials."[289] Integration at the source code level would be more akin to a 'transformation' or a 'recasting' rather than a simple compilation or arrangement of materials.[290] Therefore, the derivative works right provides the basis, at the copyright level of analysis, for the GNU GPL's restrictions against 'mixing' open and closed programs.[291]

---

[286] 8 F. Cas. 615, 619 No. 4436 (C.C.D. Mass. 1845), *excerpt reprinted in* 1 NIMMER ON COPYRIGHT § 3.01.

[287] However, since copyright law protects the *expression* of ideas and *does not* protect the ideas themselves, *see, e.g.*, *Whelan Associates*, 797 F.2d at 1234, modern courts should have little trouble in discerning between whether or not a derived work is a 'derivative work' for the purposes of copyright law.

[288] 1 NIMMER ON COPYRIGHT § 3.06 (footnotes omitted).

[289] 1 NIMMER ON COPYRIGHT § 302.

[290] For a background on how computer science concepts can play a role in this kind of copyright analysis, *see generally* Hamilton & Sabety, *supra* note 3. It should also be noted that if the limitation on integration is seen as a restriction on compilation or arrangement, rather than as a restriction against transformation, the licensor could argue, in the alternative, for protection on the basis of protection for 'collective works' afforded under copyright law, *see* 1 NIMMER ON COPYRIGHT § 3.03.

[291] The issue of whether or not a license can restrict a licensee from claiming copyright protection for derivative code – which the GNU GPL and the NPL/MozPL try to do, but the BSD-style and Artistic Licenses do not – will be dealt with later in this paper, *see* Part IV.[B.??].

The distribution right is another key element of open source licenses. Under Section 106(3) of the Copyright Act, the copyright holder has the exclusive right "to sell, give away, rent or lend any material embodiment of his work."[292] Open source licenses, unlike most closed source licenses, allow licensees to redistribute the pre-existing open source software and adaptations to the pre-existing code base. This express grant to allow the redistribution of software by users who modify the code base is important because Section 117 of the Copyright Act – following the recommendations of the National Commission on New Technological Uses of Copyrighted Works (CONTU)[293] – was amended so that, even if the user created a non-infringing derivative work, a user who created modifications to the source code base could not redistribute copies of those modifications without the authorization of the original copyright holder.[294]

Finally, are there any restrictions under copyright law that would prevent the copyright holder from licensing away both his derivative works right and his distribution right? The answer to this question is no. Intellectual property rights – like the proverbial 'bundle of sticks' in non-intellectual property law settings – can be divided up like sticks in a bundle – where each stick represents a particular right.[295] The licensor can then license away all, some, or only one of these

[292] 2 NIMMER ON COPYRIGHT § 8.11.

[293] For the CONTU Report, *see* Final Report of the National Commission on New Technological Uses of Copyrighted Works (July 31, 1978). For commentary by one of the commissioners of the CONTU Report, *see* Arthur Miller, *Copyright Protection for Computer Programs, Databases, and Computer-Generated Works: Is Anything News Since CONTU?,* 106 HARV. L. REV. 978 (1993).

[294] *See* 2 NIMMER ON COPYRIGHT § 8.08[B][3]. If the user made no adaptations to the software – beyond what is necessary for archival purposes, *see id.* § 8.08[C] – and was attempting to redistribute an exact copy, "such a transfer may occur only if made in conjunction with a transfer of all rights in the program, and then, only to the person acquiring such rights," *id.* § 8.08[B][3].

[295] *See* Neukom & Gomulkiewicz, *supra* note 121, at 780 (presenting the bundle of sticks analogy); 1 MILGRIM ON LICENSING § 5.33 ("The former doctrine of 'unity of copyright' is abolished by the 1976 Act. Copyright now consists of a group of distinct, enumerated rights which may be separately licensed, conveyed and exploited.").

rights – including the derivative works or the distributions rights – to the licensee.[296]

The question of to what extent the open source licensor can place restrictions in exchange for licensing away some of his rights is a question that will be dealt with in the contractual analysis portion of this paper.[297]

4. Is it Possible to Accomplish the Goals of Open Source Licensing Without Resorting to Licensing (i.e., Relying Only on Copyright Law)?

Even if we accept the idea that open source developers need to establish copyright over their programs,[298] do open source developers need to use licensing as a means of achieving the goals of the open source movement? To put it another way, is it possible to achieve the same results utilizing copyright principles – or exceptions to copyright principles – in order to keep source code open, allow users to modify the code base, and allow users to redistribute the code base and any derivatives thereof?

In her article, *Modifying Copyrighted Software: Adjusting Copyright Doctrine to Accommodate a Technology,* Pamela Samuelson argues in favor of creating avenues in copyright law for users to modify software.[299] Samuelson's reasons for supporting the right for users to modify their software are remarkably similar to some of the utilitarian justifications for open source software: Open source code allows for higher quality, easier fixes for bugs and other problems, and the possibility that the user can improve on the pre-existing software.[300]

---

[296] *See* note 295; *see also* 1 NIMMER ON COPYRIGHT § 3.06 n. 1.1 (authorization for creating derivative works can be given either exclusively or nonexclusively).

[297] *See* discussion *infra* Part IV.B.

[298] *See* discussion *supra* Part IV.A.2.

[299] Pamela Samuelson, *Modifying Copyrighted Software: Adjusting Copyright Doctrine to Accommodate a Technology*, 28 JURIMETRICS J. 179 (1988) [hereinafter Samuelson, *Modifying*].

[300] *Compare id.* at 179-84, 206-10, 220-21, *with* Eric Raymond, *Open Source: Programming as if Quality*

Along with a series of theoretical proposals,[301] the paper offers four possible ways to argue in favor of user modification of software under current copyright law: the language of Section 117,[302] the 'fair use' defense,[303] the 'first sale' doctrine,[304] and non-statutory personal or private use defenses.[305] With respect to the typical software copyrighting scenario, Samuelson concludes that current copyright law – including the four possible routes listed above – would usually fail to afford a user the right or the safe harbor to modify software.[306]

If we were to adapt Samuelson's arguments for the somewhat atypical scenario presented by open source software, we would still find that copyright alone – without licensing – would not be able to satisfy the goals of the open source movement. For instance, the Section 117[307] and the 'first sale'[308] arguments in favor of user modification, for open source purposes, would be unsatisfactory for the same reasons that public domain was an unsatisfactory solution for open source developers: Open source developers would lose the ability to maintain the restraints that

---

Mattered, INTELLECTUAL CAPITAL.COM, (Jan. 21, 1999) <http://www.intellectualcapital.com/issues/99/0121/icbusiness.1asp> [hereinafter Raymond, *Quality*]. For more on the qualitative advantages of open source software, *see* discussion *infra* Part VI.

[301] *See* Samuelson, *Modifying*, *supra* note 299, at 214-20. How Samuelson's proposals relate to justifications for the open source software model of software development will be dealt with in Part VI. of this paper, *infra*.

[302] *See id.* at 184-93.

[303] *See id.* at 193-95.

[304] *See id.* at 195-97.

[305] *See id.* at 198-200.

[306] Presumably, such an analysis could be analogized to exclude the possibility of open source-style redistribution on the part of users utilizing current copyright law without an express grant through licensing.

[307] Section 117 of the Copyright Act provides certain limited exemptions for otherwise infringing actions with respect to computer programs, *see* 17 U.S.C. § 117; 2 NIMMER ON COPYRIGHT § 8.08[B][1].

[308] The 'first sale' doctrine means that "once a copy of a copyrighted work has been sold [i.e., ownership interests have been fully transferred], the copyright holder's rights in that particular copy are exhausted, and the copy may be freely resold, leased or loaned," Neukom & Gomulkiewicz, *supra* note 121, at 778; *see* 17 U.S.C. § 109.

are necessary to maintain open source development. Why? Both the Section 117 and the 'first sale' justifications would only work if the open source developer fully transfers all copyright interests to the users.[309]

Presuming these arguments are even viable,[310] if an open source developer tried to take advantage of Section 117 or the 'first sale' doctrine in order to allow users to modify the software, the open source developer will have to give up all of his intellectual property rights to the program. While this may not seem all that significant since open source licenses tend to weaken intellectual property rights, open source developers – whether 'free software' idealists or utilitarians – want to utilize licensing conditions to maintain some control over their work product – almost always in furtherance of the goals of the open source movement. For example, giving up all claims to the program will not only mean that strict conditions placed by the GNU GPL cannot be implemented but will also mean that a condition that is present in all open source licenses – such as the condition that open source code base must be 'freely' redistributable – cannot be imposed as well.

This situation is rather ironic since closed source software producers use licensing – rather than an outright sale – in order to limit or prohibit the redistribution of software.[311] Open source developers, on the other hand, use the same device – licensing rather than outright transfer of all rights – in order to ensure that users can freely redistribute software.

---

[309] For Section 117: *See* Samuelson, *Modifying*, *supra* note 299, at 188 (privilege is granted only to the "owners of copies"). For first sale: *See id.* at 197 ("[L]ike § 117, § 109 provides protection only to 'owners of copies.'").

[310] *See supra* note 306 and accompanying text.

[311] *See* Neukom & Gomulkiewicz, *supra* note 121, at 778 (licenses are used to prevent the first sale doctrine from coming into play); Aldus License, *supra* note 143, at 791 (conditions restricting the distribution of software).

The 'fair use' doctrine[312] and non-statutory personal or private use defenses – presuming they are viable justifications for user modification – seems to have limited benefits in the context of open source software. The fair use and personal use defenses may or may not work for an individual who argues in front of a court. However, this sort of scenario is a far cry from a wide spread grant to users to modify and redistribute software. It would not only be more efficient to have an *ex ante* licensing of privileges to users – since users' 'uses' of the software need not be evaluated on a case-by-case basis – but it would also better serve the interests of commercial open source software producers – like Netscape – that need to protect some material from being modified while attempting to grant access to less sensitive code.

a. Analogy to 'Shareware' I.: Additional Reasons for Not Using the 'Fair Use' Justification

Another reason for not using arguments like 'fair use' is because – in the context of shareware – at least one court has ruled against a defendant trying to use such a defense to infringe upon the exclusive rights of a copyright holder.[313]

It is important to note at this point that ***shareware is not the same thing as open source software (or 'free software' – in Stallman-esque language).*** Shareware – like open source software – allow licensees[314] to redistribute the shareware[315] and tends to be distributed with little

[312] The courts and the Copyright Act recognize some limited situations where someone may act in a way that would ordinarily infringe upon the exclusive rights of the copyright holder but for the 'unfairness' of finding such a person liable for infringement, *see* 17 U.S.C. § 109; 4 NIMMER ON COPYRIGHT § 13.05. In making a determination of whether a fair use defense is valid or not, courts will use the following four factors: (1) The purpose and the character of the use; (2) the nature of the copyrighted work; (3) the amount and substantiality of the portion used; and (4) the effect of the use upon the potential market, *see* 17 U.S.C. § 109; *see, e.g.*, Campbell v. Acuff-Music, Inc., 510 U.S. 569 (1994).

[313] *See* Storm Impact, Inc. v. Software of the Month Club, 13 F.Supp.2d 782 (N.D.Ill. 1998).

[314] Like the vast majority of software – including open source – shareware is copyrighted and has its own type of licenses, *see* Software Publishers Association, *Commercial, Shareware, Freeware & Public Domain Software*, SPA ANTI-PIRACY, (last visited Mar. 26, 1999) <http://www.spa.org/piracy/share.htm> [hereinafter SPA, *Software*]. For a proper definition of 'shareware,' *see* Computer & Internet Dictionary, *supra* note 7, at 505.

[315] *See* Halloween I, *supra* note 17, at 'Software Licensing Taxonomy.'

or no nominal price,[316] but – unlike open source software – shareware cannot be modified without the licensor's permission.[317] However, for the purposes of this paper, shareware is a close enough analog to open source software in order to examine whether novel justifications under copyright law – like fair use – can be used as an alternative to open source licensing.

In *Storm Impact, Inc. v. Software of the Month Club*,[318] the court held against a defendant who attempted to use the fair use defense in order to justify the infringement of the copyrights on two shareware programs.[319] In its analysis of the fair use defense, the court seems to mention the possibility of using fair use in order to justify allowing a user to modify a copyrighted work.[320] However, because the court did not find that the defendant's modifications "sufficiently" transformed the shareware programs for the purposes of fair use analysis,[321] it is difficult to say whether or not the *Storm Impact* court would have found an infringement of the derivative works right in a scenario where the defendant *did* sufficiently transform shareware through his modifications.

Despite this uncertainty, it seems more than likely that a court that follows the logic of *Storm Impact* would rule against a fair use defense for the modification of shareware – or any other piece of software that does not license, to the user, the right to modify. In *Storm Impact*, the

---

[316] *See* Computer & Internet Dictionary, *supra* note 7, at 505. It should be noted, however, that the shareware producer does expect to be paid by the licensee for the program. The method of collecting payment is – unlike most for-profit software – based on the honor system, *see id.*

[317] *See* SPA, *Software*, *supra* note 314, at 'Shareware.' For an example of a shareware license, *see Storm Impact*, 13 F.Supp.2d 785-86.

[318] 13 F.Supp.2d 782 (N.D.Ill. 1998).

[319] *See id.* at 791.

[320] *See id.* at 786-87 ("[T]here is an inherent tension in the need to protect copyrighted material and to allow others to build upon it."); *see also id.* at 788 ("Transformation in the fair use context anticipates transformation by the user, not the copyright holder." [citing Campbell, 510 U.S. at 578-79]).

court found that the second factor of the fair use defense analysis – "the nature of the copyrighted work"[322] – was weighed against the defendant because the shareware programs involved in this case were 'creative' and 'original' works (as opposed to "facts or news").[323] According to the *Storm Impact* court, the plaintiffs works are creative and original because they are computer games.[324] It seems reasonable to conclude that a *Storm Impact* style court would find that a case involving computer software – since such a work "is at the core of intended copyright protection"[325] – would make it difficult for a user to claim fair use modifications if the software license did not allow for such a modification. This aspect of the case may mean that it is harder to justify infringing the exclusive rights – including the derivative works right – a copyright holder has over computer software through a fair use defense.

