PRELIMINARY DRAFT 2006/4/3 – FOR DISCUSSION PURPOSES ONLY
Accepted for Publication in Issue 21:4 (Fall 2006) of the
BERKELEY TECHNOLOGY LAW JOURNAL

DANGEROUS LIAISONS – SOFTWARE COMBINATIONS AS DERIVATIVE WORKS?

Distribution, Installation and Execution of Linked Programs under Copyright Law, Commercial Licenses and the GPL

By *Lothar Determann*[1]

Companies have been fighting about software interoperability and substitutability for decades. The battles have usually involved wholesale copying and significant modifications of code to achieve compatibility, and the law seems fairly settled in this respect.[2] More recently, however, software developers and users alike have started to wake up to potential problems regarding combinations of separate programs, particularly in connection with open source software.[3] Fear, uncertainty and doubt ("FUD") prevail in all quarters and have become a prominent topic in the computer lawyer community.[4]

This Article begins with a brief introduction to the issue and its context (I), examines the relevant copyright law principles in general (II) and the application of copyright law to software in particular (III), goes on to illustrate the classification of software combinations under copyright law in a few common technical and commercial scenarios (IV), and addresses the practical implications in the context of commercial (V) and open source licensing (VI), which is especially timely in light of the current debate surrounding the update of the General Public License (GPL). The article concludes that most forms of software combinations are less dangerous than commonly assumed, because they do not constitute derivative works (but instead either compilations or *sui generis* aggregations outside the scope of the copyright owner's exclusive rights), and a number of statutes and legal doctrines significantly limit a copyright owner's ability to contractually prohibit software combinations that do not also constitute derivative works under copyright law.

---

[1] Prof. Dr. Lothar Determann teaches courses on Computer and Internet law at the University of California Berkeley School of Law (Boalt Hall), University of San Francisco School of Law and Freie Universität Berlin (www.lothar.determann.name) and practices law as a partner in the international technology practice group of Baker & McKenzie LLP, San Francisco/Palo Alto office (www.bakernet.com). The author is grateful for assistance from his students, in particular Tal Lavian, Principal Scientist at Nortel Labs (valuable comments from computer science perspective), Steven B. Toeniskoetter, Lars F. Brauer, and Neda Shabahang (legal research and footnote editing).

[2] For a good introduction and overview see Mark A. Lemley, Peter S. Menell, Robert P. Merges, Pamela Samuelson, *Software and Internet Law*, 1st ed., pp. 2-45 (2000) and the references in the 2d ed. (2003).

[3] Lothar Determann and Andrew Coan, Spoilt Code? *SCO v. Linux—A Case Study in the Implications of Upstream Intellectual Property Disputes for Software End Users*, COMPUTER LAW REVIEW INTERNATIONAL 2003, 161 (hereafter "Determann, SCO"); Daniel Lyons, *Linux's Hit Men*, FORBES.COM, Oct. 14, 2003, http://www.forbes.com/2003/10/14/cz_dl_1014linksys.html (last visited Dec. 30, 2005); GPL-Violations.Org, http://gpl-violations.org/ (last visited Dec. 30, 2005); Christian H. Nadan, *Open Source Licensing: Virus or Virtue?*, 10 TEX. INTELL. PROP. L.J. 349 (2002) (hereafter "Nadan, Open Source Licensing").

[4] FUD has historically referred to a marketing strategy employed by established companies to discredit new market entrants and the open source movement in the eyes of potential customers; recently, however, even established companies complain about perceived FUD strategies by open source communities through intended vagueness and ambiguity of open source license agreements; at the 2005 ACI Conference on Software License Agreements in San Francisco, presenter Hank Jones summarized his perception with a joke: What do you get if you combine the Godfather and the GPL? An offer you can't understand. *See also* David S. Evans and Bernard J. Reddy, *Government Preferences for Promoting Open-Source Software: A Solution in Search of a Problem*, 9 MICH. TELECOMM. & TECH. L. REV. 313, 340 (2003); Klaus M. Schmidt and Monika Schnitzer, *Public Subsidies for Open Source? Some Economic Policy Issues of the Software Market*, 16 HARV. J.L. & TECH. 473 (2003). More on the GPL below in Section VI of this Article. For a summary of the history of the FUD expression, *see* Wikipedia, FUD, http://en.wikipedia.org/wiki/FUD (last visited Jan. 21, 2006). Well, the legal community does not seem to be entitled to expect much sympathy regarding the suffering of FUD from the programmer community after deciding to apply the copyright regime to software in the first place, and it also still has to play its own part in reducing uncertainties by reaching a clear consensus regarding the definitional scope of 'derivative works' in the context of software combinations.

I. INTRODUCTION

Few computer programs function in isolation–most have to be combined with other software to perform their tasks. Just consider common personal computer software packages: When one starts an application program (*e.g.*, Microsoft Word, Outlook, Adobe Acrobat, RealPlayer, etc.), it is actually an operating system program (*e.g.* Microsoft Windows) that "runs" the application program, and saves and prints data files. Application programs are often used together, *e.g.*, if a user cuts and pastes text from an Adobe Acrobat or MS Word document into an email or opens such a documents from an email attachment. Also, word processing and email processing software may use a separate program ("libraries") with definitions for fonts (*e.g.*, Times New Roman). Many applications use shared libraries instead of including the code within the application itself.[5]

From a technical perspective, in order to function in combination, programs have to be interoperable, *i.e.*, capable of exchanging and mutually using information.[6] Therefore, software manufacturers typically try to ensure that their own programs are interoperable with each other (in order to market seamlessly integrated software suites). With respect to programs made by other companies, however, software manufacturers have different agendas, depending on the market situation. For example, companies with an established platform may want to prevent interoperability with third party software in order to protect their market share for add-on programs or interests in hardware sales; new market entrants on the other hand will tend to promote interoperability with third party software in order to establish their platforms or to be able to offer add-ons or substitute programs for already established platforms.[7]

From a copyright law perspective, a combination of copyrighted programs typically requires one or more authorizations from the copyright owners. As a starting point, users normally need permission to *copy* a program in order to use it. In order to combine software or use software in combination, a user must typically first install (copy) the program(s) from the storage medium—be it a floppy disc, CD or a DVD—to a computer's hard drive. During the installation and execution of a computer program, the actual data is literally copied several times between storage medium, hard drive, computer memory, and CPU cache (as further explained in section IV.

By copying the electronic data in different storage units of the computer, the user creates additional physical manifestations of the computer program(s). Courts in the United States qualify such physical manifestations as a copy for purposes of copyright law,[8] and thus the copyright owner's duplication permission is typically required before a user may install copyrighted software in order to combine it with other programs.[9]

A copyright owner also has the exclusive right to prohibit or permit the preparation of derivative works (the "adaptation right").[10] Consequently, if and to the extent the use of two copyrighted programs

---

[5] *See generally* Wikipedia, *Library (Computer Science)*, http://en.wikipedia.org/wiki/Shared_library (last visited Jan. 13, 2006).

[6] Wikipedia, *Interoperability*, http://en.wikipedia.org/wiki/Interoperability#Software (last visited Dec. 29, 2005).

[7] *See, e.g., Sega Enterprises, Ltd. v. Accolade, Inc.*, 977 F.2d 1510 (9th. Cir., 1992) (hereafter "*Sega*"); *Sony Computer Entertainment, Inc. v. Connectix Co.*, 203 F.3d 596 (9th Cir., 2000), *cert. denied*, 531 U.S. 871 (2000) (hereafter "*Connectix*"); *Lexmark International, Inc. v. Static Control Components, Inc.*, 387 F.3d 522 (6th Cir. 2004) (hereafter "*Lexmark*"); *Chamberlain Group, Inc. v. Skylink Technologies, Inc.*, 381 F.3d 1178 (C.A. Fed. (Ill.) 2004), *cert. denied*, 125 S. Ct. 1669 (2005) (hereafter "*Skylink*").

[8] *See, e.g.*, *MAI Systems Corp. v. Peak Computer, Inc.*, 991 F.2d 511, 518 (9th Cir. 1993) (hereafter "*MAI Systems*"); *NFL v. Devcom*, 45 F.3d 231, 235 (7th Cir. 1995); *Costar v Loopnet*, 373 F.3d 544, 550-2 (4th Cir. 2004). The final report of the Congress-appointed Commission on New Technological Uses of Copyrighted Works ("CONTU") has taken the position that "[t]he text of the new copyright law makes it clear that the placement of a copyrighted work into a computer . . . is the preparation of a copy . . . . Because works in computer storage may be repeatedly reproduced, they are fixed and, therefore, copies." CONTU, FINAL REPORT 22 (1978). At least one German court came to a different conclusion, 1999 LG Mannheim, CR, 360–62.

[9] 17 U.S.C. § 106(1).

[10] 17 U.S.C. § 106(2).

in combination constitutes the preparation of a derivative work, the user needs specific permission to combine the programs,[11] and the owners of the copyrights to the two programs have a statutory right to deny granting such permission.

Traditionally, the adaptation right has been regarded as redundant and commercially irrelevant in practice given the fact that most adaptations also involve copying.[12] After all, a copy of a derivative work also constitutes a non-literal copy of the adapted original[13] and many cases involving the commercialization of non-literal copies that might also qualify as derivatives can already be resolved by finding infringement of duplication rights.[14]

In the software context, however, it can make a quite a difference whether or not an end user needs only a license to copy or additionally a license to prepare a derivative work before the end user can combine two programs. First of all, as a practical matter, commercial software programs typically come with end user licenses that expressly provide for a right to install and execute the software, but which are either silent regarding derivative works or expressly prohibit their preparation.[15] Second, purchasers who acquire ownership of particular software copies obtain certain statutory use rights, which include the right to execute the program, but not necessarily the right to create derivative works.[16] Third, a copyright owner that expressly prohibits certain software combinations is legally in a much stronger and clearer position if and to the extent it can rely on its statutory adaptation right—as opposed to contractual covenants or conditions, which require privity and may be subject to challenge under various legal theories, including copyright misuse, unconscionability, unfair competition law, antitrust law, etc.[17] And last but not least, the GPL and other open source licenses tie specific restrictions and conditions on the creation of derivative works.[18]

A few courts have had an opportunity to examine whether software combinations constitute derivative works: in a number of video game cases, the end users had lawfully purchased or licensed game software, so the game copyright owners had to rely on their adaptation right to fend off suppliers of add-on software that altered the games when executed in combination with games (*e.g.*, by running the games faster or adding new game "levels").[19] In a somewhat similar scenario, where a software

---

[11] Such permission could come in the form of a license from the copyright owner or a statutory exception to the copyright owner's exclusive rights, *e.g.*, 17 U.S.C. § 117(a)(1).

[12] REGISTER OF COPYRIGHTS, SUPPLEMENTARY REPORT ON THE GENERAL REVISION OF THE UNITED STATES COPYRIGHT LAW: 1965 REVISION BILL, p. 17 (House Comm. Print 1965) (hereafter "REGISTER OF COPYRIGHTS, SUPPLEMENTARY REPORT"); Lydia P. Loren, *The Changing Nature of Derivative Works in the Face of New Technologies*, 4 J. SMALL & EMERGING BUS. L. 57, 64 (2000). *See also* Melville B. Nimmer & David Nimmer, 2 *Nimmer on Copyright*, § 8.09.A [hereafter "Nimmer"].

[13] Hence, the Copyright Office believed that Section 106.2 of the Copyright Act was going to be largely duplicative–yet helpful for purposes of clarification. *See* REGISTER OF COPYRIGHTS, SUPPLEMENTARY REPORT, *supra*.

[14] *See, e.g., Lotus Development Corp. v. Borland International*, 49 F.3d 807 (1st Cir. 1995) (hereafter "*Lotus Development*"); *Computer Assocs. Int'l v. Altai, Inc*., 982 F.2d 693 (2d Cir. 1992) (hereafter "*Altai*"); *Whelan Associates, Inc. v. Jaslow Dental Laboratory, Inc.*, 797 F.2d 1222 (3d Cir., 1986). *But see, e.g., Worlds of Wonder, Inc. v. Veritel Learning Sys.*, 658 F. Supp. 351 (N.D. Tex. 1986) (hereafter "*Worlds of Wonder II*") and *Worlds of Wonder, Inc. v. Vector Intercontinental, Inc.*, 653 F. Supp. 135 (N.D. Ohio 1986) (hereafter "*Worlds of Wonder I*").

[15] *See, e.g., Dun & Bradstreet Software Services, Inc. v. Grace Consulting, Inc.*, 307 F.3d 197, 202 (3d Cir. 2002) (hereafter "*Dun & Bradstreet*"); also, for instance, the Microsoft Office Word 2003 SP2 End User License Agreement ("EULA") grants the user a license to use the software and make a backup copy, but nowhere mentions the preparation of derivative works, other than as it pertains to graphics and templates included with the program. The Macromedia Dreamweaver 4 EULA explicitly prohibits the user from making derivative works.

[16] 17 U.S.C. § 117 provides a very limited right to create a derivative work, but only if and to the extent the "adaptation is created as an essential step in the utilization of the computer program in conjunction with a machine and that it is used in no other manner;" given this narrow language, it is questionable whether the owner of a software copy is permitted to adapt its copy for purposes of interoperability with other software—as opposed to a machine, *see generally*, *Krause v. Titleserv*, 402 F.3d 119 (2d Cir. 2005).

[17] *See* below Section V(3).

[18] *See* below Section VI.

[19] *Micro Star v. Formgen*, 154 F.3d 1107, 1110 (9th Cir. 1998) (hereafter "*Micro Star*"); *Midway Mfg v. Artic International, Inc.*, 704 F.2d 1009 (7th Cir. 1983) (hereafter "*Midway*"); and *Lewis Galoob Toys, Inc. v. Nintendo of America, Inc.*, 964 F.2d 965 (9th Cir. 1992), *cert denied*, 507 U.S. 985 (1993) (hereafter "*Galoob*").

manufacturer sought to enjoin a maintenance services provider from offering substitute "sub-programs," the targeted customers held a valid license that allowed them to use at least unmodified versions of the plaintiff's larger software suite.[20] In the web-linking context, Internet users typically have an express or implied license to view the content on framed or linked websites, but website terms of use commonly prohibit any use of the site's content that goes beyond mere viewing.[21]

Beyond these few cases, however, courts and commentators have not yet developed general rules for the qualification of software combinations as derivative works, and the place and role of derivative works within the statutory context of compilations, collective works and other types of aggregations does not seem to have been examined in depth yet with respect to software combinations.[22] This article attempts to do so after briefly revisiting the concepts of derivative works and compilations under copyright law in general (II) and the characteristics of copyright protection for software in particular (III).

II. COPYRIGHT LAW ON DERIVATIVE WORKS, COMPILATIONS AND OTHER COMBINATIONS.

1. Statutory Definitions

The U.S. Copyright Act defines and uses the term "derivative work" separately and in contrast to the terms "compilation" and "collective work."[23]

> A "collective work" is a work, such as a periodical issue, anthology, or encyclopedia, in which a number of contributions, constituting separate and independent works in themselves, are assembled into a collective whole.
>
> A "compilation" is a work formed by the collection and assembling of preexisting materials or of data that are selected, coordinated, or arranged in such a way that the resulting work as a whole constitutes an original work of authorship. The term "compilation" includes collective works.
>
> . . .
>
> 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."

The definitions for all three categories have the primary requirement for copyright protection in common: in order to qualify, each must be an original work of authorship. Unlike patent law,[24] copyright law does not require novelty.[25] Copyright law is intended to protect creative expression. "Originality" does not require that facts or ideas be expressed in an innovative way, but merely that the arrangement of facts cannot be so logical, mechanical or routine as to require no creativity whatsoever.[26] A minimum "creative spark" is required.[27] With respect to derivative works, the *changes* to the original work must be

---

[20] *Dun & Bradstreet*, *supra* at 202.

[21] Copying, modifications, creation of derivative works, commercial exploitation and the like are typically either prohibited outright or made subject to the copyright owner's express written consent. *See, e.g.,* http://www.ticketmaster.com/h/terms.html and http://www.cnn.com/interactive_legal.html (accessed Jan. 21, 2006).

[22] *See, e.g.*, Christian H. Nadan, *A Proposal to Recognize Component Works: How a Teddy Bears on the Competing Ends of Copyright Law*, 78 CALIF. L. REV. 1633 (1990); Michael Gemignani, *Copyright Protection: Computer-Related Dependent Works*, 15 RUTGERS COMPUTER & TECH. L.J. 383 (1989).

[23] 17 U.S.C. § 101.

[24] Regarding patent protection for software *see State Street Bank & Trust v. Signature Financial Services*, 149 F.3d 1368 (Fed. Cir. 1998); Peter Weissman, Computer *Software as Patentable Subject Matter: Contrasting United States, Japanese, and European Laws*, 23 AIPLA Q.J. 525 (1995).

[25] eScholar, *LLC v. Otis Educational Systems, Inc.*, 2005 WL 2977569, p. 14 (slip copy) (S.D.N.Y. 2005); *Bucklew v. Hawkins*, 329 F.3d 923, 929 (7th Cir. 2003) (hereafter "*Bucklew*"); *Key Publications, Inc. v. Chinatown Today Publishing Enterprises, Inc.*, 945 F.2d 509, 512-513 (S.D.N.Y. 2005) (hereafter "*Key Publications*").

[26] *Feist v. Rural Tel. Serv. Co.*, 499 U.S. 340, 362 (1991) (hereafter "*Feist*").

[27] *Silverstein v. Penguin Putnam, Inc.*, 368 F.3d 77, 83 (2d Cir. 2004) (hereafter "*Silverstein*").

creative, whereas with respect to collective works and other compilations, the *selection* or *arrangement* must be creative.[28]

A compilation consists merely of the selection and arrangement of pre-existing material, without any internal changes to the compiled material.[29] A derivative work, on the other hand, is created through internal changes to existing works[30] that actually affect these works—as opposed to appearing merely in some loose context, detached from the original work.[31]

If the creator of a new work takes very little of an existing work[32] or takes only non-protectable content (*e.g.*, ideas, facts) or changes so much that the new work does not bear a close resemblance to the existing work, the new creation is simply a new work of authorship[33]—and not a derivative of the existing work.[34] After all, most new works are influenced to some extent by existing works.[35]

Thus, compilations, derivative works and entirely new works typically involve combinations of new creative materials with existing material. In the case of a compilation, the existing material remains intact and unchanged and the "combination creativity" remains separate and clearly distinguishable from the existing material.[36] In the case of a non-derivative new work, existing material may be remotely reflected in the new work, but its contribution is insubstantial. The derivative work category lies somewhere in the middle: existing creative material constitutes a substantial part of the new derivative work, and the new creative material appears in the form of inseparable changes to the existing material.

2. Examples of Combinations

Examples of derivative works include a translation of a poem into another language, an orchestra arrangement of a piano sonata, a rap version of a Beatles' song, a movie based on a comic book, or a theatre drama based on a novel. In all these cases, both the derivative (translation, arrangement, etc.) and the underlying work (poem, sonata, etc.) have to meet the originality requirement of the Copyright Act.[37] Thus, both the author of the underlying work and the creator of the derivative work have to apply at least a "creative spark."[38]

The same is true for collective works, which are creative collections of copyrighted works: Both the collection (*i.e.*, the selection thereof) and the collected works have to be creative in nature.[39] The term "compilations" includes creative collections of creative works (= collective works) and creative compilations of non-creative materials (= other compilations, *e.g.*, names and phone numbers of actual persons). In both alternatives, the compilation itself has to be creative. Thus, an arrangement of creative

---

[28] *Key Publications*, *supra* at 513; *Silverstein*, *supra* at 83.

[29] H.R. Rep No. 94-1476, 94th Cong., 2d Sess. (1976) at 57.

[30] *Id.*; *see generally* 1 Nimmer, *supra* note 15, at §3.02.

[31] *Bucklew*, *supra* at 930 (noting: "if the original expression added by the unauthorized preparer of a derivative work is clearly detachable from the original work itself, so that no confusion, or disruption of the copyright owner's plans for the exploitation of his work, would be created by allowing the unauthorized preparer to copyright his original expression, the unauthorized preparer might be allowed to do so . . . though this principle may be limited to compilations, where 'the infringing portion would be easily severable and the scope of the compilation author's own work . . . would be easily ascertainable.'" (internal citations omitted)).

[32] *Vault v. Quaid*, 655 F. Supp. 750, 759 (E.D. La. 1987), *aff'd* 847 F.2d 255, 267 (5th Cir. 1988) (hereafter "*Vault*"); *Superchips, Inc. v. Street & Performance Electronics, Inc.*, 2001 WL 1795939, p. 3, 61 U.S.P.Q.2d 1589 (M.D. Fla., 2001) (not reported in F. Supp. 2d) (hereafter "*Superchips*") ("courts consider whether there is a distinguishing variation between the derivative and underlying work and whether that variation is more than "merely trivial." (internal citation omitted)).

[33] *Castle Rock Entertainment Inc. v. Carol Publishing Inc.*, 150 F.3d 132, 143 n.9 (2d Cir. 1998) (hereafter "*Castle Rock*").

[34] *Bucklew*, *supra* at 930; *Pickett v. Prince*, 207 F.3d at 407 (7th Cir. 2000) ("works only loosely connected with some ancestral work claimed to be their original.") (hereafter "*Pickett*"); *Vault*, *supra* at 758; 1 Nimmer, *supra* note 15, at § 3.02.

[35] *Emerson v. Davies*, 8 F. Cas. 615, 619 No. 4436 (C.C.D. Mass. 1845).

[36] The creative combination is, in essence, a meta-layer; the glue that holds the existing material together.

[37] *Silverstein*, *supra* at 83.

[38] *See, e.g., Silverstein supra* at 83.

