8 Va. J.L. & Tech. 2 (2003), *at* http://www.vjolt.net
© 2003 Virginia Journal of Law and Technology Association

VIRGINIA JOURNAL *of* LAW *and* TECHNOLOGY

UNIVERSITY OF VIRGINIA	SPRING 2003	8 VA. J.L. & TECH. 2

Breaking into Locked Rooms to Access Computer Source Code: Does the DMCA Violate a Constitutional Mandate When Technological Barriers of Access Are Applied to Software?

Rod Dixon[*]

[*] Rod Dixon, Senior Attorney, U.S. Department of Education and Visiting Assistant Professor of Law, Rutgers University School of Law-Camden (LL.M. 1998 Georgetown University Law Center, J.D. 1992 George Washington University Law School, M.A. 1986 University of Pittsburgh, B.A. 1984 University of Pittsburgh). No doubt that many of my thoughts on this subject have been shaped and affected by a list of individuals far longer than my citations could represent, but I am particularly thankful for the thoughtful and helpful comments of Eric Eldred and Don Rodney Vaughan.

I. Summary

1. The popular media has taught us well. We know, for example, that the Internet is here to stay, that Content is King, that America Online has introduced most American newbies[1] to the Internet, that, despite prior conventional wisdom to the contrary, electronic mail messages may contain pernicious computer viruses, that some Internet users do not shop online because of concerns regarding privacy and that downloading a digital audio file from file-sharing services like Napster is theft. Theft? Perhaps not, but the rise and fall of the popular Internet music file-sharing service, Napster, alerted nearly every Internet user that there is a claim by some content owners that there is some connection between copyright law and the downloading of music files by users connected to the Internet.

2. During the past couple of years, some of the most important legal challenges to what may be deemed permissible conduct in Cyberspace have involved disputes over copyright, trademark, and other forms of intellectual property. At bottom, these disputes illustrate a truism of Cyberspace; namely, accompanying the increasingly perceived importance of copyright protection for digital works, the Internet is exposing the ease of violating copyright interests, the volume of those violations, and the unraveling of respect for copyright. To thwart the potential that Cyberspace could make copyright worthless, content providers and right-holders seem to have adopted an aggressive response that may not only ensure that respect for copyright is not unraveled, but may redefine the right-holder's intellectual property interests in a manner that places content providers in a better position with regard to digital works than held when wealth creation depended solely upon the analog antecedents to digital content. To achieve this objective, content providers have begun using a powerful weapon wrought from the legislative pen of Congress: the Digital Millennium Copyright Act (DMCA).[2]

3. This article extends the author's prior critical analysis concluding that aspects of computer software, particularly the source code, rarely should be regarded as a category of expression created as a result of independent and, hence, original authorship. Taking that argument a step further, it is apparent, I argue, that the DMCA's ostensible approval of locking up access to source code regardless of whether the source code meets the originality requirement violates copyright's constitutional mandate.[3]

[1] "Newbie" is a shorthand reference to an Internet user who is unfamiliar with the social customs of Internet communities or with the technologies used to provide Internet access. America Online still seems to attract the majority of American first time Internet users.
[2] Digital Millennium Copyright Act, 17 U.S.C. §1201 (1998).
[3] In this regard, the best example of when the DMCA's application of Section 1201 should be considered unconstitutional is under circumstances where a plaintiff claims protection as a result of a defendant's breaking an access control to access software lacking the requisite originality to receive copyright protection. To use a perhaps unfavorable example, if Microsoft decrypted an encryption program to obtain access to a file containing the Java programming language, a court should refuse to apply the DMCA's Section 1201 protections, since to do so would ostensibly sanction copyright protection in a work that failed to meet the constitutional requisite for originality. Although Sun Microsystems Corporation might

4. This view unfolds, first, by concluding that as a result of contemporary programming methods, a substantial portion of the source code embedded within off-the-shelf software is neither independently created nor illustrative of the minimal creative spark required to meet the constitutional requirement of originality. This conclusion follows despite the author's judgment that the refrain "source code is speech" is imprecise and too analytically debilitated to sufficiently sustain the proper balance between copyright and free expression.

5. In this regard, the case is put forward that the proper scope of copyright protection for software has been further distorted by Congress' recent enactment of the DMCA – which contains a thicket of legal lumber that substantially diminishes fair use by adopting a reverse engineering standard permitting only an exceptionally narrow use of reverse engineering – as well as by the recent trend among courts to restrict the fair use doctrine of reverse engineering in the context of copyright infringement. By authorizing civil and criminal liability against content and software users who attempt to access "locked up" source code by "reverse engineering" one form of computer code to reproduce another, more readable, form of computer code, the DMCA pushes copyright directly out of step with promoting the progress of science and useful arts, which, of course, is the fundamental purpose of copyright. Ostensibly, the DMCA not only sanctions locking software ideas in a technological room, but, in most circumstances, prohibits breaking into the room or circumventing the "door locks" to access the hidden ideas. In this manner, the DMCA's reverse engineering provision is uncharacteristic of a copyright law because it undercuts the public's access to ideas: ideas embedded in source code. Unfortunately, the DMCA confounds a pre-existing but perplexing trend within copyright law: courts fronting judicial imprimatur for increasing the scope of copyright protection for aspects of computer software that should be clearly uncopyrightable.

6. Oddly enough, this trend provides an apparent cover for content-based speech restrictions directed toward the activities of some libertarian-oriented Internet-based computer hackers and the open source community, whose activities are focused upon unleashing the locked up ideas of software, not hiding them. Indeed, it is the open source community that best demonstrates that a wide-ranging network of programmers can develop robust, popular, and reliable software,

---

beg to differ; to be useful, standard computer-programming languages must be viewed as unoriginal works, not subject to copyright protection. A computer language is an abstract, systematized formation of signs and symbols used to develop or construct computer programs. The computer language itself is not copyrightable. *See generally* Rod Dixon, *Profits in Cyberspace: Should Newspaper and Magazine Publishers Pay Freelance Writers for Digital Content?*, 4 MICH. TELECOMM. & TECH. L. REV. 127 (1998). *Cf.* Brief of Appellant at VII.A.1.e., Microsoft Corp. v. Sun Microsystems, Inc., 188 F.3d 1115 (9th Cir. 1999) (No. 99-15046) (visited Dec. 30, 2001) *available at* http://www.microsoft.com/presspass/java/contract.asp. Although Microsoft disputed Sun Microsystems' position that the plaintiff owned a copyright interest in the Java programming language itself, it is unclear whether the judge issuing rulings in the case as it progressed toward settlement accepted Sun Microsystems' position.

where source code is open and freely available to the public, not hidden by access controls or technological barriers. It also demonstrates that some of the prior assumptions about the relationship between copyright and software are no longer well-founded. At a minimum, the future will require active support by courts and Congress for the vigilant support of fair use and reverse engineering of copyright-protected works to reverse the trend toward the over-protection of software by copyright.

7. What emanates from the text of the DMCA and recent judicial interpretations of the reverse engineering doctrine is a fairly displeasing and, likely, odiferous aura of complete acceptance, if not reverence by Congress and too many courts, of the copyright holder's desire to restrict the scope of lawful reverse engineering without accommodating an important access point of the public to public domain material. The doctrine of reverse engineering is being redefined as protecting a limited practice authorized solely as an apparent economic efficiency primarily benefiting pertinent intellectual property owners. Divorcing the reverse engineering doctrine from its context in fair use and free expression removes the public's only access to the ideas and functional elements embodied in software. That this privilege of access is being replaced by a technological barrier sanctioned by Congress with the force of civil and criminal liability is demonstrative of the larger context of how close we have come to an ill-fated future for the public domain of information and information products.[4] This does not simply promise a future where the user of digital information will pay an "owner" for its use in each and every instance, but includes a potential guarantee that some owners will be more equal than others. For some, copyright law rightfully will favor the dissemination of works like CSS (an encryption program) over DeCSS (a decryption program); for others, copyright law will become an increasing patchwork of legislative favoritism dislodged from its constitutional purpose.[5]

8. On a trod toward this end, the first court to apply the DMCA to computer source code, *Universal City Studios, Inc., v. Corley*,[6] rejected the defendants' argument

[4] Broadly stated, the argument concerns whether the ill-fated future – each side of this debate views the unfolding of the other side's prediction as ill-fated – of the scope of the relationship between copyright and digital works will include the elimination of copyright (or the need for copyright) or an evolution of ubiquitous, perpetual copyright; in contemporary debates over the scope of copyright, "[t]here are those who favor thick protection and those who prefer thin. The argument in favor of balance is not a liberal vs. conservative argument. The argument is old vs. new." LAWRENCE LESSIG, THE FUTURE OF IDEAS: THE FATE OF THE COMMONS IN A CONNECTED WORLD 202 (Random House 2001) (citing Siva Vaidyanathan at chap. 11, n. 14).
[5] *See, e.g.,* JESSICA LITMAN, DIGITAL COPYRIGHT 192-915 (Prometheus Books 2001); Stephen Breyer, *The Uneasy Case for Copyright: A Study of Copyright in Books, Photocopies, and Computer Programs*, 84 HARV. L. REV. 281 (1970) (noting far earlier than the modern debates over copyright that the practical effect of Congress legislating an expansion of copyright is to impose a tax on content users and that such power should be undertaken with considerable circumspection).
[6] Universal City Studios, Inc. v. Corley, 273 F.3d 429 (2d Cir. 2001), *aff'g* Universal City Studios, Inc. v. Reimerdes, 111 F.Supp.2d 294 (S.D.N.Y. 2000) (for purposes of consistency and to reflect the proper case name, *Corley* is used throughout the text).

that under the circumstances of the case, the practice of reverse engineering copyright-protected works like a software program was supported by doctrinal as well as statutory privilege, which allows defendants to avoid the liability that would otherwise apply under the DMCA.[7] According to the court, if a software user successfully obtained access to source code by circumventing a technological "lock" or barrier to access without authorization from the copyright holder of the computer program, the circumvention would not be excused by the doctrine of reverse engineering under the DMCA or by relevant case law defining the same doctrine unless the defendant could persuade the court that her purpose for breaking the access control was to determine how two programs interoperate.[8]

9. In *Corley*,[9] the plaintiffs, eight motion picture studios, brought suit under the DMCA, inter alia, seeking an injunction against the magazine and website, known as 2600 and 2600.com, (and several other defendants) to prohibit the defendants from publishing, disseminating, trafficking in, and posting on or linking to a website with the source code to a decryption program that allowed computer users of the Linux[10] operating system (OS) to play lawfully acquired movies digitized by the copyright holder onto digital video discs (DVDs). The source code was alleged to provide a way of circumventing a technological access barrier; namely, encryption software called Content Scramble System Software (CSS).[11] This was accomplished by use of a decryption software program called, cleverly enough, DeCSS, which was developed after reverse engineering CSS.[12] Apparently, at the time DeCSS was developed, DVDs were sold with CSS to allow users to play the content on DVD-ROM drives connected to computers running only the Windows operating system developed by Microsoft Corporation.[13]

10. *Corley* both narrowed the scope of the defendants' conduct that could come within the statutory exception permitting reverse engineering,[14] and limited the range of protection allowing public access to a work for the purpose of reverse engineering. The court also took the unprecedented step of restricting the reverse engineering exception to circumstances involving the dissemination of information, solely for the purpose of achieving interoperability among disparate computer programs, which the court (mistakenly) determined was unlikely to be among the defendants' actual purposes. In the court's view, interoperability is not likely to have been the goal of the reverse engineer and it is possible that the

---

[7] 111 F.Supp.2d at 304-05.
[8] *Id.* at 319-20.
[9] *Id.* at 303.
[10] Linux is a free software operating system developed by thousands of volunteer programmers and hackers. Some commentators have argued that Linux is the only existing viable competitor to Microsoft's Windows-based operating systems. *See, e.g.,* Eben Moglen, *Microsoft's Fatal Error*, THE NATION, Nov. 29, 1999, at 5.
[11] 111 F.Supp.2d at 308.
[12] *Id.*
[13] *Id.* at 311.
[14] Reverse engineering is the common practice of disassembling a product to discover how it works. Kewanee Oil Co. v. Bicron Corp., 416 U.S. 470, 476 (1974).

immediate fruits of reverse engineering may lead to the development of a decryption program designed to operate on the same OS, rather than immediately interoperate on a different OS.[15] Indeed, the court appears to have confused the purpose of a program designed to break an access barrier with software that interacts with the content of the DVD. DeCSS does not directly allow the motion picture to play on a computer using an unauthorized operating system. Instead, it decrypts a technological control used to block access to the content on the DVD. Under the circumstances, overriding the access control is an initial step in the reverse engineering process that must occur using the native or a compatible operating system. This is the only practical manner to access the ideas contained within the source code. In this manner, the access control has the practical effect of denying all computer users access to the underlying ideas in the software used to block access as well as other software programs deactivated during the access block.[16] Remarkably, the court failed even to consider whether obtaining access to unprotectable ideas had been the basis of defendants' efforts of reverse engineering. Instead, the district court's analysis followed a lock-step argument presented by the plaintiffs concerning the copyright in the motion picture content on the DVD, which had nothing at all to do with reverse engineering the CSS.