What is not in doubt is that the *Storm Impact* court held against the use of a fair use defense to justify infringement of the distribution right of the shareware creator.[326] The court held that the shareware creators' method of distribution – placing "the material on the Internet, [and] allow[ing] limited distribution for free"[327] – did not mean that the plaintiffs "impliedly consented" to the defendant's redistribution of the shareware.[328] Therefore, the defendant was without excuse

---

[321] *See id.* at 788.

[322] 17 U.S.C. § 107(2).

[323] *See Storm Impact*, 13 F.Supp.2d at 789.

[324] *See id.*

[325] *See id.*

[326] *See id.* at 790-91.

[327] *Id.* at 791.

[328] *See id.* at 790-91.

when it "adversely affected the market for"[329] the plaintiffs' shareware programs – leading to the conclusion that "the fourth factor [in a fair use analysis] militates against a finding of fair use."[330]

All of this leads one to conclude that the fair use defense – in the absence of express conditions allowing a user to modify and redistribute software – will not serve as an effective means of accomplishing the goals of the open source movement.[331] If the example of shareware is sufficiently analogous to open source software for the purposes of copyright law, then the creator of open source software should utilize a license, rather than relying on pure copyright principles, in order to ensure that the goals of the open source movement are met.

B. The Contractual Enforceability of Open Source Licenses

As we saw in the previous discussion, copyright principles alone are usually not sufficient to accomplish the goals of the open source movement. Rather than relying on the public domain or exclusively on copyright principles, an open source developer needs to establish copyright over his program and *then* license his program in such a way as to further the goals – whether those goals are 'free software,' utilitarian, or somewhere in between – of the open source movement.[332]

This section will examine whether or not – once copyright has been established in the open source program – open source licenses can be enforced under contract law. The contractual

---

[329] *Id.* at 790.

[330] *Id.*

[331] This view is further bolstered when we consider the Second Circuit's opinion in *American Geophysical Union v. Texaco*, 60 F.3d 913 (1995), *cert. dismissed*, 516 U.S. 1005 (1995). In *American Geophysical*, the defendant made a fair use argument to justify the copying of articles from various scientific journals by its research scientists. The Second Circuit held that the copying of articles from scientific journals for the purposes of research by scientists working in industry was not justified by a fair use defense, *see* 60 F.3d at 931. If we draw an analogy between open source software and journal articles – both of which can be thought of as catalysts for intellectual advancement – then an opinion like *American Geophysical* bodes ill for those who would want to rely solely on a fair use defense, in the absence of the granting of privileges to users via licensing, in order to allow users to copy, redistribute, or create derivative works with little or no restrictions.

[332] The logic of open source licensing in general is identical to the logic of Richard Stallman's copylefting, *see supra* Part II.B.

analysis of open source licensing will also explore to what extent the conditions of the stricter licenses – like the GNU GPL and the NPL/MozPL – can be enforced on a licensee.

Although some of the law in this area is unsettled and controversial, this paper concludes that open source licenses are enforceable as a matter of contract law.[333]

1. Adequacy of Consideration

Although open source licenses do not prohibit the licensor from receiving money in exchange for the licensed software,[334] most open source licenses are formed without the use of money or the promise to pay money in the future.[335] While this situation may be changing with the emergence of for-profit open source developers – like Red Hat, Netscape, and IBM – there is no requirement that open source licenses be formed with the use of money,[336] and, therefore, there will always be some open source licenses formed without the use of pecuniary consideration.[337]

If money is not involved, what are the considerations exchanged in order to form a contractual relationship between an open source developer and the user? The developer is granting a license for the use of his program – along with the binary code, the source code, and

[333] It is worth noting here that, if a licensee violates the condition of a license, the licensor "may have an action for copyright infringement, breach of contract, or both," *see* Heffan, *supra* note 9, at 1502 & n. 96; *see also* Gilliam v. American Broadcasting Companies, Inc., 538 F.2d 14, 21(2d Cir. 1976). Therefore, the discussion in this section on the contractual nature of open source licenses will often overlap with the copyright nature of open source licenses.

[334] *See, e.g.*, Free Software Foundation, *Some Confusing and Loaded Words and Phrases that are Worth Avoiding*, (last visited Aug. 5, 1998) <http://www.gnu.org/philosophy/words-to-avoid.html> [hereinafter FSF, *Confusing*], at "For free" (stating that 'free software' is available for free – in terms of price – but can also be available for a price). The most obvious example of an open source software license that involves the exchange of money is for-profit open source software, *see supra* Part II.B.3.

[335] *See, e.g.*, Heffan, *supra* note 9, at 1489, 1507, 1510, 1511 n. 148 (in many cases, people obtain the GNU Project's programs at no cost); McHugh, *supra* note 5, at 95-100 (providing a number of examples where licenses were formed without the use of money as consideration).

[336] *See OSD (v.1.0)*, *supra* note 44, ¶ 1 (open source licenses "may not restrict any party from selling *or giving away* the software ..." [emphasis added]).

[337] *Cf.* Heffan, *supra* note 9, at 1510 (Ira Heffan, in his analysis of the GNU GPL, assumes that "the requirement of consideration will not be met by monetary payment").

provisions that allow the licensee to create derivative works and redistribute the code base and any derivatives thereof – in exchange for the user's promise to follow the terms and conditions of the open source license.[338]

The open source developer's consideration is obviously adequate; this is the kind of consideration (*sans* the open source code and various privileges to the user) that the more typical closed source developers bargain away all the time.[339] However, in a closed source scenario, the user's consideration is money (or the promise to pay money) rather than the promise to abide by the terms of the license. Is the promise to abide by the terms and conditions of the license adequate consideration to form an open source license?[340] In other words, is the distribution of open source software merely a gift,[341] and, therefore, unenforceable under contract law?

Generally speaking, monetary consideration is not necessary to form a contract.[342] According to E. Allen Farnsworth, "[v]irtually anything that anyone would bargain for in exchange for a promise can be consideration for that promise."[343] For example, a promise to refrain from smoking, drinking, and gambling until the age of 21 was considered to be adequate

---

[338] *See, e.g.*, Heffan, *supra* note 9, at 1510-11 (providing an example of the considerations being exchanged for the formation of the GNU GPL).

[339] *See, e.g.*, Neukom & Gomulkiewicz, *supra* note 121, at 777 ("Most computer software publishers transfer their software by license.").

[340] Note: This question should be distinguished from whether an offer and an acceptance has taken place – an issue that will be discussed shortly, *see infra* Part IV.B.2. A typical shrinkwrap, 'clickwrap,' and/or 'webwrap' license will not face this sort of problems since money – and not merely the promise to follow the terms of the license – serves as the consideration being exchanged by the user.

[341] Eric Raymond, who is more of a utilitarian than a 'free software' idealist, often describes the hacker culture that has spawned the open source movement as a culture of gift giving, *see* McHugh, *supra* note 5, 99.

[342] *See* P.S. ATIYAH, AN INTRODUCTION TO THE LAW OF CONTRACT 127 (5th ed. 1995), *and* E. ALLEN FARNSWORTH, CONTRACTS § 2.11, at 69 (discussing the 'peppercorn theory' – where even something of minimal value, like a peppercorn, can be considered to be adequate consideration for a contract).

[343] *See* FARNSWORTH, *supra* note 342, § 2.3, at 46-47.

consideration.[344] In fact, courts rarely inquire into the question of whether or not consideration is adequate.[345] Therefore, it seems safe to conclude that the user's promise to abide by the conditions of the license is sufficient consideration for the purposes of contract law.[346]

a. Analogy to Shareware II.: Further Support for the Enforcement of a Software License Formed Without Monetary Consideration

In *Storm Impact*,[347] the court held that the restrictions placed on users by a license[348] for shareware – posted on the Internet for free (zero price) distribution – was enforceable.[349] The court noted that this situation – the distribution of software posted on the Internet where the nominal price is zero – was a "question of first impression."[350]

While this case deals with shareware and not open source software,[351] a holding by a court in favor of the enforceability of a license for Internet-posted software not involving the exchange of money would bode well for the enforceability of open source licenses and negate any fears

---

[344] *See* Hamer v. Sidway, 124 N.Y. 538, 27 N.E. 256 (1891).

[345] *See, e.g.*, Apfel v. Prudential-Bache Securities, Inc., 81 N.Y.2d 470, 475, 616 N.E.2d 1095, 1097 (1993) (courts rarely question the adequacy of consideration); Bolton v. Madden (1873) LR 9 QB 55, at 57 (Lord Blackburn: "The adequacy of the consideration is for the parties to consider at the time of making the agreement, not for the Court when it is sought to be enforced.").

[346] *See* Heffan, *supra* note 9, at 1510-11 (concluding that the GNU GPL is "supported by adequate consideration").

[347] *See supra* Part IV.A.4.a.

[348] For the terms and conditions of the shareware license, *see Storm Impact*, 13 F.Supp.2d at 785-86.

[349] *See id.* at 791. Note: Although the court is dealing with a question of copyright infringement, as an earlier footnote pointed out, *see* discussion *supra* note 333, a violation of a copyright license can be dealt with as a violation of copyright, a breach of contract, or both, *see also* 3 NIMMER ON COPYRIGHT § 10.15.

[350] *See Storm Impact*, 13 F.Supp.2d at 791.

[351] However, shareware licensing may be a close enough analog to open source licensing for the purposes of determining the enforceability of open source licenses, *cf.* Heffan, *supra* note 9, 1501-1504 (making a similar argument when analyzing the GNU GPL).

about the lack of monetary consideration.[352]

2. Assenting to an Open Source License

Even if we conclude that open source licensing involves the exchange of adequate consideration by both sides, an open source license is not contractually enforceable until the licensor makes a valid offer and the licensee makes a valid acceptance.

The offer side of the equation is fairly simple: The open source developer/licensor makes an offer by making his open source software available for distribution on the Internet, on a CD-ROM, or through another commonly used means of software distribution[353] – a scenario that is akin to offering up a good by placing it on the shelf of a store.[354] The acceptance side, on the other hand, is complicated by the fact that most software – including open source software – is offered 'as is' without the opportunity for negotiation or review prior to acceptance.[355] Additionally, acceptance of software licenses is normally indicated by "something other than the offeree's signature."[356]

Therefore, this paper will focus in on whether or not, in a typical scenario, a user can

---

[352] To be fair, a shareware license can be said to involve monetary consideration since the licensee is, arguably, 'promising' to pay the developer at some time in the future, *see* Heffan, *supra* note 9, at 1510 (registering the shareware involves the user paying money to the licensor); SPA, *Software*, *supra* note 314, at 'Shareware' (user/licensee is promising to pay for the software upon adoption). However, the *Storm Impact* court views this case – perhaps incorrectly – as a case involving "free" (meaning zero price) distribution of software posted on the Internet and does not consider the future payment aspect of shareware licensing, *see Storm Impact*, 13 F.Supp.2d at 791. This paper, for the sake of analysis, accepts the *Storm Impact* court's logic.

[353] *See* FARNSWORTH, *supra* note 342, § 3.10, at 132-33 ("Conduct that would lead a reasonable person in the other party's position to infer a promise in return for performance or promise may amount to an offer.").

[354] *See* ProCD, Inc. v. Zeidenberg, 86 F.3d 1447, 1450 (7th Cir. 1996) ("[P]lacing the package of software on the shelf is an 'offer,' which the customer 'accepts' by paying the asking price and leaving the store with the goods."); *see also* Barker v. Allied Supermarket, 596 P.2d 870, 871-873 (Okla. 1979), *and* Peeters v. State, 142 N.W. 181, 182 (Wis. 1913) (simply placing a good up for sale and having someone come in and take the good, with at least the *intention* of paying for the good, *see* Barker, 596 P.2d at 871-73, is enough to have an offer and an acceptance).

[355] *See* David G. Post & Dawn C. Nunziato, *Shrinkwrap Licenses and Licensing on the Internet*, *in* TECHNOLOGY LICENSING AND LITIGATION 1997, 477 PLI/PAT 517, 519 (1997).

properly assent to an open source license where acceptance is suppose to be indicated without negotiation, *ex ante* review, and the use of a licensee's signature. Although the law in this area is highly controversial, acceptance of an open source license, without the presence of more traditional indications of contractual assent, is probably enforceable as a matter of law.[357]

a. 'Shrinkwrap' Licenses, 'Clickwrap' Licenses, 'Webwrap' Licenses, and *ProCD v. Zeidenberg*

Most licensed software packages that can be bought at a local retailer utilize 'shrinkwrap' licensing in order to place terms and conditions on the end user.[358] Recently, with the increasing use of the Internet as a means of distributing software, software developers have been utilizing so-called 'clickwrap' and/or 'webwrap' licenses in order to place conditions on the end user.[359] Like their closed source counterparts, open source software developers use shrinkwrap, clickwrap, and/or webwrap licensing as a means of binding users to the terms and conditions governing the use of an open source program.[360]

---

[356] *Id.*

[357] Bruce Perens, the original author of the 'Open Source Definition,' hinges the automatic application of an open source license based on a belief that this sort of "no-signature-required license" will be tested in the courts in the near future, *see* Perens, *The Open Source Definition*, *supra* note 43, at 179.

[358] *See* Post & Nunziato, *supra* note 355, at 519. 'Shrinkwrap' licenses are so named because most of the terms of the licenses are placed within the shrinkwrap packaging of computer software and acceptance is indicated by opening the software package, *see id.*

[359] *See id.*; *see also* Jerry C. Liu et al., *Electronic Commerce: Using Clickwrap Agreements*, 15 No. 12 COMPUTER LAW. 10 (1998). 'Clickwrap' and 'webwrap' licenses are similar to a 'shrinkwrap' license in binding a user to the terms of a software license, *see* Post & Nunziato, *supra* note 355, at 519. However, 'clickwrap' licenses go further than shrinkwrap licenses by offering the user a pair of on-screen buttons that gives the user the choice of accepting (by clicking the 'I AGREE' or 'I accept' button) or rejecting (by clicking the 'I DECLINE' button) the terms governing the use of the program, *see* Liu et al., *supra*, at 10. Acceptance of a 'webwrap' license can be indicated by either clicking an 'I AGREE' button (like a 'clickwrap' agreement) or by simply downloading and then using the software (which is logically similar to 'shrinkwrap' licensing), *see* Post & Nunziato, *supra* note 355, at 519.

[360] *See supra* note 357; *cf.* FSF, *GNU GPL*, *supra* note 58, at 'How to Apply These Terms to Your New Program' (the FSF's example of offering an interactive way of evaluating the license can be thought of as something akin to clickwrap licensing).