[39] *Silverstein*, *supra* at 83.

or non-creative material in a purely logical order (*e.g.*, alphabetically or in numerical order) cannot qualify as a copyrighted compilation.[40] Copyrightable compilations are, for example, creative catalogues or collections of poems.[41]

If someone applies minor changes to a work without any originality (*e.g.*, a copy with typos or typo corrections and a few missing words),[42] the result will not constitute a derivative work, but only a (non-literal) copy of the work.[43] Similarly, if someone prepares a collection of copyrighted or non-copyrighted material without any creativity (*e.g.*, in historic or alphabetical order), the result will not constitute a copyrightable compilation, but rather a series of literal copies (if the collected material is copyrighted) or a collection that is entirely outside the scope of copyright law.[44] If an author creates a new work, borrowing only minor aspects from existing works, the result qualifies as an independently created new work, not a derivative work.[45]

Thus, for copyright law purposes a combination of new and existing material may constitute one or more of the following:

- a new (non-derivative) work (if only very little of or non-protectable elements of the existing materials are present in the new work or if the new work does not bear a substantial resemblance to the existing work);[46]
- a derivative work (if new material changes the substance of the existing material and both are creative, *e.g.*, a song based on a poem);[47]
- a compilation (if existing creative or non-creative material is arranged in a creative manner, *e.g.*, a collection of songs, poems and/or facts related to the holiday season);[48]
- a non-literal copy (if new material makes insubstantial, non-creative changes to the substance of the existing material, only the existing material is creative and the end result is nearly identical, *e.g.*, publication of a poem with typo corrections);[49]

---

[40] *Feist*, *supra* at 362-364; *Silverstein*, *supra* at 83; *Matthew Bender & Co., Inc. v. West Pub. Co.*, 158 F.3d 674, 687 (hereafter "*Matthew Bender*").

[41] 1 Nimmer, *supra* note 15, at § 3.02.

[42] *See* spell-check example discussed in *Galoob*, *supra* at 969; *Silverstein*, *supra*, at 83; *Matthew Bender*, *supra* at 681, note 4.

[43] *Galoob*, *supra*; *but see Superchips*, *supra* at 3-4; *Sherry Mfg. Co. v. Towel King of Fla., Inc.*, 753 F.2d 1565, 1568 (11th Cir. 1985); *Bucklew*, *supra*.

[44] *Feist*, *supra* at 362-364; *see also*, *Silverstein*, *supra* at 83.

[45] *Pickett*, *supra* at 407 ("works only loosely connected with some ancestral work claimed to be their original").

[46] *See, e.g., Well-Made Toy Mfg. Corp v. Goffa Intern. Corp.*, 354 F.3d 112, 117 (2d Cir. 2003) (toy doll was not substantially similar to competitor's product, and therefore did not constitute infringing derivative work); *Ty, Inc. v. Publications Intern. Ltd.*, 292 F.3d 512, 521 (7th Cir. 2002) ("Beanie Babies" collectors' guide was not a derivative work, but a "public evaluation," which the toy manufacturer was not entitled to control under its copyright).

[47] *See, e.g., Shoptalk, Ltd. v. Concorde-New Horizons Corp*., 897 F. Supp. 144, 147 (S.D.N.Y.1995) (movie as derivative work of screenplay), *affirmed in part, vacated in part* 168 F.3d 586 (2d Cir. 1999), *certiorari denied* 527 U.S. 1038 (1999); *Lamb v. Starks*, 949 F. Supp. 753 (N.D. Cal. 1996) (trailer as derivative work of movie).

[48] *See, e.g., Corsearch, Inc. v. Thomson & Thomson*, 792 F. Supp. 305, 322 (S.D.N.Y. 1992) (state trademark computer database was protectable compilation where copyright proponent had selected, coordinated, arranged, enhanced, and programmed the trademark data); *Mason v. Montgomery Data, Inc.*, 967 F.2d 135, 141 (5th Cir. 1992) (collection of real estate ownership maps incorporating information from various sources were "sufficiently creative to qualify . . . as original 'compilations'" because author had engaged in "selection, coordination and arrangement of the information . . . depicted"). *Cf.* Howard B. Abrams, THE LAW OF COPYRIGHT, § 1:16 (updated 2006) ("majority of compilations will pass [minimal level of creativity] test").

[49] *See, e.g., Signo Trading International, Ltd. v Gordon*, 535 F. Supp. 362, 365 (N.D. Cal. 1981) (list of words translated into foreign language did not constitute copyrightable compilation because author of translation had not selected words on the list). For infringement liability by reason of non-literal copying *see generally Castle Rock Entertainment, Inc. v. Carol Pub. Group, Inc.*, 150 F.3d 132, 140 (2d Cir. 1998) ("Seinfeld" trivia book infringed copyright in television show, even though "direct quotations or close paraphrases . . . copied from the Seinfeld series [were] few and almost irrelevant"); *Twin Peaks Productions, Inc. v. Publications Intern., Ltd.*, 996 F.2d 1366, 1372 (2d Cir. 1993) (infringement of television series copyright shown by "comprehensive non-literal similarity" where book contained "detailed recounting of . . . episodes of the series").

- literal copies (if existing creative material is arranged and reproduced in a non-creative manner, *e.g.*, a collection of all poems by a particular author by titles in alphabetic order);[50] and/or
- an arrangement that is neither restricted nor protected by copyright law (if the combination does not involve any changes, duplication, or creative arrangement, *e.g.*, storage of books on a shelf sorted by the author's name in alphabetical order).[51]

These categories are not mutually exclusive: An author could translate poems (*i.e.*, prepare derivative works), add some poems without translation but edits for typographical corrections (*i.e.*, non-literal copies) and then creatively arrange the poems (*i.e.*, create a collective work).

3. Ownership of Derivative Works vs. Compilations

The Copyright Act treats compilations and derivative works similarly with respect to copyright subject matter and ownership rights:[52] The author of a derivative work or compilation owns the copyrights to her creative contributions, but not to the underlying work.[53] Consequently, the creator of a derivative work or compilation can exclude anyone—including the owner of the copyrights to the underlying work(s)—from copying, distributing, etc., the derivative work or compilation.[54]

If and to the extent a creator of a derivative work or compilation bases her work unlawfully on copyrighted works of others, however, she does not acquire any copyrights in the derivative work or compilation.[55] Also, a licensee who creates literal or non-literal copies of an existing work under license from the copyright owner does not acquire any copyrights to such copies.

4. Exclusive Rights to Derivatives Works vs. Compilations

Given the similar treatment of derivative works and compilations for ownership purposes, it is worth noting—and particularly relevant for the permissibility of software combinations—that the Copyright Act treats derivative works and compilations very differently with respect to the scope of exclusionary rights. A copyright owner has the exclusive right to prohibit or authorize (*i.e.*, license) the preparation of derivative works.[56] The copyright owner does not, however, have an exclusive right to prohibit or authorize the preparation of compilations or non-creative arrangements of works.[57] Thus, the Copyright Act specifically empowers the creator of artwork to prohibit a buyer of prints from selling tiles

---

[50] *Silverstein, supra* at 7 (grant of summary judgment for plaintiff on claim of infringement of poem compilation where defendant copied "nearly the entire work," including the original author's "selection and . . . guiding principles"), *reversed, vacated and remanded* 368 F.3d 77 (2d Cir. 2004) (summary judgment and injunction not appropriate because issue of fact existed as to whether plaintiff's arrangement was copyrightable in the first instance).

[51] *See, e.g., Paramount Pictures Corp. v. Video Broadcasting Systems, Inc.*, 724 F. Supp. 808, 821 (D. Kan. 1989) (Defendant had added ads for local business at beginning of movie videotapes. While both ads and movie were copyrightable works, adding the ads did not amount to "recasting, transforming or adapting the motion picture," *i.e.*, creation of a derivative work.)

[52] 17 U.S.C. § 103: "Subject matter of copyright: Compilations and derivative works. (a) The subject matter of copyright as specified by section 102 includes compilations and derivative works, but protection for a work employing preexisting material in which copyright subsists does not extend to any part of the work in which such material has been used unlawfully. (b) The copyright in a compilation or derivative work extends only to the material contributed by the author of such work, as distinguished from the preexisting material employed in the work, and does not imply any exclusive right in the preexisting material. The copyright in such work is independent of, and does not affect or enlarge the scope, duration, ownership, or subsistence of, any copyright protection in the preexisting material."

[53] 17 U.S.C. § 103(b).

[54] 17 U.S.C. §§ 103(b) & 106.

[55] 17 U.S.C. § 103. In this respect, creators of derivative works and compilations are in slightly different positions: The creator of a derivative work needs an authorization (*i.e.*, license) from the owner of the copyright to the underlying work(s) to create the derivative work in the first place—and then later to reproduce or distribute it. Such license has to specifically allow the preparation of derivative works. With respect to compilations, such a specific authorization is not required to ensure the lawfulness and thus acquisition of ownership rights in the compilation, *see infra,* Section II(6).

[56] 17 U.S.C. § 106(1)-(3).

[57] 17 U.S.C. § 106(1), (2) and (3) mention only derivative works, but not collective works or other types of compilations.

with framed prints attached,[58] but not from arranging unmodified prints in a creative or non-creative compilation, *e.g.*, side-by-side other artwork.[59] Consequently, it is the distinction between derivative works and other categories of combinations that is most crucial for software interoperability.

The special treatment of derivative works in contrast to compilations under copyright law (and in contrast to the treatment of improvements under patent law)[60] seems to have its origin in a recognition of the special relationship that authors traditionally have to their works: An author of a copyrighted novel, painting, or a symphony piece is affected in its personal and commercial interests if a publisher changes the work to add a happy ending, removes potentially offensive scenes from the painting, or a movement from the symphony. Such changes can affect the author's reputation and ability to commercialize future works if the public cannot easily separate the changes from the original work.[61] Transparent and detached combinations, on the other hand, *i.e.*, combinations that can easily be undone conceptually, such as compilations or improvements on patented inventions, affect the interests to the underlying intellectual property in a lesser way. Therefore, the requirement of internal changes to the adapted work seems to be an important definitional element for derivative works under copyright law, which justified the legislative grant of exclusion rights to the owner of the copyrights to the adapted work.

5. Fixation Requirement

The Copyright Act postulates a fixation requirement for ownership purposes, but does not specify how permanent a derivative work must be to be infringing.[62] In some cases, courts simply assumed that the same permanency threshold applies for ownership and infringement purposes.[63] In other cases, courts have asserted that a different standard applies for infringement purposes, albeit without clearly defining exactly what constitutes that standard.[64] Overall, however, it seems generally accepted that adaptations do not constitute derivative works for infringement or ownership purposes if they are fleeting and lack any significant permanency—*e.g.*, a work viewed through a pink filter.[65]

---

[58] *See Mirage Editions, Inc. v. Albuquerque A.R.T. Co*., 856 F.2d 1341 (9th Cir. 1988); *but see, also Lee v. A.R.T.*, 125 F.3d 580, 582 (7th Cir. 1997).

[59] Of course, a copyright owner can independently prohibit copying of its work, so any combinations that involve duplication require separate permission. Thus the creator of the artwork here could prohibit the buyer of the prints from arranging the unmodified prints in a book, since that would involve duplicating the prints.

[60] Mark Lemley, in discussing this issue, uses the term "improvers" to refer to those who make works or inventions based upon, to varying degrees, an underlying work or patent. He divides "improvers" into three categories: minor improvers, significant improvers, and radical improvers. *See generally* Mark A. Lemley, *The Economics of Improvement in Intellectual Property Law*, 75 TEX. L. REV. 989 (1997) (hereafter "Lemley, Economics of Improvement").

[61] This phenomenon is relevant beyond the context of continental European copyright laws that protect authors against distortion of their works, because even jurisdictions that consider an author's rights against distortion inalienable will typically allow an assignment or perpetual license to derivative works, *see, e.g*., Code de la propriété intellectuelle [Intellectual Property Code], art. L111-1, L113-1 (Fr.); *see* Lemley, Economics of Improvement, *supra,* at 1033; Ian Eagles and Louise Longdin, *Technological Creativity and Moral Rights: A Comparative Perspective*, 12 INT'L J.L. & INFO. TECH. 209 (2004); 2 Paul Goldstein, Copyright § 5.3 n.12 (2d ed. 2002); Edward J. Damich, *The Right of Personality: A Common-Law Basis for the Protection of the Moral Rights of Authors*, 23 GA. L. REV. 1, 41 (1988); *Lee v. A.R.T.*, 125 F.3d 580, 582 (7th Cir. 1997) (criticizing the 9th Circuit's decision in *Mirage Editions, Inc. v. Albuquerque A.R.T. Co*., *supra* note 62, as granting a backdoor moral right to authors via the derivative works doctrine); Amy B. Cohen, *When Does a Work Infringe the Derivative Works Right of a Copyright Owner?*, 17 CARDOZO ARTS & ENT. L.J. 623, 645 note 102.

[62] 17 U.S.C. § 101 defines when a work is "created," but 17 U.S.C. § 106.2 does not refer to the "creation," but rather to the "preparation" of derivative works, a more generic term that is used also in other sections of the Copyright Act.

[63] *See*, *e.g.*, *Sega*, *supra* at 1518.

[64] *See*, *e.g.*, *Galoob*, *supra* at 967-968; *Micro Star*, *supra* at 1110-1111. 2 Nimmer, *supra* note 15, at § 8.09[A] differentiates between infringements of the adaptation right through performances as opposed to copies.

[65] One court used the "low tech" example of a pink piece of cellophane placed in a frame over a television screen. *Micro Star*, *supra* at 1111 n.4; *see also* Tyler Ochoa, *Symposium Review: Copyright, Derivative Works and Fixation: Is Galoob a Mirage, or Does the Form(gen) of the Alleged Derivative Work Matter?*, 20 SANTA CLARA COMPUTER & HIGH TECH. L. J. 991 (2004); *but see* Lydia P. Loren, *The Changing Nature of Derivative Works in the Face of New Technologies*, 4 J. SMALL & EMERGING BUS. L. 57, 84 (2000); E. Nicolas, *Why the Ninth Circuit Added Too Much to Subtract Add-on Software from the Scope of Derivative Works Under 17 U.S.C. §106(2): A Textual Argument*, 2004 SYRACUSE SCI. & TECH. L. REP. 44, 45.

6. Summary

In addition to the right to control duplication, copyright owners have an express statutory right (adaptation right) to prohibit or permit combinations of their works with other materials if and to the extent such combinations qualify as derivative works. Combinations qualify as derivative works only if they are sufficiently permanent, contain significant amounts of existing copyrighted works and involve significant and creative changes to such pre-existing works. Combinations without internal changes can qualify as compilations (if the combination is creative), but unlike with derivative works, a copyright holder has no express statutory right to prohibit end users from making compilations.[66] Neither insignificant or non-creative changes to existing copyrighted works, nor entirely insignificant adaptations from existing works result in the creation of a derivative work—they instead result in the creation of non-literal copies or non-derivative new works respectively.

III. SOFTWARE UNDER COPYRIGHT LAW

The peculiar relationship between software and copyright law has already been well analyzed.[67] To lay the groundwork for the following analysis, however, it is helpful to briefly recall how two important principles of copyright law apply in the software context: (1) copyright law strikes a delicate balance between access and exclusion rights, and in doing so, (2) copyright law protects only creative expression, but not functionality, however valuable such functionality may be.

1. The Balance Between Access and Exclusion Rights

As contemplated by the U.S. Constitution,[68] the Copyright Act protects investments in creative works through exclusion rights in order to encourage further creation and public availability of such works.[69] Exclusion rights enable the owner to permit or prohibit the prescribed activities—and charge fees for permissions (licenses). The prospect of such license fees are intended to incentivize creators to create and adapt original works of authorship.

The public interest in creative works can be harmed by both under- and over-protection.[70] This is risk is particularly obvious with respect to the adaptation right, since most creative works borrow to some extent from existing material; overbroad adaptation rights could seriously stifle further developments.[71] Thus, legislatures and courts have over the years struck a delicate balance between granting and limiting

---

[66] The copyright holder does retain the right to authorize copying, so the compilation cannot involve any unlicensed copying of the underlying works. If an end user holds a license to execute and use two programs, however, and their execution in combination constitutes a compilation, such combination would not infringe the software copyright owners' exclusive rights under Section 106 of the Copyright Act. Where a combination qualifies as a derivative work, on the other hand, and all other things are equal, the end user would need a separate license to combine the programs; without such license, the combination would infringe the copyright owners' adaptation rights under Section 106(2) of the Copyright Act.

[67] *See, e.g.,* Mark A. Lemley*, Convergence in the Law of Software Copyright*, 10 HIGH TECH. L.J. 1, 3-6 (1995); Pamela Samuelson et al., *A Manifesto Concerning the Legal Protection of Computer Programs*, 94 COLUM. L. REV. 2308 (1994); Peter S. Menell, *Tailoring Legal Protection for Computer Software*, 39 STAN. L. REV. 1329 (1987).

[68] Article 1, Section 8 of the U.S. Constitution provides that "Congress shall have the power . . . [t]o promote the Progress of Science and useful Arts, by securing for limited Times to Authors and Inventors the exclusive Right to their respective Writings and Discoveries."

[69] *See* Pamela Samuelson, *Computer Programs, User Interfaces, and Section 102(b) of the Copyright Act of 1976: A Critique of Lotus v. Paperback*, 55-SPG LAW & CONTEMP. PROBS. 311, 339 (1992) (hereafter "Samuelson, Computer Programs").

[70] *See* Samuelson, *Computer Programs*, *supra* at 338-39.

[71] 2 Nimmer, *supra* note 15, at § 8.03.A. *See also* Daniel S. Hurwitz, *A Proposal in Hindsight: Restoring Copyright's Delicate Balance by Reworking, 17 U.S.C. § 1201*, UCLA J. L. & TECH. 1 (2005); Christine Jeanneret, Note, *The Digital Millennium Copyright Act: Preserving the Traditional Copyright Balance*, 12 FORDHAM INTELL. PROP. MEDIA & ENT. L.J. 157 (2001).

exclusion rights for authors and access rights for the public.[72] Copyright owners who upset this balance by abusing their rights can be penalized by the denial of copyrights under the doctrine of copyright misuse.[73]

2. Creative Expression vs. Functionality

As part of said balancing act, the Copyright Act takes great care to limit copyrightable subject matter to creative, artistic expression and to keep underlying ideas and functionality in the public domain:

> In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work.[74]

This express exception made software historically an unlikely candidate for copyright protection, given the fact that the value of most[75] computer programs lies in their functionality and efficiency.[76] Software licensees typically appreciate and pay for the speed, reliability and operational simplicity with which a particular program produces results, but not, for example, how creative the underlying code is written.

Yet, the threshold question whether software is protected by Copyright Law at all has long been settled in the United States.[77] For over three decades, U.S. courts have consistently found that computer programs (both object and source code versions) are generally protected under the U.S. Copyright Act.[78] The underlying code constitutes a literary work[79] whereas output such as screen displays and user interfaces can be protected separately as audiovisual works.[80] It is possible to generate the same screen displays and user interfaces with substantially different underlying code, so that a program's screen display may be infringing while the actual code is not.[81]

Despite the conceptual mismatch, copyright law has established itself globally as the primary intellectual property regime for software, and jurisdictions that initially rejected this approach have now adopted it.[82] Yet, since courts first made the decision that software was copyrightable, they have had to

---

[72] *See, e.g., Eldred v. Ashcroft*, 537 U.S. 186 (2003); *Sony Corp. of America v. Universal City Studios, Inc*., 464 U.S. 417 (1984); *Harper & Row Publishers, Inc. v. Nation Enterprises*, 471 U.S. 539 (1985). Where this line should be drawn has been the subject of extensive academic debate, especially in the last decade.

[73] *See* below Section V(3)(b); *Lasercomb America, Inc. v. Reynolds*, 911 F.2d 970, 977 (4th Cir. 1990) (hereafter "*Lasercomb*"); *see generally*, 4 Nimmer, *supra* note 15, at §13.09[A].

[74] 17 U.S.C. § 102(b).

[75] Video and computer games and other entertainment software are noteworthy exceptions.

[76] David G. Luettgen, *Functional Usefulness vs. Communicative Usefulness: Thin Copyright Protection for the Nonliteral Elements of Computer Programs*, 4 TEX. INTELL. PROP. L.J. 233, 249-60 (1996).

[77] *See*, *e.g.*, *Harcourt, Brace & World, Inc. v. Graphics Controls Corp*., 329 F. Supp. 517 (S.D.N.Y. 1971) (hereafter "*Harcourt*"); *Apple Computer, Inc. v. Franklin Computer Corp*., 714 F.2d 1240 (3d Cir. 1983), *cert. denied*, 464 U.S. 1033 (1984) (hereafter "*Franklin Computer*"); for an overview, *see*, Mark Lemley, Peter Menell, Paul Merges and Pamela Samuelson, *Software and Internet Law* (1st ed. 2000), pp. 1-45, 97-8; 2d ed., pp. 33-5.

[78] *See*, *e.g.*, *Harcourt*, *supra*; *Williams Electronics, Inc. v. Artic International, Inc.*, 685 F.2d 870 (3d Cir. 1982); *Franklin Computer*, *supra*.

[79] *Franklin Computer*, *supra* at 1246-48.

[80] *Stern Electronics, Inc. v. Kaufman*, 669 F.2d 852 (2d Cir. 1982) (hereafter "*Stern Electronics*"); *Data East USA, Inc. v. Epyx, Inc.*, 862 F.2d 204 (9th Cir. 1988); *Apple Computer, Inc. v. Microsoft Corporation*, 35 F.3d 1435 (9th Cir. 1994) (hereafter "*Apple v. Microsoft*").

[81] *See Altai*, *supra* at 703, and *Lotus Development*, *supra* at 80 (user interface command hierarchies); for an analysis of the decision, *see* Dennis S. Karjala & Peter S. Menell, *Applying Fundamental Copyright Principles to Lotus Development Corp. v. Borland International, Inc*., 10 HIGH TECH. L.J. 177 (1995).