11. The district court in *Corley* is not the only court recently to question or doubt a defendant's claim to be engaged in legitimate or credible reverse engineering conduct in the context of software. In this respect, there is a troubling trend toward increasing statutory and judicial approval of a software developer's prerogative to block public access to source code without exception, despite the appealing logic of the notion that copyright protection of the expressive quality of source code should result in its exposure, not its secrecy. In a case related to *Corley*, *DVDCCA v. Bunner*,[17] a trial court rejected the argument that a 15-year-old Norwegian programmer had engaged in lawful reverse engineering for purposes of interoperability. The court's skepticism seemed to have emanated from its concern that the 15-year-old was an alleged hacker who had publicly declared his disrespect for the law of copyright.[18] Although the plaintiff had sought injunctive relief under a state law trade secret misappropriation claim, rather than a claim under the DMCA, the court's application of the doctrine of reverse engineering, generally, would retard the public's access to source code if applied to the DMCA in *Corley*.

[15] *See* Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1527 (9th Cir. 1992). Quite the contrary to the *Corley* court's determination, the fruit of reverse engineering is not likely useful if the data dump is immediately directed toward an incompatible operating environment. The confusion for the courts may have arisen as a result of the fact that DeCSS is a utility software program that does more than circumvent an access control. Even so, understood correctly, the fact should have counted as a favorable one for the defendants since the DMCA's provisions are explicitly directed toward proscribing access controls of limited, rather than broad, purpose.

[16] It should be apparent that an access control may be used on a DVD-ROM or CD-ROM to block access to data subject to copyright protection as well as that which is not with equal force. Hence, there is special need for courts to be cautious as they tread through the DMCA's technological barriers.

[17] DVD Copy Control Association v. Bunner, (Santa Clara, CA 6th App. Dist. Nov. 1, 2001) (unreported).

[18] *Id.* at 657.

12. *Corley*, in particular, so far has had the perverse effect of invalidating reverse engineering when occurring in the context of the public dissemination of source code that contains information concerning how reverse engineering might be undertaken. Oddly enough, this trend provides an apparent cover for content-based speech restrictions directed toward the activities of some libertarian-oriented Internet-based computer hackers and the open source community, whose activities are focused upon unleashing the locked up ideas of software. Indeed, the DMCA's ostensible approval of locking up access to source code regardless of whether the source code meets the originality requirement of copyright violates the constitutional mandate that copyright protection (which includes a para-copyright statute like the DMCA) only extend to works or those aspects of works that meet the originality requirement.

II. Introduction

13. Although the appropriate scope for the copyright protection of computer software[19] has been subject to extensive commentary,[20] the boundaries between what is within the scope of copyright protection for software and what is not[21] remains fuzzy and subject to semantic invention.[22] This is particularly true in the

[19]A computer program is a set of instructions to a computer. *See generally* Michael S. Keplinger, *Computer Software – Its Nature and its Protection*, 30 EMORY L.J. 483, 484-85 (1984) (source code is "a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result"). Source code is written in a computer language, and may form the basis of a copyrightable work. Often, source code also includes non-functional phrases or sentences referred to as comments meant to be read by humans for enhanced understanding of how the computer program functions, but having no purpose regarding the intended successful execution of the software program by a machine. Since a computer language is an abstract, systematized formation of signs and symbols used to develop or construct computer programs, the computer language, itself, is not copyrightable. Source code ultimately controls the software and hardware that taken together function as computers. Computers are digital technologies, and, as such, can be used to efficiently express vastly different forms of information – such as factual databases, audio recordings, or electronic mail messages – using bits of data in the form of computer 0s and 1s. Bits (or binary digits) are essentially the smallest and most fundamental units of digital technology data; each bit has a value of 0 or 1. The bits 0 and 1 represent off and on switches, which measure the presence or absence of electrical voltage in any given memory register of the computer. Since binary digits enable fairly easy digital expression and digital technology significantly expands the amount of data that can be processed on a single silicon chip, computers have become the format of choice in electronics. *See generally* Dixon, *supra* note 3; Robert W. Gomulkiewicz, *How Copyleft Uses License Rights to Succeed in the Open Source Software Revolution and the Implications for Article 2B*, 36 HOUS. L. REV. 179, 180 (1999) (noting that source code consists of highly readable programming statements).

[20] Indeed, the debate has included disagreement over whether computer programs ought to be protectable subject matter under copyright at all. *Cf.* Robert A. Kreiss, *Accessibility and Commercialization in Copyright Theory*, 43 UCLA L. REV. 1, 55-64 (1995); Paul Goldstein, *The Future of Software Protection: Infringement of Copyright in Computer Programs*, 47 U. PITT. L. REV. 1119 (1986).

[21] *See, e.g.,* Peter S. Mennell, *An Epitaph for Traditional Copyright Protection of Network Features of Computer Software*, 43 ANTITRUST BULL. 651 (1998); Dennis S. Karjala, *The Relative Roles of Patent and Copyright in the Protection of Computer Programs*, 17 J. MARSHALL J. COMPUTER & INFO. L. 41, 50-56 (1998) (noting disagreeing views over the scope of copyright protection to computer software code).

[22] *See, e.g.,* Yochai Benkler, *Free as the Air to Common Use: First Amendment Constraints on Enclosure of the Public Domain*, 74 N.Y.U. L. REV. 354 (1999); Julie E. Cohen, *Lochner in Cyberspace: the New Economic Orthodoxy of "Rights Management,"* 97 MICH. L. REV. 462 (1998) (deconstructing the rhetoric

context of copyright infringement actions involving computer source code.[23] Even so, the global importance of the Internet and the increasing vitality of the software industry warrants further examination of the question regarding whether there really is a principled basis for delineating distinctions between copyrightable and uncopyrightable expression in software.[24]

14. As a starting point, viewing source code as a cauldron of basic ideas[25] underlying copyrightable expression resolves some of the limitations in previous debates concerning the scope of copyrightable expression within software by clearly fixating most of computer source code in the marketplace of ideas or, in other words, outside the scope of copyright protection.[26] Unfortunately, the proper scope of copyright protection for software has been further distorted by Congress' recent enactment of the DMCA, which substantially diminishes the fair use standard permitting an exceptionally narrow use of reverse engineering, as well as by the recent trend among courts to restrict the fair use doctrine of reverse engineering[27] in the context of copyright infringement.[28] Undeniably, the DMCA

---

of economic efficiency by comparing Chicago-school analysis to the analysis in Lochner v. New York, 198 U.S. 45 (1905)); Michael Madison, *Legal-ware: Contract and Copyright in the Digital Age*, 67 FORDHAM L. REV. 1025, 1053 (1998); Pamela Samuelson et al., *A Manifesto Concerning the Legal Protection of Computer Programs*, 94 COLUM. L. REV. 2308 (1994).

[23] *See* David Nimmer, *Adams and Bits: Of Jewish Kings and Copyrights*, 71 S. CAL. L. REV. 219, 223-24 (1998) (noting that the traditional copyright scheme is ill-equipped to deal with the information age, and that it is likely that some "copyright industries [will] thus watch in hopeful horror as the [Internet] revolution unfolds.").

[24] See Rod Dixon, *When Efforts to Conceal May Actually Reveal: Whether First Amendment Protection Of Encryption Source Code and the Open Source Movement Support Re-Drawing The Constitutional Line Between the First Amendment and Copyright*, 1 COLUM. SCI. & TECH. L. REV. 3 (2000), available at http://www.stlr.org/cite.cgi?volume=1&article=3; *see also* Marci A. Hamilton, *The TRIPS Agreement: Imperialistic, Outdated, and Overprotective*, 29 VAND. J. TRANSNAT'L L. 613 (1996); Jessica Litman, *The Exclusive Right to Read*, 13 CARDOZO ARTS & ENT. L.J. 29 (1994).

[25] As noted *infra*, that computer source code is speech and should be viewed as such is a remarkable determination. Notwithstanding that declaring that 'source code is speech' is simplistic in its formulation, the impact of the determination could be far reaching and profound, not the least of which includes a reconsideration of the proper scope of copyright protection in source code. *See* Dixon, *supra* note 24 (noting that "[q]uite apart from the impact of the dispute on the future use of encryption, the ultimate resolution of the question will have an astonishing level of influence on the current jurisprudence governing the balance between copyright and the First Amendment in the context of computer programs.").

[26] Once commentators and courts tagged source code as a so-called literal aspect of software and bootstrapped that description with the Copyright Act's classification of computer programs as a literary work, few doubted that conclusive proof had been established that source code is a literal aspect of software that must constitute copyrightable expression. The superstition that literal aspects of software are distinguishable from the nonliteral and that literal is expressive in the copyright sense is repeated in judicial opinion and legal commentary as if it had the quality of a mantra. As noted more fully below, it has not been overthrown yet. The classification serves as a reference to the distinction between source code and all other aspects of software not including textual source code. More properly, the distinction should serve as a shorthand reference to a continuum running from source code (literal) to the fundamental essence and structure of a software program. In this manner, the reference would [not] classify inherent aspects of software programs, but, instead, would aid in assessing what aspects of a computer program evidences infringing conduct. Somewhere along the line of the continuum, literal becomes nonliteral, but, at that point, the reference may still be to source code, but not to "literal" copying of source code.

[27] *See generally* MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT, §13.05[D][4] (1999).

is uncharacteristic of a copyright law because it undercuts the public's access to the ideas embedded in source code.

15. The first court to apply the DMCA to computer source code, *Universal City Studios, Inc., v. Corley*,[29] rejected the defendants' argument that under the circumstances of the case the practice of reverse engineering copyright-protected works like a software program was supported by doctrinal as well as statutory privilege, which allows defendants to avoid the liability that would otherwise apply under the DMCA.[30] According to the court, if a software user successfully obtained access to source code by circumventing a technological "lock" or barrier to access without authorization from the copyright holder of the computer program, the circumvention could not be considered permissible under the doctrine of reverse engineering as interpreted by the DMCA unless the defendant could persuade the court that her purpose for circumventing or breaking the access control was to determine how two programs interoperate.[31]

16. In *Corley*, the plaintiffs, eight motion picture studios, brought suit under the DMCA, *inter alia*, seeking an injunction against the magazine and website, known as 2600 and 2600.com, (and several other defendants) to prohibit the defendants from publishing, disseminating, trafficking in, and posting on or linking to a website with the source code to a decryption program that allowed computer users of the Linux operating system (OS) to play lawfully acquired movies digitized by the copyright holder onto DVDs.[32] The source code was alleged to provide a way of circumventing a technological access barrier – namely CSS. This was accomplished by use of a decryption software program called DeCSS,[33] which was developed after reverse engineering CSS. Apparently, at the time DeCSS was developed, DVDs were sold with CSS to allow users to play the content on DVD-ROM drives connected to computers running only the Windows operating system developed by Microsoft Corporation.[34]

