15 Tex. Intell. Prop. L.J. 157

Texas Intellectual Property Law Journal
Winter 2007

Article

OPEN SOURCE, FREE SOFTWARE, AND CONTRACTUAL ISSUES

Dr. José J. González de Alaiza Cardona[a1]

Table of Contents

*159 I. Introduction

"Free software" is an increasingly used method of licensing computer programs, which on the one hand gives users the rights to use, modify, and redistribute the program; and, on the other, forces any person redistributing an original or modified version of the program to license it with the same rights. Such a forced obligation is introduced through the "copyleft clause" and uses copyright in a creative way to achieve freedom instead of control.

This paper discusses free software foundations and contractual issues. The discussion is structured in two main parts and a conclusion. In Part II, the emergence of free software and its implications in different fields will be exposed. I will seek to explain how the copyleft clause affects the ways in which software is developed and distributed. Moreover, I will explain the common points and differences between free software and "open source software."

In Part III, the contractual issues raised by the peculiarity of the copyleft clause will be addressed. I will argue that the license agreement which contains the copyleft clause is not a mere non-contractual copyright license, but a contract. This argument triggers a number of contract-related questions which I will seek to resolve from the U.S. perspective. In particular, I will address concerns about lack of consideration; validity of clickwrap and shrinkwrap licenses; possible consequences of lack of privity between licensor and licensee; the enforceability of the warranty disclaimer included in most copyleft licenses; and the relation between copyright and contractual provisions.

Finally, the paper will summarize the main conclusions drawn in the two parts mentioned above.

II. General Part

A. Open Source Numbers

The terms free software and "open source" F[1] are now as familiar to every computer user as "mouse," "keyboard," or "laptop." More importantly, the use of open source software is widespread in different fields.F[2] Its success is particularly ***160** remarkable on the Internet. For example: almost 70% of web servers[3] are run by Apache,[4] a well known open source software package; a vast majority[5] of the domain name servers (DNS)[6] use BIND,[7] another open source software package; the Firefox browser counts over 150 million downloads[8] and the email client Thunderbird reported over one million downloads in the ten days following its release on December 7, 2004.[9] ***161** Beyond the Internet, open source software is often used to control electronic devices, from antilock brakes to watches to consumer electronics (i.e., mobile phones, PDAs and TV set-top boxes) to medical equipment, etc.[10]

A third area where the use of open source software is increasing, but only has a moderate market share, is software for desktop computers.[11] This is true for both operating systems and applications.[12] In relation to the former, GNU/Linux claims around 29 million users,[13] far behind the numbers of the non-open source operating systems provided by Microsoft, i.e., the Windows family (Windows XP, Windows Me, Windows 2000, etc.).[14] Among the most popular open source applications are OpenOffice[15] and MySQL,[16] which also occupy a very small market share in comparison with Microsoft Office.[17]

***162** Increased use of open source software will depend mainly on the support offered by major companies with the power to influence the technological market. On the one hand, Microsoft's fight against the open source movement is well known.[18] On the other hand, a fair number of large companies, such as IBM, Intel, Hewlett-Packard, Oracle, SAP, Sun Microsystems, Dell, Motorola, and Sony are supporting this movement.[19] Another decisive factor in the success of open source will be the attitude of public institutions. So far, the governments of Argentina, Australia, Brazil, China, Denmark, France, Germany, India, Korea, Japan, Peru, the United States, and even the United Nations have either adopted initiatives or are already using open source software.[20]

B. Technical Concepts

Understanding the legal issues related to open source software requires some basic knowledge about computer programming. First, hardware and software must be distinguished. "In information technology, hardware is the physical aspect of computers, telecommunications, and other devices. The term arose as a way to distinguish the 'box' and the electronic circuitry and components of a computer from the program you put in it to make it do things."[21]

Software is a general term for the various kinds of programs used to operate computers and related devices. . . . Software can be thought of as the variable part of a computer and hardware the invariable part. Software is often divided into application software ***163** (programs that do work users are directly interested in) and system software (which includes operating systems and any program that supports application software).[22]

This paper will focus on the legal aspects of exploiting software as open source. But what does "open source" mean?

The source code consists of the programming statements that are created by a programmer with a text editor[23] or a visual programming tool and then saved in a file. For example, a programmer using the C language[24] types in a desired sequence of C language statements using a text editor and then saves them as a named file. This file is said to contain the source code.[25]

The source code can be read and modified by other programmers. However, a computer does not understand these instructions.

To convert the source code into something readable by the computer, the programmer uses a compiler, which is a "special program that processes statements written in a particular programming language and turns them into machine language or 'code' that a computer's processor uses."[26] The resulting output, the compiled file, is known as machine code or object code.[27] In contrast to the source code, the object code file contains a sequence of instructions that the processor can understand or execute but that is almost impossible for a human to read because it consists entirely of numbers. In short, "[s]ource code is what programmers write; object code is what computers run."[28]

***164** In some legal articles, the relationship between source code and object code has been analogized to that between a recipe and a dish. The source code is like a recipe, which one cannot eat, but allows one to cook the dish.[29] On the other hand, if one has only the dish, you may enjoy it, but not improve it, because of the lack of knowledge about its ingredients and quantities.

In the vast majority of purchases and any other acquisitions of software, the object code is delivered without the source code. That may mean little for the typical user, who is only interested in running the program and would not even know what to do with the source code. However, it is easy to realize that if the source code were released, a small number of users who are computer experts could customize or improve the program.[30]

Finally, it may be asked if it is possible to extract the source code from the object code. The answer is in the affirmative, and this can be done through decompilation.[31] However, this process implicates certain technical and legal issues. First, the decompilation of object code into source code is not a straightforward process; on the contrary, it may fail for various reasons.[32] Moreover, a successful decompilation does not provide us with the original source code, but only with a source code which is functionally equivalent to the original[33] and usually much more difficult to maintain. Second, decompilation involves reproductions of the decompiled computer program, or parts of it, and often circumvention of technological-protection measures.[34] Both will most certainly violate intellectual property laws. Although most countries provide exceptions for reverse engineering and, therefore, for decompilation, such exceptions are only applicable under certain circumstances. Basically, this action has to be undertaken ***165** to obtain the information necessary to achieve the interoperability of an independently-created computer program with other programs.[35]

C. Historic Overview

In the 1960s the distinction between hardware and software was not as clear as it is today. The first computers were designed to perform one or a few specific tasks. Early programs were designed by machine manufacturers to be used in conjunction with a specific computer model or individual machine.[36] The users of computers were mainly companies and governmental institutions, which purchased the hardware and received the software as part of the deal.[37] The business model of selling the software did not exist as such.[38] In this scenario, the producers of software usually delivered the program with its source code, because they did not have any interest in hiding it. Moreover, users and computer scientists used to share the source codes of the programs without limitations.[39]

For these reasons, it has been claimed that free software preceded the distribution of software as a protected work.[40] That may have been the situation de facto, but it cannot be stated that users had a right to reproduce the programs or to access the source code without the authorization of the author or producer. It is true that the kind of legal protection deserved by computer programs was still unclear at the time,[41] but even at a very preliminary stage, the consensus was that computer programs were somehow protectable.[42]

***166** The distribution of computer programs independent from a particular machine began to gradually increase.[43] And because the production cost of a program was, and is still today, far beyond that of making copies of the program, a certain legal standard had to be achieved.

In 1980 the U.S. amended the Copyright Act extending its protection to computer programs.[44] This decision probably influenced Japan and the European Union[45] to take steps in the same direction. Finally during the 1990s, WTO[46] and WIPO,[47] the international organizations regulating copyright issues, agreed on the protection of computer programs under copyright law.

***167** At the same time, software developers stopped delivering computer programs in source code. In their view, the exclusive distribution of the object code has two advantages: first, the product is more appealing for the standard, non-sophisticated software users; and second, it protects the source code from disclosure and therefore, from a possible modification.[48]

As a reaction against the distribution of computer programs in object code, Richard Stallman founded in 1985 the Free Software Foundation,[49] which has since then coordinated the efforts of the free software movement.[50] Stallman, who developed operating systems as a researcher at the Massachusetts Institute of Technology (MIT) Artificial Intelligence (AI) Lab, greatly disliked not only the new ways of distributing software but also the transformation in the ways of producing it.[51] Now software developers have to sign a nondisclosure agreement to get access to the software, even as object code.[52]

Stallman advocates that no one should have to pay for software[53] and that access to the source code should be granted.[54] If a different system governs the production and distribution of software, a few people would dominate computing.[55] Stallman compares his free software system with a non-free or proprietary software system and considers the latter antisocial.[56] In opposition to free software, the proprietary software is characterized by the fact that "[i]ts use, redistribution or ***168** modification is prohibited, or requires you to ask for permission, or is restricted so much that you effectively can't do it freely."[57]

Taking into account that the proprietary software was flooding the market, Stallman began to create his own software, which would be controlled under the free movement principles.[58] The first step in order to build a consistent alternative to the proprietary software was to have an operating system,[59] without which he could not have even run a computer.[60] This ongoing operating system was called GNU.[61] FBuilding the operating system from scratch was a huge task, and therefore, Stallman asked at an early stage of the project for help and money.[62]

An important strategic decision in the development of GNU was to make the system compatible with Unix.[63] That had three main advantages: first, Unix design was already proven; second, it was portable;[64] and third, Unix users could easily switch to GNU.[65] The Unix operating system and its source code were delivered by its copyright owner (AT&T) to many academic and research institutions around the world to allow its study, improvement, and enlargement.[66] Unix was not free ***169** software because it was not licensed to everybody, and its redistribution was not free.[67] However, the free software group found in Unix the perfect model to develop its own operating system, replacing component by component.[68] This job took over 5 years, from 1984 to 1990. By then, only one piece of the new operating system that was being built was lacking: the kernel.[69]

The kernel is defined as:

[t]he central module of an operating system. It is the part of the operating system that loads first, and it remains in main memory [or RAM] . . . . [It provides] all the essential services required by other parts of the operating system and applications. Typically, the kernel is responsible for memory management, process and task management, and disk management.[70]

The GNU community was taking longer than expected in the creation of a kernel, but in 1991 a computer science student, Linus Torvalds, released a more modest (i.e., non-portable and often-crashing) operating system which did have its own kernel. Torvalds made this operating system free, and in 1992 the combination of Linux with the almost-complete GNU system resulted in a complete operating system:[71] the GNU/Linux system (or simply Linux).[72]

***170 D. Free Software Foundations and Perspectives**

1. Introduction

The free software philosophy and structure were triggered by practical problems. As an example, Stallman explains his dissatisfaction with the fact that the lack of access to the source code of the MIT AI Lab printer prevented him from adding any convenient features, such as the one that notifies a user when a job has been printed.[73] We must point out that Stallman is a computer scientist and that his concern may be shared exclusively by other computer scientists or highly knowledgeable users--a tiny fraction of computer users. However, if computer scientists have the right to access the source code and, consequently, to fix the bugs and to improve the programs, new ways of developing software and distributing it in society will emerge, as will be shown below.

Free software is a complex phenomenon with deep foundations and broad implications in different fields. In an extremely brief summary, we can say that the foundations of free software have a moral character. However, the pillars of the building are in legal documents that make the realization of free software principles possible. Finally, this legal approach leads to certain forms of software development and economic strategies.[74]

2. Definition

A program is free software if users have four kinds of freedom:

• Freedom 0: The freedom to run the program, for any purpose.

• Freedom 1: The freedom to study how the program works, and adapt it to your needs. (Access to the source code is a precondition for this.)

• Freedom 2: The freedom to redistribute copies so you can help your neighbor.

• Freedom 3: The freedom to improve the program, and release your improvements to the public, so that the whole community benefits. (Access to the source code is a precondition for this.).[75]

***171** 3. Moral Foundations

Stallman's point of departure is that "society needs to encourage the spirit of voluntary cooperation in its citizens. When software owners tell us that helping our neighbors in a natural way is 'piracy,' they pollute our society's civic spirit."[76]

This moral philosophy supports two main free software principles. First, free software must allow access to its source code. Second, free software must allow making copies and redistribution of them. These two principles sit in opposition to proprietary software, that is, software whose "use, redistribution or modification [that implies access to the source code] is prohibited, or requires you to ask for permission, or is restricted so much that you effectively can't do it freely."[77]

The first principle is based upon the idea that society "needs information that is truly available to its citizens--for example, programs that people can read, fix, adapt, and improve, not just operate. But what software owners typically deliver is a black box that we can't study or change."[78] In other words,

[t]he ease of modification of software is one of its great advantages over older technology. But most commercially available software isn't available for modification, even after you buy it . . . . Software development used to be an evolutionary process, where a person would take an existing program and rewrite parts of it for one new feature, and then another person would rewrite parts to add another feature; in some cases, this continued over a period of twenty years. Meanwhile, parts of the program would be 'cannibalized' to form the beginnings of other programs. The existence of owners prevents this kind of

evolution, making it necessary to start from scratch when developing a program.[79]

In relation to the lack of access to the source code, Stallman makes a very relevant point from the copyright point of view: "In any intellectual field, one can reach greater heights by standing on the shoulders of others. But that is no longer generally allowed in the software field--you can only stand on the shoulders of the other people in your own company."[80]

The freedom to make and distribute copies derives from the intangible nature of intellectual works. In opposition to material objects, like cars, chairs, or sandwiches, programs can be reproduced at almost no cost.

It is easy to show that the total contribution of a program to society is reduced by assigning an owner to it. Each potential user of the program, faced with the need to pay to use it, may choose to pay, or may forego use of the program. When a user chooses to pay, this is a zero-sum transfer of wealth between two parties. But each time someone chooses ***172** to forego use of the program, this harms that person without benefiting anyone. The sum of negative numbers and zeros must be negative.[81]

One common misunderstanding about free software is that it is forbidden to make money with its distribution; this is not correct. This misunderstanding can be traced back to the GNU Manifesto, which Stallman started with the following sentence: "GNU . . . is the name for the complete Unix-compatible software system which I am writing so that I can give it away free to everyone who can use it."[82] However, eight years later Stallman made this clarification:

[T]he wording here was careless. The intention was that nobody would have to pay for *permission* to use the GNU system. But the words don't make this clear, and people often interpret them as saying that copies of GNU should always be distributed at little or no charge. That was never the intent; later on, the manifesto mentions the possibility of companies providing the service of distribution for a profit. Subsequently I have learned to distinguish carefully between 'free' in the sense of freedom and 'free' in the sense of price. Free software is software that users have the freedom to distribute and change. Some users may obtain copies at no charge, while others pay to obtain copies--and if the funds help support improving the software, so much the better. The important thing is that everyone who has a copy has the freedom to cooperate with others in using it.[83]

The ambiguity of the term "free" has caused a search for alternatives, like "liberated," "freedom," "open," and "non-proprietary."[84] However, Stallman insists that these other words have either the wrong meaning or some other disadvantage and maintains the original term.[85]

***173** 4. Legal Approach

The legal perspective of free software cannot be underestimated. In the end, the whole free software philosophy is only possible through its legal construction.[86] As it has often been said, the characteristic freedoms of free software seem to clash with copyright. It may be argued that copyright foresees some exceptions in favor of the user of a legal copy to run the program (freedom 0) and to adapt it to the user's needs (freedom 1). However, we will not delve into this discussion, as it is clear that the redistribuition of copies (freedom 2) and the release of modified versions of the program (freedom 3) are only possible under the author's authorization.