[82] An example is Germany: Despite the fact that software had been expressly recognized in Section 2 of the German Copyright Act as a category of copyrightable works since 1985, prior to the implementation of the EC Software Directive into German law in 1993, German courts had required a very high level of originality before they would afford copyright protection for software. The leading cases are from 1985 and 1991: 1985 BGH, GRUR, 1041 (known as the collection program case—"Inkasso-Programm") and 1991 BGH, NJW, 1231 (known as the operating system case—"Betriebssystem"). Many programs that would have easily qualified as copyrightable in the United States were not found to be so in Germany. For an overview of software copyright protection in the European union, *see* Pamela Samuelson, *Comparing U.S. and EC Copyright Protection for Computer Programs: Are They More Different Than They Seem?,* 13 J.L. & COM. 279 (1994).

struggle with the fundamental problem that copyright law is designed to protect creative expression as an incentive for further creative activity, whereas the value in software is usually measured by functionality and efficiency,[83] *i.e.*, aspects that are expressly carved out from copyright protection.[84] In the process, courts complained about having to fit a square peg into the round hole[85] and developed a number of tests and approaches to separate protectable creative elements from non-protectable functional elements in software.[86] Creative elements are protected against literal and non-literal copying, whereas functional elements are in the public domain and can be freely duplicated, even where idea and expression or functionality and creativity merge—*e.g.*, because a particular technical solution can be programmed efficiently only in one particular manner.[87]

Courts have defined the dividing line on a case-by-case basis, in light of the underlying public policy considerations. Artistic screen displays of computer games (with fantasy figures and landscapes)[88] bear a greater resemblance to traditional subjects of copyright protection (like novels and paintings) than the zeros and ones that constitute object code or the functionality-driven user interfaces for application programs.[89] Therefore, computer game screen displays have generally fared better in cases where non-literal copying and the idea-expression dichotomy issue has been raised.

Some courts had to confront situations where companies used otherwise creative works in a purely functional manner, *e.g.*, as interfaces, passwords, or lock-out mechanisms. Courts denied copyright protection for such works, regardless how creative and original they were, in the interest of preserving the balance between protection and access rights described above.[90] Thus, as a general matter, copyright owners cannot rely on the protections afforded by copyright law where they deploy copyrighted works in a software context for the sole purpose of forcing others to infringe (by copying or adapting copyrighted code) in order to establish interoperability.[91]

Other courts tried to stretch the boundaries of copyright law in order to protect investments, even where the Copyright Act did not literally cover the material or activities that concerned the plaintiffs.[92] Since the U.S. Supreme Court vehemently rejected the "Sweat of the Brow"–Doctrine in 1991,[93] however, it is important to start with thoroughly examining the creativity vs. functionality dichotomy in any software copyright analysis and filter out ideas, processes, methods, facts, elements dictated by external factors or efficiency, material in the public domain, expression which has "merged" with any of

---

[83] *See Bucklew*, *supra* at 928.

[84] *Altai*, *supra* at 712. *See also* Mark A. Lemley & David W. O'Brien, *Encouraging Software Reuse*, 49 STAN. L. REV. 255 (1997); Peter S. Menell, *An Analysis of the Scope of Copyright Protection for Application Programs*, 41 STAN. L. REV. 1045 (1989); Pamela Samuelson et al., *A Manifesto Concerning the Legal Protection of Computer Programs*, 94 COLUM. L. REV. 2308 (1994); *cf.* Jane C. Ginsburg, *Four Reasons and a Paradox: The Manifest Superiority of Copyright Over Sui Generis Protection of Computer Software*, 94 COLUM. L. REV. 2559 (1994) ("hereafter Ginsburg, Four Reasons"); Arthur R. Miller, *Copyright Protection for Computer Programs, Databases, and Computer-Generated Works: Is Anything New Since CONTU?*, 106 HARV. L. REV. 977 (1993); Stacey L. Dogan and Joseph P. Liu, *Copyright Law and Subject Matter Specificity: The case of Computer Software,* 61 N.Y.U. ANN. SURV. AM. L. 203, 204.

[85] *See, e.g., Altai*, *supra* at 712.

[86] *Altai*, *supra* at 714; *Gates Rubber Co. v. Bando Chem. Indus., Ltd*., 9 F.3d 823 (10th Cir. 1993) (hereafter "*Gates Rubber*"); *Lotus Development*, *supra,* at 58; *Lexmark*, *supra; Skylink*, *supra*.

[87] *See Computer Associates*, *supra* at 712.

[88] *See*, *e.g.*, *Micro Star*, *supra* at 1110.

[89] *See*, *e.g.*, *Lotus Development*, *supra*; *Altai*, *supra*.

[90] *Sega*, *supra*; *Skylink*, *supra*; *Lexmark*, *supra*.

[91] *Sega*, *supra*, *Lexmark*, *supra.*

[92] *See*, *e.g.*, *Midway*, *supra* at 1014 ("[T]he amount by which the language of Section 101 must be stretched to accommodate speeded-up video games is, we believe, within the limits within which Congress wanted the new Act to operate.").

[93] *Feist*, *supra.* Also, *see Bateman v. Mnemonics, Inc*., 79 F.3d 1532, 1543-45 (11th Cir. 1996); *Apple v. Microsoft*, *supra*; *Engineering Dynamics, Inc. v. Structural Software, Inc.*, 26 F.3d 1335 (5th Cir. 1994); *Kepner-Tregoe, Inc. v. Leadership Software*, 12 F.3d 527 (5th Cir. 1994); *Gates Rubber*, *supra*; *Atari Games Corp. v. Nintendo*, 975 F.2d 832 (Fed. Cir. 1992); *Control Data Systems, Inc. v. Infoware, Inc.*, 903 F. Supp. 1316 (D. Minn. 1995); *CMAX/Cleveland, Inc. v. UCR, Inc*., 804 F. Supp. 337 (M.D. Ga. 1992); *Brown Bag Software v. Symantec Corp*., 960 F.2d 1465, 1475-76 (9th Cir. 1992); *MiTek Holdings Inc. v. Arce Engineering Co*., 89 F.3d 1548 (11th Cir. 1996); Mark A. Lemley, *Convergence in the Law of Software Copyright*, 10 HIGH TECH. L.J. 1 (1995).

the foregoing, and expression which is so standard or common as to be a "necessary incident" to any of the foregoing.[94]

This filtering technique is equally important for the characterization of software combinations as derivative works, compilations or other aggregations. Given the utilitarian nature of software, modifications that are dictated by external factors or functionality considerations do not generally affect the special relationship between the author and its creative work, which adaptation rights are intended to protect under copyright law (in contrast to patent law, which does not grant exclusive rights to improvements in order to avoid stifling innovation in functionality). Hence, courts have to be particularly careful to focus on creative expression—as opposed to functionality—when drawing the line between combinations that infringe adaptation rights (*i.e.*, derivative works) and combinations that do not (because they constitute compilations or other aggregations that are generally permissible without the copyright owner's consent). Hence, only internal changes to code that cannot easily be separated or distinguished by software users or other developers would seem capable of affecting the adaptation rights of an author, whereas, for example, combinations with separable add-on programs or interface modifications dictated by functional requirements should generally not be considered to create derivative works.

3. Summary

Given its typically utilitarian nature, software usually contains many elements that are excluded from copyright protection and need to be filtered out at the outset of any copyright analysis. Consequently, the coverage and strength of copyright protection for software varies depending on its different levels and functionalities. The screen output of computer games, for example, can be highly creative and thus enjoy similarly strong protection as more traditional works of authorship (such as paintings, novels or musical compositions). The underlying code of the same games, on the other hand, is usually more functional in nature, derives value from execution efficiency as opposed to artistic creativity, and contains a thinner layer of copyrightable expression (on uncopyrightable functionality). Interfaces and lock-out mechanisms are excluded from copyright protection if and to the extent duplication or adaptation is necessary to overcome such mechanisms and establish interoperability between programs (because idea and expression merge under such circumstances).

Thus, the general test formulated at the end of the previous Section II of this Article can be further refined: A combination of a copyrightable computer program with another constitutes a derivative work of the program(s) if the combination (a) is sufficiently permanent, (b) contains significant and creative portions of the program(s) (as opposed to interface information or purely functional incorporation of lock-out code), (c) is creative in its own right (as opposed to representing the only or most efficient technical combination solution), and (d) involves significant and creative internal changes to the other program that cannot be easily separated or distinguished from the other program. If the adapted portions or the changes caused by the combination are not creative, but merely functional in nature, the combination does not constitute a derivative work. If a software combination does not involve such qualified internal changes to the combined program(s), it may constitute a compilation (or fall outside the scope of copyright law altogether).

IV. SOFTWARE COMBINATIONS UNDER COPYRIGHT LAW

In this Section, the abstract rule developed in the previous Sections is illustrated and further refined through application to a few common technical and commercial scenarios. A number of definitions and general explanations (1.) are followed by five hypotheticals[95] (2. through 6.) and a summary (7.).

---

[94] *Gates Rubber*, *supra* at 837.

[95] The hypotheticals are based on factual scenarios discussed in the sparse existing case law, but simplified to address only issues relevant for purposes of this Article and expanded to allow a review of surrounding questions. The assessment of the hypotheticals generally comes to the same ultimate conclusions as the courts did in the underlying cases, except, however, that it

1. Software Combinations viewed from different Technical Perspectives

Before an attempt is made to shed some light on technical details, it is important to acknowledge the limitations of this effort. Computer science and information technology are complex and rapidly evolving disciplines and much of the applicable terminology, details and context are specific to particular programming architectures. Thus, this Article has to address technical details in a simplified and exemplary manner, guided by the objective to illustrate the relevant legal principles. Also, along those lines, this Article refers to different types of software loosely as "software" or "computer programs" and mentions characteristics of applications, libraries, drivers, operating systems and other categories of software only to the extent it is relevant for purposes of the legal analysis. Subject to these caveats, it seems important to realize at the outset of any copyright analysis that (a) the installation and execution of any computer program involves countless instances of copying, in excerpts, mixed up with bits of other software, dictated by data storage efficiency requirements, and (b) interoperations between programs can be achieved in many technical ways, which are dictated by program interoperability and efficiency requirements.

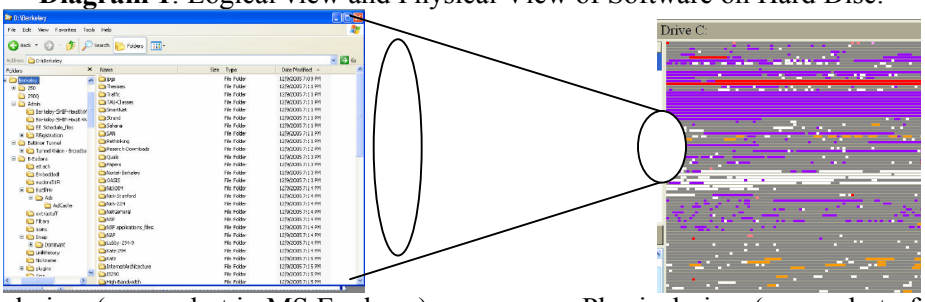
a) Physical and Logical View of Computer Programs in Combination

Software users are used to the logical view of programs, which is provided and managed by the Operating System's (OS) File System (FS).[96] In the Microsoft Windows environment, users can see each application (*e.g.*, Adobe Reader, Microsoft Word, Outlook and PowerPoint) separate and distinguishable as installed (through Windows Explorer) and as copied to RAM and executed by the CPU.

Physically, however, the bits of data comprising the various computer programs are located on various different places on the media providing the memory space, *e.g.*, a memory stick, floppy disc, CD, computer hard disc (*e.g.*, C-drive), RAM and CPU cache memory. The physical arrangement of the data bits serves purely technical storage efficiency considerations and bears little or no relationship to the logical function of the various programs. In order to save space and preserve execution speed, each of the different types of memory usually contains different amounts of code of a particular program.

A good example is a typical software installation from CD to a personal computer's hard disc, followed by execution of the program from the hard disc. The full amount of the applications' electronic data (object code, date files, libraries) comes on a CD set. Based on whether the installation is full or partial, some portion of this electronic data is installed (copied) on the hard drive, where it is parceled up and placed into various spaces on the disc (entirely unrelated to the logical context of the application; bits and pieces of code related to Word will sit right next to, and intermingled with pieces of code related to Outlook and Adobe Reader).

Diagram 1: Logical view and Physical View of Software on Hard Disc.



Logical view (screenshot in MS Explorer) Physical view (screenshot of Speed-Disc)

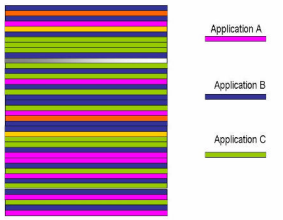
is not entirely clear whether *Dun & Bradstreet* addressed static (as assumed herein) or dynamic linking and whether the courts in *Micro Star* and *Midway* relied (or should have relied) on direct or indirect (as assumed herein) infringement.

[96] File System (FS) is a method for storing and organizing files on a storage device. FS provides service of hiding the physical structure of the disc hardware from the Operation System http://en.wikipedia.org/wiki/File_system.

During the application execution, only a relatively small portion of this data is usually loaded from the hard drive into RAM, because not all portions of the program code are typically needed in RAM.[97] The computer processor (CPU) is accessing the electronic data on the RAM and during the operation will copy some smaller amount of the data into the cache memory of the CPU (L1 or L2 cache),[98] where the data resides only for extremely short time periods and can be accessed and executed much faster than in RAM. During the operation of the CPU, much less data is copied into the CPU registers, a minuscule amount of memory inside the CPU.[99] The closer memory is to the CPU, where all data is processed via "current" and "no current" signals, the more accessible (faster) it has to be and the smaller in size it usually is, and stored to a shorter time.

If one were to draw up a physical map to identify where the various portions of a particular program reside on a hard drive or in RAM, alone or in combination with other programs, one would find them mixed up with bits and pieces from other programs all over the storage media—like dozens of CD pieces end up mixed up on a pavement after falling from a 12 story building or a truck load of books and newspaper stamped into pulp in a recycling factory.[100]

Diagram 2: The physical address space of applications is mixed on RAM.



In order to ensure that program data storage and execution is efficient and that separate programs do not interfere on a logical level, the Operating System (OS) and its components allocate different memory spaces to each program on the hard drive, in RAM and cache, and translate between the human view and the computer view. Often, the amount of memory needed by applications is not available in RAM and memory pages are copied back and forth between the RAM and the hard drive. While running many applications or applications that require more memory than is available in RAM, personal computer users can see the disc light blinking and hear the disc drive working. This copying back and forth between the RAM and the hard disc called "paging" and is part the virtual memory[101] mechanism. Virtual memory addressing allows the operating system to manage the memory with non-contiguous address space in the interest of storage and execution efficiency: the cost of address space on RAM is much more expensive than the cost of address space on hard disc (while RAM access is much faster and available to the CPU).

---

[97] In general terms, there are differences in the amounts of space and availability relatively in a hard disc, RAM, cache and registers. For example, Microsoft Office suite has many features that are not required in the day-do-day operation. Most Word installations do not need to copy Equations Editor and not all design templates are needed in Power Point. Thus, for execution (*i.e.*, use), computers create "excerpts" of software, based on purely functional and efficiency considerations.

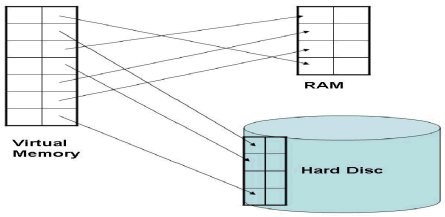
[98] In today's CPU, L2 cache is a size of about 256-512KB. L1 cache is in a smaller size of about 32KB. http://www.freescale.com/files/32bit/doc/app_note/AN2663.pdf (accessed 2/2/2006). http://en.wikipedia.org/wiki/CPU_cache.

[99] CPU Register is a very fast memory providing quick access to commonly used values, see http://en.wikipedia.org/wiki/CPU_register (accessed 2/2/2006).

[100] *See* the physical view to the right in Diagram #1.

[101] Computer Organization & Design. The Hardware Software Interface. David Patterson. & John Hennessy. Parallel Computer Architecture, Hardware/Software Approach. David Culler. Chapter 11.6, page 877; Chapter 7. page 538.

Diagram 3: Virtual memory address space on RAM and hard disc.



These examples illustrate that from a purely physical perspective, computer program code is constantly copied, in excerpts, and intermingled with excerpts of other code, on a hard disc, in RAM and in CPU cache. These activities are dictated entirely by external functionality requirements (storage and execution efficiency) and completely unrelated to particular software applications and the creative expression embodied in them. Therefore, the dissection and combination events occurring on a physical level seem generally irrelevant for purposes of copyright law analysis. Instead, the copyright analysis has to focus on the logical perspective, which views programs as separate works based on how they are written by their authors and perceived by their users.

b) Interoperations and Data Communications between Programs

Interoperability and communications between applications in a computer system are mainly realized through Inter-Process Communication (IPC). A number of different mechanisms are available, including the following: Signals provide one-way asynchronous communications between application processes; a process can send a signal to a target process, and the operating system generates an event to the target application; the target application may have signal handler to handle those signals respectively for further actions. Named pipes are a pre-defined pipeline in the computer system, and provide a one-way communication between application processes; usually two applications attach to a named pipe; one application writes to the pipeline while the other application read from the pipeline. A file can be used by an application to input and output data, in binary or text format. Application communications can be realized by opening the file to read or write data. Shared memory[102] uses a section of RAM to store data that is shared between applications; applications can access the shared memory to read and write data; it provides a communication mechanism between applications. Sockets[103] provide a data transport application program interface (API) for a network; a socket is generally bound to the network transport protocol such as TCP/IP, including a host IP address, a protocol (TCP or UDP),[104] and a port number; typically, an application opens a socket and listens on a certain port number; another application which can run locally or remotely opens a socket and connects to the listening sockets; after the connection is completed, the two applications can perform a 2-way communication, thus they can read and write to each other. All these IPC mechanisms achieve different levels of communication and interoperability efficiency, but programs that are combined through such IPC mechanisms remain separate and distinguishable in the logical program view. The selection of the most appropriate IPC is dictated by external functionality requirements (which IPC will allow efficient interoperability?). Hence, programs so combined would generally not seem to become part of a larger derivative work due to a lack of combination creativity and significant internal changes, as will be further illustrated in the following hypotheticals.

[102] David Culler, Parallel Computer Architecture, Hardware/Software Approach, Chapter 32, page 484. Douglas E. Comer. Chapter, Computer Networks and Internets. Chapter 11.6, page 877.

[103] Richard Stevens, *Unix Network Programming*, Volume 1, Chapter 3.

[104] Larry Peterson & Bruce S. Davie, *Computer Networks a System Approach*, Second Edition, Chapter 5, section 5.1 and 5.2.

2. Package with Programs of Isolated Functionality

a) Basic Hypothetical

A reseller buys application programs from two software manufacturers. The applications have isolated, independent functionality (*e.g.*, two alternative photo editors) that do not interact with each other. Each manufacturer ships its respective program on pre-packaged CDs to the reseller. The reseller takes one pre-packaged CD from each manufacturer and places them in a card-board box to sell them in combination. Consumers purchase the two-CD packages and usually install only one of the two alternative programs on a personal computer.

b) Assessment

The combination of the programs in a sales package is too loose and detachable,[105] and not sufficiently creative, to qualify as any work for purposes of copyright law. Thus, due to its lack of fixation and originality, the combination would not qualify as a derivative work, collective work or compilation. Also, neither the reseller nor any consumers make any changes to the programs and the consumers do not typically install the programs on the same hardware or execute programs in the same RAM. Thus, the programs do not modify each other even temporarily. This is another reason why neither the reseller nor the consumers create derivative works by distributing, installing or executing the programs in combination.

c) Variations

The assessment should not change if the reseller receives golden master copies from the two software manufacturers and copies two programs on one CD or the hard drive of a personal computer. The fixation requirement for a compilation would be met under such circumstances. Yet, the selection could still not be qualified as creative enough in nature to qualify as a copyrighted compilation. Physically, the programs' individual bits of data would stay fairly separate on the CD (where data is organized sequentially), but be "mixed up" on the hard drive of the personal computer. The sequential arrangement of data bits on the CD and the "mix up" of data bits on the PC hard drive, at the physical level, do not involve even a spark of creativity with respect to the software itself, because the storage arrangements are dictated entirely by technical storage efficiency and functionality requirements. Therefore, the physical arrangement lacks creative internal changes required to qualify as a derivative work. From a logical perspective, the two programs remain entirely separate and detachable on either disc (CD and hard drive) and will be perceived and interpreted by any computer user as separate, unrelated programs.

3. Modification of Screen Output through Software Combination

a) Hypothetical[106]

Company A makes video games that consist of three programs, which are delivered as separate object code files on a CD: a game engine program, a source art library and a MAP file. Consumers who purchase a game CD receive an express license to install and play the game (uploaded into RAM of a computer), but not a right to create derivative works. Printed on a piece of paper, the object code of each of the three programs would appear as thousands of lines of zeros and ones; the source code would appear as a set of instructions to a machine to create a certain screen output, which audio-visualizes the story of a fighter, who is to some extent controlled by the game player and who has to confront various enemies and challenges in numerous fantasy worlds (depicted by different background environments and game levels).

[105] *See*, *Bucklew*, *supra* at 930 and above footnote 35.
[106] The details of this hypothetical are adapted from *Micro Star*, *supra*, *Galoob*, *supra*, and *Midway*, *supra*.