As noted earlier, because these kinds of licenses do not involve negotiation, *ex ante* review of the program, the use of a signature, or many other traditional indications of assent, there is a serious question as to whether parties can adequately understand and assent to the terms of these kinds of mass-market licenses.[361] In other words, it is hard to tell whether or not there was the proper assent necessary to create a legally enforceable contractual relationship between the parties.

The question of whether or not shrinkwrap, clickwrap, or webwrap open source licenses can be properly accepted by the user is answered if we analogize open source licensing to the scenario found in *ProCD, Inc. v. Zeidenberg*.[362] In *ProCD*, the Seventh Circuit – in an opinion written by Frank Easterbrook, J. – held that the opening up of the shrinkwrap packaging of a software program (i.e., a shrinkwrap license) – as well as the acquiescence to the on-screen license encoded in the software (i.e., equivalent to a clickwrap or a webwrap license) – constituted an acceptance of the software license.[363] One commentator summarizes the contractual formation part of the holding in the following way:

> [T]he court found that Zeidenberg [the defendant] did have adequate notice of the terms on which ProCD [the plaintiff] offered him its product. The exterior of the packaging bore a notice stating that acquiring the product subjected the user to a license, the terms of which were stated inside the package. The ProCD computer program supplied a screen that referred to the license terms, which Zeidenberg had to acknowledge with a key stroke or a mouse click before he could use the

[361] *See* Post & Nunziato, *supra* note 355, at 519; *see also* Brian Covotta & Pamela Sergeeff, Note, *ProCD, Inc. v. Zeidenberg*, 13 BERKELEY TECH. L.J. 35 (1998).

[362] 908 F.Supp. 640 (W.D.Wis. 1996), *rev'd*, 86 F.3d 1447 (7th Cir. 1996).

[363] *See id.* at 1448-54; *see also* Covotta & Sergeeff, *supra* note 361, at 39.

> program to access the listings. According to the court, he was, therefore, bound by the license terms when he chose to use the product rather than to return it to the "seller." Under Wisconsin's version of the UCC, Zeidenberg accepted one and declined the other of the two options explicitly put to him by ProCD in the license: use the product and accept the license, or reject the license and return the product for a refund.[364]

It should be noted that the contractual assent holding of *ProCD* has been criticized[365] and may not be followed by other courts.[366] One could address the criticisms by pointing to commentaries that have been, more or less, favorable to *ProCD*.[367] In addition, while there have been opinions that seem to run counter to the *ProCD* holding,[368] these contrary opinions may be distinguished from the *ProCD* scenario by the fact that the *ProCD* opinion is the case that has most directly dealt with the issue of shrinkwrap and other types of mass-market technology

---

[364] Michael J. Madison, *Legal-Ware: Contract and Copyright in the Digital Age*, 67 FORDHAM L. REV. 1025, 1052 (1998) (footnotes omitted).

[365] *See, e.g.*, Brandon L. Grusd, *Contracting Beyond Copyright: ProCD, Inc. v. Zeidenberg*, 10 HARV. J.L. & TECH. 353 (1997) (criticizing Easterbrook's expansion of intellectual property protections via contract law); Apik Minassian, Note, *The Death of Copyright: Enforceability of Shrinkwrap Licensing Agreements*, 45 UCLA L. REV. 569 (1997) (criticizing the decision on contract and copyright law grounds); Note, *Contract Law – Shrinkwrap Licenses – Seventh Circuit Holds that Shrinkwrap Licenses are Enforceable. – ProCD, Inc. v. Zeidenberg, 86 F.3d 1447 (7th Cir. 1996)*, 110 HARV. L. REV. 1946 (1997) [hereinafter 'Harv. L. Rev. Note'] (ditto).

[366] *See* Covotta & Sergeeff, *supra* note 361, at 53-54.

[367] *See, e.g.*, Robert W. Gomulkiewicz, *The Licensing is the Product: Comments on the Promise of Article 2B for Software and Information Licensing*, 13 BERKELEY TECH. L.J. 891, 896 (1998) ("The court's ruling in *ProCD v. Zeidenberg* affirming the enforceability of mass market licenses may have surprised some legal scholars, but a contrary ruling would have devastated the software and electronic information industries."); *see also* Covotta Sergeeff, *supra* note 361, at 54 (crediting the *ProCD* court of "providing some semblance of certainty in the retail software market").

[368] *See* Arizona Retail Systems, Inc. v. The Software Link, Inc., 831 F. Supp. 759, 764 (D. Ariz. 1993) (holding that a pre-existing contract could not be 'modified' by a shrinkwrap agreement without express assent as laid out in UCC § 2-209); Step-Saver Data Systems, Inc. v. Wyse Technologies, 939 F.2d 91, 105-106 (3d Cir. 1991) (similar to *Arizona Retail* except relying on UCC § 2-207).

licenses.[369]

Further bolstering the *ProCD* opinion is the fact that there have been cases that either reaffirm the holding or cite the case without criticism.[370] *ProCD* is further bolstered by the fact that the proposed Uniform Computer Information Transactions Act (ex-UCC Article 2B) – which deals specifically with licensing/contracting in the 'information' age – incorporates, to varying degrees, the logic and implications of *ProCD*.[371]

In the context of mass-market software distribution, it is also worth noting that shrinkwrap, clickwrap, and webwrap licenses – as standardized contracts – may provide an economically efficient way of placing terms and conditions on the end user.[372] The legal enforceability of standardized 'take it or leave it' contracts – assuming that an opportunity for a refund is available – is certainly not without precedent.[373]

Because of these and other factors, the software and information technology industries have developed a considerable level of reliance interest in a *ProCD*-style doctrine that would

---

[369] *See* Liu et al., *supra* note 359, at 11 (distinguishing the cases cited above, *see supra* note 368, and stating that *ProCD* is the case that "squarely" addresses the issue of the enforceability of shrinkwrap license).

[370] *See* Hill v. Gateway 2000, Inc., 105 F.3d 1147, 1149-50 (7th Cir. 1997) (reaffirming *ProCD* and extending the holding to cover contracts for the sale of computer hardware). For cases that, arguably, further bolsters the holding in *ProCD*, *see* Liu et al., *supra* note 359, at 12. For decisions that cite *ProCD* without criticism, *see* Covotta & Sergeeff, *supra* note 361, at 53 n. 92.

[371] The Uniform Computer Information Transactions Act (ex-UCC Article 2B) – and how it relates to *ProCD* – will be discussed later in this paper, *see infra* Part IV.B.2.b.

[372] *See* ProCD, 86 F.3d at 1449-51 (Easterbrook, J., entering into a quasi-economic analysis of shrinkwrap licensing and recognizing such licensing as a reasonable way to prevent economic loss); FARNSWORTH, *supra* note 342, § 4.26, at 296 ("As with goods, standardization and mass production of contracts may serve the interests of both parties .... [because] they simplify operations and reduce costs.").

[373] *See* ProCD, 86 F.3d at 1451 (citing several sources to support this kind of argument); Gomulkiewicz, *supra* note __, at 897; *cf.* CHARLES FRIED, CONTRACT AS PROMISE 43 (1981) (although advocating explicit acceptance whenever possible, Prof. Fried recognizes that there are certain situations that require what he calls "tacit acceptance").

uphold the enforceability of shrinkwrap, clickwrap, and webwrap licenses.[374] Therefore, the key principles of *ProCD*, including allowing mass-market software licenses – like shrinkwrap, clickwrap, and/or webwrap licenses – to be legally enforceable, will likely be upheld in some manner for the foreseeable future.[375]

i. Analogy to Shrinkwrap III.: Further Support for *ProCD* and Extending *ProCD* to Support 'On-line' Contractual Formation

In *Compuserve, Inc. v. Patterson*,[376] Compuserve used an on-line clickwrap/webwrap shareware license[377] in order to provide shareware as a service to its customers.[378] Although the main thrust of the *Compuserve* opinion was to decide on the question of personal jurisdiction,[379] the Sixth Circuit's opinion has been read as a holding in favor of allowing clickwrap/webwrap contracts that are formed on-line to be legally enforceable.[380]

If the *Compuserve* opinion can be read as supporting the contractual enforceability of on-line agreements, this decision would further support the reasoning found in *ProCD*.[381]

---

[374] *See* Gomulkiewicz, *supra* note 367, at 892-94, 908.

[375] *Cf.* Covotta & Sergeeff, *supra* note 361, at 54 (although the article is somewhat critical of *ProCD*'s implications for copyright law [which will be discussed later in this paper, *see infra* Part IV.B.3.a., nevertheless recognizes the need for "some meaningful guidance" on the issue of shrinkwrap licensing); Madison, *supra* note 364, at 1053-54 (practitioners design software licenses based on the presumption that *ProCD*-style logic is here to stay).

[376] 89 F.3d 1257 (6th Cir. 1996).

[377] *Id.* at 1260.

[378] *Id.*

[379] *Id.* at 1268-69.

[380] *See id.* at 1260-61, 1264 (the court found that the defendant had "entered into a written contract with Compuserve"); Liu et al., *supra* note 359, at 12 ("[T]he Sixth Circuit ... held that Patterson had entered into an online contract with Compuserve .... The court concluded that Patterson *manifested assent* to the [shareware license] ... by typing "Agree" at points throughout the agreement. The court thus suggested that a contract formed online is enforceable." [emphasis added]).

[381] The *Compuserve* opinion cites *ProCD*, *see* Compuserve, 89 F.3d at 1260.

Additionally, *Compuserve* – along with some other *ProCD*-style cases and principles – would extend the logic of *ProCD* to cover contracts formed on-line (as opposed to 'off-line' as in *ProCD* itself).[382] Therefore, if we assume that there is enough of a parallel between clickwrap/webwrap shareware licenses and clickwrap/webwrap open source licenses, then the kind of rationale found in both *ProCD* and *Compuserve* would mean that, regardless of whether the licenses were formed off-line or on-line, the user can manifest the assent necessary to form a valid contractual relationship.[383]

ii. Conclusions: Implications for Open Source Licenses

What does a *ProCD*-style principle for contractual formation mean for open source licensing? First, if assent for the formation of licensing agreements can be manifested by simply opening a shrinkwrap package, clicking the 'I accept' button (or typing in words to that effect), and/or by downloading software, it would mean that the vast majority of mass-market software licensing agreements – including open source licenses – can be entered into for any of the currently popular means of software distribution. *ProCD*, and its progeny, would also mean that on-line licensing agreements – which is common for Internet distributed software – can be formed without the traditional means of showing acceptance. Since open source software is often distributed over the Internet, open source licensing would benefit from a *ProCD*-style principle favoring on-line contractual formation.

Additionally, as we shall see in Part IV.B.3., *infra*, a *ProCD*-style reasoning for the

[382] *See* Liu et al., *supra* note 359, at 12. It is fairly logically simple to extend *ProCD* to cover both on-line and off-line contractual situations, *see, e.g.*, Post & Nunziato, *supra* note 355, at 519 (noting that many webwrap licensing schemes require that users assent to the licensing terms on-line *before* they download software).

[383] Arguably, *Storm Impact* – since the court held in favor of enforcing the terms and conditions of an on-line copyright (shareware) license – can be seen as furthering supporting the notion that a user can manifest assent (or can, at the very least, be seen as having *constructively* assented, *see* Covotta & Sergeeff, *supra* note 361, at 35 n. 3) to on-line agreements, *see* Storm Impact, 13 F.Supp.2d at 791.

licensing of technology would give open source licensors – as well as their closed source counterparts – a freer hand to grant privileges and/or place restrictions on the end user. As we will see later on, this is another case in which open source licensing – ironically – depends on the same set of legal principles and theories, such as *ProCD*, that closed source software licensors depend on. Although some open source advocates think of open source software as a replacement for closed source 'shrinkwrap' software,[384] the open source movement – to a large extent – depends on the enforceability of 'shrinkwrap' licenses – as embodied by *ProCD* – in order to establish the legal enforceability of open source licenses.[385]

b. Proposed Uniform Computer Information Transactions Act (Formerly Known as UCC Article 2B. Licenses)

[**Note:** As of April 7, 1999, what was formerly known as the 'Drafts for UCC Article 2B. Licenses' is now known as the 'Uniform Computer Information Transactions Act' [hereinafter UCITA][386] This is due to the fact – that as of the time of writing – the ALI has withdrawn its support for ex-UCC Art. 2B.[387] Without this support, the NCCUSL cannot promulgate this proposed statute as a part of the UCC.[388] For the sake of clarity, the terms 'UCC Article 2B' and

---

[384] *See* Ronald Kuetemeier, *Editorial: OSS to Replace Shrink-wrapped Software?*, FRESHMEAT.NET (Jan. 19, 1999) <http://ct.us.mirrors.freshmeat.net/news/1999/01/19/916751786.html>.

[385] *See supra* note 357.

[386] *See NCCUSL to Promulgate Freestanding Uniform Computer Information Transactions Act: ALI and NCCUSL Announce that Legal Rules for Computer Information Will Not Be Part of UCC*, NATIONAL CONFERENCE OF COMMISSIONERS OF UNIFORM STATE LAWS (Chicago, Il), (Apr. 7, 1999) <http://www.nccusl.org/pressrel/2brel.html>, *and* AMERICAN LAW INSTITUTE (Philadelphia, PA), (Apr. 7, 1999) <http://www.ali.org/ali/pr040799.htm> (joint press release by the NCCUSL and ALI stating that ex-UCC Article 2B will now be promulgated as the Uniform Computer Information Transactions Act [hereinafter 'UCITA']); Dan Goodin, *Group Pulls Support of Software-Sales Rules*, NEWS.COM (Apr. 12, 1999) <http://www.news.com/News/Item/0,4,34975,00.html> [hereinafter Goodin, *Pulls Support*] (reporting that the ALI has withdrawn support for ex-UCC Art. 2B; therefore, only the NCCUSL and computer industry groups will support UCITA).

[387] *See* Goodin, *Pulls Support*, *supra* note 386.

[388] *See id.*

'Uniform Computer Information Transactions Act' (and/or derivatives thereof) are interchangeable for the purposes of this paper.]