---

[28] It is a well-established principle of fair use of copyright-protected computer programs that reverse engineering of those programs to obtain access to the underlying ideas embedded within the software's source code is a fair use of the software. As noted more fully below, courts that have reviewed the matter have interpreted the DMCA's reverse engineering exception more narrowly and, hence, have fallen out of step with the constitutional division between copyright and the free expression of ideas.
[29] Universal City Studios, Inc. v. Corley, 273 F.3d 429 (2d Cir. 2001), *aff'g* Universal City Studios, Inc. v. Reimerdes, 111 F.Supp.2d 294 (S.D.N.Y. 2000).
[30] *Corley*, 273 F.3d at 429.
[31] *Reimerdes*, 111 F.Supp.2d at 294.
[32] *Id.* DeCSS is not a DVD-playing software program.
[33] Interestingly enough, since the district court also issued an injunction prohibiting the defendants from posting DeCSS on websites, a music website that was not a party to the proceedings, mp3.com, determined it was obligated to follow the court's ruling as well. In doing so, on September 11, 2000, the website sent a letter to the author of the song at the top of its folk music chart, Joseph Wecker. The letter informed Wecker that his song, Descramble, with lyrics that contained a fourth of the source code contained in DeCSS, was being deleted from the mp3.com website. *See* Peter Maass, *The Supercool Top-Secret DVD-Decoder Song*, THE NEW YORKER, Oct. 16 & 23, 2000 at 92-94.
[34] Indeed, to thwart software users from disrupting the motion picture industry's attempt to tighten its grip on potential revenue sources from new digital media, the industry filed three different lawsuits against

17. *Corley* both narrowed the scope of the defendants' conduct that could come within the statutory exception permitting reverse engineering and limited the range of protection allowing public access to a work for the purpose of reverse engineering. The court also took the unprecedented step of restricting the reverse engineering exception to circumstances, involving the dissemination of information, solely for the purpose of achieving interoperability, among disparate computer programs, which the court (mistakenly) determined was unlikely to be among the defendants' actual purposes. In the court's view, interoperability is not likely to have been the goal of the reverse engineer and if the immediate fruits of reverse engineering may lead to the development of a decryption program designed to operate on the same OS, rather than immediately interoperate on a different OS.[35] In addition, the court, rather conspicuously, failed to consider whether obtaining access to unprotectable ideas had been the basis of defendants' efforts of reverse engineering. Instead, the district court's analysis followed a lock-step argument presented by the plaintiffs concerning the copyright in the motion picture content on the DVD, which had nothing at all to do with reverse engineering the CSS.[36]

18. The district court in *Corley* is not the only court recently to question or doubt a defendant's claim to be engaged in legitimate or credible reverse engineering conduct in the context of software. In fact, there is a troubling trend toward increasing statutory and judicial approval of a software developer's prerogative to block public access to source code without exception, despite the appealing logic of the notion that copyright protection of the expressive quality of source code should result in its exposure, not its secrecy. In a case related to *Corley*, *DVDCCA v. Bunner*, a trial court rejected the argument that a 15-year-old Norwegian programmer had engaged in lawful reverse engineering for purposes of interoperability.[37] The court's skepticism seemed to have emanated from its concern that the 15-year-old was an alleged hacker who had publicly declared his disrespect for the law of copyright.[38] Although the plaintiff had sought injunctive relief under a state law trade secret misappropriation claim, rather than a claim under the DMCA, the court's application of the doctrine of reverse engineering, generally would retard the public's access to source code if applied to the DMCA in *Corley*.[39]

---

website owners. The *Bunner* trade secret claim was initially filed against 72 website owners, many of whom were viewed as Internet-based hackers. In two other cases, filed in the federal courts of New York and Connecticut, the industry filed suit against website owners and operators for alleged violation of the Digital Millennium Copyright Act. In each case, the movie industry representatives argued that the defendants essentially attempted to obtain the benefit of using an electronic "crowbar" to force open a locked file. In the defendants' view, their conduct of posting source code on the Internet was no different than buying an electronic item and, after paying for it, opening the item to look under the "hood" to see what is inside. *Reimerdes,* 111 F.Supp.2d at 294.

[35] *Reimerdes*, 111 F.Supp.2d at 324; *Corley*, 273 F.3d at 429.

[36] *Id*.

[37] DVD Copy Control Association v. Bunner, (Santa Clara, CA 6th App. Dist. Nov. 1, 2001) (unreported).

[38] *Id*.

[39] *Id*.

19. What these recent judicial interpretations of the doctrine of reverse engineering and the text of the DMCA itself have in common is an acceptance, if not reverence, of the copyright holder's desire to restrict the scope of lawful reverse engineering without accommodating an important access point of the public to public domain material. The doctrine of reverse engineering is being redefined as protecting a limited practice authorized solely as an apparent economic efficiency primarily benefiting pertinent intellectual property owners.[40] Divorcing the reverse engineering doctrine from its context in fair use and free expression removes the public's only access to the ideas and functional elements embodied in software.[41] That this privilege of access is being replaced by a technological barrier sanctioned by Congress with the force of civil and criminal liability is demonstrative of how close we have come to an ill-fated future for the public domain of information and information products. This does not simply promise a future where the user of digital information will pay an "owner" for its use in each and every instance, but includes a potential guarantee that some owners will be more equal than others.[42] For some, copyright law rightfully will favor the dissemination of works like CSS over DeCSS; for others, copyright law will become an increasing patchwork of legislative favoritism dislodged from its constitutional purpose.

20. Part III of this article sets the technological backdrop of the primary argument that now is the time to recalibrate the balance between copyright and free expression for computer software. Specifically, Part III documents the practice of reverse engineering software and sets forth arguments why this technological process is central to our understanding of ideas embedded within software, which ostensibly resides in the source code. Part III also provides the roots for later sections of the article concerning a paradigm shift in the process of computer programming that ought to inform the reader's understanding of why previous interpretations of how copyright might promote the progress of software development require adjustment.

21. Part IV outlines why both copyright law's constitutional mandate of originality and the First Amendment's value for the free expression of ideas well serve the

[40] William M. Landes & Richard A. Posner, *An Economic Analysis of Copyright Law*, 18 J. LEGAL STUD. 325, 332-33 (1989); David A. Rice, *Public Goods, Private Contract and Public Policy: Federal Preemption of Software License Prohibitions Against Reverse Engineering*, 53 U. PITT. L. REV. 543 (1992); See generally Julie E. Cohen, *Lochner in Cyberspace*, *supra* note 22.
[41] Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1527 (9th Cir. 1992).
[42] This form of inequality is suitable in the eyes of some who have argued that the concern that some intellectual property owners may use intellectual property rights to bottle up useful ideas so that others may not have access rarely has been taken seriously by technology giants like Xerox, Hewlett Packard, Microsoft, or Motorola. Apparently, these companies have the financial wherewithal to pooh-pooh the law, find "social virtue" in committing infringement, act like "grownups" by litigating each other to a standstill, and, then, bulldog or dupe the victim of piracy into signing a cross-licensing deal. *See* Holman W. Jenkins Jr., *Busting the Intellectual Property Bubble*, WALL STREET JOURNAL, March 29, 2000, at A23 (citing a 40 year practice of Silicon Valley companies whose view of someone else's intellectual property – particularly regarding patent rights – could be described aptly as steal now, litigate later).

limits of copyright for software. In this regard, the argument is advanced that the court's prevailing conception of "source as speech" is too analytically debilitated to sufficiently sustain the proper balance between copyright and free expression. The recent case involving the DMCA is shown to be a series of strong lights illuminating support for this argument.

22. In Part V, the open source community is assessed for its impact on the paradigm shift in computer programming, as well as its success in demonstrating countervailing incentives that promote the progress of software development without some of the compromises to the public domain that the current prevailing legal regime accommodates.

23. Part VI concludes that the future progress of software development requires open access to the ideas embedded within computer programs. The vehicles for ensuring open access, such as fair use and the ability to reverse engineer closed code, have been put under unprecedented constrainment by legislative consequences arising from the enactment of the DMCA, technological isolation of courts reviewing these matters with analytic tools unsuitable for the matter at hand, and recent trends to expand intellectual property right interests for the creators of digital works. Part VI proposes a re-calibration of the scope of copyright protection for software and consistent with this proposal, a suggestion that the DMCA be interpreted structurally to fully embrace the privilege of reverse engineering. In this manner, like the open source community, all developers of software may be able to create works openly based upon the unleashed and free access to the ideas embedded in software and currently locked up and hidden by barriers of access that stand before the law.

III. Locked Rooms: How and why to obtain access to source code

A. The Process Of Reverse Engineering

24. It is a well-established principle that reverse engineering of copyright-protected computer programs to obtain access to the underlying ideas embedded within the software's source code is a fair use of the software.[43] Courts that have recently reviewed the matter have interpreted the DMCA's reverse engineering exception more narrowly and are far out of step with the constitutional division between copyright and the free expression of ideas.[44] In calibrating the degree to which reverse engineering ought to be supported for software in the fast-changing context of the Internet, one should first determine whether it is correct that reverse engineering neither threatens the public interest in supplying creative works for the public domain nor threatens the interests of copyright holders.[45] Although

[43] *Sega*, 977 F.2d at 1527.
[44] *Id*.
[45] *Sega*, 977 F.2d at 1519-27; Andrew Johnson-Laird, *Software Reverse Engineering in the Real World*, 19 U. DAYTON L. REV. 843, 901 (1994).

copyright holders are quick to see doom in the context of new technologies applied to their business models, Congress should recognize when these doomsday predictions are the sounds of crying wolves. Not long ago, for example, during the Congressional hearings held in response to the Supreme Court's fair use analysis in the *Universal City Studios, Inc. v. Sony* decision, the Motion Picture Association of America (MPAA) beseeched Congress to provide legal authorization for a licensing scheme to compensate Hollywood for the Court's decision.[46]

25. Among its many claims, the MPAA argued that fair use in home videotaping of audiovisual broadcast programs would inevitably result in a "barren wasteland" of uninspired and unentertaining television programs. It is not entirely clear whether Hollywood believed that its request for a royalty arrangement, which Congress granted, would alleviate what was inevitable or just make the painful reality less painful. Quibble, if you must, but the MPAA may have been demonstrating proof of exceptional perspicacity rather than simply crying wolf in this instance. Nevertheless, the tactic worked and it has been a consistent response to the appearance of new technology that may deliver audio or audiovisual works in a new style, mode, or technological forum.[47]

26. Whatever threat reverse engineering poses in the context of new technology, it is doubtful that the end-game consistently results in losses for copyright holders and no countervailing gains to the public interest.[48] Indeed, as shown, *infra*, new technologies, like those produced to support the Internet's infrastructure, can produce gains for the public domain as well as provide greater opportunity for potential copyright holders to produce original works.[49] Reverse engineering is important in the software context because software is a unique copyrightable work.[50] The work does not provide the "expressive" benefits that copyrightable works usually provide upon publication.[51]

[46] *See* Universal City Studios, Inc. v. Sony Corp., 464 U.S. 417, 429 (1984).

[47] *See, e.g.,* Universal City Studios, Inc. v. Reimerdes, 111 F.Supp.2d 294 (S.D.N.Y. 2000), Universal City Studios, Inc. v. Corley 273 F3d 429 (2nd Cir. 2001).

[48] For example, strictly speaking, it cannot be disputed that one exploitative "cost" to the copyright holder of the privilege of reverse engineering includes the making of an intermediate copy of a software program. *See, e.g.,* John G. Mills, *Possible Defenses to Complaints for Copyright Infringement and Reverse Engineering of Computer Software: Implications for Antitrust and I.P. Law*, 80 J. PAT. & TRADEMARK OFF. SOC'Y 101, 106 (1998).

[49] *Sega*, 977 F.2d at 1527.

[50] *See* Andy Johnson-Laird, Technical Demonstration of Decompilation, Address, in Copyright Protection of Software: Derivative Works and Reverse Engineering, 361-87 (1992); U.S. Congress, Office of Tech. Assessment, Finding a Balance: Computer Software, Intellectual Property and the Challenge of Technological Change 148 (1992); *See generally* G. Gervaise Davis III, *Scope of Protection of Computer-Based Works: Reverse Engineering, Clean Rooms and Decompilation*, 370 COMPUTER L. INST. 115, 142-43 (PLI Patents, Copyrights, Trademarks, and Literary Property Practice Course Handbook Series No. G-370, 1993) (listing teaching, writing, debugging, emulation, modification, achieving interoperability and developing competitive replacements as various rationales for reverse engineering).

[51] *See* Seth A. Cohen, *To Innovate or Not to Innovate, That is the Question: The Functions, Failures, and Foibles of the Reward Function Theory of Patent Law in Relation to Computer Software Platforms*, 5 MICH. TELECOMM. TECH. L. REV. 1 (Nov. 23, 1998) http://www.mttlr.org/volfive/cohen.html. Mark A.