In the RAM of a computer, the three programs' individual instructions are stored in different address spaces where they are accessed by the CPU for execution. During execution, the programs automatically call on each other through links embodied in the programs. The game engine program tells the computer when to read data, save and load games, play sounds and project images onto the screen. In order to create the audiovisual display for a particular game level, the game engine calls the MAP file, which contains a series of instructions that tell the game engine (and, through it, the computer) what artwork from the source art library (*e.g.*, a mountain, city or tree) to display where on the screen. For instance, the MAP file might say scuba gear goes at the bottom of the screen. The game engine then goes to the source art library, finds the image of the scuba gear, and displays it at the screen location specified by the MAP file. The MAP file describes the level environment in painstaking detail, but it does not actually contain any of the art itself; everything that appears on the screen actually comes from the source art library program. The creation of the game's audiovisual display functions similar as a paint-by-numbers kit: The MAP file tells the engine program to put blue paint in section number 565, but it does not contain any blue paint itself; the blue paint comes from a palette, which is the low-tech analog of the source art library, while the painter plays the role of the game engine.[107]

Company B enters the market and sells add-on products that enable consumers to manipulate the screen output of Company A's games, by speeding up the game sequence,[108] changing existing parameters (speed or strength of game characters),[109] or correcting errors.[110] Company C creates new MAP files that contain instructions for alternative combinations of game level environments (*e.g.*, with new creative arrangements of backgrounds and characters that have to be confronted).[111] The underlying code of the add-on programs does not bear any resemblance to Company A's code, neither at the source nor object code level.

Consumers purchase and install copies of the add-on programs developed by Company B and Company C on the same computer as Company A's games and execute them simultaneously. Each of the add-on programs is executed in a different memory address space of the computer's RAM. They provide additional instructions to the computer, which results in modified screen outputs during the execution of the programs. Depending on how the game is played, the computer will execute the programs in different sequences and variations. When consumers finish playing, they close the programs and no permanent changes remain in the copy of Company A's game that is installed on the hard disc.

b) Assessment

The object and source code of Company A's game engine program and MAP file each constitute a literary work; assuming a minimal creative spark, these programs are sufficiently original to enjoy protection against literal copying.[112] The source art library program on the other hand constitutes a collection of items (namely instructions to create the various background items), which are arranged in a manner designed to optimize access by the MAP file and game engine program; while individual items may be copyrightable, their arrangement seems purely functional and thus not sufficiently creative to constitute a collective work or other compilation.[113]

When executed in combination, the three programs create game sequences with a reasonably consistent plot, set of characters and background variations.[114] This screen output is protectable as an audiovisual work.[115]

---

[107] *Micro Star*, *supra* at 1110.

[108] In *Midway*, *supra*, this result was achieved through a chip as opposed to software, which is not relevant for purposes of copyright law, because 17 U.S.C. § 106.2 prohibits any adaptations regardless of the technical means of creation.

[109] *Galoob*, *supra* at 967.

[110] *Galoob*, *supra* at 969 addresses add-on spell-check programs in a dictum.

[111] *Micro Star*, *supra* at 1110.

[112] *See* above Section II(1).

[113] *See* above Section III(3).

[114] *See generally*, Stern Electronics, *supra*.

After the development process,[116] Companies B and C distribute their add-on programs independently from Company A's games. The add-on programs do not contain any expression taken from Company A's copyrighted games—neither at the code nor screen output level.[117] Since derivative works and compilations by definition require content from other works, the add-on product themselves cannot qualify as either.[118] Nevertheless, since B and C make, advertise and distribute their add-on programs for the sole intended purpose that users combine the add-on programs with A's games, B and C are contributorily liable for their customers' actions.[119]

Consumers execute the add-on programs simultaneously with the game programs to modify the screen output of the games. These combinations affect the games at two levels: (i) the screen output, which is protected as an audiovisual work and (ii) the underlying code, which is protected as a literary work. The combination of each add-on program with the game would be illegal if an unauthorized derivative work is created at either level. Whether the combination creates a derivative work depends on whether the combination (a) is sufficiently permanent, (b) contains significant and creative portions of the program(s), (c) is creative in its own right, and (d) involves significant and creative internal changes to the other program that cannot be easily separated or distinguished from the other program.

(i) *Screen-Output Level*

(aa) Permanence

By deploying Company B's add-on products, the consumers do not permanently modify Company A's game screen outputs; at the end of the combined use of Company A's and Company B's products, Company A's software is back to "normal" and if the users execute the games next time without the add-on programs, the screen output will be exactly as designed by Company A. Thus, the consumers do not create permanent derivative works through the combination with the add-on programs. While the games and add-on programs are in RAM, however, users can manipulate the speed and other aspects of the games through the add-on products made by Company B and the MAP files supplied by Company C provide new game level background environments. RAM copies are generally sufficiently permanent fixations for purposes of copyright law.[120] Yet the manipulations made by players through the add-ons supplied by Company B are fleeting in nature, will be different during each game session and thus, lack the minimum permanence required to qualify as derivative works.[121] The MAP files supplied by

---

[115] 17 U.S.C. § 102(a); *Stern Electronics, supra* at 855-856.

[116] It seems likely—yet outside the scope of the hypothetical—that in the process of developing its products, Company B may have modified the copyrighted software of Company A products. However, such preparatory activities are subject to a number of different considerations, including the fair use doctrine, *see*, *e.g.*, *Sega*, *supra* at 1520; *Connectix*, *supra* at 602; and 17 U.S.C. §§ 107 & 1201(f).

[117] In both *Micro Star*, *supra*, and *Midway*, *supra*, the court did not explain whether add-on products even had a user-interface or any independent screen output.

[118] *See*, *Galoob*, *supra* at 969 (distinguishing Midway by pointing out that Galoob's add-on module for the Nintendo system, the Game Genie, "does not physically incorporate a portion of the copyrighted work [such as the Nintendo console or games]." *But see* Mitchell L. Stoltz, Note, *The Penguin Paradox: How the Scope of Derivative Works in Copyright Affects the Effectiveness of the GNU GPL*, 85 B.U. L. REV. 1439, 1458 (2005) (hereafter "Stoltz, Penguin Paradox"). In both *Micro Star*, *supra*, and *Midway*, *supra*, the courts seemed less concerned with this requirement and focused on the respective "story lines" being retold by the makers of the add-on. However, the facts in either case would also seem to support a finding of infringement based on a contributory liability theory and in that context, the derivative work could have been the audio-visual work created by the end-user through the combination of the add-on program with the original game. Thus, it does not seem that these cases create authority for a general proposition that an add-on program on its own—outside the context of the intended combination work--can constitute a derivative work or compilation.

[119] *See Metro-Goldwyn-Mayer Studios, Inc. v. Grokster, Ltd*., 125 S. Ct. 2764 (2005); *A&M Records, Inc. v. Napster, Inc*., 284 F.3d 1091 (9th Cir., 2002); *Sony Corp. of America v. Universal City Studios, Inc*., 464 U.S. 417 (1984).

[120] *MAI Systems*, *supra* at 518 ("After reviewing the record, we find no specific facts (and Peak points to none) which indicate that the copy created in the RAM is not fixed.").

[121] According to *Galoob*, *supra* at 967, fixation is only relevant for the question whether derivative works are protectable, not whether they infringe; however, in order to infringe, derivative works have to exist in a "concrete or permanent" form; *see also Micro Star*, *supra* at 1110-1111. 2 Nimmer, *supra* note 15, at §8.09[A] also requires "permanence."

Company C on the other hand are permanent in nature and always create the same art combinations in conjunction with the source art library files. Therefore, the MAP files create combinations that are sufficiently permanent to qualify as derivative works or compilations.[122]

(bb) Substantial and Creative Portions

Upon execution of the add-on products, the screen outputs still consist only of artwork created by Company A, because all background items originate from Company A's source art library program. Some of the individual art items are likely sufficiently creative to qualify for copyright protection. Therefore, the screen outputs created by the combination of the game programs and any of the add-on programs contain substantial and creative portions from Company A's copyrighted programs.

(cc) Combination Creativity

The next question then is whether the changes to the screen outputs created by the combination of the games and the add-on programs are merely functional or sufficiently creative. Simple changes to the speed of program execution,[123] game character parameters[124] and error corrections[125] caused by Company B's add-on programs will typically not meet even minimal creativity requirements whereas the detailed re-arrangement of background motives from the source art library program caused by Company C's program would typically be sufficient.[126]

(dd) Significant Internal Changes

Whether or not the combination causes internal changes to existing copyrighted works determines whether the combination can qualify as a derivative work (as opposed to a compilation or other arrangement). By rearranging the items or sequence of events on the screen outputs designed by Company A, all add-on products apply internal changes to the audio-visual work(s) created by Company A. In the case of Company C's MAP files, the changes seem significant whereas the mere manipulation of the playing speed caused by Company B's add-ons would likely not suffice.[127]

(ee) Conclusion

When executed in combination with the games, Company C's add-on program creates a derivative work of Company A's screen outputs. Combinations with Company B's add-on programs on the other hand lack sufficient permanence, significant internal changes and creativity to qualify as derivative works. Therefore, consumers – and contributorily, Company C--would infringe on Company's A's adaptation right to the screen outputs by deploying Company C's add-on program, whereas Company B does not infringe on Company A's adaptation right.

(ii) *Underlying Code*

Given how different the software at issue looks at the code and screen output levels, it is not surprising that the characteristics of the respective combinations also show some distinctions:

---

[122] *Cf.*, *Stern*, *supra* at 856.

[123] *Galoob*, *supra* at 969. In 1983, the 7th Circuit came to the same conclusion regarding a lack of creativity, yet ultimately found for infringement due to a perceived need to honor the equities involved in the case outside the scope of the Copyright Act: "[T]he amount by which the language of Section 101 must be stretched to accommodate speeded-up video games is, we believe, within the limits within which Congress wanted the new Act to operate," *Midway*, *supra* at 1014; such a pure economic analysis does not seem justifiable anymore post-Feist.

[124] *Galoob*, *supra* at 967.

[125] *Galoob* , *supra* at 969, addresses add-on spell-check programs in a dictum.

[126] *Micro Star*, *supra* at 1112.

[127] For instance, a DJ would not create a derivative work by playing at 45 RPMs a musical record designated for 33 RPM speed, *see Midway* at 1013. Similarly, small add-on toolbars to Internet browsers or additional menu features for other programs may not qualify as "significant" in the generally functional user interface environment*, see generally, Lotus Development*, *supra* at 815-816.

(aa) Permanence

The combination of Company A's game engine code and the add-ons made by Companies B and C depends much on user interaction and does not meet the permanence requirement at the code level either. The same is true for any combinations that occur in CPU cache given the fleeting nature these extremely short-lived copies in the execution process.[128] The interaction between Company C's MAP file and the source art library on the other hand, occurs in a similar way whenever the two programs are executed in combination, which indicates some form of permanency. Yet, given the fact that the RAM copies and execution sequences are logically separated in RAM (in different address space) and on the hard drive of the computer, the combination of the executables appears too detached[129] and separate to qualify as a permanent combination for purposes of constituting a derivative work or compilation.

(bb) Substantial and Creative Portions

If one were to focus on the physical view and consider the functionally segmented aggregation of programs in RAM a permanent combination—contrary to the preceding assessment—the substantiality requirement would be met. During execution, the RAM contains large portions of Company A's code, which contains at least some creative portions.

(cc) Combination Creativity

The question whether the changes to the RAM copy created by the combination of the games and the add-on programs are merely functional or sufficiently creative must be examined separate from the question whether the changes to the screen output are functional or creative.[130] In this context, the idea-expression merger phenomenon[131] comes into play: If there is only one efficient mechanism to manipulate game sequence, change game character parameters, correct errors, or compile level background artwork, then the software implementation of this mechanism cannot qualify as creative for copyright purposes.

(dd) Significant Internal Changes

Whether or not the combination causes significant and creative internal changes to existing copyrighted works determines whether a combination that otherwise qualifies as a copyrightable work will qualify as a derivative work or a compilation. The screen outputs of Company A's games constitute works in their own right, and by rearranging the items or sequence of events on the screen outputs, all add-on products apply internal changes to the work(s) created by Company A.[132] On the code level, however, the situation looks quite different:[133] From a logical perspective, Company A's underlying programs and subprograms remain separate literary works, consisting of instructions to a computer which are formed and arranged for a purely functional reason: namely to cause the desired changes on the screen output level. Company C's MAP file does not make any internal changes to the source art library program, it merely calls individual items from the library into RAM in a different manner than Company A's MAP file would have. Therefore, at the code level, Company C's MAP file lacks another requirement to qualify as a derivative literary work of Company A's game program code: significant and creative

[128] *See*, *Costar v Loopnet*, 373 F.3d 544, 550-2 (4th Cir. 2004) regarding quantative and qualitative thresholds regarding the 'fixation' requirement.

[129] *See*, *Bucklew*, *supra* at 930 and above footnote 35.

[130] The latter question has been examined above, Section IV(2)(b)(i)

[131] *See* above Section III(2).

[132] *See* above Section IV(2)(b)(i)

[133] When Micro Star raised this point, the court clarified that the infringed work at issue was the audiovisual screen output and story line, thus the fact that the underlying code was not changed did not matter for the outcome of the case. *Micro Star*, *supra* at 1112. The courts in *Sega*, *supra*, and *Connectix*, *supra*, did not even discuss a potential adaptation right infringement through code combinations during the execution phase.

internal changes. Consequently, at the code level, Company C's program constitutes a new (non-derivative) work.

(ee) Conclusion

At the code level, combinations with the add-on programs do not qualify as derivative works, because the combination lacks permanency and typically also combination creativity (the means to achieve interoperability are dictated by externalities). Even if any combination creativity was present, due to the lack of internal changes, the add-on programs could only qualify as compilations (which would not affect Company A's adaptation right under Section 106.2 of the Copyright Act).

4. Add-on Product with Static or Dynamic Links, but With No Impact on Screen Output.

a) Hypothetical[134]

Company A owns and markets a payroll management software package that consists of numerous programs, including data base software in which employers can store human resources data and an application program (AW-2) that allows employers to create W-2 year-end reports for income tax purposes. Customers receive the program package under a license that permits installation and execution and prohibits the creation of derivative works.

Company B develops a substitute W-2 application program ("BW-2") and markets it to Company A's customers. BW-2 comes with a separate user interface and neither its code, nor its user interface nor any screen output of BW-2 contains any elements of AW-2. However, BW-2 must extract data from the human resources data base in order to perform its functions. Thus, BW-2 has to interact with Company A's data base software. For example, if a payroll administrator enters "Create W-2 report for John Doe," BW-2 sends commands to Company A's data base software to find the information required for the report. In order to ensure that Company A's program understands the requests from BW-2, Company B has to use certain commands and interface specifications defined by Company A. Also, BW-2 calls on other functions offered by sub-programs or libraries in Company A's overall payroll management software package, *e.g.*, scripts, macros, mathematical calculations and currency conversion. Once BW-2 has gathered all the required information, it creates the report and the payroll administrator can save and print John Doe's W-2 report.

Company B can achieve interoperability through static or dynamic links. If Company B follows the static linking approach,[135] it permanently copies and pastes necessary commands and other code lines from Company A's programs into the BW-2 code (this would usually occur at the source code level prior to compilation into object code); thus, on the CD delivered by Company B, the copies of the BW-2 code would actually contain copied portions of code written by Company A developers. In the dynamic linking option, on the other hand,[136] Company B programs call functions into BW-2 that instruct the computer to

[134] Facts are adopted from *Dun & Bradstreet*, *supra*, but subject to clarifications and intentional modifications.

[135] For an explanation of static linking in the context of a library program *see* Wikipedia, *Static Library*, http://en.wikipedia.org/wiki/Static_Library (last visited Jan. 21, 2006) ("In computer science, a static library, also referred to as a statically linked library, is a computer library in which links are resolved at compile-time by a linker. Static libraries may be merged with other libraries and executables to form a single object file, or they may be loaded at run-time into the address space of the linking executable or library, at a static memory offset determined at link-time.")

[136] For an explanation of dynamic linking in the context of a library program *see* Wikipedia, *Dynamic Linking*, http://en.wikipedia.org/wiki/Dynamic_linking (last visited Jan. 21, 2006) ("Dynamic linking means that the data in a library is not copied into a new executable or library at compile time, but remains in a separate file on disk. Only a minimal amount of work is done at compile time by the linker—it only records what libraries the executable needs and the index names or numbers. The majority of the work of linking is done at the time the application is loaded (load time) or during the execution of the process (runtime). The necessary linking code, called a loader, is actually part of the underlying operating system. At the appropriate time the loader finds the relevant libraries on disk and adds the relevant data from the libraries to the process's memory space. Some operating systems can only link in a library at load time, before the process starts executing; others may be able to wait until after the process has started to execute and link in the library just when it is actually referenced (*i.e.*, during runtime). The latter is

obtain certain data or other input by executing a Company A program (in a separate address space of the RAM), and after the Company A program produces certain data results, then to resume the execution of BW-2 with the input so obtained.[137]

b) Assessment

BW-2 does not impact the screen output or user interface of Company A's programs at all and therefore, does not create a derivative pictorial or audiovisual work. At the actual code level, however, software combinations occur, both in the static and dynamic linking scenario. Such combination would qualify as a derivative work of Company A's programs if the combination (i) is sufficiently permanent, (ii) contains significant and creative portions of Company A's programs, (iii) is creative in its own right, and (iv) involves significant and creative internal changes to Company A's program(s) that cannot be easily separated or distinguished.

(i) *Permanence*

In the static link scenario, the combination is permanent because the copied sections of Company A's code appear within the code lines of Company B's code.[138] The combination created by dynamic links, on the other hand, does not appear to be permanent for two reasons. First, the dynamic links are activated by a user requesting a particular operation from the BW-2 program and thus the precise character of the combination will be different each time the programs are executed. Second, the programs appear in separate address space of RAM and they are connected only through sequential calls; thus they do not permanently become part of a larger work.

(ii) *Substantial and Creative Portions*

In the static link scenario, BW-2 incorporates significant amounts of code from Company A's program(s) into its own code; this will typically result in the adoption of substantial and creative portions of copyrighted code owned by Company A. In the dynamic link scenario, the code lines whose functionality is required are not incorporated, however, but rather executed by the computer separately and in sequence.[139] In order to reach the desired functions in the Company A programs, BW-2 may have to copy some necessary interface code lines, but such material is typically neither substantial nor creative.[140]

---

often called "delay loading." In either case, the library is called a dynamically linked library. This term is sometimes shortened to "dynamic link library" or DLL, but this last initialism is most common in Microsoft Windows environments where dynamic libraries use the filename extension .dll.")

[137] The court decision in *Dun & Bradstreet*, *supra*, is not entirely clear on this point. It mentions "Copy and Call commands to access Geac's software" (at 204-5), which seems to describe dynamic linking, but also mentions that Grace's substitute "W-2 program actually consists of 62% Geac code" (at 208), and that "Grace admitted that the installation, testing, compiling and link editing of its W-2 programs required copying Geac's software and link editing the Geac code" (at 213), which seems to describe static linking. Grace asserted that its W-2 program accessed customer's data, not Geac's copyrighted code (at 212). *See also* 2 Nimmer, *supra* note 15, at § 8.08[D][2] and David McGowan, *Legal Aspects of Free and Open Source Software*, pp. 23-4 (2005), http://www.cogsci.ucsd.edu/~rik/courses/readings/McGowanD-OpenSource.pdf (last visited Jan. 21, 2006).

[138] Company A's duplication right is infringed as well.

[139] The computer executes BW-2 until it calls for a function that is only available from a Company A program; the computer then executes the Company A program, until the requested result is available; the computer uses this result to continue the execution of BW-2. The execution of the linked code in RAM is generally permitted under the applicable license agreements. *See also* 2 Nimmer, *supra* note 15, at § 8.08[D][2].

[140] *See Altai*, *supra* at 704-705.

(iii) *Combination Creativity*

The creativity aspect does not seem to depend on the method of linking (static or dynamic). Either way, the combination serves the sole purpose of extracting data or functions from one program and sharing it with another. Thus, if Company B limits the combination to the absolute minimum necessary for functionality purposes, the combination itself should generally be classified as purely functional and thus not sufficiently creative to constitute a derivative work or a compilation.[141]

(iv) *Significant Internal Changes*

A classification as a derivative work or compilation (assuming all other requirements are met) can depend on the linking technique: In the static link solution alternative, Company B permanently copies and pastes lines of code from Company A's copyrighted programs into the context of Company B's own code. If such code lines are taken out of a creative context (as opposed to a merely functional aggregation), this would involve internal changes to a copyrighted work owned by Company A. If, however, Company B incorporates entire subprograms, *i.e.*, works in their own right, the combination product may lack significant internal changes to Company A's works. Also, in the dynamic link solution alternative, Company A's code remains generally unchanged, viewed from a logical perspective, even in RAM where the programs are kept in different address spaces. Thus, the requirement of significant internal changes is typically not met by a software combination created through dynamic linking or by statically linking detachable subprograms.

(v) *Summary*

Software combinations that pursue functional goals and do not involve changes to, or combinations of, screen outputs or user interfaces will typically not meet the creativity requirement to qualify as derivative works. Additionally, dynamic link solutions will typically lack the "permanence," "substantial and creative elements," and "internal changes" elements of derivative works.[142] Static link solutions that do not qualify as derivative works nevertheless typically infringe the duplication right to the linked program.

5. Web Linking

a) Hypothetical

X makes various articles and other information on currencies as well as a number of currency conversion programs (with simple user interfaces) available through her home page. Y has an entirely separate website on which he recommends various websites, including X's homepage, with hyperlinks; if a website visitor clicks on the hyperlink to X's page, Y's website disappears and X's website appears in the visitor's browser. Z programs her own website with "in-line linking" references to X's currency

[141] *See Sega*, *supra*; *Connectix*, *supra*. The practical relevance of this assessment is greater for dynamically linked programs as compared to statically linked programs, because static linking involves by definition literal or non-literal copying that is typically not covered by any license—whereas the RAM copies involved in dynamic linking are usually permitted in standard EULAs, *see* Section IV below. However, U.S. courts have generally been sympathetic towards defendants who copied minor amounts of code in the interest of achieving interoperability (as opposed to substitution), *see*, *e.g.*, *Vault*, *supra* at 267-8; *Sega*, *supra* at 1518; *Lexmark*, *supra* at 544..