[**Additional Note:** Prof. Pamela Samuelson, of Boalt Hall, has informed me that it is unlikely that the ALI will reinstate support for ex-Art. 2B.[389] Therefore, in all likelihood, the NCCUSL – with the support of industry groups and major software producers – will pursue the enactment of UCITA among the states (perhaps as early as the Fall of 1999) without the support necessary to make it a part of the UCC.[390]]

Although *ProCD v. Zeidenberg* has provided some level of clarity for those software developers looking to license their programs,[391] the unsettled nature of the case law on mass-market software licenses has meant that the question of whether or not shrinkwrap, clickwrap, and/or webwrap licenses can be properly formed and enforced via contract law has yet to be definitively answered.[392] In order to definitively and uniformly address some of the concerns associated with the rise of electronic commerce, the Uniform Computer Information Transactions Act (UCITA) – ex-UCC Article 2B – was proposed by the National Conference of Commissioners on Uniform State Laws (NCCUSL) in order to deal with "transactions in information"[393] and to provide a "framework for contractual relationships among industries at the

---

[389] *See* E-mail from Pamela Samuelson, Professor, University of California, Berkeley, School of Law (Boalt Hall) (Apr. 13, 1999) [hereinafter Samuelson, E-mail] (on file with the author).

[390] *See id.*

[391] *See* Covotta & Sergeeff, *supra* note 361, at 54.

[392] *See, e.g., id.* at 53-54.

[393] [Note: Unless otherwise specified, this paper will use the April 15, 1998 draft of ex-UCC Art. 2B. This is because this draft seems to be more 'stable' than subsequent drafts. However, I will try to use language from the latest drafts whenever possible.] American Law Institute, Uniform Commercial Code Article 2B. Licenses: Tentative Draft (April 15, 1998) at Preface, 2 (1998) [hereinafter UCITA]. The latest draft(s) are available at <http://www.law.upenn.edu/library/ulc/ulc.htm>.

forefront of the information era."[394]

Not surprisingly, UCITA codifies much of the reasoning and results of the *ProCD* decision.[395] Generally speaking, like *ProCD*, UCITA recognizes the validity of mass market shrinkwrap licenses.[396] UCITA also recognizes methods for manifesting assent – like the user clicking the appropriate button or, even, using an "electronic agent" to manifest assent – that are commonly used for shrinkwrap, clickwrap, and webwrap licenses – whether they are formed on-line or off-line.[397] For example, the commentary to UCITA § 2B-111 uses the example of the *on-line* New York Times on the Web click-on pre-registration screen – where the user can click either "I Agree" or "I Decline" – as an example of an on-line mass market (i.e., webwrap) license that would be enforceable.[398]

In addition to the question of manifesting assent for information technology licenses, UCITA – as *ProCD* implicitly does by finding the shrinkwrap/click-on license enforceable – also eliminates Statute of Fraud concerns about enforcing shrinkwrap, clickwrap, and webwrap

---

[394] *Id.*; *see also* Gomulkiewicz, *supra* note 367, at 892 ("A contract statute [like the proposes UCITA] ... holds great promise for software and information licensing[,] [because] [l]icensing law can be chaotic for both licensors and licensees."); Pamela Samuelson, *Intellectual Property and Contract Law for the Information Age: Foreword to a Symposium*, 87 CAL. L. REV. 1, 1-3 (1999) [hereinafter Samuelson, *Symposium*] (UCITA, ex-UCC Art. 2B, is a "model law that aspires to provide a standard set of rules that will regulate transactions in information products and services").

[395] *Compare, e.g.*, UCITA § 2B-208 & cmts. 1, 2, 4 (recognizing the enforceability of mass-market licenses like shrinkwrap licenses and citing *ProCD*, *see id.* cmt. 4, in support), *and* UCITA § 2B-111 & cmt. 4(c), Illus. 1 (supporting a system of manifesting assent similar to most shrinkwrap, clickwrap, and webwrap licenses and providing an example of an on-screen click-on license that would be enforceable), *with* ProCD 86 F.3d at 1448-54 (holding in favor of enforcing shrinkwrap and on-screen click-on mass market licenses). As we will discuss in Part IV.B.3., *infra*, UCITA and *ProCD* have some other characteristics in common, *compare, e.g.*, UCITA § 2B-105 cmt. 4, *and* UCITA at Preface, 9-10, *with* ProCD, 86 F.3d at 1455 (apparently, both UCITA and *ProCD* would allow state contract law to negate federal copyright preemption). It is also worth noting that the *ProCD* court also cites an earlier draft of UCITA to support its holding in favor of recognizing the validity of shrinkwrap licenses, *see* ProCD 86 F.3d at 1452.

[396] *See* UCITA § 2B-208 & cmts. 1, 2, 4; Liu et al., *supra* note 359, at 13.

[397] *See* UCITA § 2B-111 & cmt. 4(c), Illus. 1; Liu et al., *supra* note 359, at 13.

[398] *See* UCITA § 2B-111 cmt. 4(c), Illus. 1.

licenses.[399] Specifically, UCITA allows electronic authentication to take the place of signatures in order to eliminate the possibility that a high-tech contract will not be legally enforced because of traditional Statute of Frauds concerns.[400]

Needless to say, the adoption of UCITA by all fifty states (and the District of Columbia) would mean that the key principles of *ProCD v. Zeidenberg* will be codified and applied in all jurisdictions. Therefore, UCITA would ensure that software licenses – including open source software licenses – can be properly formed using the methods afforded by modern technology.[401]

3. Are the Terms of Open Source Licenses Legally Enforceable?

The recognition of the novel means of contractual assent increasingly used in the 'information age' – while a critical element to the enforcement of open source licenses – would not automatically mean that the *terms* contained in a typical open source license – or for that matter, closed source licenses – would be legally enforceable. There are two reasons for the unenforceability of contractual terms that are worth analyzing in the context of mass-market software licensing (which would include open source licenses): unenforceability on public policy grounds[402] – the relevant 'public policy' being copyright law,[403] and unconscionability.[404]

---

[399] *See* Liu et al., *supra* note 359, at 13-14.

[400] *See* UCITA § 2B-113 (titled: "Legal Recognition of Electronic Record and Authentications."); Liu et al., *supra* note 359, at 13-14.

[401] Somewhat ironically, the passage of UCITA will depend in large part on closed source software producers – including Microsoft, *see* Goodin, *Pulls Support*, *supra* note 386; Samuelson, E-mail, *supra* note 389. Robert W. Gomulkiewicz, a lawyer for Microsoft, has even argued that the use of open source licenses – including the GNU GPL, the BSD licenses, and the NPL/MozPL – and open source software – like the copylefted Linux – would benefit from the passage of UCITA, *see* Gomulkiewicz, *supra* note 367, at 894-95, 899-900.

[402] *See generally* FARNSWORTH, *supra* note 342, § 5.

[403] *See id.* § 5.5 ('public policy' for contract law purposes are often derived from legislation (e.g., the federal Copyright Act) and, presumably, the interpretation of legislation by the courts).

[404] *See generally id.* §§ 4.26-4.28.

This section will explore the relevance of these two issues to open source software licensing. Although these two concerns may lead to the unenforceability of closed source mass-market software licenses, open source licenses should not be adversely affected by concerns over copyright law and unconscionability. This section will also draw analogies to the terms of some non-open source software licenses that provide support for the enforceability of open source licenses.

a. Can Contract Law Usurp Copyright Law?

As noted earlier,[405] the developing law of information technology contracting is highly controversial. The controversy surrounding this newly developed field of law centers around the criticisms directed toward both *ProCD* and UCITA.[406] One of the key reasons for the controversy over both *ProCD* and UCITA is the question of whether or not giving legal validity to shrinkwrap, clickwrap, and webwrap licensing – which, by in large, tends to favor the licensors – will allow software producers to create software licenses that may override traditional principles of copyright law.[407]

The concern that a licensor may attempt to use contract law as a tool to expand his intellectual property protections ***beyond*** what is usually granted under copyright law is an especially troublesome issue when dealing with closed source licenses. Many closed source mass-

[405] *See* discussion *supra* Part IV.B.2.

[406] Some of the criticisms of the *ProCD* decision where cited earlier, *see supra* note __ and accompanying text. For a flavor of the pros and cons of UCITA, *see* Symposium, *Intellectual Property and Contract Law for the Information Age: The Impact of Article 2B of the Uniform Commercial Code on the Future of Information and Commerce*, 87 CAL. L. REV. 1 (1999).

[407] For a small sampling of the scholarship recognizing the potential for conflict between copyright and contract law in a *ProCD*/UCITA world, *see, e.g.*, Covotta & Sergeeff, *supra* note 361, at 44-52; Niva Elkin-Koren, *Copyright Policy and the Limits of Freedom of Contract*, 12 BERKELEY TECH. L.J 93 (1997); Grusd, *supra* note __, at 363-66; Charles R. McManis, *The Privatization (or "Shrink-Wrapping") of American Copyright Law*, 87 CAL. L. REV. 173 (1999); David Nimmer et al., *The Metamorphosis of Contract into Expand*, 87 CAL. L. REV. 17 (1999); Samuelson, *Symposium*, *supra* note 394, at 5-10.

market licenses attempt to place restrictions on the licensee that limit the *user/licensee's* rights under copyright law. For example, the Aldus shrinkwrap license requires that the licensee not "[r]everse-engineer, disassemble, decompile, or make any attempt to discover the source code of the Software."[408] The Aldus License's anti-reverse engineering term flies in the face of the fact that, as a general matter, copyright law does allow reverse engineering of copyrighted materials.[409] Additionally, one could argue that a *ProCD*-style technology licensing law would allow licensors to use contract law in order to scale back or eliminate safe harbors and defenses allowed under copyright law – like the fair use defense – favorable to end users.[410]

Although it may be tempting to think that some of these concerns are merely academic jeremiads,[411] the *ProCD* decision itself has provided critics with cause for alarm. The Seventh Circuit, in *ProCD*, held that ProCD's intellectual property, a CD-ROM telephone directory, was protected from unauthorized reproduction and redistribution because the terms that limited the licensee's unauthorized use were enforceable under contract law.[412] Arguably, the *ProCD* opinion

---

[408] *See* Aldus License, *supra* note 143, at 791.

[409] *See* Atari Games Co. v. Nintendo of America, Inc., 975 F.2d 832 (Fed. Cir. 1992), *and* Sega Enterprises v. Accolade, 977 F.2d 1510 (9th Cir. 1992) (both cases holding that copyright law allows reverse engineering and decompilation of software under most circumstances); Neukom & Gomulkiewicz, *supra* note 121, at 778 (software producers often place terms prohibiting the reverse engineering and decompilation of licensed software despite copyright principles to the contrary).

[410] *See* Minassian, *supra* note 365, at 598-601 (criticizing *ProCD*-style jurisprudence because it endangers rights and privileges that users may have – like fair use and the right to reverse engineer – and thus, as the title of his article states, may lead to the 'death' of copyright law); *see also supra* notes 365, 406, 407 (the articles cited in these notes point out similar concerns). For concerns over UCITA, *see* McManis, *supra* note 407, at 176-77 (the author proposed a motion during an ALI oversight meeting expressing concern over the possibility that UCITA could be used to take away the fair use defense and other copyright principles that are more favorable toward the user/licensee).

[411] *See* Gomulkiewicz, *supra* note 367, at 902 (acknowledging the concerns of scholars and critics "that licenses can limit the user's ability to use the licensed software or information," but dismissing these concerns as being impractical).

[412] *See* ProCD, 86 F.3d at 1448-49, 1450-54; *see generally* 1 NIMMER ON COPYRIGHT § 3.04[B][3][a] (an excellent analysis of what the *ProCD* decision means to copyright law).

– providing quasi-copyright protections via contract law[413] – directly contradicts a Supreme Court opinion, *Feist Publications, Inc. v. Rural Telephone Services Co.*,[414] that held that an ordinary telephone directory ***cannot*** be protected by copyright law because it failed to meet the 'originality' requirement necessary for a work to receive copyright protection.[415] Not only "has the Supreme Court's *Feist* decision been effectively nullified"[416] via *ProCD*, other copyright decisions and principles – such as the fair use defense – have also been 'effectively nullified' if the courts and the legislatures accept *ProCD*-style contractual protection of intellectual property.[417]

According to *Nimmer on Copyright*, "in a world governed by Judge Easterbrook's radical freedom to impose terms by shrinkwrap 'contract,' .... the imagination of shrinkwrap drafters can come close indeed to achieving the type of complete control [over intellectual property] that [copyright law may deny]."[418] In short, the acceptance of *ProCD*/UCITA-style technology licensing law may allow software producers to contract away elements of copyright law that might be more favorable to the user.[419]

---

[413] *See* Covotta & Sergeeff, *supra* note 361, at 49 (the results of *ProCD* can lead to licensing provisions that "closely approximates the result under the Copyright Act"); *see also* Harv. L. Rev. Note, *supra* note 365, at 1949.

[414] 499 U.S. 340 (1991).

[415] *See id.* at 362-63 ("Rural expended sufficient effort to make the white pages directory useful, but insufficient creativity to make it original."); 1 NIMMER ON COPYRIGHT § 3.04[B][3][a], at 3-34.2 ("[*ProCD*] contravenes one of the core policies of the Copyright Act – limiting protection to works that qualify as 'original.'"). It should be noted that meeting the 'originality' requirement is critical to receiving copyright protection, *see* Feist, 499 U.S. at 348.

[416] 1 NIMMER ON COPYRIGHT § 3.04[B][3][a], at 3-34.4.

[417] *See id.* at 3-34.4 to 3-34.11; McManis, *supra* note 407, at 176-77.

[418] 1 NIMMER ON COPYRIGHT § 3.04[B][3][a], at 3-34.4. Or, to put it another way, "[i]f shrink-wrap licenses and click-here contracts are made enforceable ... the copyright industry ... will essentially be able to have its cake and eat it too," *see* McManis, *supra* note 407, at 176.