An apparently straightforward solution would be to disclaim the copyright on the program which we want to be free. In other words, we would take the necessary steps to place the program in the public domain. However, this would be unsatisfactory in order to achieve the goals of the free software movement. First, a program in the public domain can be released as object code, and that impedes the access to its source code. Second, if no copyrights are attached to the program, even being released as source code, everyone could redistribute it as proprietary--that is, turn its source code into object code and release it only as such.[87] If the redistribution as proprietary software takes place after any copyrightable modification is made to the public domain program, the author of the modification will be protected under copyright laws.[88] It is easy to realize that these actions endanger free software movement goals.[89]

Therefore, the public domain strategy is discarded and instead a copyright license is issued, called the GNU General Public License (GNU GPL).[90] The GNU GPL is based on two main clauses: first, the license provides users with the freedom to use, modify, and redistribute the software (free software clause); and second, the license forces any person redistributing the original or modified free software to do it under the same license, that is, under the free software clause (copyleft clause).[91] The license was written to guarantee that not only the original ***174** free software but also every single modification or derivative work will allow access to and redistribution of the source code.[92]

The originality and complexity from the GNU GPL lies in its copyleft clause, which basically states "that anyone who redistributes the software, with or without changes, must pass along the freedom to further copy and change it."[93] This is what is called "copyleft."[94] The concept copyleft reflects in an amusing way that the GNU GPL is achieving through the rights granted to the copyright holder the very opposite effect to the one intended by copyright: freedom instead of control.[95]

The notions of free software and copyleft are not synonymous. As we have seen before, software is free if it complies with the four mentioned freedoms. However, that does not necessarily mean that it also guarantees the access to the source codes of modified versions or its redistribution. For example, if a program is placed in the public domain as object code, it will be free software but not copylefted software. Moreover, there are some licenses that allow conversion of free software into proprietary software, for example, the BSD (Berkeley Software Distribution).[96] The works released under this kind of license "are usually called 'open' rather than 'free,' or if 'free' are qualified as 'but not copyleft.'"[97] I will expound on this difference below.

5. Software Development

Naturally, the lack of control under copyleft licenses influences the way in which software is produced. The traditional model of developing software, that is, a company employing computer scientists to sell the software that they produce, requires strict legal control over products, and it is therefore incompatible with the GNU GPL or any other copyleft license. If any third party could freely redistribute ***175** the software produced expensively by a private company, the latter would simply run its business at a loss.

The software development under the free software movement premises is based on the work of computer scientists or hackers,[98] who voluntarily and without expecting an economic benefit in return help to build a computer program. Licensing copyrights is forbidden under the GNU GPL.[99] Therefore, the authors of the program have no direct economic advantage against any persons. The incentives to cooperate in a free software project are satisfaction of programming,[100] prestige,[101] and economic gains obtained from sources different from the copyright licenses, for example, funding.[102] Other profit strategies are also possible under the free software principles,[103] as we will see in the following section.

In a very famous publication, Eric Raymond compares the proprietary and free models of software development with a cathedral and a bazaar.[104] The cathedral represents the model where a single person or a small group crafts the program and works in isolation until the release of the final product.[105] In opposition, in the bazaar, anyone can contribute to the program with different approaches, debugging solutions, and improvements.[106] Under Raymond's view, an open source project begins with the creation of a program. This program should be released to the community as soon as it constitutes a "plausible promise," even if it does not work particularly well or is still incomplete.[107] Once the program is being developed by the community, it is important to release the new versions often in order to get ***176** more corrections.[108] The free software movement strongly believes that the most effective way to produce software is by working in a community because "[g]iven a large enough beta-tester and co-developer base, almost every problem will be characterized quickly and the fix obvious to someone. Or, less formally, 'Given enough eyeballs, all bugs are shallow.'"[109]

Raymond's metaphor has found some criticisms in the literature. First, it has been argued that most open source software "[is] developed by individuals, rather than communities."[110] That may be true, but it would not change Raymond's point in relation to the big programs, like the development of an operating system. Second, many raise the point that Raymond underestimates the role of the productivity-multiplying effect of conventional management.[111] But Raymond's contention is that the community itself or the project leader of an open source project is more effective in all the tasks attributed to the conventional management team.[112] Third, it has been observed that Linux, which is the open source project on which Raymond bases his theory, "looks more like a highly centralized (cathedral) development model."[113] In fact, the most important open source projects have a leader or coordinator--some compare their role to a dictator. For example, in Linux it is Linus Torvalds who, with the support of a small group of lieutenants, makes the final decisions about the patches that will be implemented in the next versions of the program.[114] In Apache the committee directors are periodically elected in a democratic manner by the members of the Apache Foundation.[115] This criticism may affect the open source main version of a certain project; however, it is still true that nobody will prevent an independent programmer from using the source code to improve it and to make his own version available.

To sum up, open source projects are developed under a variety of models, and in certain cases, their characteristics are not so different from proprietary models. However, when an open source program is released as a final and complete product (also known as version 1.0), and assuming that it has some appeal to other computer scientists, the global review process by the community will occur. This will never happen under the proprietary approach.

***177** 6. Economic Models

For many years, the overwhelming opinion among jurists has been that granting copyright protection over software is the only way to guarantee production and quality. Along this line of thinking, the National Commission on New Technological Uses of Copyrighted Works considered that "some form of protection is necessary to encourage the creation and broad distribution of computer programs in a competitive market."[116] And at the same time it dismissed the possibility that "the creator is indifferent to cost and donates the work to the public," with the only exception for the academic and government sponsored research.[117] This idea has been promoted by the proprietary industry. Bill Gates asked the following rhetorical question in a letter dated in 1976 to open source programmers: "One thing you do is prevent good software from being written. Who can afford to do professional work for nothing?"[118]

However, as we have already seen, the free software movement has proven that it is possible to renounce all economic advantages from copyright and still produce software of a reasonable quality.[119] The question now is if, after renouncing the royalties coming from the copyright licenses, there is any economic model supporting the production, distribution, and maintenance of free software.

As we have seen, before the release of an open source computer program, some funding can come from donations. However, once the software is in the computers of final users, revenues will be made by the support sellers or redistributors. This model includes "media distribution, branding, training, consulting, custom development, and post-sales support."[120] One famous example of a company based on open source software redistribution is Red Hat, which not only distributes the software but also provides maintenance services to its customers.[121] The support seller economic model can even indirectly explain the ***178** interest of developers in working on open source projects for free, as they will gain experience that will allow them or their companies to guarantee high quality support to final users of the open source program that they developed.[122]

There are other economic models: "loss leader," in which the open source product is used to promote the sales of related proprietary software; "widget frosting," where the open source software supports the hardware, which is the actual source of revenues; "accessorizing," in which a company distributes books and other physical items associated with a certain open source software; "service enabler," where a company creates and distributes open source software to support access to revenue-generating online services; "sell it, free it," where a company releases software products first as proprietary and then converts them to open source products when the benefits of developing the software products in an open source environment outweigh the direct software license revenues they produce, typically when a new proprietary version is released; "brand licensing," where even if a company releases its software products as open source, it still retains the rights to its product trademarks (e.g., Linux and Mozilla) and logos; and "software franchising," in which the company would authorize other developers to use its brand names and trademarks in creating associated organizations doing open source support and custom software development.[123]

E. Free Software v. Open Source Software

1. Introduction

The terms free software and open source are often taken for one and the same thing. Although it may be true that the differences are not big, they are at least worth an explanation. The open source concept was born in February 1998 as a reaction to Netscape's announcement to give away the source of its browser.[124] A small group of computer scientists (Open Source Initiative) coined the term open source with the intention of making open development processes more appealing for the corporate world.[125] This concept was supported by important actors in the free movement, like Linus Torvalds, but not by Richard Stallman or his Free ***179** Software Foundation, and that caused a split.[126] Paradoxically, the expression open source has become the more popular of the two--92 matches in the titles of law review articles of the Westlaw database against only 10 for free software[127]-- but free software licenses, particularly the GNU GPL, are more used by far.[128]

The differences between free software and open source can be observed in four areas: the terminology, the definition, the philosophy, and their respective characteristic licenses.

2. Terminology

The members of the Open Source Initiative claim that open source is a much clearer term than the ambiguous free software.[129] The latter, they maintain, has many different meanings for different groups of people, from distribution for free to distribution under a copyleft license.[130] The counterargument from the Free Software Foundation is that "the obvious meaning for the expression 'open source software' is 'You can look at the source code.' This is a much weaker criterion than free software; it includes free software, but also includes semi-free programs[131] such as Xv, and even some proprietary programs. . . ."[132] To summarize, the ***180** expression open source software underlines the fact that one has access to the source code but not the possibility to modify and/or redistribute the program, whereas the term free software emphasizes the freedoms to copy, redistribute, and modify the program. However, free could mean (and is often misunderstood to mean) "at no charge;" moreover, the reference to access to the source code is only indirect.

In coherence with their respective main concepts, the Free Software Foundation uses the term non-free software, which includes semi-free and proprietary software. In opposition, the Open Source Initiative uses the term "closed" to refer to software, the access to the source code of which is restricted, and this term is expressly excluded from the Free Software Foundation terminology.[133] Both organizations consider the software proprietary when either its redistribution or access to its source code is disallowed.[134] Generally, the terms "non-free software" and "closed software" imply proprietariness.[135] However, there are some exceptions. For example, under the terminology of the Free Software Foundation, if a piece of software is allowed to be used, copied, distributed, and modified exclusively for non-profit purposes, it would be semi-free software--that is, non-free software--but not proprietary. Under the definition of the Open Source Initiative, if a piece of software allows access to and modification of the source code for private purposes but not its redistribution, it is not closed source software, but it is still proprietary.

In order to encompass both free software and open source software, the combination of both expressions has been proposed: "free and open source software" (FOSS). Another possibility is to refer to both types as the opposite of proprietary software, that is, "non-proprietary."[136] Finally, a working group within the European Union adopted the term "Libre Software," where "libre" means "free" in Spanish and French--as opposed to "freedom"--and hence does not cause any ambiguity.[137] Alternatively, the combination of the last two phrases has been used: "Free/Libre Open Source Software."[138]

***181** 3. Definitions

In order to eliminate or at least reduce these ambiguities, both groups have published definitions of the terms that they defend.[139] We have already seen that the free software movement characterizes "free software" as granting the user the freedom to run, copy, distribute, study, change, and improve the software and, therefore, to access the source code.[140] The Open Source Initiative guidelines are exposed in the Open Source Definition, which contains ten criteria.[141] First, the license must allow for free redistribution.[142] Second, "the program must include source code, and must allow distribution in source code as well as [in] compiled form."[143] Third, "the license must allow modifications and derived works."[144] Fourth, "the license may restrict source-code from being distributed in modified form only if the license allows the distribution of 'patch files'[145] with the source code for the purpose of modifying the program at build time."[146] This allows for unofficial changes that can be readily distinguished from the base source. Fifth, "the license must not discriminate against any person or group of persons."[147] Sixth, "the license must not restrict anyone from making use of the program in a specific field of endeavor."[148] Seventh, "the rights attached to the program must apply to all to whom the program is redistributed without the need for execution of ***182** an additional license by those parties."[149] This criterion forbids closing up software by indirect means, such as requiring a non-disclosure agreement. Eight, "the rights attached to the program must not depend on the program's being part of a particular software distribution."[150] Ninth, "the license must not place restrictions on other software that is distributed along with the licensed software."[151] And tenth, "no provision of the license may be predicated on any individual technology or style of interface."[152]

The four freedoms of the free software movement are almost equivalent with the ten clauses of the Open Source Definition.[153] Both definitions guarantee the possibility to run,[154] copy, distribute,[155] study, change, and improve the software and, therefore, to ensure access to the source code.[156] In some minor aspects, the Open Source Definition is less advantageous for the recipients of the program and more friendly to commercial interests.[157] For example, the Open Source Definition allows authors to restrict the distribution of source code in modified form in very specific circumstances;[158] the Open Source Definition does not allow restrictions on other software that is distributed along with the licensed software, while the free software definition is silent on this point. To sum up, the relevant differences between the position of the Free Software Foundation and that of the Open Source Initiative are not contained in the definitions that they use, but in the copyleft clause, which we will analyze later.

As a matter of fact, many of the licenses complying with the Open Source Definition can be qualified as free software as well, and the reverse is also true. Most of the 58 licenses approved by the Open Source Initiative[159] are recognized as ***183** free software by the free software movement[160] (i.e., Academic Free License; Apache License 2.0; Apache Software License; New BSD license; Eclipse Public License; X11, also called MIT license; PHP license; Python license; zlip/libpng license; and so on)--though they are not necessarily compatible with the GNU GPL because many of them fail to include a copyleft clause. The licenses that have been approved by the Open Source Initiative as open source licenses but do not satisfy the requisites of the free software definition are rare, though not completely non-existent (i.e., Reciprocal Public License).[161]

4. Philosophy

The free software movement and the Open Source Initiative describe their own philosophic background in a fair, distinguishable manner. On the one hand, the free software movement justifies the freedoms to distribute and access the source code as a matter of social fairness. The proprietary control over these acts causes different levels of harm: in particular, fewer people using the program, none of the users being able to adapt or fix the program, and other developers not being able to learn from the program or base new work on it.[162] For these reasons, the free software movement embarks on a crusade against proprietary software, which they characterize as “the enemy.”[163] Consequently, the free software movement has been depicted by the literature as idealistic[164] or even radical.[165]

On the other hand, the Open Source Initiative agrees with the superiority of an open development process, but they consider it compatible with the commercial software companies, that is, with proprietary software. They consciously distinguish themselves from the “confrontational attitude that has been associated with ‘free software’ in the past and sell the idea strictly on . . . pragmatic, business-case grounds.”[166] Consequently, the Open Source Initiative tolerates the conversion of open source software into closed or proprietary software and dislikes the use of the word “free,” which can be easily misunderstood as “free of charge.” ***184** Thus the members of the Open Source Initiative have been described as pragmatic.[167]

5. Representative Licenses: GNU GPL v. BSD License

We have already seen that the opposite philosophic foundations of the free software movement and the Open Source Initiative did not lead to very diverse definitions of free software and open source software. However, their different foundations are starkly reflected in the most representative licenses for both movements: the GNU General Public License (GNU GPL) and the Berkeley Software Distribution license (BSD),[168] respectively. In particular, the GNU GPL includes a copyleft clause whereas the BSD license does not. The copyleft clause imposes the obligation over modified or, in general, redistributed software to be licensed under the same license that allows its modification and redistribution. In other words, whoever wants to redistribute the software must pass along the freedom to further copy and change it. This might be seen as a restrictive condition for the commercial software companies or as the enlargement of the software user’s rights. In opposition, the BSD license does not have any copyleft clause, which means that it is possible to transform open source software into proprietary software.

Also, the Free Software Foundation describes the BSD license as free software, but not copyleft.[169] On the other hand, the Open Source Initiative considers the GNU GPL as open source software.

6. Conclusions

The most relevant differences that we have examined show a classification of three kinds of licenses or software strategies.

• First, proprietary license, which generally means that the access to the source code is restricted and its modification and redistribution is not allowed without authorization.

• Second, free software license or copyleft software, the access to the source code of which and the modification and redistribution of which are allowed under the condition that the modified and/or redistributed program will recognize these same rights.

***185** • Third, open source license or non-copyleft open source software, which unconditionally allows access to the source code, its modification and redistribution--and therefore its conversion into proprietary software.

The first two categories are completely irreconcilable, even antagonistic. The free software movement and the open source movement have different philosophies and goals, but still have enough common characteristics to allow collaboration and sympathy between both movements.[170]

III. Contractual Issues

A. Introduction

The non-proprietary licenses, both copyleft and non-copyleft open source software licenses, introduced a completely revolutionary way to deal with copyrights.[171] However, the most problematic issue in the free software and open source software arena is the copyleft clause.[172]

The non-copyleft open source software licenses do not generally impose any obligations on the licensors, besides those related to recognition of authorship. Therefore, the likelihood of a legal dispute for this kind of license is rather small. Meanwhile, the copyleft licenses impose on the downstream licensees the obligation to release their programs under the same copyleft clause. Due to this expansion of the effects of the copyleft clause over all the subsequent modifications and redistributions of the work, the copyleft licenses have been described as "viral ***186** contracts,"[173] that is, as "an attempt to make commitments run with a digital object."[174]

It may be useful for our discussion to quote the GNU GPL copyleft clause:

Section 2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions: . . . b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.[175]

The first debate that the copyleft clause has raised is about the legal nature of the document that introduces it. A significant number of scholars have questioned whether the GNU GPL should be treated as a license and suggest that we are rather dealing with a contract.

B. Contract v. License

1. Relevance of the Discussion

The distinction between contract and license is not merely academic. Most of the scholars entering this debate are considering at the same time the legal consequences that the qualification under one of the two categories may have for the GNU GPL.