[142] An example of dynamic linking in the "off-line" world would be a hornbook supplement for a law school textbook, which is designed to slip into the pocket part of the law textbook. The hornbook does not use any creative content from the textbook, but instead consists of a series of "links" which point the user to a specific page (*e.g.*, the hornbook might say "See Page 15 for Liquidated Damages" and list a few new cases in the field.) The hornbook also makes no internal changes to the original textbook. It should not be considered a derivative work.

conversion programs; visitors of Z's website see X's currency conversion program interfaces as part of Z's website.[143]

b) Assessment of Y's Hyperlinks

Y creates a combination between his and other websites through hyperlinks. The permanence of the links collection resembles more the dynamic link solution discussed in Section IV(4), *infra*, than the MAP file scenario discussed in Section IV(3), *infra*, because user interaction is required to realize any combination. Y's hyperlinks do not adopt "significant portions" of X's website, since the linked pages are displayed separately and after Y's home page is processed. Finally, Y's hyperlinks do not cause any internal changes to X's or others' websites. Thus, B's link collection does not constitute a derivative work in combination with the linked websites.

c) Assessment of Z's Inlinks

Any visitor of Z's website sees the same combination of Z's website and elements from X's website. This combination takes X's user interfaces out of the context of X's home page and combines them with Z's own website layout. Thus, the "permanency" and "substantial and creative portions" requirements seem generally fulfilled. If the combination is sufficiently creative, it would qualify as a derivative work or compilation; if not, as a non-literal copy. Whether or not Z's in-line links create a derivative work of X's website on the one hand, or on the other hand a compilation or non-copyrightable, functional arrangement of individual works displayed on X's websites depends on how creatively the various items are arranged on the respective websites. Given the fact that many websites look very similar these days and are designed by functional considerations without much originality or creativity,[144] it seems more likely than not that Z's website lacks the elements of a derivative work. In any event, the combination of X's and Z's website elements is not directly realized by Z, but rather by the website visitors. When they visit Z's website, their browsers create the derivative works, compilations or non-literal copies of X's website. Z could only be contributorily responsible for this result if the combination created by the website visitors' browsers constitutes a derivative work.[145]

6. Interface Modifications

For completeness sake, one point should be mentioned again at this junction: If programs are not already designed to be interoperable, their combination will often require changes to the program interfaces and possibly other code segments. Such changes typically involve copying and modification of interfaces, which is usually permissible so long as the activity remains limited to what is absolutely necessary for purposes of achieving interoperability.[146]

---

[143] For a more detailed description of the technological background of in-line linking, *see Kelly v. Arriba*, 336 F.3d 811 (9th Cir. 2003). On liability for in-line linking in general, *see* Allison Roarty, Note, *Link Liability: The Argument for Inline Links and Frames as Infringements of the Copyright Display Right*, 68 FORDHAM L. REV. 1011, 1019-20 (1999).

[144] *But see MyWebGrocer, LLC v. Hometown Info, Inc.*, 375 F.3d 190, 193-194 (2d Cir. 2004) (hereafter "*MyWebGrocer*").

[145] If X's website is generally available to the public without password or other restrictions, the website visitors should normally be deemed to receive an implied license to view X's website and the individual elements thereon. Thus, if Z's website is not qualified as a derivative work of X's website, X could not succeed in a suit against Z under copyright law – but may have other cause of actions, for example, under breach of website use contract terms or trademark law, *see Ticketmaster Corp. v. Tickets.com, Inc*., 127 Fed. Appx. 346 (9th Cir. 2005).

[146] Based on one of a number of threshold requirements and defenses regarding infringement, including the idea/expression merger and the fair use doctrine, see generally *Altai, Sega, Sony, Lexmark* and *Skylink, supra*.

7. Summary

Applying the refined test developed in the previous Section to the examples in this Section confirms results that courts have reached in similar cases[147], *i.e.*, that users do not normally create a derivative work by executing two separate programs, unless one program significantly and creatively changes the creative screen output of another program. Developers can avoid or at least substantially reduce the risk of becoming liable for infringing duplication or adaptation rights by creating add-on programs using only dynamic linking techniques as opposed to static links.[148]

Of course, every individual case is different and technology is constantly evolving. Yet, given that the above examples address the main scenarios that have been subject to court decisions or discussion in the GPL-context, it seems that the question raised in the title of this article can be answered: software combinations do not normally constitute derivative works or dangerous liaisons—at least not according to the U.S. Copyright Act. The following two sections of this article examines whether a review of commercial (V) and (VI) open source licensing practices mandates a different answer.

V. SOFTWARE COMBINATIONS UNDER COMMERCIAL SOFTWARE LICENSES.

Even though a copyright owner does not have a *statutory* right under copyright law to prohibit software combinations (unless to the extent a combination qualifies as a derivative work of the copyright owner's software), a copyright owner can try to prohibit combinations *contractually*, whether or not they constitute derivative works. Section V of this Article examines this potential danger by briefly reviewing commercial software licensing practices and then discussing laws that limit a copyright owner's ability to contractually prohibit software combinations.

For several reasons discussed in more detail below, copyright owners find contractual or quasi-contractual measures less effective than a statutory prohibition sanctioned by the Copyright Act. First of all, contract or license clauses bind only licensees and not the public at large. Second, licensees may simply refuse to agree to such clauses. Third, even if agreed upon, the clauses may be invalid as a matter of statutory law, *e.g.*, the doctrine of copyright misuse, competition law or laws against unfair contract terms.

Despite such weaknesses, contractual or quasi-contractual clauses can have similar effects as statutory copyright law itself if they appear uniformly in mass-market end-user license agreements ("EULAs"). Just imagine if a dominant software manufacturer, such as Microsoft, included a clause in the EULAs of its Windows and MS Office Suite according to which "licensee agrees that he or she will not install or execute Netscape Navigator, RealNetworks RealPlayer or any other non-Microsoft programs on any PC on which Microsoft products are used or installed." If such a clause appeared in a shrink-wrap or click-through license agreement, many—if not most—consumers would probably accept it without hesitation.[149] If it such a clause were also legally valid and enforceable, it could have serious implications for software interoperability.

---

[147] The only exception may be *Dun & Bradstreet*, *supra*, if the software at issue there did contain dynamic as opposed to static links, which is not entirely clear from the decision. If so, Dun & Bradstreet would be at odds with prior cases such as *Lotus Development*, *supra* at 78-79 (recommending add-on products as an alternative to substitute products with architectural similarities) and *Galoob*, *supra* at 967 (which expressly insisted that derivative works must contain creative expression copied from the adapted original). *Worlds of Wonder I*, *supra,* and *Worlds of Wonder II*, *supra*, dealt with a different situation in which the add-on products (tapes to be inserted in a Teddy Bear) contained creative content that adapted a creative storyline from another, copyrighted product.

[148] As noted above, even static linking may not affect the copyright owner's adaptation right to the linked program (*e.g.*, the linking program may constitute merely a compilation if it includes copies of entire programs without making internal changes; or it could constitute even an entirely unprotectable combination for lack of combination creativity). Nevertheless, static links will often affect infringe duplication rights and are, therefore, generally more problematic than dynamic links.

[149] *See*, *e.g.*, Lydia Pallas Loren, *Slaying the Leather—Winged Demons in the Night: Reforming Copyright Owner Contracting with Clickwrap Misuse*, 30 OHIO N.U. L. REV. 495, 496 n.6 (2004). Anecdotal evidence suggests that few, if any, consumers ever read such clauses, but instead merely select "I agree" and continue, with little hesitation. Based on the evidence

Given the general freedom of contract and the variety of commercial licensing practices and market situations, the following discussion can only be exemplary in nature and aim to flag potential legal issues. At the same time, it is worth broadening the jurisdictional scope of the discussion given the global reach of most software licenses. Copyright laws are territorial, but fairly harmonized among the many member states of the Berne Convention.[150] Contract and competition laws, on the other hand, vary substantially from jurisdiction to jurisdiction. Nevertheless, many software licenses and license agreements apply to software use globally and many licensors port their licensing models to other jurisdictions without regard to local laws. German law seems well-suited for comparison as its Civil law tradition often differs substantially from U.S. legal approaches, and because German courts have had opportunity to decide many software-related cases that with very similar facts to cases brought in the United States.[151]

1. First Sale Doctrine

First, it seems helpful to remember why copyright owners are even able to require end user license agreements in the software field. Copyright owners of other types of works (*e.g.*, music records, movie DVDs, books, etc.) are usually prevented from soliciting such agreements by the First Sale Doctrine. According to this Doctrine, the copyright owner's exclusive right to prohibit (or permit) further distribution of a particular copy becomes exhausted after that particular copy is sold.[152] Consequently, the first purchaser (*e.g.*, a distributor, retailer or end user) can freely resell a lawfully acquired copy to a secondary purchaser (*e.g.*, a retailer or friend or a used book or CD trader).[153]

The U.S. Supreme Court created the First Sale Doctrine to strike a balance between the interest in rewarding the copyright owner for her creativity (with a right to control initial distribution) and the public's interest in free flow of goods and access to copyrighted works (after the copyright owner has been rewarded in connection with the first sale of a particular copy).[154] In the first case to acknowledge this doctrine, the court prevented the copyright owner from controlling retail pricing through a license condition.[155]

With respect to software, the traditional first sale doctrine alone cannot have the same effect that it has with respect to books or CDs. The reason is that software end users usually need to create additional copies in order to enjoy the use of a software copy, typically one permanent copy in the process of installing the software on the hard drive of a computer and further temporary copies in RAM and CPU cache in the process of executing the program.[156] Neither the first sale doctrine, nor the first purchaser of a software copy could confer such a right on a secondary purchaser. Therefore, the legislature added the software-specific Section 117(a)(1) to the Copyright Act to complement the First Sale Doctrine:[157]

> [I]t is not an infringement for the owner of a copy of a computer program to make or authorize the making of another copy or adaptation of that computer program provided:

---

discovered in connection with recent antitrust lawsuits against Microsoft, it seems quite possible that OEMs, distributors and resellers would also have tolerated such clauses in EULAs, because they themselves accepted similar restrictions on higher levels of trade, *see U.S. v. Microsoft Corp*., 253 F.3d 34, 60-63 (D.C. Cir. 2001) (hereafter "*Microsoft I*").

[150] For more information on the Berne Convention for the Protection of Literary and Artistic Works of September 9, 1886 *see* WIPO, *Treaties and Contracting Parties: Berne Convention*, http://www.wipo.int/treaties/en/ip/berne/ (last visited Jan. 21, 2006); Sam Ricketson, *The Berne Convention for the Protection of Literary and Artist Works: 1886-1986* (1987).

[151] *See*, for examples, Lothar Determann and Aaron X. Fellmeth, *Don't Judge A Sale by its License: Software Transfers under the First Sale Doctrine in the United States and the European Community*, 36 U.S.F. L. REV. 1 (2001) (hereafter "Determann/Fellmeth, First Sale Doctrine").

[152] 17 U.S.C. § 109.

[153] Neither the first nor the secondary purchaser are permitted to make any additional copies or prepare derivative works––these rights remain with the copyright owner and are not exhausted after the first sale of a copy.

[154] *See* Determann/Fellmeth, *First Sale Doctrine*, *supra* (with references).

[155] *Bobbs-Merrill Co. v. Straus*., 210 U.S. 339, 350 (1908).

[156] *See MAI Systems, supra* at 518-519.

[157] 2 Nimmer, *supra* note 15, at § 8.08[D]; *see also MAI Systems*, *supra* at 519 n.6.

> (1) that such a new copy or adaptation is created as an essential step in the utilization of the computer program in conjunction with a machine and that it is used in no other manner . . . .[158]

Pursuant to Section 117(a)(1) of the Copyright Act, end users who own software copies are not only permitted to duplicate their software, but also to adapt it as an essential step in the utilization of the software with a machine,[159] *i.e.*, to create derivative works in the interest of interoperability.

From the outset, software companies have tried to avoid the implications of this law and the First Sale Doctrine by insisting that they never transfer ownership to their software copies.[160] With few exceptions, including a relatively recent case,[161] the software industry has been able to prevail with their position in U.S. courts.[162] Therefore, Section 117(a)(1) of the Copyright Act has not had a significant practical effect in terms of authorizing the preparation of derivative works for purposes of achieving interoperability.[163] German courts, on the other hand, have ignored the labels as well as most of the content of software license agreements and have qualified perpetual software transfers for fixed fees as sales.[164] This had the effect that the First Sale Doctrine usually applies, a (broader) equivalent of Section 117(a)(1)[165] is available to end users, developers and distributors alike, and EULAs are often invalid due to a lack of deemed acceptance.[166]

2. *Typical License Clauses*

Copyright owners commonly include a number of definitions, limitations and restrictions regarding the permitted software use in their software license agreements, for example, regarding the licensed territory in which the software may be used, the number of copies (archival and production copies) the licensee may create, the type and number of individual end users (*e.g.*, licensee's employees) who may access the software copies and the type of equipment on which the software may be used (defined by serial number or performance categories, *e.g.*, MIPS, processor speed, etc.).[167]

Additionally, most commercial software companies typically prohibit any reverse engineering and any modifications to their copyrighted object code.[168] They may also state that the limited warranty does not cover malfunctions caused by operating the licensed software in conjunction with other software

---

[158] 17 U.S.C. § 117(a)(1).

[159] *See, e.g.*, *Krause v. Titleserv*, 402 F.3d 119, 125 (2d Cir. 2005).

[160] *Novell v. Unicom Sales*, 2004 WL 1839117 at p. 7 (N.D. Cal. 2004) (not reported in F. Supp. 2d); *Softman Prods. Co., LLC v. Adobe Systems Inc.*, 171 F. Supp. 2d 1075, 1084 (C.D. Cal. 2001) (hereafter "*Softman Prods*"); *Novell v. CPU Distributing, Inc.*, 2000 U.S. DIST. LEXIS 9975, p. 17-8 (S.D. Texas 2000) (not reported in F. Supp. 2d); *DSC Communications Corp. v. Pulse Communications Inc.*, 170 F.3d 1354, 1360 (Fed. Cir. 1999); *Novell v. Network Trade Ctr.*, 25 F. Supp. 2d 1218, 1230 (D. Utah 1997); *Davidson v. Internet Gateway*, 334 F. Supp. 2d 1164, 1177 (E.D. Miss. 2004) (hereafter "*Davidson*"); *ISC-Bunker Ramo v. Altech, Inc.*, 765 F. Supp. 1310, 1314 (N.D. Ill. 1990); *Data Products v. Reppart*, 18 U.S.P.Q.2d 1058, 1601 (D. Kan. 1990); *Microsoft Corp. v. Harmony Computers & Electronics*, 846 F. Supp. 208, 212-213 (E.D.N.Y. 1994).

[161] *Softman Prods*, *supra* at 1086.

[162] *See generally,* Chris Nadan, *Software Licensing in the 21st Century: Are Software "Licenses" Really Sales, and How Will the Software Industry Respond?*, 32 AIPLA Q. J. 555 (2004).

[163] *See* Nimmer, *supra* note 15, at § 8.08 [B] and [D][2]. *But see*, recently, *Krause v. Titleserv*, 402 F.3d 119 (2d Cir. 2005).

[164] *See*, Determann/Fellmeth, *First Sale* (with references).

[165] The German Copyright Act implements Council Directive 91/250/EEC of 14 May 1991 on the legal protection of computer programs, OJ L 122 pp. 42-46, which permits, in Article 5 and 6, adaptations "where they are necessary for the use of the computer program by the lawful acquirer in accordance with its intended purpose, including for error correction."

[166] If an end user does not need a license to execute a program, licensors find it much more difficult to persuade courts that the end user should be deemed to assent by way of implication because he or she tore away a shrink-wrap or clicked on a technically compulsory "accept" button during installation. *See*, *e.g.*, J. Contreras and K. Slade, *Click-Wrap Agreements: Background and Guidelines for Enforceability*, CRi 2000, 104-109 (2000).

[167] *See, e.g.,* Microsoft, *Microsoft Windows XP Professional EULA*, http://www.microsoft.com/windowsxp/pro/eula.mspx (last visited Jan. 21, 2006); Adobe, *Adobe Acrobat Reader EULA*, http://www.adobe.com/products/acrobat/acrreula.html (last visited Jan. 21, 2006) (hereafter "Adobe Acrobat Reader EULA").

[168] *See* 4 Nimmer, *supra* note 15, at § 13.09[A][3][b].

products or programs for which the licensed software is not recommended according to the documentation.

Commercial software license agreements, however, do not commonly contain express prohibitions of executing or combining the licensed software with other products as suggested in the preamble to Section V of this article. The author of this Article has never seen such a contractual prohibition in any commercial license agreement outside the "free software" context,[169] or heard about any case where a commercial software manufacturer has complained about software combinations, except with respect to video game screen manipulations and cases involving other infringements.[170] Most EULAs are silent on this point and contain relatively broad license grants. This, coupled with the fact that accompanying documentation often expressly contemplates a combination with particular other products, may be interpreted to permit the preparation of derivative works in the program combination context.[171] If software companies want to prevent consumers from using their programs in combination with other software, the companies usually design the programs or file formats to be incompatible and then take actions based on copyright law against competitors when these try to circumvent the compatibility hurdles.[172]

Thus, it seems that prevailing commercial licensing practices do not necessitate a more cautious answer than the one given at the end of the previous Section (III.7) based on statutory copyright law. Yet, given the possibility that commercial licensing practices may change—*e.g.*, in response to the "free software" licensing practices discussed in more detail in Section VI.—applicable legal limitations are summarized in the remainder of this Section.

3. Statutory Limitations to License Restrictions

As a matter of public policy, restrictions in license agreements (and other contracts, for that matter) are subject to scrutiny under a number of different statutes and legal theories, including competition laws, the doctrine of copyright misuse, and laws against unfair contractual terms.

a) Competition Law

§1 of the U.S. Sherman Act—and numerous laws in other jurisdictions modeled after it--prohibit or restrict contractual restraints of trade.[173] A restraint of trade is any agreement wherein one of the contracting parties is restrained in the manner it conducts business with third parties.[174] Courts in the U.S. qualify only relatively few types of restraints as "per se illegal" and otherwise apply a "rule of reason,"

---

[169] The GPL and other open source license agreements contain such restrictions, but are usually not referred to as "commercial" agreements, even though commercial software companies are using these agreements; see below Section VI.

[170] The computer game cases are addressed in Section III above. *Dun & Bradstreet*, *supra*, involved numerous other alleged infringements, so that the potential infringement on Geac's adaptation right did not ultimately carry the decision, *see* 2 Nimmer, *supra* note 15, at § 8.08[D] and David McGowan, *Legal Aspects of Free and Open Source Software*, pp. 23-4 (2005), http://www.cogsci.ucsd.edu/~rik/ courses/readings/McGowanD-OpenSource.pdf (last visited Jan. 21, 2006).

[171] For example, many EULAs contain a license to "use" the program. Whereas the term "use" has a specific meaning in the patent context, it is not defined in the Copyright Act and courts have decided that a contractually granted right to "use" may imply broad rights under copyright law, including a right to prepare derivative works. *See, e.g., Kennedy v. National Juvenile Detention Ass'n*, 187 F.3d 690, 695 (7th Cir. 1999) (hereafter "*Kennedy*").

[172] *See, e.g., Sega*, *supra; Lexmark*, *supra*; *Skylink*, *supra. See also*, Open Design Alliance, http://www.opendesign.com (last visited Jan. 21, 2006) (with information on the struggle of competitors of Autodesk Inc. against Autodesk Inc.); another recent example is the fight about music file formats, *e.g.*, between Apple and RealNetworks, *see* Sean Captain, *Basics; My Songs, My Format, N.Y. Times*, Oct. 6, 2005, at C.

[173] Sherman Act, 15 U.S.C. § 1 provides "Trusts, etc., in restraint of trade illegal; penalty: Every contract, combination in the form of trust or otherwise, or conspiracy, in restraint of trade or commerce among the several States, or with foreign nations, is declared to be illegal." Article 81 of the Treaty establishing the European Community provides: "1. The following shall be prohibited as incompatible with the common market: all agreements between undertakings, decisions by associations of undertakings and concerted practices which may affect trade between Member States and which have as their object or effect the prevention, restriction or distortion of competition within the common market . . . ."

[174] 1-2 Federal Antitrust Law § 2.4, Matthew Bender & Company, Inc.

which considers market power as a threshold requirement for antitrust scrutiny.[175] The European Community (EC) follows a similar approach in its block exemption regulations based on Article 81 (3) of the EC Treaty.[176] While intellectual property laws do not provide an absolute immunity from competition law,[177] competition law does generally not prohibit intellectual property owners from exercising the very "monopoly rights" that intellectual property laws confers upon them.[178] Thus, a copyright owner is generally free to exercise its exclusionary right and prohibit others from preparing derivative works.[179] If a copyright owner grants conditional permissions to prepare derivative works, however, then such conditions are fully subject to competition law scrutiny. For example, grant-back clauses (*i.e.*, clauses that allow licensees to further develop licensed technology on the condition that they transfer ownership to all improvements back to the licensor) are considered anticompetitive, because of the potential concentration of additional intellectual property rights under the control of the licensor.[180]

Thus, a licensor can rely on copyright law to contractually prohibit licensees from creating derivative works (through software combinations or otherwise); such a restraint should generally be exempt from antitrust scrutiny as a legitimate exercise of statutory rights granted under copyright law. A licensor that contractually prohibits the combination of its software with any other programs, however, would exceed the safe haven established by copyright law and subject itself to full competition law scrutiny. If such licensor has sufficient market power, it risks violating applicable antitrust laws and providing infringers of its copyrights with a defense.[181]

---

[175] *See, e.g.,* Antitrust Law Handbook, *supra*, § 2.10 n.7.