27. For non-software literary works, once the work is published, the ideas contained in the work become apparent and conspicuous. In this manner, the work's basic ideas may provide a basis for further development of additional works by any member of the public with access to the work.[52] This is a critical function of the public domain because it serves the primary purpose of copyright.[53] Ironically, the relationship between the public domain and open access to ideas is quite like a recursive function in a computer program, wherein a call to a procedure may result in a procedure repeating itself indefinitely. The enrichment of the public domain of ideas increases the public's access to works, which further enriches the public domain.[54] Software, however, presents a slight conundrum that disrupts the flow of the recursive enrichment of the public domain served by copyright law.[55]

28. Software is executed or is run on computers in a form that often hides the source code from all but the copyright holder. Although graphical user interfaces have enabled technologically savvy end-users greater access to the ideas embedded within software, many ideas that can be discerned from software are woven into the source code. The publication or distribution of software does not alter this result. Hence, the complete bargain ostensibly accomplished by the government's grant of copyright is unfulfilled by software developers who either block access or withhold access to the ideas embedded within source code from end-users through a number of methods, including technological barriers, hardware locks, licensing restrictions, and claims that reverse engineering software is unsupportable under copyright law.

29. Often, reverse engineering requires an end-user to use a software tool that reverses the process of software development to yield the source code. Reverse engineering, in this manner (often called decompilation), does not yield an exact copy of the original source code, but it does reveal at least sub-routines showing the structure and operation of a program. In other words, these are the basic ideas embedded in the software. To effectuate reverse engineering, it is necessary to copy the underlying software. Although the copy produced might be considered an intermediate copy, the copy is, nonetheless, "fixed" in the computer's active memory or RAM. Fixation of a copyright-protected work – even when only in RAM – produces an infringing copy of the work under current interpretations of copyright law. Hence, reverse engineering software is likely to constitute

---

Lemley, *Economics of Improvement in Intellectual Property Law*, 75 TEXAS L. REV. 989, 992 (1997); Robert Merges, *Intellectual Property Rights and Bargaining Breakdown: The Case of Blocking Patents*, 62 TENN. L. REV. 75 (1994).

[52] Yochai Benkler, *supra* note 22; Julie E. Cohen, *Lochner in Cyberspace, supra* note *22*; Michael Madison, *supra* note 22.

[53] *See* Madison, *supra* note 22, at 1092-1107 (noting that the public domain is a doctrinal representation of open space implicitly emanating from the copyright statute and the Constitution).

[54] *But see* Wendy Gordon, *Fair Use as Market Failure: A Structural and Economic Analysis of the Betamax Case and its Predecessors*, 82 COLUM. L. REV. 1600 (1982) (arguing that fair use is best viewed as a remedy for market failure).

[55] *See generally* Pamela Samuelson, *CONTU Revisited: The Case Against Copyright Protection for Computer Programs in Machine-Readable Form*, 1984 DUKE L.J. 663.

copyright infringement, unless found permissible under a judicially created doctrine of reverse engineering or the statutory defense of reverse engineering.[56]

30. Is computer programming a highly creative and individualistic endeavor?[57] Do programs[58] consist predominantly of commonly known techniques and materials strung together with skill, but without significant originality?[59] And what if a program has become so popular as to set a *de facto* standard, such that users come to expect its keystrokes and command structures to be present in any program designed to accomplish the same functionality?

31. It is axiomatic that material in the public domain is not protected by copyright, even when incorporated into a copyrighted work.[60] It cannot be doubted that highly inefficient and skilled programmers often build existing public domain software or, perhaps, source code into their works.[61] In this respect, courts should be mindful under our prevailing copyright jurisprudence to limit protection only to those elements of the program that represent the author's original work – even

[56] Andrew Johnson-Laird, *Software Reverse Engineering in the Real World*, 19 U. DAYTON L. REV. 843, 901 (1994).
[57] Although the common perception that software is developed by an individual programmer has yielded to our understanding that software development often requires development teams, the debate whether software development is more science or engineering than art is unresolved. The recent adoption of 'eXtreme Programming' as a description emphasizing that software development is a social process supports the increasingly accepted view that software development and design is a creative endeavor. *See* PETE MCBREEN, SOFTWARE CRAFTMANSHIP: THE NEW IMPERATIVE 26 (Addison-Wesley, Boston) (2002).
[58] As noted *infra*, under the law of copyright, a computer program has both literal and non-literal aspects. Although this distinction may have some lasting utility, the fact that copyright law persists with the bright-line distinction further illustrates why the time has come for a refreshing re-assessment of the scope of copyright protection for software. For most, it is no surprise that literal and non-literal aspects of software can always be reduced to their source code equivalent. Hence, so-called non-literal aspects of software should be viewed independently from the software program in the context of a copyright infringement action.
[59] Advanced programming tools – sometimes called RAD or Rapid Application Development tools and sometimes called glue – have significantly simplified the mundane tasks of software development for projects that either require a graphical user interface (GUI) or significant user interaction. These tools allow developers to drag and drop objects onto a window rather than write hundreds of lines of source code to accomplish the same task. Dialog boxes that require user input, menu structures that drop down upon a user's mouse click can be programmed or glued into a program in seconds instead of painstaking hours of typing source code. Although this advancement in programming has been tremendously popular within computer programming circles, American copyright law seems immune to the advancement. Even fairly complex software programs can be produced with substantial aspects of the source code essentially pre-coded or generated by the machine in response to a drag and drop. Tim Berners-Lee, who created the software that became the basic protocols for the World Wide Web by using drag and drop RAD tools, provides a nice example; in describing the development of the Web's first web browser client, he notes that he created a significant portion of the browser "easily, just dragging and dropping" menus and windows "into place with a mouse." TIM BERNERS-LEE, WEAVING THE WEB 28 (New York: Harper Collins 1999). (Of course, the point is not that the task of programming is simplistic or easy. And, it is readily apparent that some programmers steer clear of RAD tools. Notwithstanding these concessions, the fact is that copyright laws exist unfazed by contemporary programming practices, and these practices tend to disfavor concluding that the source code for a software program is an original work of independent authorship).
[60] *See* Samuelson, *CONTU Revisited*, *supra* n. 55 at 704.
[61] *See generally* Pamela Samuelson et al., *A Manifesto*, *supra* note 22.

if doing so is an extremely difficult task for the court since the protection of the public domain today has become too important to shirk or ignore.[62] Given the vast likelihood that any particular software program contains significant portions of unprotectable material – in some instances so significant that the program itself may be unprotectable – it strains logic and common sense to believe that the case has been made that software programs ought to obtain the additional protections granted by the DMCA based on principles of copyright law.

32. The Copyright Act defines a computer program as "a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result."[63] Indeed, software authors can and do bring copyright infringement suits against other software authors.[64] In such a case, the defending author is likely to argue that if any copying at all occurred he took merely unprotected ideas from the other work. Under current copyright doctrine, a copyright infringement claim involving source code[65] entails an allegation of literal copying.[66] As such, a claim by the defendant that he only copied unprotected ideas would require the reviewing court to either apply the idea/expression dichotomy or to reject the defense on its face as inconsistent with the evidence of literal copying.[67] It is in this regard that courts require claimants to separate a software program's protected expression from its unprotected ideas, including the relevant aspects of the source code.[68]

[62] *See generally* Greg Aharonian, *Deconstructing Software Copyright, 30 Years of Bad Logic,* Internet Patent News Service (Oct. 2001), *available at* http://www.bustpatents.com/aharonian/softcopy.htm (visited Dec. 30, 2001).

[63] Copyright protection arises automatically upon creation of the work. For those who choose to register their software with the Copyright Office, only a small portion of the program's source code is deposited with the Copyright Office. 37 C.F.R. § 202.20(c)(2)(vii)(A)(1) (2002) (only first and last 25 pages of source code reproduced on paper need be deposited).

[64] *See* Computer Assocs. Int'l, Inc. v. Altai, Inc., 982 F.2d 693, 712 (2d Cir. 1992) ("We think that copyright registration – with its indiscriminating availability – is not ideally suited to deal with the highly dynamic technology of computer science"); Baker v. Selden, 101 U.S. 99, 102 (1879) ("To give to the author of the book an exclusive property in the art described therein, when no examination of its novelty has been officially made, would be a surprise and a fraud upon the public. That is the province of letters-patent, not of copyright").

[65] Source code, defined from a programmer's point of view, is a set of symbols governed by lexical rules that computer programmers use to instruct computers to perform certain actions. JOSEPH WEBER, USING JAVA 1.1 74 (Third Edition) (1997).

[66] Apple Computer, Inc. v. Microsoft Corp., 35 F.3d 1435, 1442 (9th Cir. 1994) ("We conclude that only 'thin' protection, against virtually identical copying, is appropriate"), *cert. denied*, 513 U.S. 1184 (1995).

[67] *See* Julie E. Cohen, *Lochner in Cyberspace*, *supra* note 22; Jane C. Ginsburg, *Putting Cars on the Information Superhighway: Authors, Exploiters, and Copyright in Cyberspace*, 95 COLUM. L. REV. 1466 (1995).

[68] Of course, the application of the idea/expression dichotomy to source code occurs regardless of whether the copyright infringement claim is directed toward literal aspects of software or so-called non-literal aspects of software. The distinction between literal and non-literal aspects of software has an arbitrary boundary. A potential copyright infringer may decide to "copy" the aspect of a computer program that captures user input from the screen display, for example; in doing so, the potential infringer may copy copyrightable non-literal aspects of the graphical user interface as well as literal aspects of the underlying source code that yields the display. Often, it is difficult to determine whether an infringing copy was created by reproducing literal code, non-literal code or both.

33. Although the legislative history of the 1976 Copyright Act indicates that Congress intended for the revised copyright statute to protect computer programs, courts did not agree on the contours of what constituted a computer program under the Copyright Act until Congress amended the Act by enacting the Computer Software Copyright Act of 1980,[69] which added the definition of computer program.[70]

34. Thereafter, the Copyright Office had issued a circular stating that copyright protection extends to the literary or textual expression contained in a computer program. Courts then began summarily applying the idea/expression dichotomy to software programs as a proxy for constitutional analysis of the inherent First Amendment question implicated by the idea/expression dichotomy.[71] In several cases, the inconsistent application of the idea/expression dichotomy shows that the broad concept of "expression" used in traditional literary works infringement cases seems strained when applied to software.[72] This may be particularly true because computer programs are, in significant respects, artifacts of technology.[73] Within the U.S. regime of intellectual property, copyright has not been the traditional province of protection for technological devices; that domain belonged to the law of trade secrets and patents.[74]

35. A significant source of consistency between programs arises out of programming practices and techniques that have become widely used and accepted in the computer software industry. Most programmers rely on a number of traditional solutions to recurring problems in their programming. Standard programming techniques are as much the "stock" elements of computer programming as are the common themes, incidents, and plot elements referred to in the traditional literary cases.[75]

[69] *See* Samuelson, *CONTU Revisited*, *supra* note 55 at 704-05.
[70] *Id.*
[71] *See, e.g.,* Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240 (3d Cir. 1983).
[72] *See* Paul Edward Geller, *Copyright History and the Future: What's Culture Got to Do With It?,* 47 J. COPYRIGHT SOC'Y 209, 254 (2000).
[73] Sony Computer Entertainment, Inc. v. Connectix Corp., 203 F.3d 596, 599 (9th Cir. 2000) ("Copyrighted software ordinarily contains both copyrighted and unprotected or functional elements.").
[74] *See* Dixon, *supra* note 2. The idea/expression dichotomy is by no means troublesome only in the context of software. More than just a few commentators have found the notion that an idea and its expression may be parsed with the precision required by the jurisprudence of copyright to be sophistic, at best, and, perhaps, more appropriately, intellectually objectionable. Perhaps, the better argument is that the meaning or idea conveyed by an expression may be rooted in a combination of linguistic and interpretative phenomena, including analytical pragmatics, semantics, and syntax, as well as other structural, contextual, and logical interpretative considerations. The intricacy in which meaning is tied to language renders the task of separating an idea from its expression a highly dubious endeavor. *See, e.g.,* NEIL SMITH & DIERDRE WILSON, MODERN LINGUISTICS: THE RESULTS OF CHOMSKY'S REVOLUTION, 170-171 (1980). No less a respected authority on legal doctrine than Judge Learned Hand is said to have concluded that a court's distinction between an idea and its expression will "inevitably be ad hoc." Peter Pan Fabrics, Inc. v. Martin Weiner Corp., 274 F.2d 487, 489 (2d Cir.1960). Thus, the doctrine may have no principled basis.
[75] *See also* note 59 *supra* and accompanying text noting the use of RAD tools in software development and how such use minimizes originality.