First, copyright is a highly harmonized body of law. The Berne Convention for the Protection of Literary and Artistic Works (1886);[176] the Rome Convention for the Protection of Performers, Producers of Phonograms and Broadcasting ***187** Organizations (1961);[177] the WIPO Copyright Treaty (1996);[178] the WIPO Performances and Phonograms Treaty (1996);[179] and the WTO Agreement on Trade-Related Aspects of Intellectual Property Rights (TRIPs) (1994)[180] have successfully pursued harmonization at an international level. Other supranational agreements, such as the eight European Directives passed in the copyright field,[181] are also remarkable achievements in this same enterprise. On the other hand, the law of contracts varies in a notable manner between different countries and even within the same country, as happens across the United States.[182]

Second, if one considers the GNU GPL as a contract, a considerable array of problems arise. For example, is the delivery of the software under the GNU GPL a contract without consideration and, therefore, void? How and when do the offer and acceptance take place? Are warranty waivers enforceable when the recipient is a consumer?[183] This question seems irrelevant if we would deal with the GNU GPL as a copyright license.

Third, the remedies applicable will depend on whether it is a contract or a license. If it is a contract, it seems that a person who refuses to comply with the terms of the GNU GPL could be forced to release the source code of his derivative work. Meanwhile, if it is a license, the only remedies available are to hinder or stop the use of copyleft licensed code under a non-copyleft license and, perhaps damages. However, the release of the code of a derivative work which is using copyleft code under a proprietary license cannot be legally pursued.[184]

***188** Fourth, the enforceability of a license can only be claimed by the author or the copyright holder.[185] On the other hand, contracts can be enforced by the contracting parties, but not by third persons. That may interfere with the ability of the original author to sue downstream parties, who are not in privity with him.

2. Conceptual Approach

An appealing and simple solution to determine the right category for the GNU GPL is to look up the definitions of contract and license. Eben Moglen, chairman of the Software Freedom Law Center and general counsel for the Free Software Foundation, opines:

[t]he word 'license' has, and has had for hundreds of years, a specific technical meaning in the law of property. A license is a unilateral permission to use someone else's property. The traditional example given in the first-year law school Property course is an invitation to come to dinner at my house. If, when you cross my threshold, I sue you for trespass, you plead my 'license,' that is, my unilateral permission to enter on and use my property.[186] In short, a license is permission to do something.[187]

The Free Software Foundation advocates that the GNU GPL is a license. In the GPLv3 Draft, this is even expressly stated in the title of Section 9.[188] The resulting advantages for the Free Software Foundation position in considering the GNU GPL as a license are fairly clear. First, it avoids the extreme uncertainty caused by the diverse contract regulations. Second, it avoids the multiple inconveniences caused by the law of contracts (e.g., some of the parties using the GNU GPL are not familiar with the requirements to make a valid offer). Third, it avoids the privity requirement. And the only point where a qualification of a contract could be more interesting, which is specific performance of the terms of the agreement, is not contained in the goals of the Free Software Foundation.[189]

***189** On the other hand, "[a] contract is a promise or a set of promises for the breach of which the law gives a remedy, or the performance of which the law in some way recognizes as a duty."[190] From this definition two important characteristics have been derived. The first characteristic is the presence of a promise, which is "a manifestation of intention to act or refrain from acting in a specified way, so made as to justify a promisee in understanding that a commitment has been made."[191] The second characteristic is the necessity of an exchange; under the doctrine of consideration, courts do not enforce contracts unless the promisee has given the promisor something in return.[192]

The dichotomy between permission and exchange gets a bit more complicated when viewed in light of the Uniform Computer Information Transactions Act (UCITA), which defines a license as "a contract that authorizes access to, or use, distribution, performance, modification, or reproduction of, information or informational rights, but expressly limits the access or uses authorized or expressly grants fewer than all rights in the information, whether or not the transferee has title to a licensed copy."[193]

This definition seems to be formulated for the proprietary software license strategy. The proprietary software companies have avoided selling their products and instead they usually transfer the software under a license agreement. The main advantage of this strategy is that a license is not a sale[194] and, therefore, the "first sale doctrine" is not applicable.[195] Nevertheless, the courts seem to take for granted that a software license is a sale of goods subject to the U.C.C. or, at least, the equivalent to it.[196] Moreover, the courts agree that licenses are enforceable unless ***190** their terms are objectionable on grounds applicable to contracts in general, such as for violating a rule of positive law or being unconscionable.[197]

From the preceding statements and rules, some authors have drawn the conclusion that two different definitions of license are possible: a non-contractual license--when it authorizes acts restricted by copyright--and a contractual license--when it is supported by consideration.[198]

3. The Dual Solution

The literature has tried to categorize the GNU GPL as a whole. However, the recipients of free software can be divided into two groups: mere users and redistributors. Mere users may get free software at no charge. In other words, they do not have any obligation in exchange for reproducing the software. Moglen explains that the GNU GPL

license does not require anyone to accept it in order to acquire, install, use, inspect, or even experimentally modify GPL'd software. . . . The free software movement thinks all those activities are rights, which all users ought to have; we don't even want to cover those activities by license. Almost everyone who uses GPL'd software from day to day needs no license, and accepts none.[199]

Beyond the philosophical foundations of the free software movement, it is a fact that copyright law recognizes the author's exclusive right to authorize the reproduction of his work.[200] That means that a person who downloads a computer program onto his hard drive or copies it from a CD is making a reproduction; therefore, he needs the authorization of the author. Although the GNU GPL does not expressly authorize the user to make copies for private use, a right to make private copies without restrictions can be derived from recognition of the right to make and distribute copies, provided that they are released under the terms of the same license. That is, if the redistribution of the copies demands observance of the copyleft clause, the non-redistributed copies can be made without any requisite.[201] ***191** Moreover, the license does explicitly recognize a right to run the program.[202] If that argument is not persuasive enough, a license may be implied by the conduct of the licensor--for example, uploading a computer program in a publicly available server.[203] In both cases, we are facing a mere permission, that is, a license in the strict sense.

There are also redistributors to whom the GNU GPL is directed. As we have seen, the Free Software Foundation advocates that the GNU GPL is a license to the redistributors. This position is reflected in section 5:

You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.[204]

Although the section uses the word "license" three times, and with capital "L," there are two facts suggesting that it is not a license, but a contract. First, the section speaks about "acceptance," which is the second step of the "meeting of minds" characteristic for a contract--the first step is the offer, in this case, the GNU GPL itself. Second, the section imposes obligations on the licensee by referring to the rest of the license, in particular to the copyleft clause in section 2(b).[205] The licensee may copy and distribute the work, with or without modification, but in return he has to release the copies under the same terms. That is not a mere permission, but an exchange.[206] Therefore, the relation between licensor and licensee is a contract or a contractual license.

It appears that this question has not yet been addressed by any U.S. court. Nevertheless, in two settled cases where the enforceability of the GNU GPL was at ***192** issue, breach of the GPL as a contract was alleged.[207] In German courts, the GNU GPL has been considered a contract.[208]

Taking into account what has been said in this section, it can be stated that the distribution of free software is done in two different ways. First, if the person getting the software is a mere user who does not redistribute the software, he obtains his rights through a license. Second, if the person who gets the software redistributes it, with or without modification, he will be bound by certain obligations by virtue of the copyleft clause; these duties do not follow directly from the Copyright Act, but from a contract or, to use the more common terminology in the field, from a contractual license.

The understanding of the GNU GPL as a contract in the cases mentioned above raises various contractual issues which the Free Software Foundation wanted to avoid. These uncertainties affect the contract formation, the possibility of passing obligations and rights to third parties through the license, the content of the license (in particular, the warranty disclaimer), and the complex relation between copyright and contract law. The next two sections are devoted to contract formation--specifically consideration and shrinkwrap/clickwrap licenses.

C. Consideration

1. Introduction

In the common law system, to make a promise enforceable, the promisee has to do something that is either a detriment to the promisee or a benefit to the promisor.[209] That benefit or detriment is called consideration. Without consideration, a contract cannot be formed.[210]

***193** 2. Definition

The Restatement (Second) of Contracts has defined consideration in terms of a bargain:

• To constitute consideration, a performance or a return promise must be bargained for.

• A performance or return promise is bargained for if it is sought by the promisor in exchange for his promise and is given by the promisee in exchange for that promise.

• The performance may consist of

• an act other than a promise, or

• a forbearance, or

• the creation, modification, or destruction of a legal relation.

• The performance or return promise may be given to the promisor or to some other person. It may be given by the promisee or by some other person.[211]

3. Peppercorn Theory

The consideration requirement is relaxed by the fact that "[v]irtually anything that anyone would bargain for in exchange for a promise can be consideration for that promise."[212] The courts often use the hyperbole that even a mere peppercorn can constitute consideration.[213]

4. Consideration in Copyleft Software Licenses

Under the GNU GPL or any other copyleft software license, the two parties are the copyright holder or licensor and the redistributor or licensee. On the one hand, the licensor's consideration is clearly the authorization to modify and distribute the software. On the other hand, the licensee's consideration is more obscure. Some legal scholars have argued that the licensor is not getting anything in return, which may mean that no contract is formed.[214] However, it seems more ***194** reasonable to understand that the licensee's consideration is his promise to abide by the copyleft clause.[215] Therefore, the agreement is completely enforceable.

D. Clickwrap and Shrinkwrap Agreements

1. Introduction

Proprietary software is not sold but rather is only licensed.[216] Under this strategy, the software industry aspires to design its own rights and duties, even beyond the rights granted by copyright law.[217] The release of the software is done under standard form contracts or "mass market licenses,"[218] in particular, shrinkwrap or clickwrap licenses.

***195** The literature defines a shrinkwrap as "a license agreement that is usually contained in a box of software, which states that by opening the package, you agree to the terms of the license agreement."[219] Clickwrap licenses, on the other hand,
are a form of license used in an interactive manner on a computer. Typically, the user is presented with a display on a computer screen of the license and is prevented from proceeding with downloading or installation of the software until such time that he or she has indicated assent by clicking on a radio button on the computer monitor display.[220]

Shrinkwrap licenses can be categorized into different subgroups: a) in-box licenses, where the license is enclosed with the product in a sealed envelope; b) box-top licenses, which can be read before opening the box; and c) referral licenses, where there is a sticker indicating that the CD-ROM should not be opened prior to reading the license agreement.[221] Clickwrap licenses may be presented in the following ways: a) prior to download, a scroll-box appears and the user is asked to read a license and click "I agree"; b) the license is shown in a similar way, but during the installation of the program rather than before download; and c) the so-called "browsewrap" licenses--a variation of clickwrap licenses which are ordinarily found in online transactions where the user is informed of the existence of a license but is not required to read the license in order to proceed.[222]

2. Shrinkwrap and Clickwrap Licenses in the Courts

U.S. courts have had many opportunities to deal with the enforceability of shrinkwrap licenses. The decisions are not completely uniform, but main guidelines can be drawn. An early case in this field is Step-Saver Data Systems, Inc. v. Wyse Technology,[223] where the plaintiff, Step-Saver, developed and marketed a multi-user computer system with hardware from Wyse and software from TSL.[224] Almost immediately upon installation of the system, Step-Saver began to receive complaints from most of its customers, and at least twelve of them filed suit.[225] Step-Saver reacted by suing its providers.[226] The orders were placed by telephone, but on the package of each copy of the program was printed a license with a disclaimer of warranties.[227] The issue in this case was the enforceability of ***196** such a license.[228] The court agreed with Step-Saver's contention that the contract for each copy of the program was formed when TSL agreed, on the telephone, to ship the copy at the agreed price.[229] The box-top license, as Step-Saver argued, was a material alteration to the parties' contract containing additional terms which did not become a part of the contract.[230]

Some years later, the courts changed their position. The leading case in the field of shrinkwrap licenses is ProCD, Inc. v. Zeidenberg, where the plaintiff, ProCD, compiled information from more than 3,000 telephone directories into a computer database, which could be searched according to users' criteria.[231] The box enclosing the product declared that the software came with restrictions stated in an enclosed license.[232] This license was printed in a manual inside the box and appeared on the user's screen every time the software was run.[233] The license limited the use of the application program and listings to non-commercial purposes.[234] However, one of the buyers, Matthew Zeidenberg, made the information available over the World Wide Web for a price.[235] ProCD filed suit, but the district court accepted the view of the defendant that a contract includes only the terms on which the parties have agreed and not any hidden terms.[236] As a consequence, the district court declared the license void because the terms did not appear on the outside of the package.[237] On appeal, the Seventh Circuit reversed the decision on the basis that "[n]otice on the outside, terms on the inside, and a right to return the software for a refund if the terms are unacceptable (a right that the license expressly extends), may be a means of doing business valuable to buyers and sellers alike."[238] The court pointed out that contracts in which the exchange of money precedes the communication of detailed terms are common.[239] In short, the court endorsed the shrinkwrap license.

***197** The analytical difference between Step-Saver and ProCD is whether "money now, terms later" forms a contract (i) at the time of the purchase order or (ii) when the purchaser receives the box of software, TseeTs the license agreement, and does not return the software.[240] In the first scenario (adding terms to an existing contract), U.C.C. section 2-207 applies.[241] Therefore if the offeree is a consumer, the new terms need to be expressly accepted; if the offeree is a merchant, the acceptance may be implicit under certain circumstances, but not if the new terms materially alter the agreement. In the second scenario (forming a contract), the relevant provision is U.C.C. section 2-204,[242] which emphasizes that the parties can form a contract in any manner sufficient to show agreement. Later decisions have supported the ProCD interpretation[243] with very few cases to the contrary.[244]

The clickwrap licenses present very similar questions to the ones posed by the shrinkwrap licenses. Therefore, courts have accepted their enforceability through the ProCD doctrine.[245] Nevertheless, the special technical features of the clickwrap licenses increase the chance that the offeree is not made aware of the existence of the license, which can lead to its unenforceability.[246]

This is particularly so in the case of browsewrap licenses. In Specht v. Netscape Communications Corp.,[247] the court was asked to determine

***198** whether plaintiffs-appellees ('plaintiffs'), by acting upon defendants' invitation to download free software made available on defendants' webpage, agreed to be bound by the software's license terms (which included the arbitration clause at issue), even though plaintiffs could not have learned of the existence of those terms unless, prior to executing the download, they had scrolled down the webpage to a screen located below the download button.[248]

The court concluded that consumers did not manifest assent, because the mentioned reference was not sufficient to make consumers aware of those terms.[249] Therefore, consumers were not subject to the license agreement. In an apparently similar case, Ticketmaster Corp. v. Tickets.com Inc.,[250] the defendant ignored the prohibition of "deep linking" included in the license agreement, which was linked to the plaintiff's homepage.[251] The court observed that

the terms and conditions are set forth so that the customer needs to scroll down the home page to find and read them. Many customers instead are likely to proceed to the event page of interest rather than reading the 'small print.' It cannot be said that merely putting the terms and conditions in this fashion necessarily creates a contract with any one using the web site.[252]

That being said, other courts have suggested that a link is enough to create a binding agreement[253] and that it is not always necessary to click the "I Agree" button to be obliged by the terms of the license.[254]

Although the decisions are not uniform, they allow us to understand the main guidelines used by courts to decide the enforceability of shrinkwrap and clickwrap licenses.[255] In any event, if the following measures are observed, the enforceability of the license is almost guaranteed. First, the offeror should give notice to the offeree at the time when the latter decides to acquire the product that the terms of a ***199** shrinkwrap or clickwrap license will govern their agreement.[256] However, the chance that a court will be tempted to declare unenforceable the additional terms under U.C.C. section 2-207(2) is clearly higher. Second, the offeree should be forced to undertake positive steps to accept the license.[257] The implementation of this kind of procedure is extremely easy if the product is transferred online (i.e., clicking an "I Accept" radio button). Nevertheless, other forms of acceptance, even remaining silent, are also possible, but the risk of being declared void is higher.[258] Alternatively, the recipient must have the opportunity to return the product if he does not agree with the terms.