[419] *See, e.g.*, David Nimmer et al., *supra* note 407, at 20-21 (providing a "[c]autionary [t]ale" of what the future would look like if *ProCD*/UCITA-like licensing law were to lead to the "[d]eath of [c]opyright"). According to Pamela Samuelson, some legal observers have predicted that UCITA "will entirely displace intellectual property law,"

The concerns over a *ProCD*-style usurpation of copyright law would be less worrisome if the courts ruled that ***federal*** copyright law preempted ***state*** contract law.[420] In fact, the lower court opinion in *ProCD* held that – under Section 301 of the Copyright Act – the plaintiff's contract claim was preempted by copyright law.[421] The Seventh Circuit, however, overruled the lower court and held that federal copyright law did not preempt the state contract claims.[422]

The Seventh Circuit has been heavily criticized for its holding on the issue of preemption.[423] The arguments for overturning *ProCD*'s anti-preemption ruling are strong on constitutional,[424] statutory, and policy grounds.[425] Courts, when confronted with licensing terms that take away the end user's rights under copyright law, may hold that the contractual terms are against 'public policy' because those terms attempt to create limitations and protections via contract in ways that have been preempted by federal copyright law.[426] Therefore, even if

---

see Samuelson, *Symposium*, *supra* note 394, at 5 & n. 24 (*citing* Maureen A. O'Rourke, *Copyright Preemption After the* ProCD *Case: A Market-Based Approach*, 12 BERKELEY TECH. L.J. 53, 77-80 (1997)).

[420] *See, e.g., id.* at 40-41, 58-59; Covotta & Sergeeff, *supra* note 361, 48-51, 54.

[421] *See* ProCD, 908 F. Supp. at 656-59; *cf. also* ProCD, 86 F.3d at 1453-54 (discussing the District Court's holding on § 301).

[422] *See* ProCD, 86 F.3d at 1453-55.

[423] The following is a small sample of the criticism directed at the Seventh Circuit for its preemption holding, *see, e.g.*, 1 NIMMER ON COPYRIGHT § 3.04[B][3][a]; Covotta & Sergeeff, *supra* note 361, at 50-54; Grusd, *supra* note 365, at 363-66; McManis, *supra* note 407, at 181-87; Nimmer et al., *supra* note 407, at 40-63. For a pre-*ProCD* criticism of *ProCD*-style preemption analysis – as it relates to 'no reverse engineering' clauses, *see* David A. Rice, *Public Goods, Private Contract and Public Policy: Federal Preemption of Software License Prohibitions Against Reverse Engineering*, 53 U. PITT. L. REV. 543 (1992).

[424] For background on the constitutional aspects of copyright preemption, *see generally* 1 NIMMER ON COPYRIGHT § 1.01[B] (dealing with how the Copyright Act, especially § 301, relates to the Supremacy Clause of the U.S. Constitution).

[425] *See, e.g.*, Covotta & Sergeeff, *supra* note 361, at 50-54; McManis, *supra* note 407, at 181-87; Nimmer et al., *supra* note 407, at 40-63.

[426] At least one case seems to have adopted this line of reasoning, *see* Vault Corp. v. Quaid Software Ltd., 847 F.2d 255, 268 (5th Cir. 1988). It should be noted that the statute involved in *Vault* – a special Louisiana statute that

ProCD's preemption ruling is codified by the states via UCITA,[427] the courts may find against a licensor's terms if those terms attempt to override federal public policy – as represented by the copyright statutes and case law – regarding intellectual property.

What does this mean for open source licenses? Probably not much, because the terms of open source licenses – unlike most closed source licenses – grant rights and privileges to the end user rather than taking them away.[428] The terms of open source licenses – where the copyright holder actually gives up many of the intellectual property rights he his entitled to[429] – can be easily distinguished from the terms of closed source licenses that place restrictions on the end user that are *not* allowed under copyright law.[430] For example, in the court's discussion of preemption, Judge Easterbrook praises terms in closed source licenses that prevent source code from being revealed – even if those attempts at intellectual 'property' protection via contract conflict with

---

allowed the enforceability of shrinkwrap licenses – allowed agreements that had terms that would prohibit reverse engineering, decompilation, and disassembly.

[427] **Note:** It is unclear at this point whether UCITA would allow terms that would attempt to override copyright law in ways favorable to the copyright holder. It seems – especially after the ALI pulled its support for the measure – that a *ProCD*-style principle that would allow contracts to negate copyright preemption would probably be okay under UCITA, *see* UCITA § 2B-105, cmt. 4 (although, on its face, § 2B-105 seems to make contractual terms preemptible by federal copyright law, comment 4 makes it clear that – by drawing a distinction between federal copyright law's creation of 'property rights' as "against all the world" and contract as defining the "rights between [the] parties to the contract" – "copyright (or patent) do not generally preclude or preempt contract law"); *see also* Samuelson, E-mail, *supra* note __ (citing a more recent version of § 2B-105, Prof. Samuelson pointed out to me that UCITA will probably not deviate significantly from *ProCD*'s anti-preemption ruling).

[428] *Cf.* Gomulkiewicz, *supra* note 367, 894-95, 899-900, 902 & n. 61 (supporting the idea that some licenses – like the GNU GPL, the BSD licenses, and the NPL/MozPL – that grant the end user the exclusive rights under copyright, are examples of licenses that "are [not] merely tools to take away rights," but serve as examples of how licensing can be "necessary to convey many affirmative rights as well").

[429] Of course these freedoms are in exchange for accepting certain restrictions, *see* discussion *supra* Part IV.2.A.. However, the restrictive terms of an open source license are different from restrictive terms of a closed source license because open source licensing restrictions are well-within the licensor's rights under copyright, and, unlike closed source licensing restrictions, are not devices to eliminate rights that users may have under copyright law. For more on the restrictive terms of specific (as opposed to generic) open source licenses, *see infra* Part IV.4.

[430] *Compare* Neukom & Gomulkiewicz, *supra* note 121, at 777-78 (closed source licenses with restrictive terms favorable to licensors that may be preempted by copyright law), *with id.* at 783, *and* Gomulkiewicz, *supra* note __, at 894-95, 899-900, 902 & n. 61 (licenses that grant rights to the user).

copyright law.[431] Clearly, the terms of all open source licenses *reveal*, rather than hide, the source code of a program. Therefore, the terms of a generic open source license would not run into the kinds of problems with copyright preemption that a *ProCD*-style closed source license would face because open source licensing terms cannot be seen as attempts to override pro-user copyright principles.

The question of whether or not contractual terms that conflict with copyright law will be negated on public policy grounds – while extremely important to closed source licenses – has no effect on the enforceability of open source licenses. If a *ProCD*-style anti-preemption principle stands up in court, open source licenses are enforceable because *ProCD*/UCITA would allow for generous licenses as well as restrictive ones.[432] Even if the courts overturn *ProCD*'s (and UCITA's) questionable negation of federal copyright preemption of state contract law, the terms of a generic open source license should survive because of its tendency to weaken, rather than strengthen, the open source developer's intellectual property rights.

b. Unconscionability

Another danger with allowing the enforceability of shrinkwrap, clickwrap, and webwrap licenses is the possibility that the terms of the contract are unfair – or 'unconscionable' – in some way.[433] The ease with which assent can be established under a *ProCD*/UCITA regime increases

---

[431] *See* ProCD, 86 F.3d at 1455 (according to Easterbrook, J., "[t]o the extent licenses facilitate distribution as object code while *concealing the source code* (the point of a clause forbidding disassembly)," (emphasis added) shrinkwrap licenses lead to lower prices and other benefits to the consumer). A "clause forbidding disassembly" might conflict with, and thus be preempted by, copyright law, *see supra* notes 409, 410 and accompanying text. Under *Vault*, this clause would be unenforceable because it violates public policy, *see* Vault, 847 F.2d at 268.

[432] *See* ProCD, 86 F.3d at 1455 (the anti-preemption holding applies regardless of "whether a particular license is generous or restrictive"); Gomulkiewicz, *supra* note 367, at 902 (UCITA would benefit licenses that grant rights as well as licenses that places restrictions).

[433] *See* Madison, *supra* note 364, at 1117-21 (recognizing the possibility of unconscionable contracts under a *ProCD*-style regime). Unconscionable contracts, or 'contracts of adhesion,' are prohibited under contract law, *see* UCC § 2-302; Williams v. Walker-Thomas Furniture Co., 350 F.2d 445 (D.C. Cir. 1965). For background on the issue of contractual unconscionability, *see* Todd Rakoff, *Contracts of Adhesion: An Essay in Reconstruction*, 96 HARV. L. REV.

the probability that mass-market information technology licenses are 'contracts of adhesion.'[434]

Because closed source licensing is more likely to impose unfair terms on the user, closed source licenses might be vulnerable to the charge of unconscionability. For example, some of the terms we looked at in the previous section – like the clause prohibiting reverse engineering or fair use – that take away the rights that users have under copyright law, can be characterized as being unfair.[435] UCITA implicitly incorporates these concerns by prohibiting the enforcement of unconscionable licensing agreements.[436]

The enforceability of open source licenses will not be affected by the issue of unconscionability. To reiterate, open source licenses, unlike most closed source licenses, *give* rights to the user rather than taking them away. It would seem bizarre to argue that the right to see the source code, the right to freely redistribute, and other rights granted through open source licenses – even when they are tempered by the restrictive terms of some types of open source licenses – are 'unfair' to the user.

Even if the terms of open source licenses were somehow unfair, it would be extremely unlikely that any mass-market software license – open or closed – will be unenforceable because of unconscionability. According to Michael Madison, in *Legal-Ware: Contract and Copyright in the Digital Age*, even if the terms of these licenses were clearly unconscionable, these sort of claims are rarely upheld by the courts.[437]

1173 (1983).

[434] *See* Madison, *supra* note 364, at 1117-19.

[435] Many of the scenarios described in Part IV.B.3.a. can be characterized as being unenforceable because of unconscionability rather than on public policy grounds, *see, e.g.*, Vault, 847 F.2d at 268 (court ruling license is 'contract of adhesion' because a term prohibited reverse engineering).

[436] *See* UCITA § 2B-208(a)(1) (unconscionable terms do not become a part of the contract).

[437] *See* Madison, *supra* note 364, at 1118 n. 389; *see also* Rakoff, *supra* note 433, at 1193-97.

It is worth emphasizing that *closed* source licenses may undergo close scrutiny by the courts on the grounds of public policy and unconscionability. Additionally, the courts may decide to bypass contractual analysis and go straight to constitutional and statutory interpretation issues like the supremacy of federal copyright law over state contract law. These dangers to the enforceability of closed source mass-market software licenses, however, are unlikely to affect open source software licenses because open source licensing is a means for empowering users rather than taking away their rights.

c. A Few Helpful Analogies to Closed Source Licenses

The enforceability of open source licensing terms are further bolstered by the fact that there are several examples of non-open source licenses that – to some extent – grant to the licensee some of the rights that open source licenses give to the end user. For example, shareware licenses – while not granting access to the source code and prohibiting the creation of derivative works – grant the user the right to (more or less) freely redistribute the shareware software.[438] Shareware licenses – as we have already seen – are enforceable.[439] Therefore – if we can draw an analogy between shareware licenses and open source licenses – the enforceability of open source licenses is bolstered by the enforceability of shareware licenses.

In addition to shareware licenses, there are other licenses – such as source code escrow licenses,[440] licenses that allow users to make multiple copies for personal use, etc.[441] – that grant

[438] *See supra* Part IV.A.4.a.

[439] *See* Storm Impact, 13 F.Supp.2d at 791; Compuserve, 89 F.3d at 1260-61, 1264; *see also* discussions *supra* Parts IV.A.4.a., B.1.a., B.2.a.i.

[440] Source code escrow licenses are licenses where source code is held by a third party, *see* 3 MILGRIM ON LICENSING § 24.14. Source code escrow licenses are used in order to ensure that licensees are allowed to have some level of 'access' to the source code in order to maintain or update software, *see id.* It is also possible to license source code directly, *see, e.g.,* Colleen M. Pouliot, *Checklists*, *in* TECHNOLOGY LICENSING AND LITIGATION 1998, 514 PLI/PAT 123, 125-29. However, these source code licenses are not necessarily open source licenses if they do not meet the full definition of open source licenses. Some of these source code licensing schemes seem to be a response to the

some, but not all of the rights that open source licenses give to the licensee. If these licenses are enforceable, then open source licenses should be enforceable as well.

d. Conclusions

The enforceability of open source licenses would be helped by the 'positive' aspects of *ProCD* and UCITA, but would not be hurt by the 'negative' aspects of either *ProCD* or UCITA. This observation provides additional support to what this paper calls 'the irony of open source licensing':[442] A device for weakening the licensor's intellectual property rights, open source licensing, depends on legal principles that *strengthen* the licensor's intellectual property rights. With *ProCD* and UCITA, the strengthening of intellectual property rights is accomplished by using contract law as a way to override the restraints that copyright law places on the copyright holder's rights over the user.

The irony of open source licensing is further reinforced by the fact that *ProCD*, which bolsters the argument favoring the enforceability of open source licenses, actually praises licensing terms that *prevent* source code from being revealed![443] Perhaps this sort of irony is appropriate. After all, just as Richard Stallman's copylefting turns the tools of the 'software hoarders' into the tools of the software 'liberators,' open source licensing can turn Judge Easterbrook's 'Law of the Horse' – his tongue-and-cheek way of advocating greater intellectual property rights for information technology producers through 'cyberspace law'[444] – into the 'Law of GNU' – the

---

kinds of concerns raised by Pamela Samuelson, in *Modifying Copyrighted Software*, *see* Samuelson, *Modifying*, *supra* note 299, at 206-21.

[441] *See, e.g.,* Neukom & Gomulkiewicz, *supra* note 121, at 783.

[442] *See supra* Part IV.A.2.

[443] *See* ProCD, 86 F.3d at 1455; *see also supra* note 431 and accompanying text.

[444] *See* Frank H. Easterbrook, *Cyberspace and the Law of the Horse*, 1996 U. CHI. LEGAL F. 207. It should be noted that Judge Easterbrook does not like to use the term 'cyberspace law' since, to him, it makes as much sense to call

utilization of legal regimes of intellectual property protection, whether they be copyright law or contract law, in order to advance the aims of the open source movement.