36. The open source/free software community represents a clear paradigm shift in programming.[76] Traditionally, computer programming was a solitary task performed by a programmer on a single machine. Other programmers or other machines could not understand most of the programmer's code. The instructions to a computer, or program, must be given to the computer in the form of "machine language" notation. Machine language is, however, difficult for humans to comprehend.[77] Generally, instead of writing machine language instructions that the processor can execute directly, programmers write programs in a programming language, which then is translated mechanically to machine language by a compiler program.[78] Machine-readable object code, incomprehensible to people, consists of a string of ones and zeros, which are the only two symbols a digital computer can understand.[79]

37. In some sense, software design is a process that programmers learn more through practice than from books, a process that cannot easily be formulated as a set of rules. Often, the end product, the program, is generally the result of numerous conscious choices by the programmer.[80]

38. Microcomputers significantly changed the task of the programmer. Software engineers designed programming languages that could be understood by others and run on computers built under specifications.[81] What is more, as Cyberspace has become an increasingly useful environment for computer programming, programmers are developing a reluctance to systematically "hide" their source code from each other. Instead, programming began undergoing a paradigm shift away from viewing source code as the province of secrecy and toward sharing source code in the form of reusable modules or objects.[82] In an attempt to build more complex applications that could run on desktop computers and local networks, programmers routinely share and exchange code that can be used and re-used for various projects.[83] In this regard, a wealth of source code is used and

[76] Richard Stallman, *The GNU Operating System and the Free Software Movement*, *in* OPEN SOURCES: VOICES FROM THE OPEN SOURCE REVOLUTION 59 (Chris DiBona et al. eds., 1999).
[77] Andrew Johnson-Laird, *Software Reverse Engineering in the Real World*, 19 U. DAYTON L. REV. 843, 856-59 (1994) (describing the process by which source code is created by a programmer, and then translated into computer-readable object code).
[78] *Id.*
[79] *Id.*
[80] *See* Arthur R. Miller, *Copyright Protection for Computer Programs, Databases, and Computer-Generated Works: Is Anything New Since CONTU*?, 106 HARVARD L. REV. 978, 984 (1993) (comparing "the communicative precision required of a computer programmer" with "the discipline that a poet must achieve to convey a complex message within the confines of a tightly constrained meter," and the programmer with the "composer who must work within the limited ranges of musical instruments or the human voice").
[81] Johnson-Laird, *supra* note 77.
[82] *See* Mark A. Lemley & David W. O'Brien, *Encouraging Software Reuse*, 49 STAN. L. REV. 255 (1997) (discussing value of modularity to software reuse).
[83] *See supra* note 78. Johnson-Laird, *supra* note 77. The description of software design or computer programming is not intended to encompass software engineering, which many computer scientists view as entirely distinct from programming.

re-used, copied and shared as raw code and often as libraries and modules, which are distributed in the public domain.[84]

39. The most significant constraint on an open source code project may involve finding enough programmers available and interested in contributing their time jointly authoring freely available software projects. In this respect, Cyberspace has provided the tools necessary to bring together enough people to harness the intellectual efforts required to create serious software programs sufficient to support the paradigm shift in programming. It is quite possible that the growth of the Internet will complete the programming paradigm shift since enough Cyberspace-based programmers will be available to make both large scale projects and small programming alliances viable and routine. Open source code collaborative programming efforts may become standard. In this regard, the existence of the open source code community amply supports a conception of copyright that provides sufficient incentive for software developers to create works from a vast public domain. This is a public domain that would provide access to source code for authors and, in turn, encourage software authors to create works that could promote the progress of science, thereby, further enrich the public domain.[85]

B. Reverse Engineering under Section 1201[86] of the DMCA

40. It has been often said that the DMCA[87] was enacted to accommodate two important priorities: promoting the continued growth and development of

[84] Scripting languages blur the traditional distinctions between source code and object code because most programs written in scripting languages are executed and compiled at the same time. While this distinction has unclear relevance to copyright, it very well may be a significant factor in assessing the government's regulation of the export of an encryption program.
[85] This is consistent with the Supreme Court's analysis in Feist Publications Inc., v. Rural Tel. Serv. Co., where the Court required "some minimal degree of creativity," or a "minimal creative spark" before finding copyrightability in a compilation of a telephone book's white pages. 499 U.S. 340, 362 (1991). Notably, excluding copyright protection in source code is not tantamount to eliminating software programs from the scope of copyright protection. Computer software, like a book or a screenplay, may contain both copyrightable and uncopyrightable aspects. A computer program's screen output may be copyrightable, although the source code would not be. *See, e.g.,* Gates Rubber Co., v. Bando Chem. Indus., Ltd., 9 F.3d 823 (10th Cir. 1993). Since courts have long held that software programs contain both literal and nonliteral elements subject to copyright protection, it is highly doubtful that removing source code from the copyrightable aspect of a computer program would have a perceptible adverse impact on Congress' ability to promote the progress of computer science, should such congressional action be considered necessary.
[86] 17 U.S.C.A §1201(a)(2): "No person shall manufacture, import, offer to the public, provide, or otherwise traffic in any technology, product, service, device, component, or part thereof, that … (A) is primarily designed or produced for the purpose of circumventing a technological measure that effectively controls access to a work protected under this title; (B) has only limited commercially significant purpose or use other than to circumvent protection afforded by a technological measure that effectively controls access to a work protected under this title…."
[87] Section 1201(a) prohibits the circumvention of a measure that controls access "to a work protected under this title" and Section 1201(b) prohibits the manufacture of a device that can be used to circumvent a copy control that "effectively protects a right of the copyright owner under this title."

electronic commerce, and protecting intellectual property rights.[88] It is not apparent, however, whether Congress intended the growth of one priority to be at the expense of the other.[89] Even so, the aim of protecting copyright in Cyberspace should be grounded by the same purpose that animates the protection of copyright under any context; namely, the promotion of the progress of the public's access to the public domain of works. To the extent that the growth of the public domain is not achieved or is in doubt by one legislative proposal as opposed to another, Congress ought not to choose that proposal. Remarkably, courts have not been out of step with this fundamental principle when seeking to define the contours of permissible reverse engineering of software programs.[90] When applying the provisions of the DMCA, courts should continue to view their function broadly in interpreting copyright, as a well-balanced compromise proportioned toward the enhancement of the pubic domain.[91]

41. For courts that have considered the question,[92] it has been well-established that reverse engineering functions as an act of fair use when reverse engineering is the only way to gain access to the ideas embedded in the source code of a computer program.[93] As such, reverse engineering is legally permissible as a fair use[94] of a

[88] *But see* Pamela Samuelson, *Intellectual Property and the Digital Economy: Why the Anti-Circumvention Regulations Need to be Revised*, 14 BERKELEY TECH. L.J. 519, 521, 533-534 (1999) (arguing that the DMCA reflects Hollywood's preferences to the detriment of the public).

[89] Although the legislative history of the DMCA is long and its text convoluted, the legislation emanates from a treaty provision, known as the World Intellectual Property Organization Copyright Treaty, which, among other things, sought to harmonize anti-circumvention provisions under copyright laws among many nations, *see, e.g.*, 144 CONG. REC. H7099 (daily ed. Aug. 4, 1998) (statement of Rep. Berman); 144 CONG. REC. S4884 (daily ed. May 14, 1998) (statement of Sens. Hatch and Aschcroft).

[90] Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1527 (9th Cir. 1992).

[91] Courts ought to apply their common law powers and views of fair use to broadly scale back the anti-copyright anti-circumvention provisions of the DMCA, and to do so irrespective of the narrowly drawn reverse engineering privilege set forth under Section 1201(f). Notably, courts are not without guidance in this regard; both the first sale doctrine and the fair use defense began as judicial constructions that Congress later codified. *See* Quality King Distribs., Inc. v. L'Anza Research Int'l, Inc., 523 U.S. 135, 140 (1998) ("We first endorsed the first sale doctrine in a case involving a claim by a publisher that the resale of its books at discounted prices infringed its copyright on the books."); Castle Rock Entertainment, Inc. v. Carol Publ'g Group, Inc., 150 F.3d 132, 141 (2d Cir. 1998) ("Until the 1976 Copyright Act, the doctrine of fair use grew exclusively out of the common law.").

[92] *See, e.g.,* Sony Computer Entm't, Inc. v. Connectix Corp., 203 F.3d 596 (9th Cir. 2000); DSC Communications Corp. v. DGI Techs., Inc., 81 F.3d 597, 601 (5th Cir. 1996); Bateman v. Mnemonics, Inc., 79 F.3d 1532, 1539 n.18 (11th Cir. 1996); Lotus Dev. Corp. v. Borland Int'l, Inc. 49 F.3d 807, 817-18 (1st Cir. 1995) (Boudin, J., concurring); Atari Games Corp. v. Nintendo of America, Inc., 975 F.2d 832, 843-44 (Fed. Cir. 1992); Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1527-28 (9th Cir. 1992); Vault Corp. v. Quaid Software Ltd., 847 F.2d 255, 270 (5th Cir. 1988); DSC Communications Corp. v. Pulse Communications, Inc., 976 F. Supp. 359 (E.D. Va. 1997); Mitel, Inc. v. Iqtel, Inc., 896 F. Supp. 1050 (D. Colo. 1995), *aff'd on other grounds*, 124 F.3d 1366 (10th Cir. 1997); *cf.* DSC Communications Corp. v. Pulse Communications, Inc., 170 F.3d 1354 (Fed. Cir. 1999) (acknowledging the right to reverse engineer for some purposes, but holding it unjustified in that case).

[93] It is clear that an author cannot acquire "patent-like protection by putting an idea, process, or method of operation in an unintelligible format and asserting copyright infringement against those who try to understand that idea, process, or method of operation." Atari Games Corp. v. Nintendo of Am. Inc., 975 F.2d at 842.

copyright-protected work.[95] There is no bar to reverse engineering software by a programmer who has lawful possession of the software.[96] In this manner, reverse engineering well-serves copyright law by assuring the public that its access to ideas about computer software will not be arbitrarily cut off by copyright holders. More to the point, reverse engineering acts as a technological check that keeps the copyright protection of software in the business of protecting copyrightable aspects of computer software only. The powerful tools of copyright cannot be directed toward uncopyrightable expression.

42. Of course, a computer user can see the ideas embodied in the external interfaces of a program merely by using the program. To learn about and understand the non-displayed ideas and functions contained in a computer program, such as interface specifications and protocols, however, a computer programmer must study and analyze the source code. Generally, applications programs such as word processors will contain more expression than an operating system. Most vendors, however, do not want to disclose their source code because it contains the interface specifications on its face. Thus, the vendors only distribute their unreadable object code and not the human-readable source code. When the source code is unavailable, the programmer must transform the available object code into readable code via reverse engineering in order to examine the unprotected, underlying ideas.[97]

43. Although courts consistently examine the question of permissible reverse engineering as if its scope might be restricted depending upon the reason for a programmer's use of reverse engineering, courts have not set forth a standard for limiting the scope of reverse engineering when the programmer has lawful possession of the software under review. This may be for a sound reason. If the expression uncovered by decompilation includes only or primarily unprotectable ideas or functional elements not subject to copyright protection, then the copyright holder has no basis under copyright law to complain of the

[94] As noted more fully below, however, the only place where the term "fair use" appears in the DMCA is under Section 1201(c).
[95] At least one commentator has noted that part of the difficulty in comprehending the limits of fair use may lie in the broad, and to some extent contradictory, manner in which the courts defined the concept prior to its codification.
[96] *See generally* NIMMER ON COPYRIGHT §13.05[D][4].
[97] *See generally* Dennis S. Karjala, *Copyright Protection of Computer Documents, Reverse Engineering, and Professor Miller*, 19 U. DAYTON L. REV. 975, 1016-18 (1994); Maureen A. O'Rourke, *Drawing the Boundary Between Copyright and Contract: Copyright Preemption of Software License Terms*, 45 DUKE L.J. 479, 534 (1995); David A. Rice, *Sega and Beyond: A Beacon for Fair Use Analysis...At Least as Far as It Goes*, 19 U. DAYTON L. REV. 1131, 1168 (1994); Pamela Samuelson, *Fair Use for Computer Programs and Other Copyrightable Works in Digital Form: The Implications of Sony, Galoob and Sega*, 1 J. INTELL. PROP. L. 49 (1993); Tyler G. Newby, Note, *What's Fair Here Is Not Fair Everywhere: Does the American Fair Use Doctrine Violate International Copyright Law?*, 51 STAN. L. REV. 1633, 1657-58 (1999); Timothy Teter, Note, *Merger and the Machines: An Analysis of the Pro-Compatibility Trend in Computer Software Copyright Cases*, 45 STAN. L. REV. 1061 (1993).

decompilation.[98] Since this conclusion is a plausible outcome of the court's review, it would hamper the court's analysis and disserve the purported distinction between ideas and expressions to establish a restriction on reverse engineering that would limit the court's review at the outset of the inquiry.