3. Enforceability of Copyleft Licenses as Shrinkwrap or Clickwrap Contracts

The questions raised by using the GNU GPL as a shrinkwrap or clickwrap license have been considered the most serious argument against its enforceability.[259] Normally, the way in which a work is licensed can be observed in practice, but it is not described anywhere. However, since the Free Software Foundation wants to encourage the application of the GNU GPL to downstream licensors, it provides some recommendations of how to attach a notice informing downstream licensees of the application of the license.[260] Following the Free Software Foundation guidelines, this notice[261] should be attached "to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the 'copyright' line and a pointer to where the full notice is found."[262] The notice should also indicate where to write in order to obtain a copy of the GNU GPL.[263]

These recommendations are not part of the GNU GPL itself. Therefore, authors can release their software under the GNU GPL without following the ***200** proposed way to attach the license to the software.[264] The general counsel of MontaVista Software, Inc., a company which provides a Linux-based operating system to its customers, states that

[m]ost licensors get the GPL in one of two ways: they get a piece of paper with the GPL printed on it (but not normally "wrapped" around any box or piece of software) [physical delivery] or they get, along with the software, an electronic file containing the GPL (but normally without the file being designed as a clickwrap) [digital delivery].[265]

The vulnerabilities of the GNU GPL licensing method are: (1) the recipient of the software may not receive a notice of the license before delivery, and (2) no signature or other manifestation of assent is generally required.[266] Moreover, it should be noted that the recipient may not get a copy, physical or digital, of the GNU GPL license. All of these factors, especially when they are combined, increase the risk that a court will declare a license unenforceable. Given that increased likelihood, some authors suggest that licensees are probably not bound by the terms of the license.[267]

Nevertheless, there are arguments to the contrary. First, the American Law Institute does not seem to consider the lack of delivery of a copy of the license (referral license) as a decisive factor in excluding its application.[268] Second, the National Conference of Commissioners on Uniform State Laws has drafted the Uniform Computer Information Transactions Act (UCITA), which has introduced provisions to guarantee the enforceability of shrinkwrap and clickwrap ***201** agreements.[269] However, only Maryland and Virginia have enacted the UCITA so far.[270] Finally, the license actually affects only the persons who intend to modify the software, redistribute the software, or both, as the license does not impose any obligations on mere users. If we are dealing with a programmer who wants to modify software released under the GNU GPL, he will necessarily open a source file and there is placed according to the recommendations of the Free Software Foundation a pointer to the full notice, which directs in turn to the license.

4. Unenforceability of Copyleft Licenses as Shrinkwrap or Clickwrap Contracts

The enforceability of the GNU GPL will depend on the variables that have been discussed before, and there will be a considerable number of border cases. Therefore, the question arises as to what will happen if the license is declared unenforceable. Some authors argue that in such a situation, the licensees will have an implied license for modifying the free software and redistributing it as proprietary.[271]

In my opinion, the situation is not that disastrous for the interests of the Free Software Foundation. In the absence of a contract because the license is declared void, the open source software will still be protected by the Copyright Act.[272] If the program is made available to the public, it could be argued that there is an implied license to run the software, but the implied license cannot be read more extensively. Therefore, there is no right to modify and redistribute the software. It is clearly contradictory to state that the licensee can take advantage of the rights granted in the GNU GPL, but not be compelled by any of its obligations.

The Free Software Foundation relies on the enforceability of the license. But it may be presumed that if the license was unenforceable, they would claim the applicability of copyright law. The GNU GPL states that "[y]ou are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License."[273] One could read ***202** the license to say that these actions are prohibited by copyright law if the license is declared void.

E. Privity

1. Introduction

As an elementary principle of the common law system, contractual rights and duties can only be conferred or imposed on the parties to a contract.[274] In legal terms, that is described as the doctrine of "privity of contract."[275] "The common law doctrine of privity means that a contract cannot, as a general rule, confer rights or impose obligations arising under it on any person except the parties to it."[276] In other words, if X sells a car to Y under the condition that the car will not be sold to Z, and Y does sell the car to Z, X will not be able to file suit against Z to recover the car because X is not in privity with Z.

2. Privity Concerns in Copyleft Software Licenses

The intention of the Free Software Foundation is to leave in the hands of the author of the program the power of filing suit against any possible infringer, both licensees and downstream sublicensees. With this goal, the GNU GPL states that "[e]ach time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. . . . You are not responsible for enforcing compliance by third parties to this License."[277]

However, the copyleft clause implies passing obligations and rights to third parties through the license. The legal feasibility of this process has been questioned by a possible lack of privity.[278] That means that the copyright holder could only file suit against his licensee, but not against any downstream sublicensee.

***203** 3. Relativization of Privity Concerns in Copyleft Software Licenses

Although the lack of privity between the author of the program and the downstream sublicensees seems clear, its consequences are alleviated through different paths. First of all, there are certain exceptions to the doctrine of privity.[279] Particularly relevant to our discussion is the exception for third party beneficiaries. However, to determine who has interest in the performance of a contract entered into by others so sufficient as to allow the author to enforce the provisions of the contract presents some difficulties.
Perhaps the most widely used test is that a person who wishes to enforce a contract to which he is not a party must show that the contract was intended for his benefit in either all or part of its contemplated performance. Some courts . . . add that if there is doubt concerning such intent the doubt will be resolved against the existence of the required intent, since parties are

presumed to contract for themselves.[280]

Therefore, the exception of third party beneficiaries does not TseeTm a safe way to solve the problem.

Second, the requirement of privity has been relaxed under modern laws[281] and generally replaced by the doctrines of implied warranty and strict liability, which allow a third party beneficiary or other foreseeable user to sue the seller of a defective product.[282] However, this relaxation only goes upstream from consumer to manufacturer. While it seems that the sublicensees will have a strong case against the copyright holder if the software is defective--warranty waivers apart--it is more doubtful that the copyright holder will have a comparable interest to sue the sublicensees.

Finally, even assuming that the lack of privity is a defense for the downstream sublicensees against the contractual claims of the author of the program, the result would not be as dramatic as expected. In such a situation, a third party who does not respect the terms of the license does not infringe the contract with the author--because there is no such contract at all--but he does infringe the author's exclusive rights of copyright. Consequently, the author cannot demand that any user of the software observe the terms of the license, but he is able to block anyone from using the software under copyright law. Moreover, the Free Software Foundation does ***204** not force anyone to transform proprietary software into free software because he or she used copyleft code. Instead, it gives the infringer the choice between ceasing to use the stolen code or releasing the software under the GPL.[283] Given that policy, the outcome of simply applying copyright is not that different from having an enforceable agreement.

F. Warranty Disclaimer

1. Introduction

A warranty disclaimer is included in the GNU GPL[284] and in other open source licenses.[285] That is also a common practice in proprietary software licenses. The waiver may apply in different relationships where a transfer of open source software occurs. The most common ones are between the author of the program and a redistributor (i.e., Red Hat); the author of the program and an end user (e.g., the programmer makes the software available for download); and a redistributor and an end user (e.g., Red Hat and its customers). The differences between the three kinds of subjects taking part in the transfer of the open source software are consequential to the validity or invalidity of the waiver, as shown below.

2. Merchants

If we accept that the U.C.C. is applicable to the sale of software,[286] it is necessary to direct the attention to section 2-314, which states: "(1) [u]nless excluded or modified (Section 2-316), a warranty that the goods shall be merchantable is implied in a contract for their sale if the seller is a merchant with respect to goods of that kind."[287]

Therefore, the implied warranty exclusively applies when only one of the parties is a "merchant," which is defined as

a person who deals in goods of the kind or otherwise by his occupation holds himself out as having knowledge or skill peculiar to the practices or goods involved in the transaction or to whom such knowledge or skill may be attributed by his employment of an agent or broker or other intermediary who by his occupation holds himself out as having such knowledge or skill.[288] ***205** Under this provision, a person will be a merchant based on specialized knowledge of either the goods or the practices involved.[289]

Although it would be necessary to conduct a case-by-case analysis to determine who is a merchant in every particular situation, it is generally true that companies such as Red Hat, which sell open source software for a fee, would be seen as merchants, while programmers who make their works available on the Internet for free would not.[290]

The merchants are not trapped by the implied warranty though. They may exclude it as long as the language is conspicuous.[291] In order to fulfill this requirement, the GNU GPL introduces the two sections of the warranty disclaimer with a bold and capitalized title, which reads: "NO WARRANTY."[292] The text of both sections is capitalized as well.[293] That seems to meet the exclusion requirement. Moreover, the U.C.C. states that "unless the circumstances indicate otherwise, all implied warranties are excluded by expressions like 'as is."'[294] This precise expression is used and emphasized in GNU GPL section

11.[295] Therefore, the GNU GPL warranty disclaimer apparently meets the spirit and requisites of the U.C.C.[296] This conclusion is coherent with the UCITA.[297]

Nonetheless, two warnings are worth attention. First, if the license is void as a shrinkwrap or a clickwrap license, according to the requirements which have been explained before, the warranty disclaimer will not pass the exclusion test. Second, a few states have enacted statutes prohibiting disclaimers of implied warranties.[298]

3. Non-Merchants

Common sense tells us that if a merchant can avoid the application of the implied warranty, a non-merchant should be in an even better position. However, a more accurate analysis is still warranted.

***206** In the relationship between the author of the program and the redistributor, we have to assume that there is a contract under the copyleft software license; otherwise, the redistributors would be infringing the copyrights of the author. As we have seen before, the license meets the U.C.C. requirements to exclude an implied warranty.[299] Moreover, the author of the program is generally not a merchant; therefore, the implied warranty of merchantability is not applicable.[300] In conclusion, the author of the program has no responsibility to the redistributors not only because the warranty disclaimer is valid, but also because of the lack of any responsibility due to his non-merchant position.

The conclusion drawn in the preceding paragraph is supported by UCITA, which states:

(a) [Free software defined.] In this section, "free software" means a computer program with respect to which the licensor does not intend to make a profit from the distribution of the copy of the program and does not act generally for commercial gain derived from controlling use of the program or making, modifying, or redistributing copies of the program. (b) [Implied warranties inapplicable.] The warranties under Sections 401 [Warranty and obligations concerning noninterference and noninfringement] and 403 [Implied warranty. Merchantability of computer program] do not apply to free software.[301]

Finally, the relationship of the copyright holder and the end user differs from the one between redistributor and end user because the former is arguably not bound by a contract, as stated earlier.[302] Implied warranties can be modified by contract, but not by a condition to a license.[303] Therefore, if there is no agreement between the parties, the warranty disclaimer does not have any effect. However, it is necessary to emphasize that the implied warranty will not emerge in the first place. The author of the program and the end user will be third parties to each other; therefore, the author could only be responsible under the much more relaxed standards of tort law.[304]

*207 G. Copyright v. Contract

1. Introduction

The author of a program has three different and cumulative methods to control the exploitation of his work: copyrights, contracts, and technological-protection measures (e.g., copy-protection devices). Undoubtedly, the protection of the author's interests is crucial to promote the creation of works. However, it is a non sequitur that the more protection, the better. Copyright law tries to strike the right balance between authors' protection and the public interest to access information. If the authors use contracts and/or technological-protection measures to go beyond copyright borders, the balance may break down.[305] The copyleft license uses both copyrights and contracts to achieve its goal; therefore the question of whether copyright can be overruled by the content of the copyleft license arises.

The means by which authors try to use contracts to bypass copyright are many.[306] A paradigmatic example is to avoid the application of the first sale doctrine by not accepting that the software is sold, but instead insisting that it is only licensed.

2. Preemption

In a conflict between copyright provisions and contract clauses, one could think of adopting one of the two following extreme solutions: always applying either copyright law or the agreement between the parties. However, neither approach is satisfactory.[307] To determine which rule should prevail, one needs to ask if the lack of protection under the copyright regime is because a certain act should remain unprotected, or if the copyright provision is simply a default rule that does not prevent protection under any other regimes, such as patents and contracts.

An optimal solution for the sake of certainty would be that the statute should distinguish between mandatory and default copyright rules. However, this task ***208** seems to be infeasible for the legislator, as such distinctions are not found in U.S. law and only rarely in foreign legislation.[308]

As a consequence, hard cases about when copyright law will preempt the content of the contract are left to the courts, which have had some difficulty in establishing clear guidelines in preemption cases based on contractual rights.[309] The leading case in this field establishes that "[a]lthough Congress possesses power to preempt even the enforcement of contracts about intellectual property . . . courts usually read preemption clauses to leave private contracts unaffected."[310] This tendency to favor contracts over copyright has been supported by the general acceptance of shrinkwrap/clickwrap licenses and the approval of the UCITA.[311] However, on occasions the courts have decided to preempt the content of the agreement, for example, in relation to a license prohibiting reverse engineering.[312]

In any event, if the copyright holder achieves the extension of his rights through a contractual agreement, the infringement of these rights will result in only one remedy against the infringing party under the rules of contract law.[313] In other words, third parties who fail to comply with the agreement while complying with copyright law will escape responsibility.

3. Deviations of the Copyleft Licenses from the Copyright Language

Generally, the GNU GPL uses the same language as the Copyright Act. However, there are some discrepancies. In particular, the GNU GPL gives its own definition of derivative work,[314] which differs from the one provided in the ***209** Copyright Act.[315] These deviations will probably be more important in the third version of the GNU GPL, where, for example, the concept of "propagation" is introduced,[316] which is completely alien to copyright law.

The observed deviations seem to be minor and do not restrict access to information. Therefore, they would probably be upheld by the courts. However, being part of a private agreement, these innovative definitions do not affect the rights of third parties, whose activities will be controlled exclusively by copyright provisions.

IV. Conclusion

Since the release of the GNU Manifesto in 1985, the free software presence has continuously increased. Today, free software represents a real alternative to proprietary software.

In the first part of the paper, I exposed the foundations and differences among the three main methods to license software: free, open source, and proprietary.

"Free software" has two main features: first, it is free because it authorizes anyone to copy, distribute, and/or modify (i.e. access the source code) the software; and second, it is copyleft because it forces any redistributor to recognize the right to copy, distribute, and/or modify the software.

The concept of "open source software" emerged after "free software," and it is used to refer to the software licenses which authorize the licensees to copy, distribute, and/or modify the software. However, open source software licenses do not necessarily oblige redistributors to recognize these same rights of their licensees. Therefore, it is generally true that "free software" has become a subspecies of "open source software," although there are some minor exceptions.

"Proprietary software," in contrast with free and open source software, does not allow either modification, redistribution, or, most commonly, a combination of those actions.

In the second part of the paper, I focused on the contractual issues in relation to free software and, in particular, the copyleft clause. I also referred to the peculiarities introduced by the copyright nature of the subject matter.

The existence of a contract is characterized by the presence of agreement and exchange, which are not present in mere licenses. Therefore, I consider that free ***210** software licenses are Janus-like. On the one hand, they are a non-contractual copyright license that allows end users private utilization of the program. On the other hand, they are a contractual license that allows redistributors to copy, distribute, and/or modify the software under the condition of granting to any sublicensee the right to copy, distribute, and/or modify the software.

As a consequence of the Janus-like nature of copyleft licenses, the relationship between the author of the program and the end user is exclusively governed by the Copyright Act. In contrast, the agreement between the author and redistributor must be examined not only under copyright law but also under contract law provisions.

The first issue arising due to the applicability of contract law concerns the element of consideration in the agreement, which is indispensable for the existence of a valid contract. In my opinion, the consideration requirement is met in the relationship between the author of the program (licensor) and redistributors (licensee) because the licensor authorizes modification and distribution of the software, and, in exchange, the licensee promises to abide by the copyleft clause. In the relationship between the author of the program and the end user, consideration is not necessary if one accepts that it is governed by a non-contractual copyright license, and therefore only copyright rules are applicable.

The second issue, also related to the existence of a contract, refers to the enforceability requirements of shrinkwrap and clickwrap licenses. Although the analysis of the relevant law does not allow for drawing very precise rules, the following three elements must be taken into account: making conspicuous a notice related to the applicable license before delivery of the program, delivery of the license itself, and the licensee's manifestation of assent. Meeting all of these requirements guarantees the enforceability of the shrinkwrap or clickwrap license. Moreover, the courts have shown some flexibility with a partial compliance with these requirements. In any event, if the shrinkwrap or clickwrap license is not enforceable, the author of the program will still be protected by copyright law.

The third issue puts into question the capability of the author of the program to file suit against downstream licensees, noticing that they lack privity of contract. The proposition that the author is incapable of filing suit may have some merit, although the privity doctrine has been recently relaxed. In any event, the author would still be protected by contract law against his licensee and by copyright law against any other sublicensee.

The fourth issue addresses the validity of the warranty disclaimer included in most copyleft licenses. The question must be answered by distinguishing between the position of the redistributor who licenses a computer program for a fee and the author of a program who posts it on a website and does not charge anything in return. In the former case, the redistributor is a merchant under the U.C.C. and an implied warranty crops up. However, this warranty may be disclaimed through a conspicuous notice. Therefore, if the shrinkwrap or clickwrap license is ***211** enforceable, it is most probable that the warranty disclaimer will displace the implied warranty. In opposition, the author of the program is generally not a merchant, and therefore the implied warranty does not emerge in the first place.