4. Are the Restrictive Terms of Some Types of Open Source Licenses Enforceable?

In the previous sections, we dealt with the question of whether or not open source licenses, in general, are enforceable. Answering this question, however, does not necessarily answer the question of whether or not restrictions specific to certain types of open source licenses are enforceable. As was noted in Part III.C., *supra*, the GNU GPL and the NPL/MozPL had more restrictive terms than the Artistic license and the BSD-style licenses. Both the GNU GPL and the NPL/MozPL restricted the licensee from creating closed source derivative works off of the open source code base.[445] Only the GNU GPL prevented the licensee from integrating closed source software with copylefted open source software.[446]

Are the restrictive terms of the stricter open source licenses enforceable? If we simply accept the logic and the consequences of *ProCD* (and UCITA in its current form), then the answer is a simple yes. Under Judge Easterbrook's reasoning, the principle of contractual freedom as a device for intellectual property protection would mean that the licensee is free to accept – by using the software, clicking 'I accept,' etc. – or reject the terms of the license.[447]

What would happen if *ProCD*-style logic was at least partially overturned on the grounds

---

areas of the law where the questions of fact involves horses, the 'Law of the Horse,' as it does to call areas of law involving 'cyberspace,' the 'Law of Cyberspace,' *see id.*

[445] *See supra* Part III.C.5.

[446] *See id.*

[447] *See supra* note 364 and accompanying text; David Nimmer et al., *supra* note 407, at 20-21 (providing a hypothetical scenario where technology licensing can be used as a means to place onerous conditions on the licensee); *see also* Gomulkiewicz, *supra* note 367, 894-95, 899-900 (suggesting that the enforceability of open source licenses – including the GNU GPL and the NPL/MozPL – would be bolstered by the adoption of UCITA, assuming it closely mirrors the logic of *ProCD*).

of copyright preemption or unconscionability? Even if the courts reject the parts of *ProCD* and UCITA that can be used to enforce restrictive terms in closed source licenses that attempt to override the licensee's rights and safe harbors under copyright law, the restrictive terms of open source licenses will probably remain enforceable. Why? Both the clause that prohibits the creation of closed source derivative works and the clause that prohibits the integration of closed and open source programs arise out of the exclusive derivative works right of the open source developer/licensor.[448] Under the 'bundle of rights' theory of intellectual property, all of the subordinate rights associated with the exclusive derivative works right can be divided up like sticks in a bundle where the copyright holder can then give, take away, or withhold those sub-divided rights.[449] Therefore, since both of the restrictive clauses in the stricter open source licenses are simply devices for retaining some of the sub-divided rights stemming from the developer's larger derivative works right, the restrictive terms of an open source license do not override copyright principles and, therefore, are not preempted by copyright law. Additionally, the restrictive terms of open source licenses cannot be said to unconscionably take away the rights of licensees because the rights granted – which include the right for users to create derivative works – were rights carved out and granted by the copyright holder and were never the user's 'rights' to begin with.[450]

One may wonder at this point whether giving up some of the rights under copyright law – such as allowing the user to create derivative works – means that the licensor has given up *all* of

[448] *See supra* Part IV.A.3. Arguably, the clause prohibiting the integration of closed and open source programs may be covered by the copyright holder's 'collective works' right, *see supra* note 290. The reasoning supporting enforceability, however, should remain the same.

[449] *See* Neukom & Gomulkiewicz, *supra* note 121, at 780. For an excellent explanation of the 'bundle of rights' theory of property, *see* GREGORY S. ALEXANDER, COMMODITY & PROPRIETY: COMPETING VISIONS OF PROPERTY IN AMERICAN THOUGHT 319-23 (1997).

[450] *See supra* Part IV.A. 4. (the user does not have a 'right' to modify software under copyright law).

his rights to restrict the user. Under the 'bundle of rights' theory of intellectual property, the licensor can subdivide one of his exclusive rights and then simultaneously give away and/or retain some of these sub-rights. That is what is happening with the GNU GPL and the NPL/MozPL: The open source developer is giving up many of the subset of rights under the superset of the exclusive derivative works right, but is retaining some of the subset of rights under the derivative works right necessary to prevent the user from blurring the lines between the open source and closed source worlds.[451] Thus, regardless of whether or not *ProCD* is upheld in the future, the open source developer is free, in true contractarian fashion, to place the kind of restrictions that the GNU GPL and the NPL/MozPL place on licensees.

V. THE 'ENFORCEABILITY' OF OPEN SOURCE LICENSES WITHOUT THE LAW

The preceding sections went through the steps necessary to show that open source licenses are legally enforceable. We concluded that: a.) Open source licenses, in general, are enforceable; and b.) The restrictive terms of open source licenses – particularly the stricter licenses like the GNU GPL and the NPL/MozPL – are legally enforceable as well.

However, all of this begs the question: Even if these licenses are legally enforceable, how effective are the more lenient licenses – like the Artistic and the BSD-style licenses – in maintaining the viability of the open source software development model? While an affirmative answer to the question of legal enforceability may be helpful in maintaining open source development for those who use stricter licenses, it does little to explain how open source licenses

[451] Further support for this argument, beyond the usual 'bundle of rights' theory arguments, comes from analogizing this situation to shareware licenses. In *Storm Impact*, the shareware developers allowed users to freely redistribute their software – which means that the developer surrendered some part of his exclusive distribution right, *see* Storm Impact, 13 F.Supp.2d at 785-86, 790-91. However, the developers did place some restrictions on the distribution of their software – thus placing "express reservations [on their] distribution rights," *Id.* at 791; *see id.* at 785-86, 790-91. The *Storm Impact* court held that the "express reservations of distribution rights are valid[] [and] enforceable," *Id.* at 791. Therefore, if we draw an analogy between open source licenses and shareware licenses, the restrictive terms that allow the creation of derivative works while, at the *same* time, placing restrictions on the creation of those derivative works (e.g., they have to be open source) are legally enforceable.

with only limited legal provisions to prevent the proprietization of derivative code can succeed in sustaining a viable open source community.[452]

This section will attempt to provide some possible explanations as to how the 'spirit' of open source licensing – maintaining open code – can be 'enforced' without the 'letter' of the law.

A. The Culture of the Open Source Hacker

The importance of cultural norms to behavior in cyberspace is not a new idea. Lawrence Lessig, in *The Constitution of Code*, listed "norms" – along with "law" and "code" – as a way of regulating cyberspace behavior.[453]

Cultural norms have also been seen as a way of explaining the behavior of people in "real space"[454] in both the presence and the absence of law.[455] For example, in Robert C. Ellickson's now-classic *Order Without Law*, Prof. Ellickson describes and analyzes how the residents of Shasta County, California, are able to settle their disputes in the absence of law – or when law is simply in the background – due largely to the set of cultural norms that have evolved to settle these disputes.[456] Extra-legal cultural norms, therefore, provides one possible analytical approach to answering the question of whether or not the culture of the open source movement, as partially expressed through the various licenses, can be enough to keep derivative code open even without

[452] If the reader will recall, the BSD and the Artistic licenses allow the licensee to 'close-off' code derived from the original open source code base, *see supra* Part III.C.5.

[453] *See* Lessig, *The Constitution of Code*, *supra* note 37, at 183.

[454] I borrowed this term from Prof. Lessig, *see id.* at 181.

[455] *See id.* at 181 ("But not only law regulates ['real space']. Social norms also regulate."). For an overview of the current scholarship dealing with the relationship between law and norms, *see* Symposium, *Law, Economics, & Norms*, 144 U. PA. L. REV. 1643 (1996).

[456] *See* ROBERT C. ELLICKSON, ORDER WITHOUT LAW: HOW NEIGHBORS SETTLE DISPUTES 57 (1991) ("Not only are most trespass disputes in Shasta County resolved according to *extralegal* rules, but most enforcement actions are also extralegal." [emphasis added]).

express legal provisions to accomplish that goal.

The culture of the open source movement is directly descended from a pre-existing 'hacker'[457] culture that emanated from places like the MIT AI Lab.[458] The 'Hacker Ethic' – the cornerstone of hacker culture – is based on "[t]he belief that information-sharing is a powerful positive good, and that it is [the] ethical duty of hackers to share their expertise by writing free [meaning open source] software and facilitating access to information and to computing resources wherever possible."[459] To put it another way, hacker culture is based on the idea that "[a]ll information should be free."[460]

With this kind of cultural backdrop, it is easy to see how open source developers – as hackers – continue to maintain the goals of the open source movement by keeping their derivative works open even when they are not forced to do so.[461] Like the good people of Shasta County, open source developers accept the information 'freeing' aspects of the Hacker Ethic as a necessary form of extra-legal social control in order to maintain the uniqueness of being a 'true'

[457] By the term 'hacker,' one should not confuse this term with the meaning often given to that word – someone who maliciously tampers with computer systems and/or software. The proper term for someone who maliciously tampers with computer systems 'cracker.' A 'hacker' – for the purposes of this paper – is used to describe someone who "enthusiastically ... enjoys programming," Raymond, TNHD, *supra* note 15, at 233. For an enlightening discussion of the meanings of, and the differences between, the words 'cracker' and 'hacker', *see* Raymond, TNHD, *supra* note 15, at 130 (defining 'cracker'), 233-34 (defining 'hacker').

[458] *See* Eric S. Raymond, *A Brief History of Hackerdom*, *in* OPEN SOURCES, *supra* note 62, 19 (drawing a direct relationship between the early MIT hackers and the current open source developer hackers). For an excellent historical account of the roots of hacker culture, *see generally* LEVY, *supra* note 67, at 15-152, 413-30.

[459] Raymond, TNHD, *supra* note 15, at 234.

[460] LEVY, *supra* note 67, at 40.

[461] *See, e.g.*, Lash, *Source Code for the Masses*, *supra* note 13 (one group of software developers who use open source code in their products claim that – while they are "not legally required to share" their derivative code (since they are modifying Apache Web Server code, which is under the lenient Apache/BSD-style license) – they feel a "moral obligation" to share their work); *see also* Laird & Soraiz, *supra* note 181 (programmers still contribute to FreeBSD even though they are not legally obligated to do so).

hacker.[462]

Even beyond its hacker roots, the open source movement has also developed other cultural norms in order to encourage even those who are using lenient licenses to contribute to the open source community. One of the key cultural norms of the open source movement – the coalescing of shared code to form a unified piece of open source software – is described in Eric S. Raymond's *The Cathedral and the Bazaar*.[463]

The basic idea behind *The Cathedral and the Bazaar* is this: Traditional (closed source) software development was based on the idea that, like building a cathedral, software could be developed by a few select group of programmers – isolated from the larger community of programmers – and, thus, not be able to share with, or receive input from, other programmers.[464] The inability to get 'peer review' often leads to the development of software of poor quality and functionality.[465] With the open source development model, however, software development is like a "a great babbling bazaar"[466] – where programmers can freely exchange source code and ideals with each other in order to develop a piece of software.[467] Like a bazaar – an unfettered market guided by the Smithian 'invisible hand' that coalesces individual motivations to achieve an exchange that is beneficial to society[468] – the "great babbling bazaar of differing agendas and

---

[462] *See, e.g.*, Raymond, TNHD, *supra* note 15, at 234 (subscribing to the Hacker Ethic is one way of being rightfully called a 'hacker').

[463] For the importance of Eric Raymond's article to the open source community, *see supra* Part II.B.3. & note __.

[464] *See* Raymond, *The Cathedral and the Bazaar*, *supra* note 112.

[465] *See* Raymond, *Evangelists*, *supra* note 242, at 48; Raymond, *Quality*, *supra* note 300.

[466] Raymond, *The Cathedral and the Bazaar*, *supra* note 112.

[467] *See id.*

[468] This allusion to Adam Smith is mine. For a description of how Adam Smith's idea of an 'invisible hand'

approaches" that is open source software development leads to "a coherent and stable system [that] could seemingly emerge only by a succession of miracles."[469]

The cultural norms embodied in *The Cathedral and the Bazaar* means that open source developers will have two additional reasons to keep their code open even under permissive licenses like the BSD and the Artistic licenses. First, if a developer wants to see a project done taking advantage of the 'bazaar' rather than the 'cathedral' model of development, that developer has to contribute code back into the community.[470] Open source activists often describe this situation as a 'gift culture':[471] "[Eric] Raymond compares hacker culture to the culture of the Native American tribes of the Pacific Northwest .... In these tribes ... the central social event was the *potlatch*, where tribal chiefs would gain status – and recruit new tribe members – by lavishing gifts and feasts on neighboring tribes."[472]

This situation with the Native American tribes is eerily similar to the situation in *Order Without Law*, where the residents of Shasta County had a norm of reciprocal gift-giving as a way to create incentives to accomplish collaborative projects.[473] This 'reciprocity' – or 'Even-Up' –

---

works to tame individual self-interest in order to achieve exchanges that benefit society as a whole, *see, e.g.*, TODD G. BUCHHOLZ, NEW IDEAS FROM DEAD ECONOMISTS 19-25 (1989).

[469] Raymond, *The Cathedral and the Bazaar*, *supra* note 112.

[470] *See, e.g.*, Jakob 'sparky' Kaivo, *Giving Back*, FRESHMEAT.NET, (Apr. 1, 1999) <http://www.freshmeat.net/news/1999/04/01/922945388.html> (encouraging people to 'give back' to the open source community by contributing code to open source software projects; the author points out as one of the reasons for contributing to open source projects is to recognize that people would not be able to enjoy the benefits of open source software if some group of people had not contributed back to the community).

[471] *See* McHugh, *supra* note 5, at 99; *see also* Robert Levin, *Agalmics: The Marginalization of Scarcity*, (Mar. 30, 1999) <http://agalmics.nu/> (discussing 'algamics' – as defined by the paper's author, the study and practice of 'gift culture' – and how it relates to the culture of the Internet and the open source movement).

[472] McHugh, *supra* note 5, at 99.

[473] *See* ELLICKSON, *supra* note 456, at 77-78 (the residents of Shasta County often shared in the costs of building boundary fences; these "mutually undertaken boundary-fence project[s]" are "similar to an exchange of gifts, which helps to maintain cooperative interneighbor relations").

strategy, a slight take off on Robert Axelrod's 'Tit-for-Tat' strategy,[474] is where people are expected to reciprocate what other people have given to or done for them.[475] The aim of the reciprocity strategy is to enable a gift-giver to create a sense of obligation in the recipient,[476] and to encourage cooperation among members of a community.[477]

Like the Native American tribes – and like the people of Shasta County – open source programmers have an incentive to contribute code back to the community – even if the license does not require it – because the only way open source software can be developed is by participating in the 'gift culture' of the open source community. This 'gift culture' is premised on the strategy of reciprocity: Like a Smithian marketplace, the open source bazaar is able to harness individual efforts, ideas, and motivations in order to create a unified piece of software as a result of the incentives created by the norm of reciprocity practiced within the open source community.