44. There simply is no way to predict to what extent the source code of any particular program contains copyrightable expression before its examination. Hence, restricting the programmer's use of reverse engineering based upon disputed claims of purpose would undermine the application of reverse engineering and limit its reach to uncontroversial or undisputed circumstances. Indeed, courts applying the DMCA[99] to DeCSS have made that very error by causing the purpose of reverse engineering to super-abound beyond its relative importance in the calculus of permissible uses of reverse engineering.[100] The district court in *Corley*, for example, flatly rejected the defendants' claim that CSS was decompiled to make DVDs interoperate with software and operating systems other than Windows.[101]

45. Some courts have recently acknowledged the critical distinction between straightforward copyright infringement and decompilation,[102] which makes the creation of additional original works feasible. Consequently, it has become common for some courts to recognize that the creation of interface specifications requires considerations of interoperability and compatibility. Compatibility and interoperability are beneficial to end-users because they provide increasing options for useful functionality with other devices and lower the cost of engaging in these optional uses. Obviously, interoperability also provides a decrease in consumer frustration with competing standards that would otherwise result in

[98] Reverse engineering is the common practice of disassembling a product to discover how it works. Kewanee Oil Co. v. Bicron Corp., 416 U.S. 470, 476 (1974).
[99] Of course, problems with the DMCA did not simply arrive at the time of legal challenges to its provisions. Far earlier, thoughtful critics predicted that the DMCA's provisions would provoke significant disruption to settled copyright principles. *See, e.g.,* Samuelson, *Intellectual Property and the Digital Economy*, *supra* note 88 at 534-46; *see also* Julie E. Cohen, *Some Reflections on Copyright Management Systems and Laws Designed to Protect Them*, 12 BERKELEY TECH. L.J. 161, 172-75 (1997) (criticizing the anti-circumvention provisions in the DMCA's predecessor, the National Information Infrastructure Copyright Protection Act). In a related analysis that warranted the careful attention of Congress before it implemented the United States' copyright treaty obligations by enacting the DMCA, See Julie E. Cohen, *A Right to Read Anonymously: A Closer Look at "Copyright Management" in Cyberspace*, 28 CONN. L. REV. 981 (1996).
[100] For example, Section 1201 of the DMCA prohibits acts that are considered fair use under *Sega* and *Connectix*. As noted below, circumventing copy controls and reverse engineering is considered a fair use in these cases, if it remains the only viable means of gaining access in order to design a non-infringing work.
[101] As noted *infra*, the district court's rejection of this claim was baseless because the initial operating system that DeCSS was made to run on was a non-Windows operating system. The court may have misunderstood that the decryption program, DeCSS, was needed to gain access to the content on the DVD, but was not, itself, software that played DVDs. Even so, rather than excusing the court's blunder, this error reinforces the point that an examination of purpose is often outcome-determinative and disserves the basis for permitting reverse engineering.
[102] "Decompilation" is used in its common sense, meaning "disassembly," rather than in its technological sense, which refers to reverse compilation.

consumer confusion and inefficient use of technology.[103]

46. Copyright holders often oppose the use of reverse engineering because of an unfounded fear that reverse engineering will be used to undertake efforts at "piracy" or unlawful reproduction and distribution of software. Some copyright holders also doubt that courts can set acceptable limits to lawful or authorized reverse engineering or to restrict its scope to avoid abuse.[104] Even if these fears and concerns were well-founded, they would not outweigh the countervailing interests involved in ensuring that access to the underlying unprotected ideas embedded in software does not remain hidden during the lifetime of copyright protection – especially when reverse engineering is the only practical means of obtaining access to those ideas.

47. Using reverse engineering software to obtain access to the underlying ideas embedded within the software is a fair use of the software. Not surprisingly, copyright law has traditionally encouraged, rather than discouraged, the public to build upon the ideas and information contained in a work. Restricting the scope for reverse engineering of software retards the opportunity for growth of the public domain because of the increased costs to the public for access to unprotectable ideas embedded within software works. Even those who exalt economic efficiency as the touchstone for rational transfer and distribution of information should be concerned by recent legislative and judicial trends concerning software ideas.[105]

48. Failing to protect the broad privilege of reverse engineering software will surely leave the software industry – as well as consumer electronic products that interface with software – with less competition and inefficient product development. Indeed, the most pernicious effect of divorcing reverse engineering from its context of fair use and thereby narrowing its scope significantly might be an unimaginable increase in the ability of consumer electronic manufacturers to impose artificial conditions on the interoperability of software running in conjunction with the electronic hardware.[106] No doubt, the public suffers when the ideas embedded in the software that end-users use to obtain full use of a consumer electronic product is not under their own control. Neither end-users nor anyone

[103] *See* Sony Computer Entertainment, Inc. v. Connectix Corp., 203 F.3d 596 (9th Cir. 2000) (holding that it was lawful to reverse engineer a video game system as an intermediate step to creating a computer program that would allow games designed for that system to run on a PC); Bateman v. Mnemonics, Inc., 79 F.3d 1532, 1539-40 n.18 (11th Cir. 1996) (endorsing the use of reverse engineering to gain access to the unprotectable ideas in a program, as well as access to copyrighted expression that might be used fairly).
[104] Notably, failing to protect the privilege of reverse engineering could lead to "abuses" by a purported copyright holder who has "pirated" source code or claimed copyright for what a court may determine is not copyrightable.
[105] *See* Sony Computer Entertainment, Inc. v. Connectix Corp., 203 F.3d 596.
[106] One obvious example discussed here is the MPAA's restrictions on electronic products that run software that plays DVDs; another example, discussed more fully below, is Sony Corporation's attempt to prevent open source development of software programs that add functionality to the company's entertainment robots.

else would have legal access to the ideas or functional elements embedded within source code, which most commentators would likely agree would raise serious questions – even under copyright law.[107] In light of the federal courts' attempts to provide an interpretative gloss on the DMCA, these issues are no longer simply interesting "academic" questions.[108]

1. Statutory Exception to circumvention under Section 1201(f)

49. Section 1201(f) authorizes the circumvention of an access control by reverse engineering a copyright-protected work if: [a] the reverse engineering does not otherwise constitute copyright infringement, [b] the statutory definition of reverse engineering is met, which permits reverse engineering for the sole purpose of achieving interoperability, and [c] the defendant does not otherwise have a readily available alternative manner to access the information sought through reverse engineering.[109] Since the DMCA is silent on whether the reverse engineering exception applies to copy controls, a computer user or software developer may circumvent a copy control, if the control is identifiable as such, for the purpose of reverse engineering and need not rely upon the DMCA's narrowly drawn statutory definition of reverse engineering.[110] It is, however, difficult to distinguish an access control from a copy control. Even in *Corley*, the plaintiff's claims[111] alleged copy control violations and access control violations as if the two were interchangeable and treated similarly by the DMCA. Of course, they are not and the Act does not treat them so.[112] If one were attempting to obtain the benefit of the DMCA's lack of a prohibition or silence on circumventing copy controls – assuming he had the skill to create his own tool, since the market for these tools is precluded by Section 1201(b) – he would not be limited to the exception provided in Section 1201(f), which presumably only applies to Section

[107] *See* Pamela Samuelson et al., *A Manifesto*, *supra* note 22 at 2319; Dennis S. Karjala, *Copyright, Computer Software, and the New Protectionism*, 28 JURIMETRICS J. 33 (1987).
[108] Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1527 (9th Cir. 1992).
[109] 17 U.S.C. § 1201(f).
[110] Section 1201(f)(2) could be read to: allow development of circumvention tools to aid in accomplishing reverse engineering a work protected by access controls, provide implicit authority to develop circumvention tools to aid in accomplishing reverse engineering of a work protected by copy controls, and authorize the aforementioned two activities as long as the reverse engineering does not constitute copyright infringement under the Copyright Act. Even so, few could confidently invoke the protections of Section 1201(f)(2) until the DMCA is revised to provide a clearer definition of reverse engineering. In particular, the DMCA needs to harmonize its restrictive definition of reverse engineering with the broader view by courts of reverse engineering under the Copyright Act. More important, Section 1201(f)(2) appears to adopt a rather solipsistic-styled definition of reverse engineering by limiting lawful reverse engineering to non-infringing uses. A defendant could lose the protection of Section 1201(f)(2) if a court summarily determines that the reverse engineering was done for the purpose proclaimed (e.g., interoperability). Section 1201(f)(2) also conflates two distinct issues; namely, the infringement of the access control and that of the work that is the target of reverse engineering. This two-tier approach to precluding reverse engineering imposes a heavy burden on the potential use of reverse engineering.
[111] Universal City Studios, Inc. v. Reimerdes, 111 F.Supp.2d 294 (S.D.N.Y. 2000), Universal City Studios, Inc. v. Corley 273 F3d 429 (2nd Cir. 2001).
[112] 17 U.S.C. § 1201(f).

1201(a)(1)(A) issues (unless the defendant could identify the distinction between access and copy controls).[113]

50. One ambitious implementation of technological barriers of access controls for digital works is Microsoft's authentication scheme, which includes both access and copy controls implemented through encryption known as ".NET Passport." This system is intended to be ubiquitous in Cyberspace (or, more precisely, to control access to *any* web-enabled device) by following Internet users as they move about in Cyberspace. Each participating web site can authenticate the user's identity transparently without the need for user intervention or interaction. Although such a system may make it convenient for the user to shop online without keeping abreast of a number of different passwords, .NET Passport can create this advantage only by storing a great deal of personal information[114] and personal profiles about Internet users in a vast database of "passports."

51. Once the user signs in to one of the .NET Passport participating web sites during an Internet session, at some point in the process, a file will be downloaded to the user's computer or device. The file is a locked, licensed file, if a digital work is involved in a user's purchase or electronic transaction. Some of these access controls are also copy controls, which prevent the user from making more than the authorized number of copies of the work. More to the point, if one could distinguish an access control from a copy control, the question remains whether the DMCA's prohibition from circumventing access controls would chill users from the practice of reverse engineering. In other words, the failure of the DMCA to provide specific guidance on how an access control differs from a copy control ostensibly renders the reverse engineering exception in Section 1201(f) inaccessible to software developers and computer users in circumstances where the copyright holder does not make it evident what type of control has been established.[115]

[113] Pub. L. No. 105-304, § 101, 112 Stat. 2860, 2861 (1998); 17 U.S.C. § 1201(a)(1)(A). How one may distinguish between an access control and a copy control is far from clear in the text of the DMCA. Even as a practical matter, the two types of controls seem fungible, and their distinction highly dependent upon the view of the right-holder.

[114] According to Microsoft, some of the information stored in the .NET database includes how much is being purchased, how much is being spent, or whether the transaction was successful. The data are tied to the user's e-mail address, password, name, web client IP address, and other information supplied by the user while online at one of the participating web sites. What .NET Means, http://www.passport.net/consumer/consumerqa.asp (Last visited March 25, 2003). In apparent response to the privacy and data-collection concerns raised by a number of Internet users and consumer advocates, Microsoft announced plans to discontinue its .NET Passport "express" service in 2003. http://www.passport.net/Consumer/WalletLetter.asp?lc=1033 (Last visited March 25, 2003).

[115] 17 U.S.C. §1201(f). Indeed, there is no incentive for the right-holder to make it apparent what type of control is used to "protect" his work. Although some right-holders have found it useful to notify users that a technological barrier exist to protect the copyright interest in the work sold or licensed to the user, oftentimes the description used to provide the notice is carelessly drafted so as to refer to access and copy control interchangeably.