The fifth issue deals with the interrelation between copyright and contract law. Court decisions show a clear trend to accept contract clauses that deviate from copyright law provisions. The courts only opt for preemption in rare cases. However, it must be underlined again that the lack of privity could impede the enforcement of the license against third parties. In this case, copyright law would be applicable.

Footnotes

[a1] This paper has been written with the financial support of the Real Colegio Complutense. Dr. José J. González de Alaiza Cardona is a professor at the University of the Balearic Islands. He received an LLM from Harvard in 2006. He would like to thank William W. Fisher for his guidance and valuable suggestions, Gary Lam for his editing on the final draft, and the staff of the Texas Intellectual Property Law Journal for their careful editing.

[1] "'Free software' and 'Open Source' describe the same category of software, more or less, but say different things about the software, and about values." Richard Stallman, The GNU Operating System and the Free Software Movement, in Open Sources--Voices from the Open Source Revolution 53, 70 (Chris DiBona et al. eds., 1999) [hereinafter Stallman, The GNU Operating System], reprinted in Richard Stallman, Free Software, Free Society: Selected Essays of Richard M. Stallman 15 (Joshua Gay ed., 2002) [hereinafter Stallman, Free Software, Free Society]. In section II.E, infra, I will discuss the differences between the two concepts.

2 Mainstream products “get applications, trained users, and momentum that reduces future risk.” David A. Wheeler, Why Open Source Software/Free Software (OSS/FS, FLOSS, or FOSS)? Look at the Numbers!, Nov. 14, 2005, http:// www.dwheeler.com/oss_fs_why.html (last visited Oct. 1, 2006).

3 Whatis.com, What is Web Server?, http://whatis.techtarget.com (search “Look it up” for “web server”; follow “Read definition of ‘web server”’ hyperlink) (last visited Oct. 1, 2006) (“A Web server is a program that, using the client/server model and the World Wide Web’s Hypertext Transfer Protocol (HTTP), serves the files that form Web pages to Web users (whose computers contain HTTP clients that forward their requests). Every computer on the Internet that contains a Web site must have a Web server program.” In other words, a web server gets the user the web page that she has requested.)

4 A recent Netcraft survey (March 2006) found that of all the web sites they could find, counting by name--web server hostnames rather than physical computers--Apache had 68.70% of the market, Microsoft had 20.51%, Sun had 2.43%, and Zeus had 0.74%. Netcraft.com: March 2006 Web Server Survey, http:// news.netcraft.com/archives/2006/03/06/march_2006_web_server_survey.html (last visited Oct. 1, 2006).

5 Representing 95%, according to data compiled by Bill Manning, a staff member at the University of Southern California’s Information Sciences Institute. Bill Manning, In-addr version distribution (2q2000), http:// www.isi.edu/~bmanning/in-addr-versions.html, (last visited Oct. 1, 2006).

6 Webopedia, What is DNS?, http://www.webopedia.com/term/d/dns.html (last visited Oct. 1, 2006) (“Domain Name System (or Service or Server), an Internet service that translates domain names [e.g., www.harvard.edu] into IP addresses. Because domain names are alphabetic, they’re easier to remember. The Internet however, is really based on IP addresses. Every time you use a domain name, therefore, a DNS service must translate the name into the corresponding IP address.”).

7 Internet Systems Consortium, Inc., ISC BIND, http:// www.isc.org/index.pl?/sw/bind (last visited Oct. 1, 2006) (“BIND (Berkeley Internet Name Domain) is an implementation of the Domain Name System (DNS) protocols and provides an openly redistributable reference implementation of the major components of the Domain Name System, including: a Domain Name System server (named); a Domain Name System resolver library; and tools for verifying the proper operation of the DNS server.”).

8 Spread Firefox, 150 Million and Counting!, http:// www.spreadfirefox.com/node/22360 (last visited Oct. 1, 2006). Firefox was released on November 9, 2004 and 99 days later had been downloaded 25 million times. On March 3, 2006, the numbers of downloads was already 150 million. However, Firefox still has a modest market share (7%) in relation with its main competitor, Internet Explorer (90%, adding versions 6.x and 5.x). TheCounter.com, The Full-Featured Web Counter, http:// www.thecounter.com/stats/2006/March/browser.php (last visited Oct. 1, 2006). The use of Netscape, the other open source browser, is under 1% and decreasing. Id.

9 Spread Firefox, Thunderbird1.0 Reaches 1,000,000 Downloads, http:// www.spreadfirefox.com/node/8864 (last visited Oct. 1, 2006). Unlike numbers for web servers, a central site cannot get statistics about email clients unless it receives mail from the client. That makes it much more difficult to obtain this information. However, due to the fact that non-open source web clients such as Microsoft Outlook and Microsoft Outlook Express are installed simultaneously with the Windows operating system, one may assume that Thunderbird is clearly behind its opponents.

10 Linus Torvalds, Just for Fun - The Story of an Accidental Revolutionary 161 (2001); Jason B. Wacha, Taking the Case: Is the GPL Enforceable, 21 Santa Clara Computer & High Tech. L.J., 451, 452 (2005); see also Paul Gustafson & William Koff, Open Source: Open for Business, Leading Edge F. (CSC, El Segundo, Cal.), Sept. 2004, at 55-59, available at http:// www.csc.com/features/2004/uploads/LEF_OPENSOURCE.pdf (last visited Oct. 1, 2006).

11 In a survey conducted of 1,452 companies and public institutions in Germany, Sweden, and the UK from June 2001 to June 2002, it was observed that “the use of Open Source software on client or desktop computers is not very widespread. Only about 20% of those establishments that use OSS have some form of OSS installed on their desktops.” Thorsten Wichmann, Part I: Use of Open Source Software in Firms and Public Institutions, in FLOSS Final Report, Free/Libre Open Source Software: Survey and Study, at 41 (2002), http:// www.infonomics.nl/FLOSS/report (last visited Oct. 1, 2006).

12 Whatis.com, What is Operating System?, http://whatis.techtarget.com (search "Look it up" for "operating system"; follow "Read definition of 'operating system'" hyperlink) (last visited Oct. 1, 2006) ("An operating system [i.e., GNU/Linux, Windows, etc.] is the program that, after being initially loaded into the computer by a boot program, manages all the other programs in a computer. The other programs are called applications or application programs [e.g., OpenOffice and Microsoft Office]. The application programs make use of the operating system by making requests for services through a defined application program interface.").

13 That was an approximate guess of the Linux Counter Project in March 2005. Linux Counter: Estimates on the Number of Linux Users, http:// counter.li.org/estimates.php (last visited Oct. 1, 2006). Richard Stallman, the main developer of the GNU/Linux operating system, stated in a speech given at New York University on May 29, 2001:
In general, in business most users are not using GNU/Linux. Most home users are not using our system yet. When they are, we should automatically get 10 times as many volunteers and 10 times as many customers for the free software businesses that there will be. And so that will take us that order of magnitude.
Stallman, Free Software, Free Society, supra note 1, at 179.

14 Ted Shadler, Your Open Source Strategy, Tech Strategy Rep. (Forrester Research, Inc., Cambridge, Mass.) Sept. 2003, at 2, available at http://www.redhat.com/whitepapers/forrester/Forrester_OSS_Sep.pdf. Nevertheless, it may be surprising that there is a high percentage--70%--of large companies that use the GNU/Linux operating system. Id. Moreover, GNU/Linux claim very notable customers such as NASA, Torvalds, supra note 10, at 161, and the U.S. Army, Gustafson & Koff, supra note 10, at 59.

15 OpenOffice is an application suite--a set of applications that are designed to work together, which includes a word processor, spreadsheet, presentation graphics and drawing program and provides access to popular databases. OpenOffice runs on Linux, Windows, Mac and other operating systems.

16 MySQL is a database management system.

17 Gustafson & Koff, supra note 10, at 60. ("First released in 2000, OpenOffice has a 14 percent share of the large enterprise office systems market, according to Forrester Research, suggesting it is becoming a true alternative to Microsoft Office, which holds a 94 percent share of the overall office market, has been around for over a decade, and claims 300 million users worldwide. Microsoft, which in general has scoffed the open source approach, nonetheless recognizes the threat."). On the other hand, MySQL was declared to be used by 6% of U.S. large companies. Shadler, supra note 14, at 3. However, MySQL claims to be the number three "Top Deployed Database" after SQL Server and Oracle. See MySQL AB, Market Share, http://www.mysql.com/why-mysql/marketshare (last visited Oct. 1, 2006).

18 A leaked confidential document, known as Halloween I (author: Vinod Valloppillil; date: August 1998), explained the threat of open source for Microsoft and the ways to compete with it. Steven Weber, The Success of Open Source 126, 127 (2004). This memorandum has become a series which has been annotated and published by Eric S. Raymond. Eric S. Raymond, The Halloween Documents, http://www.catb.org/~esr/halloween (last visited Oct. 1, 2006).

19 Gustafson & Koff, supra note 10, at 3; see also Wacha, supra note 10, at 452.

20 Gustafson & Koff, supra note 10, at 2; see also Jordi Mas i Hernàndez, Software Libre en el Sector Público, FUOC, Oct. 2003, at 9, 10, available at http://www.uoc.edu/dt/20327/20327.pdf (last visited Oct. 1, 2006).

21 Whatis.com, What is Hardware?, http://whatis.techtarget.com (search "Look it up" for "hardware"; follow "Read definition of 'hardware'" hyperlink) (last visited Oct. 1, 2006).

22 Whatis.com, What is Software?, http://whatis.techtarget.com (search "Look it up" for "software"; follow "Read definition of 'software'" hyperlink) (last visited Oct. 1, 2006).

23 "A text editor is a computer program that lets a user enter, change, store, and usually print text." Whatis.com, What is Text Editor?, http:// whatis.techtarget.com (search "Look it up" for "text editor"; follow "Read definition of 'text editor'" hyperlink) (last visited

Oct. 1, 2006). “The distinction between editors and word processors is not clear-cut, but in general, word processors provide many more formatting features. Nowadays, the term editor usually refers to source code editors that include many special features for writing and editing source code.” Webopedia, What is Editor?, http://www.webopedia.com/TERM/e/editor.html (last visited Oct. 1, 2006).

24 Webopedia, What is Programming Language?, http:// www.webopedia.com/term/p/programming_language.html (last visited Oct. 1, 2006) (A programming language is a “vocabulary and set of grammatical rules for instructing a computer to perform specific tasks. The term programming language usually refers to high-level languages, such as BASIC, C, C++, COBOL, FORTRAN, Ada, and Pascal. Each language has a unique set of keywords (words that it understands) and a special syntax for organizing program instructions. High-level programming languages, while simple compared to human languages, are more complex than the languages the computer actually understands, called machine languages.”).

25 Whatis.com, What is Object Code?, http://whatis.techtarget.com (search “Look it up” for “object code”; follow “Read definition of ‘object code”’ hyperlink) (last visited Oct. 1, 2006).

26 Whatis.com, What is Compiler?, http://whatis.techtarget.com (search “Look it up” for “compiler”; follow “Read definition of ‘compiler”’ hyperlink) (last visited Oct. 1, 2006).

27 Stallman, Free Software, Free Society, supra note 1, at 3.

28 Jonathan Zittrain, Normative Principles for Evaluating Free and Proprietary Software, 71 U. Chi. L. Rev. 265, 271 (2004).

29 Id.; Hernàndez, supra note 20, at 1-2.

30 Christian H. Nadan, Open Source Licensing: Virus or Virtue?, 10 Tex. Intell. Prop. L.J., 349, 351 (2002).

31 “To decompile is to convert executable (ready-to-run) program code (sometimes called object code) into some form of higher-level programming language so that it can be read by a human. Decompilation is a type of reverse engineering that does the opposite of what a compiler does.” Whatis.com, What is Decompile?, http://whatis.techtarget.com (search “Look it up” for “decompile”; follow “Read definition of ‘decompile”’ hyperlink) (last visited Oct. 1, 2006).

32 Id. (“Decompilation is not always successful for a number of reasons. It is not possible to decompile all programs, and data and code are difficult to separate, because both are represented similarly in most current computer systems. The meaningful names that programmers give variables and functions (to make them more easily identifiable) are not usually stored in an executable file, so they are not usually recovered in decompiling .... Programs can be designed to be resistant to decompilation through protective means such as obfuscation.”).

33 Zittrain, supra note 28, at 271.

34 Zittrain, supra note 28, at 271.

35 See Digital Millennium Copyright Act, 17 U.S.C. §1201(f) (2000); Council Directive on the Legal Protection of Computer Programs 91/250/EEC, art.6, 1991 O.J. (L 122) 45 (EC).

36 Final Report of the National Commission on New Technological Uses of Copyrighted Works 11 (1978), available at http://digital-law-online.info/CONTU/PDF/chapter3.pdf.

37 Jesús González Barahona et al., Introducción al Software Libre 32 (2003).

[38] Id.

[39] Id.; see also Stallman, The GNU Operating System, supra note 1, at 53.

[40] Stallman, The GNU Operating System, supra note 1, at 53; see Miquel A. Zarza, ?Qué es el Software Libre? Regulación y Autoorganización, in Internet y Pluralismo Jurídico: Formas Emergentes de Regulación 297 (Pompeu Casanovas ed., Comares 2003).

[41] Before the adoption of a legal response, the literature was divided between protection through the copyright laws or through the patent system. Pamela Samuelson et al., Manifesto Concerning the Legal Protection of Computer Programs, 94 Colum. L. Rev. 2308, 2310 (1994).

[42] Although the 1976 Copyright Act did not address the issue of the copyrightability of computer programs, "[i]n 1964, the Register of Copyrights announced that computer programs would be accepted for registration, provided that (1) they contained sufficient original authorship, (2) they had been published, and (3) copies submitted for registration were in human-readable form." Nat'l Comm'n on New Technological Uses of Copyrighted Works, Final Report of the National Commission on New Technological Uses of Copyrighted Works 15 (1978), available at http://digital-law-online.info/CONTU/PDF/Chapter3.pdf. The National Commission on New Technological Uses of Copyrighted Works stated that "it was clearly the intent of Congress to include computer programs within the scope of copyrightable subject matter in the Act of 1976." Id. at 16.

[43] The National Commission on New Technological Uses of Copyrighted Works described this process in a very detailed form:
As the number of computers has increased dramatically, so has the number of programs with which they may be used. While the first computers were designed and programmed to perform one or a few specific tasks, an ever increasing proportion of all computers are general-purpose machines which perform diverse tasks, depending in part upon the programs with which they are used. Early programs were designed by machine manufacturers to be used in conjunction with one model or even one individual computer. Today, many programs are designed to operate on any number of machines from one or more manufacturers. In addition, and perhaps even more importantly, there is a growing proportion of programs created by persons who do not make machines. These people may be users or they may be--and increasingly are--programmers or small firms who market their wares for use by individual machine owners who are not in a position to write their own programs.
Nat'l Comm'n on New Technological Uses of Copyrighted Works, Final Report of the National Commission on New Technological Uses of Copyrighted Works 10-11 (1978), available at http://digital-law-online.info/CONTU/PDF/Chapter3.pdf.

[44] Act of Dec. 12, 1980, Pub. L. No.96-517,94 Stat. 3015, 3028. Although computer programs are not expressly quoted in the works of authorship list of section102(a) of the U.S. Copyright Act, the introduction of a computer software definition in section101 ("A 'computer program' is a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result") and an exception to make copies or adaptations to use the computer program or for archival purposes in section117 were enough for the courts to consider that "the copyrightability of computer programs is firmly established after the 1980 amendment to the Copyright Act." Williams Elecs., Inc. v. Artic Int'l, Inc., 685 F.2d 870, 875 (3d Cir. 1982); TseeT also Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240, 1248 (3d Cir. 1983).

[45] Council Directive on the Legal Protection of Computer Programs91/250/EEC, art.5.2, 1991 O.J. (L 122) 44 (EC).

[46] "Computer programs, whether in source or object code, shall be protected as literary works under the Berne Convention (1971)." Agreement on Trade-Related Aspects of Intellectual Property Rights art. 10(1), Apr. 15, 1994, Marrakesh Agreement Establishing the World Trade Organization, Annex 1C, Legal Instruments--Results of the Uruguay Round, 33 I.L.M. 1125, 1201, available at http://www.wto.org/English/docs_e/legal_e/27-trips.pdf.