[176] *See*, *e.g.*, Polley, Softwareverträge und ihre kartellrechtliche Wirksamkeit, CR 2004, 641; Andreas Heinemann, Kartellrecht und Informationstechnologie, CR 2005, 715, 719.

[177] *See*, *e.g.*, *Microsoft I*, *supra* at 63; U.S. Department of Justice, *Antitrust Guidelines for the Licensing of Intellectual Property Issued by the U.S. Department of Justice and the Federal Trade Commission*, p. 3 (1995), http://www.usdoj.gov/atr/public/guidelines/0558.pdf (last visited Jan. 21, 2006) (hereafter "DOJ Antitrust Guidelines"); Guidelines on the application of Article 81(3) of the EC Treaty to technology transfer agreements (TTBER), OJ 2004, C 101/02 Ann. 7–9.

[178] *See* Gerald Spindler, *Rechtsfragen der Open Source Software, Rechtsgutachten im Auftrag des Verbandes der Softwareindustrie Deutschlands e. V.* (VSI), http://www.vsi.de/inhalte/aktuell/studie_final_safe.pdf (last visited Jan. 21, 2006) (hereafter "Spindler") [specific in GPL context] and generally William C. Holmes, *Intellectual Property and Antitrust Law*, Chapter 36 § 36:1, p. 36-5 (2005); Judgment of the European Court of Justice of 6 April 1995, European Court Reports 1995 I-743—Magill TV Guide; Order of the President of the Court of First Instance of 22 December 2004, OJ 2005, C 69/31—*Microsoft Corporation v. Commission*.

[179] Spindler, *supra*; Noerr Pennington doctrine, *see* 4 Nimmer, *supra* note 15, at § 13.09[A][1]; DOJ Antitrust Guidelines, *supra* Section 2.2.

[180] *See*, *e.g.*, DOJ Antitrust Guidelines, *supra* at p. 30; *Transparent-Wrap Machine Corp. v. Stokes & Smith Co.*, 329 U.S. 637, 642 (1947); *Santa Fe-Pomeroy, Inc. v. P&Z Co*., 569 F.2d 1084, 1101 (9th Cir. 1978); *United States v. Associated Patents, Inc.*, 134 F. Supp. 74, 82 (E.C. Mich. 1955), *cert. denied*, 350 U.S. 960 (1956); Holmes, *Intellectual Property and Antitrust Law*, Vol. 2, 2002, § 23.02; *Old Dominion Box Co. v. Continental Can Co*., 273 F. Supp. 550 (S.D.N.Y. 1967), *aff'd*, 393 F.2d 321 (2d Cir. 1968); *Old Dominion Box Co. v. Continental Can Co*., 273 F. Supp. 550, 572 (S.D.N.Y. 1967), *aff'd*, 393 F.2d 321 (2d Cir. 1968); *Chandler v. Stern Dental Laboratory Co*., 335 F. Supp. 580 (S.D. Tex. 1971); *Robintech, Inc. v. Chemidus Wavin, Ltd.*, 450 F. Supp. 817, 822 (D.D.C. 1978), *aff'd*, 628 F.2d 132 (D.C. Cir. 1980); Nimmer, *The Law of Computer Technology*, 3d ed. 2002, Vol. 1, § 7.71; Article 5 (1) lit. a) and b) Commission Regulation (EC) No 772/2004 of 27 April 2004 on the application of Article 81 (3) of the Treaty to technology transfer agreements, OJ 2004, L 123/11, *see* TTBER Guidelines OJ 2004, C 101/02 Ann. 107-111; Fair Trade Commission of Japan, *Guidelines for Patent and Know-how Licensing Agreements under the Antimonopoly Act of Japan*, p. 27-8, http://www.jftc.go.jp/e-page/legislation/index.html (accessed Jan. 23, 2006). The concern is that more and more related intellectual property rights and thus market power accumulates under the control of the original licensor, which seems particularly dangerous in the patent context where competition cannot develop based on independent development. Whether the same concerns apply in the copyright law context can be questioned, however, to the extent copyright law is used to protection functional software, the situation seems at least more similar to patent protection than in the context of more traditional works of authorship (*e.g.*, novels or music).

[181] Nimmer, *supra* note 15, at § 13.09[A][2].

b) Copyright Misuse

According to the doctrine of copyright misuse, copyright owners may not, through a contract or otherwise,[182] magnify their rights beyond those sanctioned by the Copyright Act.[183] The policy behind the doctrine of copyright misuse bears some resemblance to competition law policies, however, two main differences apply: The copyright misuse doctrine provides only a defense, never an offensive claim, and its viability does not depend on market power of the misuser.[184]

In the seminal case, *Lasercomb America, Inc. v. Reynolds*,[185] Lasercomb and Holliday Steel were competitors in the manufacture of steel rule dies. Lasercomb developed a software program that directed the mechanized creation of a steel rule die from a computerized template. Reynolds, a computer programmer for Holliday Steel, made unauthorized copies of the program and then marketed a new program that was extremely similar version to Lasercomb's program. A clause in Lasercomb's standard licensing agreement prohibited the development and distribution of any competing programs.[186] The court considered this clause to amount to misuse, because the licensor used its limited exclusionary rights under copyright law to prohibit activities entirely outside the scope of its copyright, namely the independent development of programs with similar functionality. As a consequence, the court denied copyright protection to the copyright owner even though in the case at hand, defendant sold what would otherwise be considered pirated copies and had not even concluded the questionable license agreement.[187]

Other U.S. courts have considered or found copyright misuse where a copyright owner agreed to license a product in exchange for an agreement by the licensee not to use a competitor's product,[188] where a copyright owner used its software copyrights to prevent licensees from substituting hardware supplied by the copyright owner,[189] where a copyright owner tried to prohibit criticism in the context in which its copyrighted video clips would appear on licensee's website,[190] and where an owner of copyrighted software attempted to use its copyright to prevent access to non-copyrightable data.[191] In each instance, the courts were concerned that copyright owners could have used their copyrights as leverage to control matters that the legislature intentionally kept outside the scope of the limited copyright "monopoly."[192]

Courts who have embraced the doctrine of copyright misuse,[193] initially adopted a rationale developed in the patent context.[194] The doctrine of patent abuse penalizes patent holders who try to

---

[182] *Assessment Technologies of WI, LLC v. WIREData, Inc*., 350 F.3d 640, 647 (7th Cir. 2003) (hereafter "*Assessment Technologies*").

[183] *Lasercomb*, *supra* at 976; 4 Nimmer, *supra* note 15, at § 13.09[A][c]; David Nimmer, et al., *The Metamorphosis of Contract Into Expand*, 87 CAL. L. REV. 17 (1999) (hereafter "Nimmer, Metamorphosis").

[184] *See* 4 Nimmer, *supra* note 15, at § 13.09; *see also Alcatel USA, Inc. v. DGI Technologies, Inc.*, 166 F.3d 772, 792 (5th Cir. 1999) (hereafter "*Alcatel*") (doctrine of copyright misuse "bars a culpable plaintiff from prevailing on an action for the infringement of the misused copyright"); *Video Pipeline, Inc. v. Buena Vista Home Entertainment, Inc.*, 342 F.3d 191, 205 (3d Cir. 2003) (copyright misuse possible even absent anti-competitive behavior).

[185] *Lasercomb*, *supra*.

[186] "Licensee agrees . . . that it will not permit or suffer its directors, officers and employees, directly or indirectly, to write, develop, produce or sell computer assisted die making software." *Lasercomb*, *supra* at 973.

[187] Neither Holliday Steel nor Reynolds had signed the licensing agreement or were bound by it; however, the Court held that, due to public policy concerns, the copyright misuse defense was available to defendants who had not been injured by the misuse. *Lasercomb*, *supra* at 979.

[188] *Practice Management Information Corp. v. American Medical Ass'n*, 121 F.3d 516, 520 (9th Cir.), *cert denied*, 522 U.S. 933 (1997).

[189] *DSC Communications Corp. v. DGI Technologies, Inc.*, 81 F.3d 597 (5th Cir. 1996); *Alcatel*, *supra* at 739. For additional information regarding the history and the results of the proceedings in these related cases, *see* 4 Nimmer, *supra* note 15, at § 13.09[A][2][b], fn. 45.

[190] *Video Pipeline, Inc. v. Buena Vista Home Entertainment, Inc.*, 342 F.3d 191 (3d Cir. 2003), *cert defined*, 540 U.S. 1178 (2004).

[191] *Assessment Technologies, supra*.

[192] For an overview, *see* 4 Nimmer, *supra* note 15, at § 13.09.

[193] According to *Davidson*, *supra* at 1179, and *Schoolhouse, Inc. v. Anderson*, 2001 WL 1640081 at 7 (D. Minn. 2001) (not reported in F. Supp. 2d), "[a]buse of copyright is generally recognized as an equitable affirmative defense to a copyright infringement claim." Other courts have been more reluctant, *see*, 4 Nimmer, *supra* note 15, at § 13.09.

expand their limited legal monopoly over the patented invention beyond the "four corners of the patent" and thus upset the balance that patent law has struck between protection and access.[195] In patent cases, courts have found a number of licensing practices abusive, including royalty requirements for components, territories or time periods outside the scope of the patent grant, covenants not to deal in competing products and package licensing.[196]

A licensor who contractually prohibits the combination of its software with other programs in situations where adaptation rights are not affected would exceed the scope of its copyright and seek to control external activities and subject matter (namely the use of independent programs). Depending on the context, such a clause could in effect constitute a prohibition on using competing products. In any event, such a clause would limit a licensee's right to create compilations and non-creative combinations, rights that the Copyright Act declared to be free, in contrast to the right to prepare derivative works.[197] Thus, a copyright owner would seem to run a significant risk that a court would classify such a clause as copyright misuse with the dramatic result that the copyright owner would be denied copyright protection against even outright piracy. A licensor who merely prohibits licensees from creating derivative works (through combinations or otherwise) would generally remain within the scope of its statutory rights and generally not risk being found to engage in copyright misuse.

The need to prevent an abuse of intellectual property law is internationally recognized, *e.g.*, in TRIPS Articles 8.2 and 40.2.[198] Nevertheless, a doctrine similar to copyright misuse does not seem common in national copyright laws outside the United States.[199] One reason for this may be that courts in other jurisdictions find more effective tools to sanction abuse in stricter competition or unfair contract term laws.

c) Unfair Contract Terms

Contracts are free[200] and paper is patient.[201] Since consumers and even businesses often ignore or helplessly agree to the small print, many mass-market contracts contain quite strange provisions, in particular standard software license agreements.[202]

---

[194] *See United States v. Loew, Inc*., 371 U.S. 38 (1962); *United States v. Paramount Pictures*, 334 U.S. 131, 157-59.

[195] *See Morton Salt Co. v. G.S. Suppiger Co.*, 314 U.S. 488. The legislature somewhat limited the applicability of the patent abuse doctrine in the 1988 Patent Misuse Reform Act, which provides that "No patent owner otherwise entitled to relief for infringement or contributory infringement of a patent shall be denied relief or deemed guilty of misuse or illegal extension of the patent right by reason of his having done one or more of the following: . . . (5) conditioned the license of any rights to the patent or the sale of the patented product on the acquisition of a license to rights in another patent or purchase of a separate product, unless, in view of the circumstances, the patent owner has market power in the relevant market for the patent or patented product on which the license or sale is conditioned."

[196] *See generally*, Donald S. Chisum, *Chisum on Patents*, § 19.04 (2005). Whether it is appropriate to transfer the abuse doctrine from patent law to copyright law may seem questionable in the context of traditional works of authorship, given the relatively smaller impact of the copyright exclusion rights on innovation and commerce; yet, in the context of protection for functional software, it seems more likely that similar problems can arise from misuse as those in the patent context.

[197] 17 U.S.C. §§ 103 and 106.2 and above Section II(5).

[198] Agreement on Trade-Related Aspects of Intellectual Property Rights, Including Trade in Counterfeit Goods, Apr. 15, 1994, 1869 U.N.T.S. 299, 33 I.L.M. 1125 (hereinafter "TRIPS"). *See also*, L. Guibault "Limitations found outside of copyright law"—*The Exceptions and Limitations to Copyright*, ALAI Study Days, Cambridge, September 14-17, 1998, S. 1. For a perspective on the EU position regarding TRIPS obligations and competition law, see Ullrich, JOURNAL OF INTERNATIONAL ECONOMIC LAW 2004.7(401)

[199] *See* P.E. Geller/M.B. Nimmer, *International Copyright Law and Practice*, Release NO. 17, October 2005, *e.g.*, *France* (A. Lucas/P. Kamina), *Germany* (A. Dietz) and *United Kingdom* (L. Bently).

[200] Any lawful obligations may be subject of a contract. *See*, *e.g.*, CALIFORNIA CIVIL CODE §§ 1595, *et seq*.

[201] This is a literally-translated German saying ("Papier ist geduldig"), meaning one can write down many things, *e.g.*, in a contract, but time will tell what writings are actually relevant, *see* Wörterbuch Deutsch, http://www.redensarten-index.de/suche.php?suchbegriff=Papier+ist+geduldig+&bool=relevanz&suchspalte%5B%5D=rart_ou (last visited Jan. 21, 2006).

[202] *See* Lydia Pallas Loren, *Slaying the Leather—Winged Demons in the Night: Reforming Copyright Owner Contracting with Clickwrap Misuse*, 30 OHIO N.U. L. REV. 495, 496 n.6 (2004) and the excerpt from terms published by a Canadian Software Company: "Should you fail to register any of the evaluation software available through our web pages and continue to use it, be

Clauses in license agreements, as in other contracts, are subject to scrutiny under doctrines such as contract of adhesion[203] and unconscionability.[204] Unilateral licenses—*i.e.*, permissions that are subject to certain conditions but do not contemplate the conclusion of a negotiated contract—are generally subject to similar interpretation and limitation rules.[205]

Some foreign jurisdictions apply relatively strict scrutiny when evaluating contracts, especially consumer contracts and contracts between businesses that are not individually negotiated. For example, the European Community has enacted the Directive on Unfair Terms in Consumer Contracts,[206] Germany has extremely rigid rules on "standard contract terms,"[207] and the UK passed the Unfair Contract Terms Act in 1977.[208] Courts in these jurisdictions usually strike contract clauses in their entirety if they contain sections that are considered to be unfair; they will not blue-pencil overreaching clauses, in order to discourage overreaching contract drafting.[209]

Courts in the United States, on the other hand, generally restrain themselves and abstain from questioning contract terms, except in extreme circumstances. This has resulted in seemingly unreflected and one-sided licensing practices, in particular in the context of shrink-wrap and click-through agreements.[210] Litigators have so far attacked such agreements commonly on procedural grounds,[211] as the law seems less developed or favorable on substantive challenges. However, as software companies are updating their contract formation procedures, taking existing case law into consideration, more and more substantive challenges can be expected.

Under general principles of contracts law in the United States, a contract of adhesion is a "standardized contract prepared entirely by one party to a transaction for the acceptance of the other." [212] A party presented with a contract of adhesion has no opportunity for bargaining.[213] It must accept or reject the contract terms on a "take it or leave it basis" and usually has no other way of obtaining the desired product or service.[214] A contract is not automatically unenforceable simply because it is formed per one

---

advised that a leather-winged demon of the night will tear itself, shrieking blood and fury, from the endless caverns of the nether world, hurl itself into the darkness with a thirst for blood on its slavering fangs and search the very threads of time for the throbbing of your heartbeat. Just thought you'd want to know that. Alchemy Mindworks accepts no responsibility for any loss, damage or expense caused by leather-winged demons of the night, either." Alchemy Mindworks, Details, http://www.mindworkshop.com/alchemy/alchemy6.html, (last visited Jan. 21, 2006).

[203] An adhesion contract is "[a] standard-form contract prepared by one party, to be signed by the party in a weaker position, usu. a consumer, who adheres to the contract with little choice about the terms." BLACK'S LAW DICTIONARY (8th ed. 2004), *contract*.

[204] Unconscionability is "[t]he principle that a court may refuse to enforce a contract that is unfair or oppressive because of procedural abuses during contract formation or because of overreaching contractual terms, esp. terms that are unreasonably favorable to one party while precluding meaningful choice for the other party." BLACK'S LAW DICTIONARY (8th ed. 2004), u*nconscionability*.

[205] *See, e.g.,* Nimmer, *Metamorphosis*, *supra* at 30; *Kennedy*, *supra* at 694; *Fantastic Fakes, Inc. v. Pickwick Int'l, Inc.*, 661 F.2d 479, 483 (5th Cir. Unit B 1981); *Womack+Hampton Architects, L.L.C. v. Metric Holdings Ltd.*, 102 Fed. Appx. 374, 378 (5th Cir. 2004).

[206] Council Directive 93/13/EEC of 5 April 1993 on Unfair Terms in Consumer Contracts.

[207] *See* §§ 305-310 BGB (GERMAN CIVIL CODE).

[208] Unfair Contract Terms Act 1977 §§ 3-7 (1977 c50).

[209] Examples for standard clauses in software license agreements invalidated by German courts: BGH, NJW 1980, 832 (clause voiding warranty in case of reassembly of software); OLG Nürnberg, CR 1990, 118, 121 (prohibition of resale); OLG Hamm, NJW 1989, 1041 (deemed acceptance upon delivery); BGH, NJW 1981, 1501 (warranty disclaimer); OLG Köln, NJW-RR 1987, 1192 (maintenance contract in lieu of warranty rights).

[210] 1 Nimmer, *supra* note 15, at § 3.04[B][3][a], footnote 126; *see generally* Mark A. Lemley, *Intellectual Property and Shrinkwrap Licenses*, 68 S. CAL. L. REV. 1239 (1995).

[211] Many shrink-wrap and click-wrap cases focus on whether contract formation mechanisms were flawed, *e.g.*, because customers did not receive sufficient notice of the contract terms, *see*, *e.g.*, *Specht v. Netscape Comms*, 306 F.3d 17 (2d Cir. 2002); see also Lothar Determann and Saralyn Ang-Olson, Comment on *Specht v. Netscape*, 6 No. 2 IPLB 39(2001).

[212] *Painters Dist. Council No. 33 v. Moen*, 128 Cal. App. 3d 1032, 1039 (Cal. Ct. App. 1982) (hereafter "*Painters*").

[213] *Painters, supr*a at 1039.

[214] *Painters, supra*, at 1039.

party's initiative, without negotiation and/or using boilerplate terms.[215] However, courts subject contracts of adhesion to heightened scrutiny, inquiring whether their terms fall within the "adhering" party's reasonable expectations.[216] Similarly, the doctrine of unconscionability allows courts to refuse to enforce an unconscionable provision in a one-sided contract.[217] The purpose of this doctrine is to make it possible for the courts to police explicitly against the contracts or clauses which they find to be unconscionable. The test for unconscionability is "whether, in light of the general background and the needs of a particular case, the clauses involved are so one-sided as to be unconscionable under the circumstances existing at the time of the making of the contract."[218]

As with respect to the doctrine of copyright misuse and competition law, copyright owners that maneuver within the express grant of rights afforded by the Copyright Act are far less likely to be successfully challenged, because it is usually not unfair to contractually exercise rights specifically granted by the legislature.[219] Thus, a copyright owner who simply prohibits the preparation of derivative works in an end-user license agreement is generally on safe ground, whereas a licensor who prohibits program combinations without regard to copyright law seems more likely subject to scrutiny under unfair contract term law.[220]

4. License Scope Definitions vs. License Conditions vs. Contractual Covenants

Copyright owners can largely stay clear of the risks described in the preceding section (arising under competition law, the doctrine of copyright misuse and unfair contract term regimes) if they follow a simple recipe: defining license scope limitations within the concepts expressly provided by copyright law is generally permissible (*e.g.*, prohibiting the preparation of derivative works)—whereas the application of conditions and covenants that are alien to copyright law (*e.g.*, prohibiting the execution of one program with another independent program or a grant-back of copyright to modifications) are subject to scrutiny and can be invalid, illegal and/or copyright misuse.

Conceptually, restrictions can be introduced through different types of mechanisms. The least intrusive of these mechanisms seems to be a clause that simply strives to prevent granting an implied license, by clarifying that the copyright owner does not intend to authorize a certain activity.[221] Most intrusive would be a clause that conditions the license grant on compliance with all contractual prohibitions in the license agreement.[222] Somewhere in between is a contractual covenant that is

---

[215] *See* RESTATEMENT (SECOND) OF CONTRACTS § 211(1) (1979) ("[W]here a party to an agreement signs or otherwise manifests assent to a writing and has reason to believe that like writings are regularly used to embody terms of agreements of the same type, he adopts the writing as an integrated agreement with respect to the terms included in the writing.").

[216] *Ericksen, Arbuthnot, McCarthy, Kearney & Walsh, Inc. v. 100 Oak Street,* 673 P.2d 251, 257 n.7 (Cal. 1983).

[217] *See, e.g.,* CAL. CIVIL CODE § 1670.5 (2004); *Am. Online, Inc. v. Superior Court*, 90 Cal. App. 4th 1, 5 (Cal. Ct. App. 2001).

[218] CAL. CIVIL CODE § 1670.5; *see also Davidson*, *supra* at 1179; *Cooper v. MRM Inv. Co.*, 367 F.3d 493, 503-504 (6th Cir. 2004) (hereafter "*Cooper*"). Unconscionability has both a procedural and substantive element. *Freeman v. Wal-mart Stores, Inc.*, 111 Cal. App. 4th 660, 669 (Cal. Ct. App. 2003) (hereafter "*Freeman*"); *Cooper*, *supra* at 503. Similar to the doctrine of adhesion contracts, the procedural element focuses on oppression or surprise due to unequal bargaining power, and substantive unconscionability focuses on overly harsh or one-sided results. Generally, courts require both procedural and substantive unconscionability be present, although not to the same degree, before a court will refuse to enforce a contract or clause due to unconscionability; *See Pardee Construction Co. v. Superior Court of San Diego County*, 100 Cal. App. 4th 1081, 1088 (Cal. Ct. App. 2002) ("In other words, the more substantively oppressive the contract term, the less evidence of procedural unconscionability is required to come to the conclusion the term is unenforceable, and vice versa."); *Freeman*, *supra* at 669.