The other reason why derivative code might stay open even without the strict licensing provisions of the GNU GPL or the NPL/MozPL is the fact that open source developers gain tremendous status within the open source community by contributing to the 'bazaar.'[478] Again, like Eric Raymond's Native American tribes example, obtaining prestige in exchange for contributing code can be seen as a rational motivation for contributing to the community even

---

[474] *See generally* ROBERT AXELROD, THE EVOLUTION OF COOPERATION (1984).

[475] *See* ELLICKSON, *supra* note 456, at 225-29.

[476] *See id.* at 154.

[477] *See id.* at 225-29; AXELROD, *supra* note 474, at 55-69, 118-20, 136-39.

[478] *See* Wayner, *Glory*, *supra* note 262 (having one's contribution be included in an open source software project is considered to be a tremendous honor among open source programmers); Economist, *Hackers Rule*, *supra* note 159, at 63 ("The programmers are motivated not chiefly by money, but by reputation. It is a coup to write 'patches' that pass the scrutiny of fellow hackers and get incorporated in the next version of a program.").

without a legal reason to do so.[479]

Therefore, we can conclude that the extra-legal cultural norms of the open source community – while not a perfect substitute for legal restrictions[480] – does provide a mechanism to check the desire to proprietize code derived from an open source code base rather than contributing the derivative code back into the open source community.

B. Open Source Hacker Culture under the Shadow of the Law

Utilizing a cultural norms explanation for the 'enforceability' of the spirit of open source licenses – even when such licenses lack express restrictions – does not mean, however, that the presence of law or legal language is irrelevant to the norms-based enforcement mechanisms of the open source community.[481] The best example of how the terms of even lenient open source licenses can help to induce compliance with open source cultural norms is the Apache License.

As was discussed in Part III.C.3.a, *supra*, the Apache License is a slight variation on the generic BSD-style license. Although – like other BSD-style licenses – the Apache License does not contain a term that would force a licensee to keep derivative code open source,[482] the Apache License makes it clear that the Apache Group welcomes and encourages "voluntary contributions" – or code 'check-ins' – to the development of the Apache Web Server.[483] This kind

---

[479] *See* Wayner, *Glory*, *supra* note 262; McHugh, *supra* note 5, at 99. For a more theoretical examination of the importance of 'reputation' for a rational actor, *see generally* REPUTATION: STUDIES IN THE VOLUNTARY ELICITATION OF GOOD CONDUCT (Daniel B. Klein ed., 1997).

[480] *See infra* Part V.C.

[481] *See, e.g.*, Berman, *Linux Legal*, *supra* note 39 (Ralph Levien, the developer of the open source graphics program, GIMP, believes that using an open source license is a "cultural marker showing you have a real commitment to free software").

[482] *See supra* Part III.C.5.

[483] *See* discussion *supra* Part III.C.3.a; *see also* Apache License, *supra* note 107, at Coda; Halloween I, *supra* note 17.

of language is important because, like other open source projects, the Apache Group depends on the voluntary contributions of software 'patches'[484] – which is how Apache got its name[485] – in order to maintain and improve the Apache Web Server code.

The incentives created by the legal language of the Apache License[486] serves to reinforce the cultural mechanisms discussed in the previous section. For example, although the licensee is free to close the code derived from the Apache code base, the licensee would not be able to benefit from the goodwill built up around the Apache name,[487] nor can the licensee avoid giving credit to the authors of the Apache code,[488] regardless of the choice the licensee makes. Considering these conditions in light of the Apache License's express encouragement of "voluntary contributions," the language of the Apache License may provide additional incentives for the licensees to participate in the 'gift culture' of the open source community.

In the case of Apache, the cultural pressure to contribute back to the open source 'bazaar' – helped by the language of the license – seems to have worked. Numerous contributors – including companies like IBM – have made significant contributions to the building up of Apache code.[489] These contributions have been made despite the fact that there is no legal obligation to do

---

[484] For the purposes of open source software development, a 'patch' simply means code added on to the existing body of code in order to improve the overall program or to fix a bug.

[485] *See* Computer & Internet Dictionary, *supra* note 7, at 21 (named 'Apache' because "it was developed from existing NCSA code plus various patches" which led to it being called "*a patchy server*" [emphasis in original]); *see also* Larry Seltzer, *Key Open Source Projects*, *supra* note 19, at 172 (ditto).

[486] The potential incentives created by the licensing terms were discussed in Part III.C.3.a., *supra*.

[487] *See supra* Part III.C.3.a. (cannot use the names "Apache Server" or "Apache Group" without the permission of the Apache Group). Being able to take advantage of the goodwill of the Apache name is important because of the wide spread adoption of the Apache Web Server, *see supra* Part II.B.2.

[488] *See supra* Part III.C.3.a.

[489] *See* Andrew Leonard, *Open Season*, WIRED, May 1999, at 140, 143 [hereinafter Leonard, *Open Season*]; McHugh, *supra* note 5, at 95, 100; Stephen Somogyi, *Free Enterprise*, WIRED, May 1999, at 146, 148 [hereinafter Somogyi, *Free Enterprise*].

so.[490] It should also be noted that – unlike the various versions of BSD – the Apache code base has yet to suffer the problem of 'code forking'[491] – largely due to extra-legal pressures like cultural norms and the practical constraint of making code work.[492]

Although the more lenient licenses – like the Artistic and the BSD-style (including Apache) Licenses – may not expressly forbid the closing of derivative code, the language and the general tenor of these licenses may serve as a guidepost to help ensure that licensees are reminded of the spirit that these open source licenses embody – the spirit of the hacker culture.

C. Reality Check

Despite the significant role that cultural norms play in providing an extra-legal enforcement mechanism for open source licenses, a savvy open source developer that wants to ensure that any code that is derived from his open source code base remains open and available to all should use a license that prohibits the closing of derivative code.[493] There is considerable pressure to close off derivative code and not return it to the open source community.[494] The best way to counteract such pressures – especially as open source software steps out of the world of hackers into the world of commerce – is to utilize law as a way of cementing the cultural norms and values that the

---

[490] *See supra* Part III.C.5; Somogyi, *Free Enterprise*, *supra* note 489, at 148. **Note:** IBM may have some sort of agreement – beyond the usual Apache License – that would force them to contribute code back into the Apache development community, *see* McHugh, *supra* note 5, at 95. Even if there is no formal agreement, there may well be reliance, *see* RESTATEMENT (SECOND) OF CONTRACTS § 90, on the part of the Apache Group that IBM – in exchange for using the Apache Web Server in IBM's WebSphere product, *see id.*; Somogyi, *Free Enterprise*, *supra* note 489, at 148 – will contribute code back into the open source code base.

[491] *See* discussion *supra* Part III.D. (dealing with the problem of code forking); *see also* Economist, *Hackers Rule*, *supra* note 159, at 64 (code forking has been a problem with the various versions of BSD Unix – leading to incompatible versions of programs that arose out of the *same* code base).

[492] *See* Somogyi, *Free Enterprise*, *supra* note 489, at 148, 151 (the author giving his opinion on why he thinks Apache has not suffered from code forking).

[493] *See* discussion *supra* Part III.D.

[494] *See, e.g.*, Lash, *Source Code for the Masses*, *supra* note 13 (some developers who develop products based on open source software do not always give back to the community in order to profit from proprietary derivative works).

open source community cherishes.

In addition to the possibility of derivative code becoming closed source, the danger of 'code forking' – with the notable exception of Apache – is more prevalent with the more lenient licenses like the BSD-style licenses. In fact, one of the main advantages of the stricter licenses, like the GNU GPL, is – as Linus Torvalds, the creator of Linux, pointed out[495] – that the risk of 'code forking' is significantly reduced because of the licensing terms preventing the proprietization of derivative code.[496]

While the choice is ultimately up to the open source developer, the concerns about code forking and about maintaining the viability of the open source community should be a part of the decision-making process when choosing an open source license.[497]

VI. WHY USE AN OPEN SOURCE LICENSE?

In the Introduction, this paper posed the following question: Why would anyone want to deliberately give up important intellectual property rights? In other words, why would anyone want to use the open source development model?

Part V. provided a partial answer to this question: The open source community has a strong set of cultural norms that encourages the sharing of intellectual property. This section will provide some additional reasons for adopting the open source model through the legal device of an open source license.

---

[495] *See* Eric Luening, *Linux Creator: We Will Crush Microsoft*, NEWS.COM, (Apr. 19, 1999), <http://www.news.com/News/Item/0,4,35328,00.html> [hereinafter Luening, *Linux Creator*] (Linus Torvalds dismissed the idea that forking was a problem because any modifications to Linux – which is copylefted – must be kept open source).

[496] One can speculate that the GNU GPL's restrictions against the integration of open and closed source codes may also help to deter code forking.

[497] *Cf.* Perens, *The Open Source Definition*, *supra* note 43, at 185 (listing the steps that an open source developer should go through in order to decide which license is best for him).

A. The Benefits of Loosening Intellectual Property Protections

Our usual understanding of intellectual property rights is based on the idea that providing authors with exclusive rights in their creative works provides the best set of incentives for the promotion of the "Progress of Science and the Useful Arts."[498] However, there is a countervailing understanding that intellectual property protections that are too strong may hinder innovation and progress.[499] As noted earlier, Richard Stallman, and the 'free software movement' element of the open source community, whole-heartedly agree with the latter set of views.

However, we need not adopt a Stallman-esque loathing of intellectual property protections for information technology in order to recognize that there may well be times when the loosening of intellectual property protections is a good idea: First, it may well be economically efficient and profitable for software producers to loosen their exclusive distribution (and reproduction) rights under copyright law by either *give away* free samples[500] or implicitly allowing

[498] U.S. CONST. art. I, § 8 cl. 8; *see, e.g.*, Sony Corp. of America v. Universal City Studios, Inc., 464 U.S. 417, 477 (1984); POSNER, *supra* note 10, § 3.3, at 43; Easterbrook, *supra* note 444.

[499] *See, e.g.*, Campbell v. Acuff-Rose Music, Inc., 510 U.S. 569, 575 (1994) (recognizing the tension between the need to protect copyrighted material and the need for others to be able to build upon pre-existing work in order to promote progress and innovation); Rochelle Cooper Dreyfuss, *Do You Want to Know a Trade Secret? How Article 2B Will Make Licensing Trade Secrets Easier (But Innovation More Difficult)*, 87 CAL. L. REV. 191 (1999) (arguing that the strengthening of intellectual property protections via UCITA – ex-UCC Art. 2B – may mean that innovation is hindered); I. Trotter Hardy, *Project Looking Forward: Sketching the Future of Copyright in a Networked World*, U.S. COPYRIGHT OFFICE (May 1998) <http://lcweb.loc.gov/copyright/cpypub/thardy.pdf> (arguing, in a report prepared for the U.S. Copyright Office, that there has been, historically, an over-reaction to the perceived threats to intellectual property rights – often resulting in 'too' strong a level of intellectual property protections via copyright law); Dennis S. Karjala, *Copyright, Computer Software, and the New Protectionism*, 28 JURIMETRICS J. 33 (1987) (arguing that the copyright protections provided to makers of computer software exceed the level of protection beyond that necessary for optimal social policy). There is also a school of thought arguing that – regardless of the merits of intellectual property protections for information technology – intellectual property laws in the 'Digital Age' are largely futile and not worth the effort of enforcing them, *see, e.g.*, Stewart Alsop, *Copyright Protection Is for Dinosaurs*, FORTUNE, Apr. 26, 1999, at 399, 399-400.

[500] *See* SHAPIRO & VARIAN, *supra* note 106, at 84-87; *see also* Carl Shapiro & Hal R. Varian, *The Logic of the Free Version*, HARV. BUS. REV., Nov.-Dec. 1998, at 108, 108-109. This is the basic idea behind shareware and shareware licenses, *see* Storm Impact, 13 F.Supp.2d at 785-86.

people to make 'unauthorized' copies.[501] Second, as Pamela Samuelson recognized in *Modifying Copyrighted Software*, there may well be some normative reasons – like being able to repair flaws and add new functions[502] – for the copyright holder to loosen his/her derivative works right.[503] Finally, there may well be a segment of creative individuals – such as artists – that may want to 'give away' their work, or allow for collaborative contributions by the public, but still be able to maintain some sort of artistic control over how their creative works are utilized.[504]

Open source licensing is an ideal way of loosening intellectual property rights under the circumstances mentioned above. A copyright holder can benefit from giving away some of the rights from the bundle of rights while not losing the benefits of utilizing the law, including retaining some controls to ensure that the goals behind giving away certain rights are met, rather than taking a chance with extralegal norms.

B. Quality Assurance Through Peer Review

In the world of science and engineering, the utilization of 'peer review' is generally seen as a way of reducing the risks that erroneous results will be accepted as 'fact' by society at large. As Eric Raymond correctly points out, "Physicists do not hide their experimental plans from each

---

[501] *See* Alsop, *supra* note 499, at 399 (Stewart Alsop provides an example of how his technology industry newsletter business benefitted from 'illicit' copying and redistribution). The 'Even-Up'/reciprocate strategy, discussed in Part V, *supra*, may also support the idea that it may be better, at times, to not strongly enforce one's reproduction and distribution rights, *see* ELLICKSON, *supra* note 456, at 258-64 (describing how academic publishers often turn a blind eye to illicit copying and redistribution in the academy because "professors, like the residents of rural Shasta County, know how to get even").

[502] *See* Samuelson, *Modifying*, *supra* note 299, at 179-80.

[503] *See supra* Part IV.A.4.

[504] *See* Heffan, *supra* note 9, at 1488-89, 1511, 1513-14 (this is one of Ira Heffan's normative rationales for the use of the GNU GPL). The utilization of copyright licenses – like open source licenses – may be necessary for artists to retain some modicum of artistic control because of the fact that current U.S. law, generally, does not recognize an artist's 'moral rights' (*droit moral*), *see generally* 3 NIMMER ON COPYRIGHT, *supra* note __, at § 8D; *but cf.* Gilliam, 538 F.2d at 18-26 (although the court notes that American law does not recognize 'moral rights,' the court ruled in favor of the creators of the BBC comedy program, *Monty Python*, by arguing that American courts have utilized other theories in the past to protect the integrity of an author's work).

other; instead they, skeptically check each others' works."[505]

Just as peer review helps in the Academy or in the field of engineering, the peer review of code that one can get with the open source model of development – via the open source license – can provide a cost-effective[506] way to improve the quality and the reliability of a program.[507] Closed source programs – generally – do not benefit from peer review. This often means that a typical commercial computer program has a number of quality problems associated with it.[508] With open source programs, however, the accessibility of source code means that quality problems that might have gone unnoticed within the confines of a software company, will become noticed when evaluated by the objective eyes of the programming public.[509]

The prospect that software can be created without the kind of bugs, crashes, and other quality problems that many users have come to expect, may be reason enough to support the open source model of development and the use of open source licensing.