[47] "Computer programs are protected as literary works within the meaning of Article2 of the Berne Convention. Such protection applies to computer programs, whatever may be the mode or form of their expression." WIPO Copyright Treaty art.4, Dec. 20, 1996, 36 I.L.M. 65, available at http:// www.wipo.int/treaties/en/ip/wct/pdf/trtdocs_wo033.pdf.

48 Nadan, supra note 30, at 351.

49 FSF--The Free Software Foundation, http://www.fsf.org (last visited Oct. 1, 2006).

50 "Richard Stallman is the God of Free Software.... Basically, he pioneered the notion of free source-code availability as something intentional, not just an accident, the way it happened with original Unix open development." Torvalds, supra note 10, at 58; see also Eric S. Raymond, A Brief History of Hackerdom, in Open Sources--Voices from the Open Source Revolution 19, 24 (Chris DiBona et al. eds., 1999).

51 Stallman, The GNU Operating System, supra note 1, at 54-55.

52 Stallman, The GNU Operating System, supra note 1, at 54-55.

53 Stallman, Free Software, Free Society, supra note 1, at 41-42 ("[Y]ou should be free to redistribute copies, either with or without modifications, either gratis or charging a fee for distribution, to anyone anywhere. Being free to do these things means (among other things) that you do not have to ask or pay for permission." In other words, "[y]ou may have paid money to get copies of free software, or you may have obtained copies at no charge. But regardless of how you got your copies, you always have the freedom to copy and change the software ....").

54 "In order for freedoms 1 and 3 (the freedom to make changes and the freedom to publish improved versions) to be meaningful, one must have access to the source code of the program. Therefore, accessibility of source code is a necessary condition for free software." Richard Stallman, Free Software, Free Society, supra note 1, at 41.

55 Chris DiBona et al., Introduction to Open Sources--Voices from the Open Source Revolution 1, 2 (Chris DiBona et al. eds., 1999).

56 Stallman, The GNU Operating System, supra note 1, at 54.

57 Categories of Free and Non-Free Software, http:// www.gnu.org/philosophy/categories.html (last visited Oct. 1, 2006).

58 "So that I can continue to use computers without violating my principles, I have decided to put together a sufficient body of free software so that I will be able to get along without any software that is not free." Richard Stallman, Initial Announcement of the GNU Project (1983), http:// www.gnu.org/gnu/initial-announcement.html (last visited Oct. 1, 2006).

59 Whatis.com, What is Operating System?, supra note 12.

60 Stallman, The GNU Operating System, supra note 1, at 55.

61 "The name GNU was chosen following a hacker tradition, as a recursive acronym for 'GNU's Not Unix."' Stallman, The GNU Operating System, supra note 1, at 56.

62 In September 1983, Stallman made the Initial Announcement of the GNU Project. A longer version called the GNU Manifesto was published in September 1985. Overview of the GNU System http://www.gnu.org/gnu/gnu-history.html (last visited Oct. 1, 2006). Moreover, in 1985 the Free Software Foundation, a tax-exempt charity for free software development, was created. Stallman, The GNU Operating System, supra note 1, at 60.

63 Webopedia, What is UNIX?, http://www.webopedia.com/TERM/u/unix.html (last visited Oct. 1, 2006) ("Pronounced yoo-niks, a

popular multi-user, multitasking operating system developed at Bell Labs in the early 1970s. Created by just a handful of programmers, UNIX was designed to be a small, flexible system used exclusively by programmers. UNIX was one of the first operating systems to be written in a high-level programming language, namely C. This meant that it could be installed on virtually any computer for which a C compiler existed.... Due to its portability, flexibility, and power, UNIX has become a leading operating system for workstations. Historically, it has been less popular in the personal computer market.").

64 "When used to describe software, portable means that the software has the ability to run on a variety of computers. Portable and machine independent mean the same thing--that the software does not depend on a particular type of hardware." Webopedia, What is Portable?, http:// www.webopedia.com/TERM/p/portable.html (last visited Oct. 1, 2006).

65 Stallman, The GNU Operating System, supra note 1, at 56; Overview of the GNU System, supra note 62.

66 Barahona et al., supra note 37, at 35.

67 Barahona et al., supra note 37, at 35.

68 Stallman, The GNU Operating System, supra note 1, at 62. The recursive acronym GNU pays tribute to the importance of Unix in the GNU Project.

69 Overview of the GNU System, supra note 62.

70 Webopedia, What is Kernel?, http:// www.webopedia.com/TERM/k/kernel.html (last visited Oct. 1, 2006). It is still fair to make the point that "[a]n operating system does not mean just a kernel, barely enough to run other programs. In the 1970s, every operating system worthy of the name included command processors, assemblers, compilers, interpreters, debuggers, text editors, mailers, and much more." Stallman, The GNU Operating System, supra note 1, at 56.

71 Overview of the GNU System, supra note 62; Barahona et al., supra note 37, at 45-46.

72 There is some controversy in using the term GNU/Linux or only Linux. Stallman states that "[i]f you call our operating system 'Linux,' that conveys a mistaken idea of the system's origin, history, and purpose. If you call it 'GNU/Linux,' that conveys (though not in detail) an accurate idea." Stallman, Free Software, Free Society, supra note 1, at 51. Moreover, Stallman states:
[o]ne CD-ROM vendor found that in their 'Linux distribution,' GNU software was the largest single contingent, around 28% of the total source code, and this included some of the essential major components without which there could be no system. Linux itself was about 3%. So if you were going to pick a name for the system based on who wrote the programs in the system, the most appropriate single choice would be 'GNU.'
Richard Stallman, Linux and the GNU Project, http://www.gnu.org/gnu/linux-and-gnu.html (last visited Oct. 1, 2006); see also Zittrain, supra note 28, at 268. However, Linus Torvalds suggests that the use of the term Linux is just fine. Torvalds, supra note 10, at 163.

73 See Stallman, Free Software, Free Society, supra note 1, at 125.

74 "[O]pen source can be seen as a revolutionary new way of doing and distributing software ...." Mikko Välimäki, The Rise of Open Source Licensing--A Challenge to the Use of Intellectual Property in the Software Industry 6 (2005), available at http://pub.turre.com/openbook_valimaki.pdf.

75 Stallman, Free Software, Free Society, supra note 1, at 41.

76 Stallman, Free Software, Free Society, supra note 1, at 48.

[77] Categories of Free and Non-Free Software, supra note 57.

[78] Stallman, Free Software, Free Society, supra note 1, at 47-48.

[79] Stallman, Free Software, Free Society, supra note 1, at 124-126.

[80] Stallman, Free Software, Free Society, supra note 1, at 126.

[81] Stallman, Free Software, Free Society, supra note 1, at 122. In comparison, thinks Stallman, the cost of production of material objects is significant, and therefore it is fair to add a share of the development cost, which does not make a qualitative difference. More moderated free software supporters are as well against the strict enforcement of copyright laws in case of illegal reproductions. Torvalds, supra note 10, at 97.

[82] Richard Stallman, The GNU Manifesto, http:// www.gnu.org/gnu/manifesto.html (emphasis added) (last visited Oct. 1, 2006).

[83] Id. Another term that causes some confusion is "freeware," that is, "[c]opyrighted software given away for free by the author. Although it is available for free, the author retains the copyright, which means that you cannot do anything with it that is not expressly allowed by the author. Usually, the author allows people to use the software, but not sell it." Webopedia, What is freeware?, http://www.webopedia.com/term/f/freeware.html (last visited Oct. 1, 2006). Freeware is not free software, as its source code may not be delivered, and it may not permit its redistribution.

[84] Stallman, The GNU Operating System, supra note 1, at 57.

[85] Stallman, The GNU Operating System, supra note 1, at 57.

[86] See Brian W. Carver, Share and Share Alike: Understanding and Enforcing Open Source and Free Software Licenses, 20 Berkeley Tech. L.J. 443, 447 (2005); Raymond Nimmer, Legal Issues in Open Source and Free Software Distribution, in 861 Critical Issues in Today's Corporate Environment 7, 11 (Practising L. Inst. 2006).

[87] Nadan, supra note 30, at 358.

[88] Nadan, supra note 30, at 358.

[89] Nadan, supra note 30, at 358-59.

[90] GNU Project, GNU General Public License, http:// www.gnu.org/copyleft/gpl.html (last visited Oct. 1, 2006).

[91] Id. (The Preamble of the GNU GPL reads: "We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.").

[92] Zittrain, supra note 28, at 268-69.

[93] Stallman, Free Software, Free Society, supra note 1, at 89.

94 Stallman, The GNU Operating System, supra note 1, at 59 (Stallman explains the origin of the term: “In 1984 or 1985, Don Hopkins (a very imaginative fellow) mailed me a letter. On the envelope he had written several amusing sayings, including this one: ‘Copyleft-all rights reversed.’ I used the word ‘copyleft’ to name the distribution concept I was developing at the time.”).

95 The literature has described this maneuver in the most diverse forms: “flips [copyright law] over,” Richard Stallman, The GNU Operating System, supra note 1, at 59; “turn[s] normal copyright on its head,” John C. Yates & Paul H. Arne, Open Source Software Licenses: Perspectives of the End User and the Software Developer, in 823 Twenty-Fifth Ann. Inst. On Computer & Internet L. 97, 104 (Practising L. Inst. 2005); or, even, as “a form of legal jujitsu,” Zittrain, supra note 28, at 269.

96 See Open Source Initiative, The BSD License, http:// www.opensource.org/licenses/bsd-license.php (last visited Oct. 1, 2006); see also Barahona et al., supra note 37, at 76 (The BSD license allows both the distribution as source code or as object code, and the single demand is to give credit to the original authors.).

97 Zittrain, supra note 28, at 269.

98 Whatis.com, What is Hacker?, http://whatis.techtarget.com (search “Look it up” for “hacker”; follow “Read definition of ‘hacker’” hyperlink) (last visited Oct. 1, 2006) (“Hacker is a term used by some to mean ‘a clever programmer’ and by others, especially journalists or their editors, to mean ‘someone who tries to break into computer systems.’”).

99 GNU Project, GNU General Public License, supra note 90, §§1-4.

100 Eric S. Raymond, The Cathedral & the Bazaar--Musings on Linux and Open Source by an Accidental Revolutionary 60, 61 (revised 1st ed. 2001); Stallman, Free Software, Free Society, supra note 1, at 127-128.

101 Raymond, supra note 100, at 84.

102 Stallman, Free Software, Free Society, supra note 1, at 128-129.

103 See Frank Hecker, Setting Up Shop: The Business of Open-Source Software (2000), http://www.hecker.org/writings/setting-up-shop (last visited Oct. 1, 2006); Sandeep Krishnamurthy, Cave or Community? An Empirical Examination of 100 Mature Open Source Projects, First Monday, vol.7, num.6, June 3, 2002, http://www.firstmonday.org/issues/issue7_6/krishnamurthy.

104 Raymond, supra note 100, at 27.

105 Raymond, supra note 100, at 27-30.

106 Raymond, supra note 100, at 27-30.

107 Raymond, supra note 100, at 58.

108 Raymond, supra note 100, at 29-30.

109 Raymond, supra note 100, at 41; see also Stallman, The GNU Operating System, supra note 1, at 55; Torvalds, supra note 10, at

226-227.

110 Krishnamurthy, supra note 103.

111 Raymond, supra note 100, at 67-68.

112 See Raymond, supra note 50, at 70-73.

113 Nikolai Bezroukov, A Second Look at the Cathedral and the Bazaar, First Monday, vol.4, num.12, Dec. 6, 1999, http:// www.firstmonday.org/issues/issue4_12/bezroukov.

114 Barahona et al., supra note 37, at 203.

115 Barahona et al., supra note 37, at 204.

116 Nat'l Comm'n on New Technological Uses of Copyrighted Works, Final Report of the National Commission on New Technological Uses of Copyrighted Works 11 (1978), available at http://digital-law-online.info/CONTU/PDF.

117 Id.

118 See Torvalds, supra note 10, at 227.

119 Stallman, The GNU Operating System, supra note 1, at 54.

120 Hecker, supra note 103.

121 Whatis.com, What is Red Hat?, http://whatis.techtarget.com (search "Look it up" for "red hat"; follow "Read definition of 'red hat"' hyperlink) (last visited Oct. 1, 2006) ("Red Hat is a leading software company in the business of assembling open source components for the Linux operating system and related programs into a distribution package that can easily be ordered. Red Hat provides over 400 different software packages .... The advantages to buying the distribution from Red Hat rather than assembling it at no cost yourself from various sources is that you get it as a single assembled package. Red Hat also offers service that isn't provided as quickly by the individual component developers, including members of the Free Software Foundation. Like all free software, Red Hat's packages allow the buyer to modify and even resell modified versions of code as long as they do not restrict anyone else from further modification.
Red Hat was one of the first companies to realize that 'free' software could be sold as a product. After examining the successful marketing campaign of Evian water, Red Hat concluded that to achieve success, the company had to create more Linux users and brand Red Hat as the Linux name that customers preferred. Today, the 'Red Hat Plan' is discussed as a model in business schools.").

122 Barahona et al., supra note 37, at 118-19.

123 Hecker, supra note 103.

124 Open Source Initiative, History of the OSI, http:// www.opensource.org/docs/history.php (last visited Oct. 1, 2006).

125 Id.

126 Id.

127 Open Source Initiative, Why “Free” Software is Too Ambiguous, http://www.opensource.org/advocacy/free-notfree.php (last visited Oct. 1, 2006)

128 Two of the most popular Open Source Software websites, Freshmeat.net and Sourceforge.net, confirm this fact. See Freshmeat.net, Statistics and Top 20, http://freshmeat.net/stats (last visited April 29, 2006) (out of approximately 43,000 projects, up to 66.79% use the GNU GPL, the LGPL is used in up to 6.29% and BSD licenses both original and revised, are used in up to 5.58%); see also SourceForge.net: Software Map, http:// sourceforge.net/softwaremap (last visited Oct. 1, 2006) (of 118.615 registered users, 51,533 are using the GNU GPL (43.44%); 9,055 the LGPL (7.63%); and 5,716 (4.82%)).

129 Open Source Initiative, Why “Free” Software is too Ambiguous, http://www.opensource.org/advocacy/free-notfree.php (“Some software is called ‘free’ because it costs no money to download or use--but source code is not available. The license that covers Microsoft Internet Explorer is a good example. Some software is called ‘free’ because it (and the source code for it) has been placed in the ‘public domain,’ free from copyright restrictions. A lot of software is called ‘free’ even though the source code for it is covered by copyright and a license agreement. The license usually includes a disclaimer of reliability, and may contain additional restrictions. The restrictions on non-public-domain ‘free’ software range from mild to severe. Some licenses may prohibit (or require a fee for) commercial use or redistribution. Some licenses may prohibit distributing modified versions. Some licenses may contain ‘copyleft’ restrictions requiring that the source code must always be made available, and that derived products must be released under the exact same license. Some licenses may discriminate against individuals or groups.”) (last visited Oct. 1, 2006).

130 Open Source Initiative, supra note 129.

131 Categories of Free and Non-Free Software, supra note 57 (“Semi-free software is software that is not free, but comes with permission for individuals to use, copy, distribute, and modify (including distribution of modified versions) for non-profit purposes. PGP is an example of a semi-free program.”).

132 Stallman, Free Software, Free Society, supra note 1, at 56.

133 Why “Free Software” Is Better Than “Open Source,” http:// www.gnu.org/philosophy/free-software-for-freedom.html (last visited Oct. 1, 2006) (“To prevent people from thinking we are part of them, we take pains to avoid using the word ‘open’ to describe free software, or its contrary, ‘closed,’ in talking about non-free software.”).

134 See Open Source Initiative OSI, The Open Source Definition §§1-2, http://www.opensource.org/docs/definition/php (last visited Sept. 8, 2006); see also Categories of Free and Non-Free Software http:// www.gnu.org/philosophy/categories.html#ProprietarySoftware (last visited Oct. 1, 2006).

135 See Proprietary Software, http://en.wikipedia.org/wiki/Closed_ software (last visited Oct. 1, 2006).

136 Andrés Guadamuz González, Viral Contracts or Unenforceable Documents? Contractual Validity of Copyleft Licences, 26(8) Eur. Intell. Prop. Rev. 331, 332-33 (2004).