[219] Spindler, *supra* at 51.

[220] Spindler, *supra* at 51.

[221] Such a clause could follow the actual license grant and be phrased, for example: "Licensor reserves all other rights, including without limitation, the right to prepare derivative works of the Licensed Software. Licensee understands that Licensor does not positively authorize the installation or execution of any programs made by other manufacturers in combination with the Licensed Software. Whether or not such combinations are permissible depends on applicable law, including, without limitation, 17 U.S.C. § 106.2. Licensee agrees to apply with all applicable laws."

[222] Such a clause could follow the actual license grant and be phrased, for example: "Aforesaid license grant is made under the condition that Licensee refrains from installing or executing any programs made by other manufacturers in combination

independent of the license grant.[223] The stronger the mechanism, the greater the chance that a court will invalidate an overreaching restriction.

5. Summary

The author of this Article is not aware of any commercial software licenses that try to control software combinations beyond general, unspecific contractual prohibitions of adaptations (which are quite common). If this practice were to change and copyright owners were to use their rights under copyright law to prohibit software combinations regardless of whether they qualify as derivative works, they would risk running afoul of competition laws, the doctrine of copyright misuse and laws prohibiting or invalidating unfair contract terms. The extent of the actual risks depends on a number of circumstances, including the restriction mechanism (license condition vs. limitation vs. contractual covenant), the licensor's market power and the applicable law. For example, licensors would face relatively high risks under German competition and contracts law and the doctrine of copyright misuse under U.S. law.

VI. SOFTWARE COMBINATIONS UNDER THE GENERAL PUBLIC LICENSE

Since courts first decided to afford copyright protection to computer programs, commercial software development companies have had a strong incentive to avoid reusing existing code.[224] Independent creation is a defense to copyright infringement,[225] and so software development companies often opt for creating programs from scratch, ideally in a "clean room" environment, so they can prove that their products are not copies of existing programs with similar functionality.[226] Thus, the decision in favor of software copyrightability had a rather dramatic impact on the professional lives and day-to-day activities of programmers: instead of being asked to further develop and improve the "state of the art" and to focus on cutting-edge problems, programmers were asked to spend most of their time recreating the wheel. All that just because lawyers did not have the energy or wit to come up with a more fitting intellectual property law regime tailored to software.[227]

---

with the Licensed Software. If Licensee violates this condition, the License grant shall immediately expire." GPL § 4, discussed in more detail in Section VI of this Article, also contains such a condition: "You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License." More commonly in practice—and less clear for licensees and courts—copyright owners preface the license grant sections with a phrase such as "subject to all terms and conditions of this Agreement, Licensor grants Licensee a license to . . ." and include the limitations as separate restrictions in different clauses without express designation as a license condition.

[223] Such a clause could be phrased, for example "Licensee shall refrain from installing or executing any programs made by other manufacturers in combination with the Licensed Software." If a licensee violates such a clause, its license rights would not necessarily be affected and the licensor would have to seek recourse based on breach of contract remedies. *See*, *e.g.*, *Sun Microsystems Inc. v. Microsoft Corp*., 188 F.3d 1115 (9th Cir., 1999).

[224] *See* Mark A. Lemley & David W. O'Brien, *Encouraging Software Reuse*, 49 STAN. L. REV. 255 (1997).

[225] 3 Nimmer, *supra* note 15, at §§ 12.10[B][2][a] and 12.11[D].

[226] *See* Wikipedia, *Cleanroom (Software Engineering)*, http://en.wikipedia.org/wiki/Cleanroom_(Software_Engineering) (last visited Jan. 21, 2006).

[227] Many practitioners and academics have proposed various types of sui generis protection regimes for software in both the U.S. and abroad. *See, e.g.,* Pamela Samuelson, et al., *A Manifesto Concerning the Legal Protection of Computer Programs*, 94 COLUM. L. REV. 2308 (1994); Richard H. Stern, *A Sui Generis Utility Model Law as an Alternative Legal Model for Protecting Software*, 1 U. BALT. INTELL. PROP. L.J. 108 (1993); John C. Phillips, *Sui Generis Intellectual Property Protection for Software*, 60 GEO. WASH. L. REV. 997 (1992); Steven B. Toeniskoetter, *Protection of Software Intellectual Property in Europe: An Alternative Sui Generis Approach*, 10 NO. 1 INTELL. PROP. L. BULL. 65 (2005). *But see* Ginsburg, *Four Reasons*, *supra*; Brett A. Carlson, *On the Wrong Track: A Response to the Manifesto and a Critique of Sui Generis Software Protection*, 37 JURIMETRICS J. 187 (1997).

So, the programmers[228] had their revenge on the lawyers:[229] The programmers invented "copyleft"[230] to fight copyright law, published a manifesto[231] and created a new set of license terms that was going to free software from the shackles and chains of copyright protection to become the new standard for licensing software to the public: the General Public License (GPL).[232] The ultimate goal was to "free"[233] software and spread the "freed software" (*i.e.*, copyrighted software licensed under the GPL) to replace and ultimately eliminate[234] all proprietary software (*i.e.*, copyrighted software that is commercialized through restrictive licensing).[235] The means to reach this goal was going to be—fight fire with fire—copyright law: Any copyright owner who releases software under the GPL would require anybody else to apply the GPL to any new versions of this code and bring copyright infringement actions against anyone who breaches the GPL. Anybody who distributes software outside the scope of the applicable license agreements lacks a valid authorization required by the Copyright Act[236] and would thus commit copyright infringement.[237] The cunning revenge plan worked quite well—judging by the increasingly publicized success of the free software movement[238] as well as the outcries in the legal community[239] and from software companies with proprietary licensing models.[240]

---

[228] More specifically, the leader of the free software movement is Richard M. Stallman, Founder of The Free Software Foundation and the GNU Project, and primary author of the current GNU GPL, *see generally* The Free Software Foundation, Leadership, http://www.fsf.org/about/leadership.html. (last visited Jan. 21, 2006).

[229] For a more objective restatement of the GPL's history *see* Wikipedia, GPL, http://en.wikipedia.org/wiki/Gpl. (last visited Jan. 21, 2006).

[230] Richard M. Stallman, *Copyleft: Pragmatic Idealism*, http://www.fsf.org/licensing/essays/pragmatic.html (last visited Jan. 21, 2006).

[231] Richard M. Stallman, *The GNU Manifesto*, http://www.gnu.org/gnu/manifesto.html (last visited Jan. 21, 2006).

[232] GNU Project, Text of the GNU General Public License, version 2, http://www.gnu.org/licenses/gpl.html (last visited Jan. 21, 2006) (hereafter "GPL License"). Richard M. Stallman has recently unveiled a draft of the next revision of the GPL. *See* Mark Baard, *New GPL is Free at Last*, Wired, Jan. 17, 2006, http://wired.com/news/technology/0,70028-0.html?tw=wn_tophead_1 (last visited Jan. 21, 2006).

[233] *See* Free Software Foundation, *The Free Software Definition*, http://www.fsf.org/licensing/essays/free-sw.html (last visited Jan. 21, 2006) ("Free software is a matter of the users' freedom to run, copy, distribute, study, change and improve the software.")

[234] "The enemy is proprietary software." Richard M. Stallman, *Why "Free Software" is better than "Open Source*, http://www.fsf.org/licensing/essays/free-software-for-freedom.html. (last visited Jan. 21, 2006).

[235] *See* Free Software Foundation, *Categories of Free and Non-Free Software*, http://www.fsf.org/licensing/essays/categories.html#ProprietarySoftware (last visited Jan. 21, 2006). Other supporters of the movement have taken more moderate positions, stressing the importance of both proprietary and open source software. *See, e.g.*, Lawrence Lessig, *Open-Source, Closed Minds*, Eweek, Oct. 1, 2003, http://www.eweek.com/article2/0%2C1895%2C1309535%2C00.asp (last visited Jan. 21, 2006) ("[A] balance between proprietary and nonproprietary property is better than either extreme.").

[236] 17 U.S.C. § 106.3. *See*, *e.g.*, *Apple v. Microsoft*, *supra* at 1441.

[237] *See* The GPL-Violations.Org Project, http://gpl-violations.org/ (last visited Jan. 21, 2006) (cataloging past and present GPL infringers).

[238] *See*, *e.g.*, Peter Galli, *GPL Could Put Heat on Microsoft*, Eweek, Nov. 29, 2004, http://www.eweek.com/article2/0,1759,1732567,00.asp (last visited Jan. 21, 2006); James Boyle, *The public domain: The second enclosure movement and the construction of the public domain*, 66 LAW & CONTEMP. PROBS. 33 (2003).

[239] *See*, *e.g.*, Michael Tsur and Shay David, *A License to Kill (Innovation): On Open Source Licenses and their implications for Innovation*, http://www.shaydavid.info/papers/shaydavid_a_license_to_kill_043005.pdf (last visited on Jan. 27, 2006), p. 37; Heidi S. Bond, *What's So Great About Nothing? The GNU General Public License and the Zero-Price-Fixing Problem*, 104 MICH. L. REV. 547 (2005); 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 (1997); Mathias Strasser, *A New Paradigm in Intellectual Property Law? The Case Against Open Sources*, 2001 STAN. TECH. L. REV. 4 (2001); Dan Hunter, *Culture War*, 83 TEX. L. REV. 1105 (2005); Greg Vetter, *Infectious Open Source Software: Spreading Incentives or Promoting Resistance?*, 36 RUTGERS L.J. 53, 162 (2004); Marc Kaufman et al., *Licensing Open Source Software*, http://www.nixonpeabody.com/copyright_article.asp?ID=1&PubType=A (last visited Jan. 21, 2006).

[240] *See*, *e.g.*, Thomas C Greene, Ballmer: *"Linux is a cancer" Contaminates all other software with Hippie GPL rubbish*, THE REGISTER, June 2, 2001, http://www.theregister.co.uk/2001/06/02/ballmer_linux_is_a_cancer/ (last visited Jan. 21, 2006); Microsoft anti-GPL fine print threatens competition, THE REGISTER, April 17, 2002, http://www.theregister.co.uk/2002/04/17/microsoft_antigpl_fine_print_threatens/; (last visited Jan. 21, 2006) and Remarks by Bill Gates, Seattle, Washington, Apr. 17, 2002, available online at http://www.microsoft.com/billgates/

In order to determine whether combinations with GPLed code are as dangerous as some say,[241] this Section of the Article reviews (1) the mechanism of Section 2(b) of the GPL; (2) possible interpretations of the definitional scope of Section 2(b); (3) the potential impact of the First Sale Doctrine; and (4) the potential impact of statutory and other legal limitations on license restrictions in the GPL. It concludes with an assessment that combinations with GPLed code are less dangerous than the GPL's drafters claim (5.) and advances a few recommendations for the current GPL update process (6.).

1. Effects of GPL § 2(b.

Section 2(b) of the GPL requires "derived works" to be licensed under the GPL also:

> 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.[242]

This requirement has a "viral"[243] or "immunizing"[244] effect on licensees, depending on their objectives.

Section 4 of the GPL clarifies that all its restrictions operate as conditions and that non-compliance results in a termination of all rights, *i.e.*, the most intrusive of the various types of license restrictions discussed in Section V.4 of this Article:

> You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License.

Section 7 of the GPL manifests what the Free Software Foundation refers to as the "liberty or death" principle:[245]

---

speeches/2002/04-17glc.asp (last visited Jan. 21, 2006); Mike Ricciuti, *Gates wades into open-source debate*, http://news.com.com/2100-1001-268667.html (last visited on Jan. 24, 2006).

[241] David S. Evans and Bernard J. Reddy, *Government Preferences for Promoting Open-Source Software: A Solution in Search of a Problem*, 9 MICH. TELECOMM. & TECH. L. REV. 313, 340 (2003); Klaus M. Schmidt and Monika Schnitzer, *Public Subsidies for Open Source? Some Economic Policy Issues of the Software Market*, 16 HARV. J.L. & TECH 473 (2003); Ulrich Wuermeling/Thies Deike, *Open Source Software: Eine juristische Risikoanalyse*, CR 2003, 87, 91; Gerald Spindler, at: *German Society for Law and Informatics (Deutsche Gesellschaft fuer Recht und Informatik e.V.), Committee for Contract Law*, conference on March 28, 2003, topic: *"Vertragsrechtliche Fragestellungen der Open Source Software"* (contractual questions concerning open source software), *see* http://www.dgri.de hyperlink; then follow "Site Map"; then follow "Vertragsrecht" (last visited Jan. 21, 2006) (thesis: GPL's "viral effect" causes legal insecurity for software innovation).

[242] GPL License, *supra*, § 2(b).

[243] "Viral" because code that is combined with GPLed code may become "infected," *see*, *e.g.*, Nadan, *Open Source Licensing*, *supra*; Andreas Guadamuz, *Viral Contracts or Unenforceable Documents? Contractual Validity of Copyleft Licenses*, E.I.P.R. Vol. 26, Issue 8, pp. 331-339, 2004, available online at http://papers.ssrn.com/sol3/papers.cfm?abstract_id=569101 (last visited Jan. 21, 2006) (hereafter "Guadamuz, Viral Contracts"); Marc Kaufman et al., *Licensing Open Source Software*, Nixon Peabody Website, http://www.nixonpeabody.com/copyright_article.asp?ID=1&PubType=A (last visited Jan. 21, 2006); *see also OpenSource.Org, the Open Source Definition*, http://web.archive.org/web/20010330062316/www.opensource.org/docs/definition.html (web-archive version); in the 9th version of the Open Source Definition, which is currently in effect, the OSI dropped the word "contamination." *See* http://www.opensource.org/docs/definition.php for latest version (last visited Jan. 21, 2006); James Gibson, *Once and future copyright*, 81 NOTRE DAME L. REV. 167, 204 (2005).

[244] "Immunizing" because this clause prevents GPLed code from being incorporated into proprietary—*i.e.*, "non-free," derivative works, *see* Kenneth J. Rodriguez, *Closing the Door on Open Source: Can the General Public License Save Linux and Other Open Source Software?*, 5 J. HIGH TECH. L. 403, 414 (2005); Michael Torrie, *new Samba license?*, available online at http://galactus.ximian.com/pipermail/mono-list/2002-April/004787.html (last visited Jan. 24, 2006); Greg R. Vetter, *The Collaborative Integrity of Open-Source Software*, UTAH L. REV. 563, 634 (2004).

> If . . . conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all.

With these two GPL clauses, copyright owners permit licensees to adapt and distribute copies of the GPLed software under the condition that licensees grant the copyright owner and everybody else a license—under the terms of the GPL, which contain a number of other restrictions and requirements—to any works that are "derived from" the GPLed program at no charge and in object and source code form.

The GPL does not prohibit licensees from charging money for selling software copies.[246] However, the GPL itself already provides everyone with a license free of charge. As a result, licensees have no realistic chances to commercialize their copyrights in works that they "derive" from GPLed code.[247] The GPL intentionally deprives copyright owners from the very economic incentive the Copyright Act has made available in the interest of furthering the development of original works of authorship—the right to exclude the public from copying, adapting, distributing, etc., and to charge a fee for individual licenses.

It is worth noting, however, that the GPL generally permits end users to execute GPLed code in any combination they want. According to Section 0 of the GPL, "[t]he act of running the Program is not restricted." As a result, software companies do not have to be concerned about invoking the "viral" effect of the GPL based on a contributory liability theory if they distribute their add-on programs separately, *i.e.*, not in context[248] with any GPLed code, even if the add-on programs are intended for combination with a particular version of GPLed code.[249] End users who run add-on programs with the GPLed code would not infringe, because the GPL allows execution without any restrictions. And without direct infringement,

---

[245] Richard M. Stallman, *Microsoft's New Monopoly*, http://www.fsf.org/licensing/essays/microsoft-new-monopoly.html (last visited Jan. 21, 2006): "The "General Public License forbid[s] publication of a modified version if it isn't free software in the same way. (We call that the "liberty or death" clause, since it ensures the program will remain free or die)."

[246] Free Software Foundation, *Selling Free Software*, http://www.fsf.org/licensing/essays/selling.html (last visited Jan. 21, 2006).

[247] The copyright owner to the original program does not have to comply with the GPL, so long as she does not accept any modifications from others, and can thus still charge fees for her software, *e.g.*, under a dual licensing model, which may involve releasing software under the GPL free of charge (for marketing purposes) and for a license fee under a proprietary license (to licensees that would rather pay a fee than comply with the GPL for their own modifications). Also, an increasing number of companies have found other ways to generate revenue in the context of "free software," *e.g.*, by using "free software" as a means to sell proprietary software, services or hardware, see, *e.g.*, John Koenig, Seven open source strategies for competitive advantage, IT Manager's Journal, May 15, 2004, http://www.management.itmanagersjournal.com/print.pl?sid=04/05/10/2052216 (last visited Jan. 21, 2006); Wikipedia, *Open Source vs. Closed Source*, http://en.wikipedia.org/wiki/Open_source_vs._closed_source (last visited Jan. 21, 2006); Jens Sieckmann, *Freie Software in der Wirtschaft* § 9.2, available online at http://www.bravehack.de/html/node45.html (last visited Jan. 21, 2006); Open Source Biotechnology Project, *Open Source as a Business Approach*, at http://rsss.anu.edu.au/~janeth/OSBusMod.html (last visited Jan. 21, 2006).

[248] If a company offers copies of GPLed code and add-ons for such code to a customer, it would need a license to distribute the GPLed code—and such a license is granted per Section 2(b). In this respect, it should not make a difference whether the vendor delivers the GPLed program and the add-on product at different times or on different CDs.

[249] This conclusion is also reached in what seems to be the first German commentary on the GPL: Till Jaeger, Olaf Koglin, Till Kreutzer, Axel Metzger, Carsten Schulz, Die GPL Kommentiert und Erklärt, pp. 66-69 (2005). However, in its FAQ, the Free Software Foundation takes the contrary position that distributors of add-on products for GPLed programs have to comply with the GPL merely because they make a product that is designed to interact with GPLed products (even though it does not contain any copied code). *See* Free Software Foundation, *Frequently Asked Questions About the GNU GPL*, http://www.fsf.org/licensing/licenses/gpl-faq.html#GPLAndPlugins (last visited Jan. 21, 2006) (hereafter "GPL FAQ") This position could only be viable in reliance on a contributory liability theory (which can be ruled out, given the expressive broad grant of end user license rights under the GPL) or if—as a matter of copyright law—the distributor of an add-on product needed a distribution license—which is usually not the case, based on the view explained in this Article, *see above* Section IV. The opposite view could try to emphasize that the courts in *Micro Star*, *supra*, and *Midway*, *supra*, did not expressly rely on a theory of contributory infringement and thus, arguably, must have negated the requirement that a derivative work must incorporate significant expression from the original work; however, such arguments would have to overcome the facts that (1) the *Micro Star* and *Midway* courts were addressing screen outputs and "story lines" with a high level of non-functional creativity that is unusual in the software context and (2) the vast majority of references in legislative history, cases and commentaries emphasize the definitional requirement that derivative works must actually incorporate copyrighted expression from the adapted work. *See* Section II.1 of this Article with further references.

there could not be any contributory liability. Yet, given implementation and integration difficulties, end users typically expect packages from their software suppliers, so separate distribution is often not an option for practical purposes.

Thus, the question remains important: What exactly is a "derived work"? At first sight, a number of possibilities come to mind: "Derived work" could mean "derivative work as this term is defined by the Copyright Act," "derivative work or collective work as these terms are defined by the Copyright Act," "any combination of programs, whether or not the combination qualifies as a derivative work" or something else. The answer to the question what the term "derived work" means determines how much a licensee has to give up when it distributes GPLed code in combination with improvements or other programs—and whether a licensee may distribute GPLed code in combination with third party code at all.

2. GPL Terminology and Interpretation

a) "Works based on the Program"

The first operative Section of the GPL (Section 0) reads as follows:

> This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The 'Program,' below, refers to any such program or work, and a 'work based on the Program' means either the Program or 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. (Hereinafter, translation is included without limitation in the term 'modification.') Each licensee is addressed as 'you.'
>
> Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

As is common in commercial contracting practice, the first Section of the GPL contains a number of definition and specifications that apply to the document as a whole (*e.g.*, the first sentence and the first ten words of the second sentence define the meaning of the capitalized term "Program"). Less common, however, are the explanatory notes that the GPL drafters interwove with the legally binding definitions (*e.g.*, the last sentence acknowledges that the conditions in the preceding half-sentence may not always be met in practice). The cause for this anomaly seems to lie in the genesis of the document: it was written by programmers for programmers. In order to make the document useful for non-lawyers (and projects without a budget for legal advice), and to establish the GPL as a standard, the GPL drafters tried to draft it as user-friendly and accessible to programmers as possible.

Along these lines, the second sentence of Section 0 defines "works based on the Program" as the Program itself or "any derivative work under copyright law" followed by a (not entirely accurate) interpretive explanation regarding what the term "derivative works" means under copyright law. This explanation, introduced with "that is to say," gives an indication of what the GPL drafters thought, hoped or may argue in a dispute, is the meaning of the term "derivative works." Section 2 of the GPL contains additional explanations and declarations of intent, which even include "collective works," *i.e.*, a term defined by the Copyright Act in contrast to the term "derivative work."[250] In order to resolve these text-internal contradictions, it would seem appropriate to rely on the "operative" portion of the definition in Section 0 (which contains the reference to the Copyright Act) and treat the "explanatory notes" as statements of opinion that have been added for convenience purposes only.[251] Accordingly, the GPL

[250] *See* above Section III.