52. What is more, it is quite easy for a defendant to lose Section 1201(f)[116] protection if a court summarily determines that the reverse engineering was not done for the purpose set out in Section 1201, which was the result of the courts' analysis in both *Corley* and *Bunner*. The reverse engineer has to show that there is no infringing conduct in the act of reverse engineering (other than the excused reverse engineering, itself), and that the reverse engineering does not infringe the copyright holder's interest in the access or copy control. Since some copyright holders may claim that a decryption program created for the purpose of overriding an encryption program's digital key lock is itself an infringement of the encryption program (i.e. the decryption program was derived from the encryption program), Section 1201(f) would be unavailable to the defendant since the reverse engineering was not accomplished without otherwise infringing a copyright.[117]

53. The district court in *Corley* clearly erred in finding that the hackers improperly reverse engineered the CSS software. There is an acknowledged industry-wide process of disassembling object code in order to abstract ideas and functionality, particularly under circumstances where the goal of reverse engineering is to permit the lawful owner of the DVD to view the DVD on a player of his own choosing. Copyright law has never been interpreted to preclude this type of consumer choice and the DMCA should not be the vehicle for that type of preclusion now.

54. Historically, programmers of all types, including video game developers, have been permitted to write program code compatible with personal computers from publicly-available, published materials regarding the operating system of the computer upon which the computer program or video game software is designed to run. That practice should not be altered or disregarded simply because an access control acts as a barrier to content that may or may not be protectable by copyright.[118] Indeed, neither the straightforward interpretation of the text of the DMCA nor the weight of precedent on reverse engineering requires a per se rule that the existence of an access barrier ought to block the defendants in cases like *Corley* for all times and in all instances. Certainly, no per se rule is suitable in a case where the defendant reverse engineered the code to understand the ideas, function, and operation of programs.

55. Although there can be some confusion about the word "disassemble," which simply means to make the ones and zeros that constitute the resulting object code easier to understand by creating a human-readable version of the software code,

[116] *See* 17 U.S.C. 1201(f)(1) (allowing the exemption "notwithstanding the provisions of Subsection (a)(1)(A)").
[117] Notably, this issue arises only in cases similar to *Reimerdes*, wherein the access control itself is subject to copyright protection.
[118] In considering the extent and limit of copyright protection, in Sony Corp. v. Universal City Studios, 464 U.S. 417 (1984), the U.S. Supreme Court stated that copyright protection has never accorded an author complete control over all possible uses of his work. Indeed, despite the limitations of copyright, there is a "natural tendency of legal rights to express themselves in absolute terms to the exclusion of all else." *Id*. at 433 n.13.

the copyright law requires that all ideas in general circulation be dedicated to the common good. Hence, the DMCA's sanction of the use of technological barriers of access to copyright-protected works cannot be extended to cover ideas and subject matter not entitled to exclusive protection.[119] The Supreme Court in *Kewanee* recognized that reverse engineering is a "fair and honest" means of discovery '"by starting with the known product and working backward to divine the process which aided in its development and manufacture."[120]

56. Under the DMCA, Section 1201(a)(1)(A) prohibits circumventing a technological measure that effectively controls access to a protected work subject to seven statutory exceptions.[121] Section 1201 also regulates technologies with circumvention enabling capabilities by prohibiting the "manufacture, import, offer to the public, provide, or otherwise traffic in any technology, product, service, device, component, or part thereof" primarily designed or produced for the purpose of circumventing the technological barrier if the device has a limited commercially significant purpose or use other than circumvention, and is ostensibly marketed for purposes of circumventing controls on copyright-protected works by the manufacturer or someone else on behalf of the manufacturer.

57. Given the badly drafted language in the statute, one task of courts in interpreting the act will be to distinguish between circumvention aimed at getting unauthorized access to a work and circumvention aimed at making non-infringing uses of a lawfully obtained copy. Section 1201(a)(1) is aimed at the former, not the latter. If Section 1201(c)(1)'s preservation of fair use and other defenses to infringement are to be given their plain meaning, it would seem that broadly defined reverse engineering practices should be permissible.[122]

58. One response to this criticism might be to assert that copyright owners will generally not sue when these or other legitimate circumvention activities occur. However, in some of the examples given above, the technical protector might well have incentives to sue the circumventor. Given that there are serious criminal penalties for willfully violating Section 1201, the overbreadth of this provision

[119] *But see* Triad Sys. Corp. v. Southeastern Express Co., 64 F.3d 1330 (9th Cir. 1995), in which the Ninth Circuit determined that copying the plaintiff's software into RAM by an organization that serviced the plaintiff's computers was not a fair use. The court distinguished its decision in Sega on the basis that the defendant Southeastern created nothing of its own. *Id*. at 1336. "Southeastern is simply commandeering its customers' software and using it for the very purpose for which, and in precisely the manner in which, it was designed to be used." *Id*. at 1337.

[120] Kewanee Oil Co. v. Bicron Corp., 416 U.S. 470, 476 (1974). Furthermore, the district court ignored, as not compelling, the court's ruling in NEC Corp. v. Intel Corp., 10 U.S.P.Q. (BNA) 1177 (N.D. Cal. 1989).

[121] In addition, this provision was restricted by a two-year moratorium during which the Librarian of Congress is supposed to study the potential impact of the anti-circumvention ban on non-infringing uses of copyrighted works which may lead to further limitations on the act-of-circumvention rule.

[122] "Courts interpreting Section 1201 may either be forced to find liability in some situations in which it would be inappropriate to impose it or to stretch existing limitations. Congress may eventually need to revise this provision to recognize a broader range of exceptions." Samuelson, *Intellectual Property and the Digital Economy*, *supra* note 88 at 538.

and the narrowness of existing exceptions will put many legitimate circumventors at unnecessary risk. Of course, if such suits are brought, courts may find other ways to reach just results. Even so, Section 1201 needs added flexibility, adaptability, and fairness, particularly with fair use issues so that courts will not have to thrash to reach appropriate results.[123]

59. More importantly, while the interoperability privilege exempts necessary tools from both device provisions of Section 1201, the encryption and security research privileges exempt tools only from the access-device provision, not from the control-device provision. Yet, it would seem that encryption and security research would often require testing both access and control components of technical protection systems. In addition, Section 1201 contains no provision enabling the development or distribution of circumvention tools to enable fair use or other privileged uses in terrain which Section 1201(a)(1)(A) does not reach (e.g., making fair uses of lawfully acquired copies). If Congress intended to recognize a right to "hack" a technical protection system to make fair uses, this right could be undermined if it could not be exercised without developing a tool to bypass the technical protection system or otherwise getting access to such a tool.[124]

2. Fair uses of copyright-protected material[125]

60. Not all unauthorized copying of copyright-protected works is impermissible.[126] In assessing whether given conduct constitutes copyright infringement, courts have long recognized that certain acts of copying are defensible as fair use.[127] The Copyright Act codifies a fair use standard in Section 107.[128] The standard provides courts with a set of factors to aid in consistent application of a doctrine

[123] Under the DMCA, the Librarian of Congress (LoC) is empowered to promulgate regulations and with consultation of the Department of Commerce make periodic formal recommendations to Congress with regard to the impact of the DMCA upon fair use. Unfortunately, it is unlikely that the LoC's authority will lead to notable improvement in the DMCA's troublesome provisions. The LoC's regulations will not apply at all to Section 1201(a)(1)(B). Hence, only Congress can provide relief from the prohibition in Section 1201(a)(2) against manufacturing or distributing the technology necessary to defeat access controls. *See* Universal City Studios, Inc. v. Reimerdes, 111 F. Supp. 2d 294, 324 (S.D.N.Y. 2000) ("The fact that Congress elected to leave technologically unsophisticated persons who wish to make fair use of encrypted copyrighted works without the technical means of doing so is a matter for Congress….").
[124] Under some interpretations of Section 1201(b)(1), development or distribution of such a tool would be unlawful.
[125] To some extent, it is difficult to parse fair use from other limiting doctrine such as scenes-a-faire, the first sale doctrine, and the idea/expression dichotomy. Hence, a reference to fair use in this context should be interpreted broadly.
[126] The United States Supreme Court, in Baker v. Selden , 101 U.S. 99 (1879), found that plaintiff's copyright in an account book did not extend to cover the accounting system described in the book. Rather, since the system is open to public use, "the methods of an art are the common property of the whole world[.]" *Id.* at 100.
[127] *See, e.g.,* Campbell v. Acuff-Rose Music, Inc., 510 U.S. 569, 575 (1994) ("From the infancy of copyright protection, some opportunity for fair use of copyrighted materials has been thought necessary to fulfill copyright's very purpose, 'to promote the Progress of Science and the useful Arts….'") (quoting U.S. CONST. art. I, § 8, cl. 8).
[128] Benkler, *supra* note 22 at 422-29; Campbell v. Acuff-Rose Music, Inc., 510 U.S. 569 (1994).

of equity that is not easily susceptible to a rigid contour or restrictive definition.[129]

61. Fair use is determined by applying relative value to a number of factors,[130] including the nature of the copyrighted material, the effect of the use under review upon the market value of the original work, the volume of copying, and the particular use of the unauthorized copying. With these broadly stated factors in mind, courts have substantial latitude in determining how best to effectuate the purposes of copyright.[131] Courts err in this area when their fair use analysis is overridden by an overwhelming concern for assessing the harm to the copyright holder's monetary reward. They do this rather than being concerned about whether the use benefits the predominant purpose of copyright, which is to effect enrichment of the public domain in a way that warrants excusing the unauthorized use and supercedes the copyright grantee's interest in avoiding diminished profits.[132]

62. The doctrine of fair use supports the same competing needs of copyright law generally, but balances those interests considerably less favorably for the copyright holder.[133] This balance is deeply embedded in the history of copyright law, wherein the government has taken on the role of advancing the body of knowledge, works, information, and discovery by providing provisional incentives to those who render such goals possible. However, this action limits the incentive to a claim for services rendered to the public.[134]

IV. Copyright and Free Expression

63. In the borderless virtual dimension of Cyberspace, the shift from a mere idea to the communication of an idea occurs automatically almost as a transparent instinctive response. Yet, the conceptual distinction between ideas and the communication or expression of ideas is fundamental in copyright doctrine.[135] Since the proverbial moment the law of copyright first recognized that computer

[129] *See, e.g.,* Princeton Univ. Press v. Michigan Document Servs., Inc., 99 F.3d 1381 (6th Cir. 1996).
[130] *See* Dan L. Burk, *Muddy Rules for Cyberspace*, 21 CARDOZO L. REV. 121, 144 (1999) (arguing that fair use in copyright may facilitate bargaining).
[131] The fair use doctrine operates as a limitation on the rights of copyright holders. One source of this limitation is the First Amendment to the U.S. Constitution. *See* Keep Thomson Governor Committee v. Citizens for Gallen Committee, 457 F. Supp. 957, 960, 199 U.S.P.Q. (BNA) 788 (D.N.H. 1978).
[132] "The doctrine of fair use…permits courts to avoid rigid application of the copyright [law] when…it would stifle the [production of works that copyright] law is designed to foster." Rubin v. Boston Magazine Co., 645 F.2d 80, 83 (1st Cir. 1981); Meeropol v. Nizer, 560 F.2d 1061, 1068 (2d Cir. 1977). Admittedly, the fair use conception of "transformative work" eventually runs head-on into the copyright law conception of derivative work. Castle Rock Enter. v. Carol Pub. Group, Inc., 150 F.3d 132, 143 (2d Cir. 1998).
[133] Campbell v. Acuff-Rose Music, Inc., 510 U.S. 569 (1994).
[134] That it may be odd to view copyright holders as some of the best paid public servants reinforces how unsuitable copyright protection is for certain classes of works as well as draws attention to whether the copyright system in its current form suitably serves the purposes for which it stands. It cannot be denied that a significant body of works currently protected by copyright law is created for the benefit of the public domain with little or no regard for copyright.
[135] Harper & Row v. Nation Enterprises, 471 U.S. 539, 560 (1985).

programs could be subject to copyright protection, courts have struggled with setting the boundaries of what aspects of a computer program are copyrightable.[136] Clearly, both the Copyright Act and the First Amendment prohibit the application of copyright protection to ideas, but applying that constitutional and statutory doctrine to actual allegations of copyright infringement is neither simple nor precise.[137]

64. The United States Supreme Court has determined that First Amendment freedoms of speech include the collective interest in protecting an individual's right to freely express almost any idea known to man. Copyright law directly affects the free expression of ideas because the United States Constitution secures for "limited [t]imes" to copyright holders "the exclusive Right to their respective Writings and Discoveries."[138] The copyright statute gives copyright owners a variety of exclusive rights: the rights to make copies of their works; to create derivative works; to distribute the works; and to publicly perform or display them.[139] In other words, the law of copyright grants to authors the right to control, restrict, or thwart public access to their expressive product.[140]