137 Working Group on Libre Software, Free Software/Open Source: Information Society Opportunities for Europe? 5 n.2 Info. Soc’y Directorate Gen. of the European Comm’n, Working Paper Version 1.2, Apr. 2000, http:// eu.conecta.it/paper.pdf.

138 Wichmann, supra note 11, at 41.

[139] Open Source Initiative, OSI Certification Mark and Program, http:// www.opensource.org/docs/certification_mark.php (last visited Oct. 1, 2006) (The Open Source Initiative has even created a certification for the licenses complying with the conditions of the Open Source Definition: "We think the Open Source Definition captures what the great majority of the software community originally meant, and still mean, by the term 'Open Source.' However, the term has become widely used and its meaning has lost some precision. The OSI Certified mark is OSI's way of certifying that the license under which the software is distributed conforms to the OSD; the generic term 'Open Source' cannot provide that assurance, but we still encourage use of the term 'Open Source' to mean conformance to the OSD.").

[140] Free Software Foundation, The Free Software Definition, http:// www.gnu.org/philosophy/free-sw.html (last visited Sept. 8, 2006).

[141] The Open Source Definition, supra note 134.

[142] The Open Source Definition, supra note 134.

[143] The Open Source Definition, supra note 134.

[144] The Open Source Definition, supra note 134.

[145] Whatis.com, What is Patch?, http://whatis.techtarget.com (search "Look it up" for "patch"; follow "Read definition of 'patch"' hyperlink) (last visited Sept. 8, 2006) ("A patch (sometimes called a 'fix') is a quick-repair job for a piece of programming. During a software product's beta test distribution or try-out period and later after the product is formally released, problems (called bug[s]) will almost invariably be found. A patch is the immediate solution that is provided to users.... A patch is usually developed and distributed as a replacement for or an insertion in compiled code (that is, in a binary file or object module).").

[146] The Open Source Definition, supra note 134.

[147] The Open Source Definition, supra note 134.

[148] The Open Source Definition, supra note 134.

[149] The Open Source Definition, supra note 134.

[150] The Open Source Definition, supra note 134.

[151] The Open Source Definition, supra note 134.

[152] The Open Source Definition, supra note 134.

[153] The literature has pointed out that "[t]he four freedoms are consistent with (albeit broader than) the criteria employed by the Open Source Initiative." Nadan, supra note 30, at 357.

[154] Nadan, supra note 30, at 357 ("The freedom to run the program...."); see The Open Source Definition, supra note 134.

155 Nadan, supra note 30, at 357 ("The freedom to redistribute copies of the program."); The Open Source Definition, supra note 134.

156 Nadan, supra note 30, at 357 ("The freedom to access the source code, and modify it.... The freedom to release your modifications to the public."); The Open Source Definition, supra note 134.

157 Nimmer, supra note 86, at 16-17.

158 Open Source Initiative OSI, The Open Source Definition, supra note 134.

159 Open Source Initiative, The Approved Licenses, http:// www.opensource.org/licenses (last visited Oct. 1, 2006).

160 Various Licenses and Comments about Them, http:// www.gnu.org/licenses/license-list.html (last visited Oct. 1, 2006).

161 Id. ("The Reciprocal Public License is a non-free license because of three problems. 1. It puts limits on prices charged for an initial copy. 2. It requires notification of the original developer for publication of a modified version. 3. It requires publication of any modified version that an organization uses, even privately.").

162 Stallman, Free Software, Free Society, supra note 1, at 122-126.

163 Stallman, Free Software, Free Society, supra note 1, at 55.

164 César Iglesias Rebollo, Otra forma de distribuir software: las licencias de software libre o de código abierto, 2 Revista General de Legislación y Jurisprudencia 219, 223 (2004).

165 González, supra note 136, at 332.

166 Open Source Initiative, History of the OSI, http:// www.opensource.org/docs/history.php (last visited Oct. 1, 2006).

167 Rebollo, supra note 164, at 223.

168 The BSD license has an original and a modified version, with the difference that the modified version has removed the obligation to acknowledge the development done by the University of California, Berkeley and its contributors. Xfree86.org, Other Copyrights, http://www.xfree86.org/3-3.6/COPYRIGHT2.html (discussing the original version in the section titled "UCB/LBL" and the modified version in the section titled "General").

169 Various Licenses and Comments about Them, supra note 160.

170 Stallman, Free Software, Free Society, supra note 1, at 55.

171 Richard Stallman and the copyleft advocates had been criticized for using copyright to achieve a system which is apparently contrary to copyright itself. González, supra note 136, at 334. The answer that Stallman gives to this criticism is that the use of the copyleft clause doesn't mean that we are in favour of copyright law as a general matter. We're not totally against copyright law, in a simple or blanket sense either, but we're not defending the global copyright system that has mostly been imposed on the world merely because we use it because it's there. And that has to be very clear. We are not endorsing the Berne plus WTO system of copyright law as it stands as a good thing, but it exists and whatever harm it may do in other areas, we're trying to do some good

with it when we can.
Transcript of Opening Session of First International GPLv3 Conference, Boston, January 16, 2006, http://www.ifso.ie/documents/gplv3-launch-2006-01-16.html (last visited Oct. 1, 2006). Even under a more conservative view, a fair point that may be made is that copyright is not as much about control of the exclusive rights as about promoting the progress of science and useful arts. And from this latter perspective copyright and copyleft may not be that distant.

172 Nimmer, supra note 86, at 53; González, supra note 136, at 334.

173 Some authors prefer to use the term "reciprocal." E.g., Carver, supra note 86, at 447; Nimmer, supra note 86, at 15.

174 Margaret Jane Radin, Humans, Computers, and Binding Commitment, 75 Ind. L.J. 1125, 1132 (2000); Stephen Mutkoski, Open Source Software Issues in Acquisitions and Other Inbound Transactions, in 861 Open Source Software: Critical Issues in Today's Corp. Env't 337, 342-47 (Practising L. Inst. 2006) (The term "viral" must be used carefully, as the literature has attributed to it a second meaning. In particular, referring to a copyleft license as "viral" may intend to evoke the belief that using free software code in a proprietary program can lead to the obligation to release the entire work as open source software.).

175 In the GPLv3 First Discussion Draft, the copyleft provision is moved to section5b and three clarifications are introduced: it states that the GPL applies to the whole work; it removes the words "at no charge," which were often misinterpreted by commentators; and the last sentence of section5b explicitly recognizes the validity of disjunctive dual-licensing. Free Software Foundation, GPLv3 First Discussion Draft Rationale 12 (2006), available at http://gplv3.fsf.org/gpl-rationale-2006-01-16.pdf.

176 See World Intellectual Property Organization, Treaties and Contracting Parties: Berne Convention for the Protection of Literary and Artistic Works, http://www.wipo.int/treaties/en/ip/berne (last visited Sept. 30, 2006) (listing 162 contracting parties).

177 See World Intellectual Property Organization, Treaties and Contracting Parties: Rome Convention for the Protection of Performers, Producers of Phonograms and Broadcasting Organizations, http:// www.wipo.int/treaties/en/ip/rome (last visited Sept. 30, 2006) (listing 83 contracting parties).

178 See World Intellectual Property Organization, Treaties and Contracting Parties: WIPO Copyright Treaty, http:// www.wipo.int/treaties/en/ip/wct (last visited Sept. 30, 2006) (listing 60 contracting parties).

179 See World Intellectual Property Organization, Treaties and Contracting Parties: WIPO Performances and Phonograms Treaty, http:// www.wipo.int/treaties/en/ip/wppt (last visited Sept. 30, 2006) (listing 58 contracting parties).

180 See World Trade Organization, Understanding the WTO: Members and Observers, http://www.wto.org/english/thewto_e/whatis_e/tif_e/org6_e.htm (last visited Sept. 30, 2006) (listing 149 member states as of December 11, 1995).

181 See European Commission, Copyright and Neighbouring Rights, http:// ec.europa.eu/internal_market/copyright/index_en.htm (last visited Sept. 30, 2006) (describing the eight directives: Enforcement, Resale Right, Copyright in the Information Society, Protection of Databases, Term of Protection, Satellite and Cable, Rental and Lending Right, and Protection of Computer Programs).

182 Wacha, supra note 10, at 455-56.

183 Nimmer, supra note 86, at 25.

184 Eben Moglen, Enforcing the GNU GPL, Gnu Project, Sept. 10, 2000, http://www.gnu.org/philosophy/enforcing-gpl.html; Pamela Jones, The GPL Is a License, not a Contract, LWN.net, Dec. 3, 2003, http://lwn.net/Articles/61292/.

185 Lawrence E. Rosen, Open Source Licensing: Software Freedom and Intell. Prop. L. 55 (Prentice Hall 2005).

186 Jones, supra note 184. A similar definition states that “[a] license is a unilateral abrogation of rights. The licensor has, by law, the ability to enforce certain rights against the licensee, and the license functions as a promise not to enforce those rights.” Wacha, supra note 10, at 456.

187 Rosen, supra note 185, at 51.

188 Free Software Foundation, GNU General Public License: Discussion Draft 1 of Version 3, §9 (Jan. 16, 2006), http://gplv3.fsf.org/gpl-draft-2006-01-16.html. In the subsequent discussion draft, the title of section9 was changed from “Not a Contract” to “Acceptance Not Required for Having Copies.” Compare id. with Free Software Foundation, GNU General Public License: Discussion Draft 2 of Version 3, §9 (Jul. 27, 2006), http://gplv3.fsf.org/gpl-draft-2006-07-27.html.

189 The Free Software Foundation gives to the infringer the following choice: “you can stop using the stolen code and write your own, or you can decide you’d rather release under the GPL.” Jones, supra note 184. “In approximately a decade of enforcing the GPL, I have never insisted on payment of damages to the Foundation for violation of the license, and I have rarely required public admission of wrongdoing.” Moglen, supra note 184.

190 Restatement (Second) of Contracts §1 (1981).

191 Restatement (Second) of Contracts §2.

192 E. Allan Farnsworth, Contracts 3 (4th ed., Aspen Publishers 2004).

193 Unif. Computer Info. Transactions Act §102(a)(41) (amended 2002), 7(2) U.L.A. 208 (Supp. 2006).

194 ISC-Bunker Ramo Corp. v. Altech, Inc., 765 F. Supp. 1310, 1331 (N.D. Ill. 1990).

195 “Notwithstanding the provisions of section106(3) [distribution right], the owner of a particular copy or phonorecord lawfully made under this title, or any person authorized by such owner, is entitled, without the authority of the copyright owner, to sell or otherwise dispose of the possession of that copy or phonorecord.” Copyright Act of 1976, 17 U.S.C. §109(a) (2000). House Report94-1476 underlines that §109(a) of the Copyright Act “restates and confirms the principle that, where the copyright owner has transferred ownership of a particular copy or phonorecord of a work, the person to whom the copy or phonorecord is transferred is entitled to dispose of it by sale, rental, or any other means.” H.R. Rep. No.94-1476, at 79 (1976), as reprinted in 1976 U.S.C.A.N. 5659, 5693.

196 I.LAN Sys., Inc. v. Netscout Serv. Level Corp., 183 F. Supp. 2d 328, 331-32 (D. Mass 2002) (internal citations omitted) (“So is the purchase of software a transaction in goods? Despite Article 2’s requirement of a sale, courts in Massachusetts have assumed, without deciding, that Article 2 governs software licenses.... Admittedly, the UCC technically does not govern software licenses,... [but] the UCC best fulfills the parties’ reasonable expectations.”); Phillip Johnson, All Wrapped Up? A Review of the Enforceability of “Shrink-wrap” and “Click-wrap” Licenses in the United Kingdom and the United States, 25 Eur. Intell. Prop. Rev. 98, 100 (2003).

197 ProCD, Inc. v. Zeidenberg, 86 F.3d 1447, 1449 (7th Cir. 1996).

198 Diane Rowland & Andrew Campbell, Supply of Software: Copyright and Contract Issues, 10 Int’l J.L. & Info. Tech. 23, 25-26

(2002).

[199] Moglen, supra note 184.

[200] 17 U.S.C. §106(1) (2000).

[201] This license to use the work for private purposes has been included in the GPLv3 First Discussion Draft: "Propagation of covered works is permitted without limitation provided it does not enable parties other than you to make or receive copies." GNU General Public License: Discussion Draft 1 of Version 3, supra note 188, §2. The term "propagation" is an invention of the GPLv3 drafter and includes approximately all the exclusive rights of the copyright holder. GNU General Public License: Discussion Draft 1 of Version 3, supra note 188, §0 ("To 'propagate' a work means doing anything with it that requires permission under applicable copyright law, other than executing it on a computer or making private modifications. This includes copying, distribution (with or without modification), sublicensing, and in some countries other activities as well.").

[202] GNU General Public License: Discussion Draft 1 of Version 3, supra note 188, §0.

[203] Rosen, supra note 185, at 52.

[204] Free Software Foundation, GNU General Public License, Version 2, §5 (June 1991), http://www.gnu.org/licenses/gpl.html.

[205] Wacha, supra note 10, at 482 ("Under the GPL, one of the main obligations of the licensee is to disclose the source code. This obligation could only be enforceable under a contract theory, as opposed to a license theory.").

[206] See Peter Brown, Legal Issues in the Open Source Community, in 780 Twenty-Fourth Ann. Inst. on Computer L. 309, 317-18 (Practising L. Inst. 2004) ("In exchange for the rights granted by the open source licensor, the licensee has certain obligations."); see also Thomas Hoeren, Open Source und das Schenkungsrecht--eine durchdacte Liason?, Recht und Risiko, Festschrift für Helmut Kollhosser zum 70 Geburtstag 229, 237 (Reinhard Bork ed., 2004) (supporting the exchange concept).

[207] See Wacha, supra note 10, at 456 (citing pleadings in Progressive Software Corp. v. MySQL AB, 195 F. Supp. 2d 328 (D. Mass. 2002) and MontaVista Software, Inc. v. Lineo, Inc., No.2:02 CV-0309J (D. Utah filed July 23, 2003), which settled in 2003).

[208] "[T]he panel has no doubt whatsoever that the general business conditions have been effectively incorporated into a possible contractual relationship between the defendant and the plaintiff pursuant to German Civil Code Section305 Para.2." Landgericht München I [LG] [District Court of Munich I], No.21 O6123/04, May 19, 2004, at 9, http://www.jbb.de/judgment_dc_ munich_gpl.pdf (unofficial English translation) (official German opinion available at http://www.jbb.de/urteil_lg_muenchen_gpl.pdf); see LG München I: Wirksamkeit einer GPL-Lizenz, 20 Computer und Recht 774, 774-80 (2004); see also Peter Brown, Beyond SCO v. IBM: Other Legal Issues in the Open Source Community, in 808 Open Source Software: Risks, Benefits & Practical Realities in the Corp. Env't. 103, 113-14 (Practising L. Inst. 2004).

[209] Mills v. Wyman, 20 Mass. (3 Pick.) 207 (1825) ("It is only when the party making the promise gains something, or he to whom it is made loses something, that the law gives the promise validity.").

[210] Id.

[211] Restatement (Second) of Contracts §71 (1981).

[212] Farnsworth, supra note 192, at 48.

213 Whitney v. Stearns, 16 Me. 394 (1839) ("A cent or a pepper corn, in legal estimation, would constitute a valuable consideration.").

214 Rosen, supra note 185, at 63-65. This author considers that the GNU GPL would be enforceable under the doctrine of promissory estoppel. See Restatement (Second) of Contracts §90 (1981).

215 Wacha, supra note 10, at 475; Carver, supra note 86, at 458.

216 See Microsoft Corp. v. Harmony Computers & Elecs., Inc., 846 F. Supp. 208, 210 (E.D.N.Y. 1994) (Microsoft successfully argued that it never sells but rather only licenses its products.); see also Adobe Sys., Inc. v. Stargate Software Inc., 216 F. Supp. 2d 1051, 1052 (N.D. Cal. 2002) ("Adobe claims all Adobe software products are subject to a shrink-wrap End User License Agreement ('EULA') that prohibits copying or commercial re-distribution.").