[251] If the GPL drafters had wanted to apply their own definition, independent of the Copyright Act, they could have omitted the reference to the Copyright Act concept of derivative works or added it as an explanation what the GPL definition would include.

would be interpreted to define "work based on the Program" to mean "derivative work as defined by the Copyright Act."[252]

b) Derived Works

The first sentence of Section 2 of the GPL permits modifications to the GPLed program in reference to the defined term "work based on the Program." The following sentences of Section 2 contain a number of license conditions and explanations and use a number of other terms to describe the result of modifications besides "work based on the Program," including modified files,[253] modified program,[254] and modified work.[255] The critical Subsection (b) refers to "any work . . . that in whole or in part contains or is derived from the Program or any part thereof."

Taken out of context, each of these terms seems to go well beyond the statutory definition of derivative works in the Copyright Act, because the statutory definition is not satisfied by every modification or any work that contains any part of another work, or that is derived from any part of another work. As discussed, under the Copyright Act, a combination of code with a GPLed program constitutes a derivative work of the GPLed program only if the combination (i) is sufficiently permanent, (ii) contains significant and creative portions of the GPLed program, (iii) is creative in its own right, and (iv) involves significant and creative internal changes to the GPLed program.[256]

In context, however, it appears that the drafters of the GPL randomly chose substitutes to the somewhat awkward term "work based on the Program" and used the substitute terms synonymously to improve the sentence flow and readability. This impression is confirmed throughout the document, which also uses other substitutes, including the "derivative or collective works based on the Program"[257] and "derivative works."[258]

Some of the explanations throughout the GPL as well as the Free Software Foundation's FAQ[259] and "Lesser General Public License"[260] imply that the drafters of the GPL intended to cover software combinations that would not qualify as derivative works under the Copyright Act according to the test developed in this Article.[261] This is primarily evidence of a difference of opinion in the application of copyright law—and does not have to mean that the condition in Section 2(b) of the GPL covers more than

---

[252] Implicitly assumed by Mitchell L. Stoltz, *Penguin Paradox*, *supra* at 1458.
[253] GPL License, *supra*, §2(a).
[254] GPL License, *supra*, §2(c).
[255] GPL License, *supra*, §2 after (c).
[256] *See* above Section III(3).
[257] GPL License, *supra*, §2.
[258] GPL License, *supra*, §5.
[259] See, GPL FAQ, *supra*, in particular the answer to the Question: "What is the difference between "mere aggregation" and "combining two modules into one program"?
[260] Free Software Foundation Website, *GNU Lesser General Public License*, http://www.gnu.org/copyleft/lesser.html (last visited Jan. 21, 2006) ("hereafter LGPL License").
[261] Most notably, the LGPL contains specific exceptions for dynamic linking of libraries, which—according to this article—would not qualify as derivative works and thus "works based on the Program" under the GPL regardless. On this topic, *see also* Brian W. Carver, *Share and Share Alike: Understanding and Enforcing Open Source and Free Software Licenses*, 20 BERKELEY TECH. L.J. 443, 459 (2005). The preamble of the LGPL explains the relationship between the two licenses as follows: "This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs. When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library." LGPL License, *supra*. The preamble is based on the assumption that a combination of dynamically linked programs constitutes a derivative work of such programs, which is contrary to the findings of this Article. Yet, it is worth noting that even this paragraph of the LGPL acknowledges that both GPL and LGPL ultimately defer to the definition of "derivative work" under copyright law—and the explanations constitute merely explanations regarding the drafters' understanding of what kinds of software combinations constitute "derivative works."

derivative works as defined by the Copyright Act. Yet, uncertainties remain given the fact that the "explanations" appear within the license text.

c) Rules of Contract Interpretation

Given the prevailing controversies and uncertainties regarding the exact scope of Section 2(b) of the GPL, it seems worth exploring whether presumptive rules of contract interpretation would favor one interpretation over another. In this context it is important to note that quite different rules could apply depending on the context of a particular licensing relationship. The GPL contains neither a contractual choice of law nor a forum selection clause. Under statutory and common law conflicts of law principles, which vary from jurisdiction to jurisdiction, the governing law of a licensing relationship subject to the GPL will be determined by the residency of the licensor and licensee, and various other factors. Thus, in practice, there is not one GPL that applies to all free software globally. Instead, thousands of different versions provide for slightly different rights and obligations of the licensing parties based on peculiarities of the governing contract law.[262]

Nevertheless, two principles of contract interpretation are likely to apply in most jurisdictions in one form or another: Courts try to (i) determine the parties' intent and (ii) interpret ambiguous clauses against the party who caused the ambiguity .

(i) *Parties' Intent*

Courts typically try to determine the intent of the contracting parties as objectively evident to each other at the time of contract formation.[263] Where the contract language is clear and unambiguous, courts will usually not look to extrinsic evidence of intent.[264] Given the uncertainties around Section 2(b) of the GPL, however, it seems likely that courts would feel tempted to look beyond the four corners of the document. Even though the GPL emphasizes that it constitutes a license as opposed to a contract,[265] courts would likely apply contract interpretation rules and try to determine the intent of the copyright owner who selected the GPL and the licensee who selected the program. In many cases, courts will probably find that neither party really had a choice—the GPL came to apply because a developer of a previous program version had opted for the GPL. Where the Free Software Foundation itself is involved as a party, it may be appropriate to take the various examples, explanations and programmatic and ideological statements on its website into consideration. Where the Free Software Foundation is not involved, however, it will often be difficult to confirm that parties to a dispute were familiar with these materials at the time the licensing relationship was formed.

(ii) *Interpretation against the Drafter*

Another common principle of contract interpretation is that in case of uncertainties, courts should interpret the contract against the party who caused the uncertainty to exist.[266] This could help licensees in cases against the Free Software Foundation. But in cases where neither party has selected the GPL to

---

[262] *See*, *e.g.*, Till Jaeger, Olaf Koglin, Till Kreutzer, Axel Metzger, Carsten Schulz, *Die GPL Kommentiert und Erklärt*, p. 4-6 (2005). The Free Software Foundation acknowledges the risk of providing official translations of the GPL (see GPL FAQ, *supra*, http://www.gnu.org/licenses/gpl-faq.html#MereAggregation), however, the variations introduced by different applicable bodies of contract law seem far greater.

[263] *See*, *e.g.*, CALIFORNIA CIVIL CODE § 1636 and GERMAN CIVIL CODE §§ 157, 133; *but see Pacific Gas & Elec. Co. v. G.W. Thomas Drayage & Rigging Co.*, 69 Cal. 2d 33, 39; 442 P.2d 641, 645 (Cal. 1967); *Trident Center v. Conn. Life Ins.*, 847 F.2d 564, 568 (9th Cir. 1988).

[264] *See*, *e.g.*, CALIFORNIA CIVIL CODE § 1639.

[265] Eben Moglen, *Enforcing the GNU GPL*, http://www.gnu.org/philosophy/enforcing-gpl.html (last visited Jan. 21, 2006) ("Licenses are not contracts: the work's user is obliged to remain within the bounds of the license not because she voluntarily promised, but because she doesn't have any right to act at all except as the license permits.").

[266] *See*, *e.g.*, CALIFORNIA CIVIL CODE § 1654.

apply for a particular modified program, it is not clear that either party is to blame for the GPL's uncertainties.

d) Summary

The GPL permits end-users to combine software and execute software in combination without any restrictions, even if the combinations constitute derivative works of the GPLed programs. As a consequence, contrary to the views expressed by the Free Software Foundation, distributors can separately distribute add-on products intended for combination with GPLed code without fear of incurring contributory liability. Thus, the distribution of add-ons to computer games discussed in Section IV.3 of this Article should be unproblematic in the GPL context, since the suppliers of the add-on products do not also sell the actual games.

The GPL strictly prohibits the distribution of program combinations that qualify as "derived works" of GPLed programs, unless the entire combination can be subjected to the license terms of the GPL. This has two serious consequences for licensees: First, they are prohibited from distributing GPLed programs in combination with proprietary third party programs whose copyright owners do not agree to the GPL terms. Second, if licensees create "derived works" of GPLed programs, they cannot commercialize such "derived works" through proprietary license models as contemplated by the Copyright Act.

Despite remaining uncertainties, the context of the GPL favors an interpretation of the term "derived work" to mean a "derivative work as defined by the Copyright Act." Consequently, the combinations discussed in Section IV.2 and IV.4 should also be permissible under the GPL, because they do not involve a creation of derivative works. However, the language of the GPL also allows broader interpretations and its drafters take the position—in documents that do not seem determinative for contract interpretation purposes—that dynamically linked programs fall under Section 2(b) if distributed in combination with GPLed code.

3. First Sale Doctrine

The drafters of the GPL did not take the same precautions to avoid the First Sale Doctrine as drafters of commercial licenses do. To the contrary, the Free Software Foundation expressly states that the GPL allows licensees to sell copies.[267] As a result, it is conceivable that a company could try to circumvent Section 2(b) of the GPL by causing copies of GPLed code to be sold[268]--which would seem to allow a secondary purchaser to resell such copies (and its purchaser to execute such copies) without regard to the GPL[269] even if the secondary purchaser also sells its own proprietary add-on programs to its customers simultaneously.[270]

4. Statutory Limitations on License Restrictions

The same legal principles that can affect the validity of commercial licenses can also affect the validity of the GPL as it applies in a particular licensing relationship. Yet, a few particular considerations apply given the somewhat unusual characteristics of the GPL.

---

[267] GPL FAQ, *supra*, http://www.fsf.org/licensing/licenses/gpl-faq.html#DoesTheGPLAllowMoney. (last visited Jan. 21, 2006) ("Does the GPL allow me to sell copies of the program for money? Yes, the GPL allows everyone to do this. The right to sell copies is part of the definition of free software. Except in one special situation, there is no limit on what price you can charge. (The one exception is the required written offer to provide source code that must accompany binary-only release.)")

[268] For instance, from a development company to a distribution company within a group of wholly owned subsidiaries.

[269] 17 U.S.C. §§ 109 and 117.

[270] Without such a "first sale" scheme, the distribution company would need to obtain a license for the distribution right to the GPLed code, and such licenses would only be available under the GPL terms. Of course, this scheme would not change the situation for scenarios involving derivative works under the Copright Act, because the adaptation right does not become exhausted through a first sale. Also, it should be noted that courts often try to sanction circumvention schemes under various legal theories.

a) Competition Law

As previously observed, the applicability and effects of competition law depend largely on the situation (*i.e.*, on the affected markets and the parties' market power.)[271] Thus, competition laws would probably play an insignificant role with respect to a relationship between two individual developers, but they could well come into play if a number of dominant suppliers[272] or purchasers[273] pushed to establish the GPL as a standard with the intent to drive "software only" companies from the market. Independent of such case-by-case considerations, however, one observation seems to apply generally: Section 2(b) of the GPL does not have the same anti-competitive effect that grant-back clauses typically have, because it does not require an exclusive license or assignment of ownership rights, and the license is granted to anyone. Thus, Section 2(b) of the GPL does not result in a concentration of intellectual property rights or market power in the hands of one particular licensor.

b) Copyright Misuse

By imposing GPL § 2(b) on licensees, copyright owners try to magnify their rights beyond those sanctioned by the Copyright Act in two different ways. First, Section 103 of the Copyright Act allocates ownership rights to authorized derivative works to the author to incentivize further investment in additional creativity.[274] In contrast, Section 2(b) of the GPL, requires creators of derivative works to forfeit their exclusion rights and any chance to generate licensing revenue.[275] Second, if the term "derived work" were found to encompass more than "derivative works" and included, for example, compilations and other forms of software combinations, Section 2(b) of the GPL would seek to prohibit activities that Section 106 of the Copyright has not reserved for copyright owners and thus exponentially increase the impact caused by the first copyright magnifying mechanism.[276]

Given the fact that copyright misuse is an equitable concept under U.S. law, it is difficult to predict if and how a court would apply this doctrine in the context of the GPL. On one hand, the non-profit status and idealistic goals pursued by the proponents and original adopters of the GPL may sway courts in favor of the GPL. On the other hand, the "copyleft" policy manifested in the GPL seems a more direct attack on the delicate balance between access and protection in the Copyright Act[277] than any other licensing practice that has so far caused courts to find copyright misuse.[278] In fact, the intended objective behind Section 2(b) of the GPL is to eliminate the effects of copyright protection for computer programs and generally replace it by the rules of the GPL.[279] This flies in the face of the many decisions by U.S. courts that found it necessary to protect economic interests of software copyright owners who pursued proprietary licensing models.[280] Also, more and more companies use the GPL for purposes other than idealism. If courts enforce clauses like Section 2(b) of the GPL, they would probably also have to accept it if proprietary software companies start prohibiting combinations of their programs with other software beyond the boundaries of the Copyright Act. This could have potentially significant implications for interoperability.

---

[271] *See* above Section V(3)(a).

[272] For example, technology conglomerates that have traditionally sold hardware and/or services in addition to software could decide to "give away" software in order to drive "software only"-companies from the market.

[273] For example, government units or large purchasing cooperatives could unite to insist on GPLed code in order to drive software prices down or distort competition by favoring "solution providers" who can afford to "give away" software so long as they can raise hardware and/or services prices.

[274] 17 U.S.C. § 103.

[275] GPL License, *supra* §2.

[276] 17 U.S.C. § 106.

[277] *See* above Section III(1).

[278] Stoltz, *Penguin Paradox*, *supra* at 1458, 1477 (referring to "copyleft" as a "hack on the copyright system" and discussing the quasi-contractual regime established under the GPL as an alternative to the statutory regime established in the Copyright Act.)

[279] *Id.*

[280] *See generally* above Section III.

Thus, for purposes of U.S. copyright law, Section 2(b) of the GPL seems to raise significant issues under the doctrine of copyright misuse, particularly, if it were interpreted to cover more than derivative works as defined by the Copyright Act.

c) Unfair Contract Terms

As in the commercial licensing context, it will depend on the situation of the particular parties to a licensing relationship whether Section 2(b) of the GPL could be found invalid as an unreasonable or unconscionable contract term. In general, unilateral software licenses easily meet most of the elements required for procedural unconscionability. Not only do licensees have no opportunity to negotiate; if they refuse to accept the license terms, they are expressly prevented from exercising the rights conveyed by the license.[281] The bargaining power differential depends on the exact circumstances of a particular licensing relationship. Consumers will typically not be affected by the GPL, because it allows combinations on the end user level without any restrictions.[282] As between businesses, on the other hand, courts might well find enough bargaining power differential to allow licensees to invoke doctrines such as contracts of adhesion or unconscionability under U.S. laws, *e.g.*, in situations where large businesses offer code under the GPL, where smaller vendors are required to comply with the GPL by large software purchasers (such as government units), or simply because the terms of the GPL are usually non-negotiable due to the ideology behind it and the numerous copyright owners involved.[283]

Substantively, however, it will often appear far-fetched for licensees to argue that Section 2(b) of the GPL is unreasonable as a commercial matter, given the fact that the licensee does not have to pay for its license.[284] Nevertheless, the "no charge" aspect would have the opposite effect where licensees require licensors to use the GPL. Also, under less flexible contract laws in other jurisdictions, Section 2(b) of the GPL seems to run a substantial risk of invalidity merely because it derogates so sharply from statutory law to the disadvantage of the licensee.[285]

5. Summary

Combinations of programs with GPLed software are somewhat dangerous for companies with proprietary license models if they want to distribute also the GPLed code itself. Distribution of separate add-on programs, however, should be permissible, even where the combination results in the creation of a derivative work as defined by copyright law because the GPL's broad end user licenses stand in the way of contributory liability theories.

Another risk is that courts could interpret Section 2(b) of the GPL in a manner to extend not only to derivative works as this term is defined under applicable copyright law, but also to compilations and other types of combinations that are not otherwise subject to the copyright owner's exclusive rights. Any courts that interprets Section 2(b) so broadly would have to seriously consider a licensee's challenges and

---

[281] *See, e.g.,* GNU License, *supra* § 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.").

[282] *See* above Section VI(1).

[283]

[284] *But see* Guadamuz, *Viral Contracts*, *supra* at 336.

[285] *See*, *e.g.*, Guadamuz, Viral Contracts, *supra*. LG München, 21 O 6123/04 (2004) expressed approval for Section 2 of the GPL in general, but this ocurred in a dictum and in reference to the unofficial, non-binding German translation of the GPL (even though the English, official version applied in the case at hand, *see* Case Note by Hoeren, CR 2004, 776, 777). It is far from clear, however, how a German court would approach the "derived works" definition if and when confronted with a case actually raising the "viral/immunizing" effect of this clause; based on general principles of German contracts law, it would seem more likely that a German court would invalidate Section 2(b) on the basis that it is surprising, overbroad, or ambiguous, or interpret "derived works" to mean "derivative work under German copyright law," which would likely result in a relatively narrow application; like U.S. law, German copyright law requires derivative works to incorporate significant expression from the adapted work and would not, *e.g.*, classify software combinations created through dynamic links as derivative works, see Spindler, *supra* at 51 and Wuermeling/Deike, *Open Source Software: Eine juristische Risikoanalyse*, CR 2003, 87, 90.

defenses under the doctrine of copyright misuse, competition law or unfair contract term laws. In light of these potential legal limitations and the context of the GPL, a narrower interpretation of Section 2(b), limited to derivative works as defined by copyright law, seems more appropriate and likely. Consequently, dynamic linking to GPLed programs would not normally trigger the application of the GPL to the linking program, even if both programs are distributed together.

6. Recommendations for GPL Version 3[286]

It seems unrealistic to ask of the Free Software Foundation to give up the "viral" / "immunizing" concept of Section 2(b), according to which licensees have to make derivative works available under the GPL because this concept is the centerpiece of the Free Software Foundation's strategy. Yet, going forward, the Free Software Foundation should consider clarifying in the actual GPL text that the various and varying references to "derived works" etc. in the GPL are meant to refer to derivative works as defined by applicable copyright law.[287] This would go a long way towards minimizing the risk of diverging GPL standards around the world due to varied local laws, because by "dynamically linking" to "applicable copyright law," the GPL would largely adopt the statutory boundaries set by copyright law in the various jurisdictions where code may be created or used.[288] This is turn should significantly reduce any risks of offending competition law, unfair contract terms law or doctrines similar to the copyright misuse doctrine in other jurisdictions.[289] The risks arising from unfair contract term laws could be minimized further by adding a governing law clause referring to the laws of a jurisdiction that respects freedom of contract in general and severability clauses in particular.

VII. CONCLUSION

Software combinations are less dangerous liaisons as some have recently argued, particularly in the context of the GPL.

Under the U.S. Copyright Act, a combination of a computer program with other software results in the preparation of a derivative work only if the combination (a) is sufficiently permanent, (b) contains significant and creative portions of the other software, (c) is creative in its own right, and (d) involves significant and creative internal changes to the other software. Most software combinations fail to meet one or more of these requirements and constitute either compilations, collective works, or non-copyrightable aggregations, and neither affect copyright owners' adaptation rights under Section 106 of the U.S. Copyright Act.

Software combinations involving dynamic links usually lack permanency, combination creativity and internal changes. Even software combinations through static links do not necessarily affect adaptation rights, because such linking often results in the creation of a compilation or non-creative aggregation of programs or sub-programs. Nevertheless, under the U.S. Copyright Act, software developers typically have to obtain a license before they may combine programs through static linking because this affects the duplication rights of the linked program's copyright owner. Also, adaptation rights may be affected where

---

[286] This article was completed just prior to the release of the draft GPL Version 3 text, *see* Free Software Foundation, *GPLv3 Draft*, http://gplv3.fsf.org/draft (last visited Jan. 21, 2006) and Robert Gomulkiewicz, *General Public License 3.0: Hacking the Free Software Movement's Constitution*, 42 HOUS. L. REV. 1015 (2005).

[287] The author acknowledges that the legal community (1) still has to play its own part by reaching a clear consensus regarding the definitional scope of "derivative works" in the context of software combinations, and (2) may not deserve any help with reducing FUD after applying copyright protection to software in the first place, at least not in the eyes of the programmer community.

[288] Such a "dynamic link" to "applicable copyright law" would work well even in situations and jurisdictions where it is not yet clearly defined which types of software combinations might infringe adaptation rights, because a court in such jurisdiction would come to decide the copyright law question at the same time as it would apply the GPL under such circumstances.

[289] An express contractual choice of law would further support and strengthen uniformity and reliability of the GPL also with respect to other clauses—*e.g.*, warranty disclaimers and limitations of liability—in the interest of all licensors worldwide. Ironically, however, a selection of one jurisdiction's contract law—*e.g.*, the laws of a State in the USA—would probably be perceived in the developer community as too local and not global enough.

software combinations (regardless of the code linking method) result in significant and creative changes to original screen output (*e.g.*, in the context of computer games).

Under common commercial licensing conditions, end users typically receive an express or implied license to execute proprietary software in combination with other software, regardless of whether the combination would qualify as a derivative work. Under the GPL, end users are free to combine GPLed code with any other code. Developers and distributors do not have to be concerned about contributory liability, so long as they distribute add-on software separately and the end-users are not legally restricted in combining the intended programs with the add-on software.

Anybody who wants to distribute programs in combination and alongside with GPLed code, however, will have to closely examine the reach and consequences of the various conditions and restrictions in the GPL. The term "derived work" in the GPL should be interpreted to mean "derivative works as defined by copyright law," and as a consequence, most programs could be distributed in combination with dynamically linked GPLed code without the necessity of subjecting the linking programs to the GPL.

It seems possible, however, that courts may interpret the GPL in a broader way, which would increase concerns regarding the validity of the GPL under copyright misuse doctrines, competition laws and unfair contract term laws; such concerns can be greater or smaller depending on the circumstances of the licensing parties and jurisdictions involved. If such broad interpretations were to prevail—but the resulting validity concerns were not—the software industry might move more generally to GLP-like restrictive licensing practices that permit and prohibit certain software combinations. This would potentially have a serious impact on interoperability. Then, software combinations could become dangerous liaisons.

Version 2006-4-3

SFODMS/6471175.22