C. Improving the Development Process

From the perspective of a commercial software producer, another reason for utilizing open source licensing and the open source model of development is the positive effects that opening up code has on the development process. In addition to the improvements in the quality and

---

[505] Raymond, *Quality*, *supra* note 300.

[506] It is cost-effective since open source development often means that the reviewers of code are doing their jobs free of charge, *see* Leonard, *Open Season*, *supra* note 489, at 142.

[507] *See, e.g.*, Economist, *Hackers Rule*, *supra* note 159, at 63-64; Levy, *Code Warriors*, *supra* note 110, at 60-61; Raymond, *Evangelists*, *supra* note 242, at 48; Raymond, *Quality*, *supra* note 300.

[508] *See* Economist, *Hackers Rule*, *supra* note 159, at 64; Raymond, *Quality*, *supra* note 300.

[509] *See* Raymond, *The Cathedral and The Bazaar*, *supra* note 112 ("Given enough eyeballs, all bugs are shallow."). An excellent example of a real life scenario where the need for reliability and stability necessitate the use of the open source model of development is NASA. NASA – for obvious reasons – need software that has near perfect reliability. NASA, therefore, utilizes open source software rather than closed source software. For a retelling of this story, *see* Robert Young, *Giving it Away: How Red Hat Software Stumbled Across a New Economic Model and Helped Improve an Industry*, *in* OPEN SOURCES, *supra* note 62, 113, 120.

reliability of software, the open source model enables a producer to tap into a virtually limitless pool of low-cost labor: the open source development community.[510]

Another benefit that the open source model of development brings to the software development process is the fact that any improvements and modifications leading to a finished product is often based on stable and widely accepted technology. Consider Apache: As we noted earlier, Apache is the web server software that runs on most of the web sites on the World Wide Web.[511] Any product that is based on the Apache Web Server will be able to take advantage of all of the intellectual capital pored into – in the form of source code – and all of the goodwill associated with Apache – so long as the licensee abides by the terms of the Apache license and honors the cultural norms associated with the open source community.[512]

VII. CONCLUSION: "DO YOU BELIEVE IN THE INTERNET?"

This paper has argued that open source licenses – whether they are lenient, like the BSD and the Artistic licenses, or they are strict, like the GNU GPL and the NPL/MozPL – are legally enforceable. This paper has also discussed how the open source movement has cultural norms and qualitative advantages that make the open source development model viable, with or without the law.

One may ask, however, whether any of this really works. Yes, open source licenses are enforceable – legally and extra-legally. Yes, there are benefits in utilizing open source software, the open source model, and open source licensing. But, in a world dominated by closed source

---

[510] *See* CUSUMANO & YOFFIE, *supra* note 11, at 321 ("This virtual workforce is huge and far exceeds Microsoft's cast of developers."); Leonard, *Open Season*, *supra* note 489, at 142 ("There's never been a more cost-effective way to contract out services than to use this type of free labor ...").

[511] *See supra* note 107.

[512] For an example of a company that is basing their business on the kind of idea described in the text, *see* Somogyi, *Free Enterprise*, *supra* note 489, at 147 (reporting on Cobalt Networks, which bases its turn-key web server on Linux and Apache because of their stability and the momentum of goodwill built around open source software).

producers like Microsoft, can the open source model work as advertised?

The answer to this question is the same as the answer to the following question: "Do you believe in the Internet?"[513] Microsoft – a company that apparently 'believes' in the Internet – had this to say about the viability of open source software: "The largest case study of [open source software] is the Internet. Most of the earliest code on the Internet was and is still based on [open source software] ..."[514]

Open source software works. It may not be obvious as we search for information on Yahoo! (running on FreeBSD), send e-mail (probably on an e-mail server running on Sendmail) to a Hotmail account (running on Apache), type in "amazon.com" instead of "208.216.182.15" (thanks to BIND), or read the *Harvard Law Record* (running on Linux), but those of us who use and believe in the Internet *depend* on the viability of open source software and open source licensing everyday.

[513] Levy, *Code Warriors*, *supra* note 110, at 60 (Tim O'Reilly posed this question when asked how he responds when people ask him about the credibility of open source software).

[514] Halloween I, *supra* note 17.

APPENDIX A.: THE OPEN LITERATURE LICENSE

As noted in the credits, this paper is licensed under *both* the GNU GPL and the Open Literature License. The Open Literature License is an attempt by the author of this paper to create an open source license – inspired and influenced by the open source software licenses analyzed in this paper – that is more specific to literary, academic, dramatic, and other creative works that may differ significantly from computer software. The following is the full text of the Open Literature License:

The Open Literature License, Draft Version 1.0, April 28, 1999

Everyone is permitted to copy and distribute verbatim copies of this license, *see infra* Clause 2, so long as the integrity of the author's work is respected, *see infra* Clause 3. If anyone has suggestions for improving this license, you can contact the creator of this license at <grokopen@email.com>.

1. Preamble

Authors of literary, dramatic, and other creative works, may feel that unfettered accessibility to their works may be beneficial for both themselves and for society at large. Authors may also feel that a collaborative, community-based process of development may be in the best interest of everyone concerned. Some possible examples of these sorts of scenarios are scholarly settings (e.g., peer review), interactive art or theater, and papers that attempt to influence policy and democratic discourse (which may require feedback from the public as well as the ability to make modifications to such a work in order to reflect that feedback).

An author in these types of situations could place his/her work in the public domain or simply not enforce his/her intellectual property rights. However, while this may provide the intended benefits to well-intentioned members of the public, this may allow less scrupulous individuals to obtain control over the author's works in a way that violates the spirit behind the author's actions.

The best way to strike a balance between allowing open access to – and free expression using – the author's works, and the need to maintain some level of control over the use of the work in order to ensure that the original goals are being met, is by utilizing an open source license. The only problem with using an open source license is that most open source licenses were created with computer software in mind, rather than the more traditional copyrighted works like literature, dramatic works, etc. While open source software licenses have been used to achieve the results described above with non-technological creative works, it may be more suitable to use an open source license specifically designed for literary, dramatic, and other 'traditional' works that may or may not be expressed or distributed utilizing information technology.

The Open Literature License is a license designed for creative works that are not automatically associated with information technology. It is suitable for works – such as some types of academic writings, novels, poems, plays, music, etc. – that may be significantly different in character than computer software (of course, computer software can be licensed under any one of the existing open source licenses). Traditional creative works, like literature, drama, etc., may have some issues associated with them that a license covering software does not directly deal with, *see infra* Clause 7.

Some of the other advantages of the Open Literature License is that: (1) It is compatible and consistent with all open source licenses; (2) Allows an author to license his/her works under both the Open Literature License and another open source license; and (3) Does not preclude the author from selling his/her work (Note: This is also true with other open source licenses, *see* 'The Open Source Definition,' Para. 1).

The Open Literature License may not be right for everyone. However, if your goals are consistent with allowing open access to your creative works, the Open Literature License may work for you.

2. Consistency and Compatibility with Existing Open Source Licenses

The licensee may reproduce, redistribute, and modify this work in accordance with the standards for open source licensing as laid out in 'The Open Source Definition,' 'The Debian Free Software Guidelines,' and/or the standards for 'free software' as formulated by the Free Software Foundation, Inc [hereinafter 'the generally accepted definitions of open source licenses']. The Open Literature License is completely and totally consistent with any and all open source licenses. If any term of this license may be interpreted as being inconsistent with the generally accepted definitions of open source licenses, then such a term can either be interpreted as being consistent with the generally accepted definitions, or it can be excised completely – allowing the remaining consistent and compatible terms of the license to survive.

3. Giving Credit Where Credit Is Due and the Integrity of the Author's Work

In accordance with the generally accepted definitions of open source licenses, the licensee must give credit to the author of the licensed work and must honor the integrity of the author's work by clearly and expressly identifying those distributions of modified works as being modifications to the original work, unless the author expressly permits the licensee to redistribute these modifications without giving such credit [this can be done by adding a subclause to this section or through another method that is acceptable to the licensor]. However, this clause does NOT prohibit the distribution or redistribution of modified works; this clause is simply designed to give credit where credit is due and to ensure that licensees do not impinge on the integrity of an author's creation.

4. Derivative Works

This license allows licensees to create modifications and derivative works, and allows modified and derived works to be distributed under this license.

4.1. Default Term: Derivative Works Must Be Kept Open Unless ....

The default term – which means that the licensor can opt out of this term if he/she wants to – is that derivative works (which includes modifications) must be kept 'open' and freely (in terms of being free of barriers rather than 'zero price') available to the public. However, the licensor can opt out of this term [by adding a subclause or by some other means acceptable to the licensor] and, thereby, allow the licensee to 'close-off' access to derivative works. One way in which the licensee can opt out is by 'double licensing' his/her work with this license plus a license that does not prohibit the proprietization of derivative works (such as the BSD License), *see infra* Clause 6 ('Multiple Licenses').

5. Aggregation (Mandatory Term) and Integration (Default Term)

This license allows licensees to aggregate – e.g., placing several works on the same medium of distribution – or integrate – e.g., combining several works at a fundamental level to form a unitary whole – works under this license with works not under this license, whether 'open source' or 'closed source.' Note, however, that the 'aggregation' part of this clause is mandatory – it cannot be opted out of by the licensor; the 'integration' part is default.

5.1. Multiple License Exception

The Clause 5, Integration term is a default term, and, therefore, can be opted out of, *see* description *supra* Clause 4.1. If the licensor 'double licenses' his/her work with this license and another license that prohibits the integration of open and closed source works (e.g., the GNU GPL), the prohibition trumps the default term of this license permitting such integration.

6. Multiple Licenses

This license permits the author/licensor to cover his/her works under this license (the Open Literature License) and another open source license, simultaneously. However, closed source licenses cannot take advantage of this multiple licensing clause.

6.1. A Recommendation: 'Double Licensing'

While the licensor is free to use as many open source licenses as he/she wants in order to cover the same works, it is highly recommended that any multiple licensing scheme be limited to just two licenses: this license and one other open source license. This may be practically and legally necessary since the specific terms of specific open source licenses often conflict with each other (e.g., GNU GPL versus BSD Licenses).

7. Analogy to Open Source Code: Intellectual Property Rights Must Not Be Used to Block Access To Licensed Works (But The Licensor Can Sell His/Her Works)

Since this license is designed to deal specifically with works that are significantly different, from a technical perspective, from computer software – such as literary, dramatic, and artistic works – an open source software license's provisions to make computer source code available is of little

direct consequence. However, we can analogize to the idea of source code availability by making it a condition of this license that any work licensed under the Open Literature License cannot be made inaccessible via the use of intellectual property rights the licensor may have. For example, if William Shakespeare had licensed 'Hamlet' under this license, neither Shakespeare nor his descendants can utilize their exclusive rights as copyright holders to prevent access to 'Hamlet.' This should not be interpreted, however, as meaning that Shakespeare – or any other author who uses this license – cannot sell his works. Under the generally accepted definitions of open source licenses, open source licenses cannot prohibit someone from selling the licensed 'software,' or, in this case, literary/dramatic work, *see, e.g.,* The Open Source Definition, Para. 1.

7.1. Regardless of Statutory Extensions of Time Limitations

This 'open accessibility to works' clause applies regardless of whatever the current time limitations are under intellectual property statutes. Therefore, in our Shakespeare example, Shakespeare and his kin are prevented from using their intellectual property rights to block access to 'Hamlet' – regardless of whether the limits on their copyright runs out at 70, 90, 100, or a 1000 years. Clause 7, in other words, is in effect indefinitely.

8. This License is 'Viral'

This license applies to each recipient and each distributor along the chain of distribution and/or redistribution – including, but not limited to, the original licensor and the original licensee.

9. In Case of Preemption

If any of the clauses or subclauses of this license are preempted by federal copyright (or other intellectual property) law or voided as being against public policy (public policy, in this sort of scenario, might be federal copyright law), then all of the other clauses and subclauses that are not legally negated will continue to be in effect.

10. Disclaimer of Warranty [adapted from the Mozilla PL, Version 1.1]

THE LICENSED WORK IS PROVIDED UNDER THIS LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, WARRANTIES THAT THE LICENSED WORK IS FREE OF DEFECTS, MERCHANTABLE, FIT FOR A PARTICULAR PURPOSE OR NON-INFRINGING. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LICENSED WORK IS WITH YOU. SHOULD ANY LICENSED WORK PROVE DEFECTIVE IN ANY RESPECT, YOU (NOT THE INITIAL AUTHOR/DEVELOPER/ETC. OR ANY OTHER CONTRIBUTOR) ASSUME THE COST OF ANY NECESSARY SERVICING, REPAIR OR CORRECTION. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF ANY LICENSED WORK IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

11. Limitation of Liability [adapted from the Mozilla PL, Version 1.1]

UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT

(INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL YOU, THE INITIAL DEVELOPER, ANY OTHER CONTRIBUTOR, OR ANY DISTRIBUTOR OF THE LICENSED WORK, OR ANY SUPPLIER OF ANY SUCH PARTIES, BE LIABLE TO ANY PERSON FOR ANY INDIRECT, SPECIAL, GENERAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF SUCH PARTY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES.

12. The Licensed Work Does NOT Constitute Legal Advice

ANY WORK, WHETHER IN WHOLE OR IN PART, UNDER THIS LICENSE DOES NOT CONSTITUTE LEGAL ADVICE OF ANY KIND. EVEN IF A WORK UNDER THIS LICENSE DEALS WITH LEGAL MATTERS – INCLUDING, BUT NOT LIMITED TO, THIS LICENSE ITSELF – IT DOES NOT CONSTITUTE, IMPLIEDLY OR EXPRESSLY, LEGAL ADVICE.