217 Christian H. Nadan, Software Licensing in the 21st Century: Are Software "Licenses" Really Sales and How Will the Software Industry Respond?, 32 AIPLA Q.J. 555, 559-60 (2004) ("Today, copyright protection for software is well established. Yet software developers still license software to address the two considerations highlighted in the introduction--using licensing to defeat the first sale doctrine and effect price discrimination, and to impose liability-limiting contract terms. Simply put, licensing enables software owners to maximize the value of their software while minimizing their liability."); Michael D. Scott, Protecting Software Transactions Online: The Use of "Clickwrap" Licenses, in 482 First Ann. Internet L. Inst. 101, 103-04 (Practising L. Inst. 1997) ("Although the enforceability of such agreements has been in doubt for many years, software vendors have continued to use them, instead of merely relying on the protection provided by copyright law, for a number of reasons: 1. To negate the copyright law 'first sale doctrine,' which provides that once a copy of a copyrighted work has been sold, the copyright holder's rights in that copy are exhausted, and the copy may be freely resold, leased, rented, lent or otherwise disposed of without the copyright holder's consent. 2. To limit[] or disclaim warranties, remedies and liability as permitted by the provisions of the Uniform Commercial Code (UCC). 3. To impose other limitations on the transaction, such as use limitations on the software, prohibitions on reverse engineering, choice of law and forum, shortened statute of limitations, export control provisions, etc.").

218 Terry J. Ilardi, Mass Licensing--Part 1: Shrinkwraps, Clickwraps and Browsewraps, in 831 Pat. & High Tech. Licensing 251, 257 (Practising L. Inst. 2005) (defining the three most remarkable characteristics mass market licenses: "1) Their acceptance is indicated by some act other then a signature on writing; 2) they are not negotiated but are, rather a 'take it or leave it' type of agreement; and 3) they are intended for use with mass market products or services by a large end user community for which negotiated licenses would be financially, administratively or otherwise infeasible."); see Jean Braucher, The Failed Promise of the UCITA Mass-Market Concept and Its Lessons for Policing of Standard Form Contracts, 7 J. Small & Emerging Bus. L. 393 (2003) (discussing the mass-market concept under UCITA).

219 Nadan, supra note 30, at 362; TseeT Mark A. Lemley, Shrinkwraps in Cyberspace, Jurimetrics J. 311, 312 (1995).

220 Ilardi, supra note 218, at 256; see Nadan, supra note 30, at 362.

221 Johnson, supra note 196, at 98.

222 Johnson, supra note 196, at 98.

223 Step-Saver Data Sys., Inc. v. Wyse Tech., 939 F.2d 91 (3d Cir. 1991).

224 Id. at 94.

225 Id.

[226] Id.

[227] Id. at 95-96.

[228] Id. at 93.

[229] Step-Saver Data Sys., Inc. v. Wyse Tech., 939 F.2d 91, 98 (3d Cir. 1991).

[230] Id. at 97.

[231] ProCD, Inc. v. Zeidenberg, 86 F.3d 1477, 1449 (7th Cir. 1996).

[232] Id. at 1450.

[233] Id.

[234] Id.

[235] Id.

[236] Id.

[237] ProCD, Inc. v. Zeidenberg, 86 F.3d 1447, 1450 (7th Cir. 1996).

[238] Id. at 1451.

[239] Id. (discussing, e.g., insurance policies where the policy is received after payment of premium, airline tickets with terms received after ordering via telephone, etc.)0.

[240] I.LAN Sys., Inc. v. Netscout Serv. Level Corp., 183 F. Supp. 2d 328 (D. Mass. 2002) (Chief Justice Young elucidates and distinguishes both Step-Saver and ProCD and finally adapts the ProCD analysis.).

[241] U.C.C. §2-207(2) (1998) ("The additional terms are to be construed as proposals for addition to the contract. Between merchants such terms become part of the contract unless: (a) the offer expressly limits acceptance to the terms of the offer; (b) they materially alter it; or (c) notification of objection to them has already been given or is given within a reasonable time after notice of them is received.").

[242] U.C.C. §2-204 (1998) ("(1) A contract for sale of goods may be made in any manner sufficient to show agreement, including conduct by both parties which recognizes the existence of such a contract. (2) An agreement sufficient to constitute a contract for sale may be found even though the moment of its making is undetermined. (3) Even though one or more terms are left open a contract for sale does not fail for indefiniteness if the parties have intended to make a contract and there is a reasonably certain basis for giving an appropriate remedy.").

[243] Hill v. Gateway 2000, Inc., 105 F.3d 1147 (7th Cir. 1997); M.A. Mortenson Co., Inc. v. Timberline Software Corp., 998 P.2d 305 (Wash. 2000); Adobe Sys., Inc. v. Stargate Software Inc., 216 F. Supp. 2d 1051 (N.D. Cal. 2002).

[244] Klocek v. Gateway, Inc., 104 F. Supp. 2d 1332, 1341 (D. Kan. 2000) ("[I]f either party is not a merchant, additional terms are proposals for addition to the contract that do not become part of the contract unless the original offeror expressly agrees.").

[245] I.LAN Systems, Inc., 183 F. Supp. 2d at 338.

[246] Softman Prods. v. Adobe Sys., Inc., 171 F. Supp. 2d 1075, 1088 (C.D. Cal. 2001) (The court found "that Adobe's EULA cannot be valid without assent. Therefore, Softman is not bound by the EULA because it never loaded the software, and therefore never assented to its terms of use.").

[247] 306 F.3d 17 (2d Cir. 2002).

[248] Id. at 20.

[249] Id. at 32.

[250] 54 U.S.P.Q.2d 1344 (C.D. Cal. 2000).

[251] Id. at 1346.

[252] Id.

[253] EF Cultural Travel BV v. Zefer Corp., 318 F.3d 58, 63 (1st Cir. 2003).

[254] Register.com, Inc. v. Verio, Inc., 356 F.3d 393, 403 (2d Cir. 2004) ("We recognize that contract offers on the Internet often require the offeree to click on an 'I agree' icon .... But not in all circumstances. While new commerce on the Internet has exposed courts to many new situations, it has not fundamentally changed the principles of contract. It is standard contract doctrine that when a benefit is offered subject to stated conditions, and the offeree makes a decision to take the benefit with knowledge of the terms of the offer, the taking constitutes an acceptance of the terms, which accordingly become binding on the offeree.").

[255] Johnson, supra note 196, at 102 ("It does not seem fair to state that [t]he only thing that is certain when considering the enforceability of shrink-wrap and click-wrap software licence is that the law is uncertain.").

[256] Ilardi, supra note 218, at 273 (This may be accomplished through a statement in the box displayed in a shop, forcing the offeree to go through the terms of the license to download the software from a website. It may even be enough to display in a visible place of the website a link which directs to the terms of the license. A later notice (i.e., when the offeree receives the product) may be sufficient0.).

[257] Nadan, supra note 30, at 362.

[258] Ilardi, supra note 218, at 273.

[259] Wacha, supra note 10, at 488.

[260] Free Software Foundation, GNU General Public License, Version 2 (June 1991), http://www.gnu.org/licenses/gpl.html#SEC4 (providing suggested content of the notice).

[261] Id.

[262] Id.

[263] Id.

[264] The only requirement included in the GNU GPL that affects the downstream licensees-licensors states that they must "keep intact all the notices that refer to this License ... and give any other recipients of the Program a copy of this License along with the Program." GNU Project, GNU General Public License, supra note 90, §1.

[265] Wacha, supra note 10, at 489-490; see also Terry J. Ilardi, Mass Licensing--Part 2: Open Source Software Licensing, in 831 Pat. & High Tech. Licensing 279, 287 (Practising L. Inst. 2005).

[266] The GNU GPL considers that "by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it." GNU Project, GNU General Public License, supra note 90, §5. However, it seems rather dubious that the unilateral declaration of one of the parties can override the legal effect of shrinkwrap and clickwrap agreements.

[267] Nadan, supra note 30, at 362-63.

[268] Restatement (Second) of Contracts §211 (1981). In opposition, European countries are more strict with this requisite. See article 5.2 of the Spanish Contractual Standard Terms Act (B.O.E. April 14, 1998, 7 [p. 12304]) (stating that the standard terms will not be incorporated into the contract unless the offeror has informed the offeree of their existence and has provided him with a copy of them); see also Bügerliches Gesetzbuch [BGB] [Civil Code] Aug. 18, 1896, as amended Jan. 1, 2002, §305(2) (allowing more flexibility in only demanding that the offeror inform of the standard terms at the time of contract formation and facilitate the offeree gaining knowledge of its content in a reasonable way).

[269] See Unif. Computer Info. Transactions Act §§209-10 (amended 2002), 7 U.L.A. 208 (Supp. 2006) (listing the provisions for Mass Market License and Terms of Contract Formed by Conduct). Paradoxically, the Free Software Foundation advocates against the UCITA. They believe the GNU GPL is not a shrinkwrap or clickwrap license, in opposition to the ones used by proprietary software companies. Free Software Foundation, Why We Must Fight UCITA (June 30, 2005), http://www.gnu.org/philosophy/ucita.html.

[270] See Unif. Computer Info. Transactions Act §101 (amended 2002), 7 U.L.A. 208 (Supp. 2006) (listing only Maryland and Virginia in the Table of Jurisdictions Wherein Act Has Been Adopted).

[271] Nadan, supra note 30, at 362-67.

[272] Rosen, supra note 185, at 54.

[273] GNU Project, GNU General Public License, supra note 90, §5.

274 Patrick S. Atiyah, An Introduction to the Law of Contract 356 (5th ed. 1995).

275 "Privity of contract. The relationship between the parties to a contract, allowing them to sue each other but preventing a third party from doing so." Black's Law Dictionary 1237 (8th ed. 2004).

276 G.H. Treitel, The Law of Contract 539-40 (10th ed. 1999).

277 GNU Project, GNU General Public License, supra note 90, §6. Under U.S. law, non-exclusive copyright licenses are personal and not assignable unless expressly permitted by the license or with permission of the copyright holder: "[copyright licenses] are not transferable as a matter of law. Under the 1909 Act, 'Absent any contractual limitations, an assignee [of the whole contract] had the right to re-assign the work. A licenTseeT, however, had no right to re-sell or sublicense the rights acquired unless he has been expressly authorized to do so."' Harris v. Emus Records Corp., 734 F.2d 1329, 1333 (9th Cir. 1984). Thus, lacking an express permission in the license, the subsequent licensors will be forced to look for the direct permission of the copyright holder, which may make the transferability of copyleft software difficult.

278 The copyleft clause "purports to restrict subsequent transferees who receive software from a licenTseeT.... As this new transferee is not in privity with the original copyleft licensor, the stipulation TseeTms unenforceable." Robert P. Merges, The End of Friction? Property Rights and Contract in the "Newtonian" World of On-line Commerce, 12 Berkeley Tech. L.J., 115, 129 (1997).

279 Guenter Treitel, An Outline of The Law of Contract 253-262 (Oxford University Press 6th ed., 2004).

280 Clearwater Constructors, Inc. v. Gutierrez, 626 S.W.2d 789, 791 (Tex. 1981) (citations omitted).

281 It has been also argued that privity considerations have all but disappeared after the introduction of the Uniform Commercial Code (U.C.C.) and the various state statutes related to the U.C.C. Wacha, supra note 10, at 475.

282 See Black's Law Dictionary 1238 (8th ed. 2004) (defining the implied warranty and strict liability doctrines).

283 Moglen, supra note 184; Jones, supra note 184.

284 See GNU Project, GNU General Public License, supra note 90, §§11-12.

285 See, e.g., Open Source Initiative, supra note 96.

286 Wacha, supra note 10, at 471. However, Nimmer expresses deep doubts. Nimmer, supra note 86, at 88.

287 U.C.C. §2-314 (2004).

288 U.C.C. §2-104 (2004) (amended 2003).

289 Farnsworth, supra note 192, at 38.

290 Nimmer, supra note 86, at 89.

291 U.C.C. §2-314(2) (2004).

292 GNU Project, GNU General Public License, supra note 90, §11.

293 See GNU Project, GNU General Public License, supra note 90, §§11-12.

294 U.C.C. §2-316(3)(a) (2004).

295 GNU Project, GNU General Public License, supra note 90, §§11-12.

296 Wacha, supra note 10, at 472.

297 See Unif. Computer Info. Transactions Act §406 (amended 2002), 7(2) U.L.A. 208 (Supp. 2006) (discussing the UCITA disclaimer of modification of warranty).

298 Farnsworth, supra note 192, at 309. This waiver would be also considered void in many jurisdictions around the world, like Germany. Axel Metzger & Till Jaeger, Open Source Software und deutsches Urheberrecht, 10 GRUR Int, 839, 846-47 (1999).

299 See supra note 291 and accompanying text.

300 See supra note 291.

301 Unif. Computer Info. Transactions Act §410 (amended 2002), 7(2) U.L.A. 208 (Supp. 2006).

302 See discussion supra Part III.B entitled "Contract v. License." This departure seems to be assumed by the drafters of the UCITA: "Many such transactions [among contributors to the development of open source software and users] may not involve a contractual relationship and would, on that basis, fall outside of the scope of this Act." Unif. Computer Info. Transactions Act §410 cmt.2 (amended 2002), 7(2) U.L.A. 208 (Supp. 2006).

303 Wacha, supra note 10, at 482; Nimmer, supra note 86, at 89.

304 A similar outcome has been reached in the German literature, where the open source software transfer from the programmer to the end user is considered by some authors as a donation and the liability standard is only met in cases of recklessness or malicious intent. Metzger & Jaeger, supra note 298, at 847.

305 Lucie Guibault, Pre-emption Issues in the Digital Environment: Can Copyright Limitations be Overridden by Contractual Agreements under European Law?, in Opstellen over Internationale Transacties en Intellectuele Eigendom, 225-62 (F.W. Grosheide & K. Boele-Woelki eds., Europees Privaatrecht [11 Molengrafica Series] 1998), available at http://www.ivir.nl/publications/guibault/article2.doc.

306 See, e.g., Vault Corp. v. Quaid Software Ltd., 847 F.2d 255, 257 (5th Cir. 1988) (quoting Vault's software license as stating: "You [the user] may not transfer, decompile, or disassemble the Licensed Software for any purpose without VAULT's prior written consent."). This license sample likely conflicts with the "first-sale" doctrine, the right to make a backup copy, and the right to reverse engineering. Copyright Act, 17 U.S.C. §§109, 117, 1201(f) (2000).

[307] Mark A. Lemley, Beyond Preemption: The Law and Policy of Intellectual Property Licensing, 87 Cal. L. Rev. 111, 136 (1999).

[308] For example, "[t]he making of a back-up copy by a person having a right to use the computer program may not be prevented by contract insofar as it is necessary for that use." Council Directive on the Legal Protection of Computer Programs91/250/EEC, art.5.2, 1991 O.J. (L 122) 44 (EC); "[t]he maker of a database which is made available to the public in whatever manner may not prevent a lawful user of the database from extracting and/or re-utilizing insubstantial parts of its contents, evaluated qualitatively and/or quantitatively, for any purposes whatsoever." Council Directive on the Legal Protection of Databases96/9/EC, art.8.1, 1996 O.J. (L 77) 26 (EU).

[309] Maureen A. O'Rourke, Drawing the Boundary Between Copyright and Contract: Copyright Preemption of Software License Terms, 45 Duke L.J. 479, 479 (1995).

[310] ProCD, Inc. v. Zeidenberg, 86 F.3d 1447, 1454 (7th Cir. 1996).

[311] Pamela Samuelson, Copyright and Freedom of Expression in Historical Perspective, 10 J. Intell. Prop. L. 319, 340-41 (2003).

[312] Vault Corp. v. Quaid Software, Ltd., 847 F.2d 255 (5th Cir. 1988). But see Bowers v. Baystate Techs., Inc., 320 F.3d 1317 (Fed. Cir. 2003).

[313] Mark A. Lemley, Intellectual Property and Shrinkwrap Licenses, 68 S. Cal. L. Rev., 1239, 1273 (1995).

[314] GNU Project, GNU General Public License, supra note 90, §0 ("any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language.").

[315] Copyright Act, 17 U.S.C. §101 (2000) ("A 'derivative work' is a work based upon one or more preexisting works, such as a translation, musical arrangement, dramatization, fictionalization, motion picture version, sound recording, art reproduction, abridgment, condensation, or any other form in which a work may be recast, transformed, or adapted. A work consisting of editorial revisions, annotations, elaborations, or other modifications, which, as a whole, represent an original work of authorship, is a 'derivative work."').

[316] See GNU General Public License: Discussion Draft 2 of Version 3, supra note 188, §0.