65. Despite the compelling language of the Constitution's copyright provision, it is apparent that the founding fathers only intended to permit Congress to protect a copyright holder's right to her original expression.[141] In the clash of competing

[136] *See* Amy B. Cohen, *Copyright Law and the Myth of Objectivity: The Idea-Expression Dichotomy and the Inevitability of Artistic Value Judgments*, 66 IND. L.J. 175, 207-10 (1990).
[137] Copyright law protects original authorship, but sets up a dichotomy between the protected work and the idea within the work. Any original work of authorship that exists in tangible form is copyrightable. Copyright protection, however, extends only to the particular expression of the ideas contained within the work, not to the ideas themselves.
[138] U.S. CONST. art. I, § 8, cl. 8. The Constitution seemingly prefers the interests of consumers over those of producers; copyright owners have enjoyed a decided advantage in the political contest over copyright's proper scope.
[139] These rights are necessarily limited in scope, duration, and subject matter.
[140] Principled and conceptually based distinctions in the law of copyright are not without their apparent contradictions and compromises. Any complete commentary on contemporary trends in the law of copyright must recognize that some doctrinal difficulty is due to political compromise. It is unremarkable to acknowledge that as a result of the growing demand for digital content, copyright owners are in a race to try and get Congress to pass laws that benefit one group over another. *See, e.g.,* David Landis, *Catching Some Entertainment, in Bits and Pieces*, USA TODAY, Aug. 25, 1994, at 8D (quoting Jay Berman, President of the Recording Industry Association of America). Often, when copyright infringement is alleged, courts must balance the constitutionally competing aims of promoting human creativity and original expression through the strict enforcement of the copyright law with ensuring that broad copyright protections do not unfairly or unnecessarily prevent the development of our knowledge base – particularly, the nation's development of practical uses of…information. Rod Dixon, *Profits in Cyberspace: Should Newspaper and Magazine Publishers Pay Freelance Writers for Digital Content?,* 4 MICH. TELECOMM. & TECH. L. REV. 127, 140 (1998).
[141] Although there is historical support that the early eighteenth-century European system of copyright was a tool of censorship, which primarily was used to restrict the flow of knowledge and information in a manner that benefited those viewed favorably by the State, the founding fathers took a different of copyright by providing for Congress' power to authorize copyright grants in Article I of the U.S. Constitution; notably, rather than adopt the view of their European predecessors, the language of the

constitutional provisions and almost strictly as a conceptual matter, the First Amendment trumps Article I, Section 8, Clause 8 in its significant limitation upon the scope and function of the law of copyright.[142]

A. Copyright: The Constitutional Copyright Requirement of Originality

1. Originality and the DMCA

66. Several doctrines, including the originality requirement, the idea-expression dichotomy, and the fair use doctrine, limit the scope and availability of copyright protection. It is often repeated that the sine qua non of copyright is originality.[143] A copyright-protected work must be original to the author. "Originality" is a term of art in copyright; it means that the work was *independently* created by the *author* (in other words, that the work was created by the author and that the work is the author's own work, not a work that emerges from the efforts of the author's and others), and that the work possess at least some minimal degree[144] of creativity.[145] Although "originality" does not signify novelty, it also does not signify "sweat of the brow."[146] Even so, a work may be original even though it closely resembles other works so long as the similarity is fortuitous, not the result of copying.

67. The DMCA provides authors with copyright-like protections without imposing upon the presumptive right-holder any of the "burdens" that usually accompany the rights granted to the copyright holder, including the constitutionally

Constitution supports the judgment that the founding fathers must have viewed copyright as an important constituent element of democracy and civilized culture.

[142] The paradox is that the public can only benefit if it has access to a work. Access is restricted, at least for a limited time, by granting the author property rights in her work, for only by restricting access can the author charge users and earn a profit. *See generally* Jessica Litman, *The Public Domain*, 39 EMORY L.J. 965 (1990). The artifacts of Cyberspace are largely intellectual property, and the owners of the intellectual property have a right to control how their property is communicated. In this respect, despite the open and public nature of Cyberspace, it is the province of an inherent and basic tension flowing from the goal of access to information between those that communicate and those that own the artifacts of communication. Of course, reliance on copyright law is not the only manner an author may reliably restrict access to his work. She may, for instance, license use of her computer software. Or, she may rely upon technological barriers – often referred to as digital copyright management systems – to prevent unfettered access to her works. Notably, some authors use a variety of factors to restrict public access to a given work.

[143] *See* Feist Publications, Inc. v. Rural Tel. Serv. Co., 499 U.S. 340, 345 (1991).

[144] *See id.* at 358 (1991) ("Originality requires only that the author make the selection or arrangement independently … and that it display some minimal level of creativity."); L. Batlin & Son, Inc. v. Snyder, 536 F.2d 486, 490-91 (2d Cir. 1976) (en banc) ("[W]hile a copy of something in the public domain will not, if it be merely a copy, support a copyright, a distinguishable variation will." (citing Gerlaen-Barklow v. Morris & Bendien, Inc., 23 F.2d 159, 161 (2d Cir. 1927))).

[145] 1 M. NIMMER & D. NIMMER, COPYRIGHT §§ 2.01[A], [B] (1990).

[146] Granting copyright protection for work on the basis of effort or sweat of the brow rather than originality was repudiated by the Supreme Court in *Feist, supra*. Oddly enough, Congress seems to have reinvigorated this constitutionally suspect doctrine in enacting the DMCA, and providing authors with copyright-like protections without an originality requirement.

mandatory originality requirement. Originality is a constitutional requirement.[147] The source of Congress' power to enact copyright laws is Article I, Section 8, Clause 8 of the Constitution, which authorizes Congress to "secure for limited Times to Authors…the exclusive Right to their respective Writings…."[148] In this regard, the constitutional text renders "writings" and "authors" critical to establishing the constitutional scope of intellectual property. In the copyright context, the constitutional provision has been held to establish that an "author" of a copyright-protected work is someone to whom a work owes its origin.[149] As such, the Supreme Court has concluded that the touchstone of copyright protection is the constitutional originality requirement. Without proof of originality, the work becomes a part of the public domain and is available to every person.[150] To claim copyright authorship in the course of litigation, an author must prove the existence of independent intellectual conception.

68. In this respect, the DMCA's ostensible approval of locking up access to source code regardless of whether the source code meets the originality requirement of copyright violates the constitutional mandate that copyright protection (which includes a para-copyright statute like the DMCA) only extend to works or those aspects of works that meet the originality requirement.

69. In most circumstances, a great deal of the content of source code should *not* be considered copyrightable.[151] As a result of contemporary programming methods, a substantial portion of source code is neither independently created nor illustrative of the minimal creative spark required to meet the constitutional requirement of originality. Copyright law protects original authorship, but sets up a dichotomy between the protected work and the idea within the work.[152] Any original work of authorship that exists in tangible form is copyrightable.[153] Copyright protection, however, extends only to the particular expression of the ideas contained within the work, not to the ideas themselves.[154] Under the law of copyright, an idea is thought to ordinarily encompass many means of expression.

---

[147] *See* Sheldon v. Metro-Goldwyn Pictures Corp., 81 F. 2d 49, 54 (2d Cir. 1936).
[148] U.S. CONST. art. I, § 8, cl. 8.
[149] Miller v. Universal City Studios, Inc., 650 F.2d 1365, 1368 (5th Cir. 1981).
[150] *Id.* What might appear to be a harsh rule because a failure to prove originality means the putative author has created a work that may be freely used by others is entirely consistent with essence of copyright law. The primary objective of copyright law is not to achieve the goals of a labor statute by ensuring that authors are rewarded for their labor but, rather, to encourage others to build freely upon the ideas and information conveyed by a work. This principle is supported by the idea/expression dichotomy and applies to all works of authorship. *See, e.g.,* Feist Publications, Inc. v. Rural Telephone Service Co., 499 U.S. 340 (1991).
[151] *See, e.g.,* Mitel, Inc. v. Iqtel, Inc., 124 F.3d 1366, 1373-74 (10th Cir. 1997) (some technological works lack sufficient originality to warrant copyright protection).
[152] *See* Feist Publ'ns, Inc. v. Rural Tel. Serv. Co., 499 U.S. 340, 350-51 (1991); Bateman v. Mnemonics, Inc., 79 F.3d 1532, 1547 (11th Cir. 1996) ("External considerations such as compatibility may negate a finding of infringement.").
[153] *See* 17 U.S.C. § 102(a) (1994) ("Copyright protection subsists...in original works of authorship fixed in any tangible medium of expression....").
[154] *See* Harper & Row, Publishers., Inc. v. Nation Enters. 471 U.S. 539, 546 (1985) ("The rights conferred by copyright are designed to assure contributors to the store of knowledge a fair return for their labors.").

Consequently, the idea/expression dichotomy, along with other limiting doctrine, thwarts the unintended effect of copyright to allow an author to gain control over an idea simply by expressing it in one tangible form.[155]

70. In its sum and substance, the law of copyright both advances and encumbers the manner in which an author may express himself.[156] Copyright is a constitutional promise granted by the Federal government that must yield, in instances of conflict, to the First Amendment.[157] To that end, courts have engrafted onto the law of copyright a presumptively dispositive test often referred to as the idea/expression dichotomy. Even so, as explained *supra*, the idea/expression dichotomy[158] may have no doctrinal application in the context of computer software copyright infringement litigation.[159] The doctrine developed from case law unrelated to software or technological expression. Moreover, even with regard to traditional literary works, courts have consistently acknowledged the need to apply other limiting doctrines, such as the doctrine of merger and the doctrine of *scènes à faire* that serve First Amendment purposes precisely when the idea/expression dichotomy ceases to serve its objective.

[155] *Id.*

[156] Some commentators note that patent law is similarly a troublesome province for software protection. Software innovations tend to be incremental or poorly documented and competition by software developers may require both that they be able to engage in reverse engineering in order to analyze existing programs, and that they have the freedom to design new products without undue risk of liability. Patent law poses obvious obstacles to both of these objectives in the context of software.

[157] U.S. CONST. art I. § 8, cl. 8.

[158] Although the formulation of the idea/expression dichotomy is largely inadequate for the purpose for which it is directed, the principles upon which the dichotomy is based are not only consistent with recognizing a First Amendment limitation on the scope of copyright, but also are consistent with the goals of copyright itself. In this regard, it is noteworthy that a basic purpose underlying the idea/expression distinction – as it applies to software – is to allow copyright protection beyond the literal computer code, and provide the proper incentive for programmers by protecting their most valuable efforts, while not giving the copyright holder a stranglehold over the development of similar software programs that accomplish the same end. *See, e.g.,* Herbert Rosenthal Jewelry Corp. v. Kalpakian, 446 F.2d 738, 742 (9th Cir. 1971) (The idea of a jeweled bee pin was held to be inseparable from the expression of the idea and thus this “stranglehold” by copyright was held invalid). The merger doctrine may account for a similar result as well. Moreover, while the idea/expression dichotomy in its actual application is of dubious use as a proxy for First Amendment analysis of the scope of copyright protection for computer source code, its value in other contexts is less open to doubt.

[159] Although among courts there may be increasing support for the suggestion that a computer program may contain many “ideas” at many levels of abstraction (or specificity), the various tests used are too divergent conceptually to conclude that courts are beginning to appropriately restrict the scope of copyright protection afforded to computer programs. Overly broad copyright protection for computer programs may actually hamper advancement in the field of computer programming. In addition, excessive grants of copyright protection are not necessary to promote advancement in the art and science of computer programming. Allowing a programmer to obtain copyright protection for elements of the program available in the public domain and other elements necessitated by industry standards and hardware compatibility could limit both the ability and the incentive for software authors to create or market competitive products.

2. Whether Source Code is Speech[160]

71. For the few courts that have addressed this question, most have conflated the inquiry into a single inquest rather than acknowledge that two questions are before the court; namely, whether source code is speech and what level of First Amendment constitutional protection should source code be afforded if it is speech. A court might conflate these distinct questions because it is no better suited than anyone else to answer the former while it is the court's constitutional province to answer the latter.[161] In the normative sense, to answer the question whether source code is speech is to determine what *is* speech, which is no simple matter.[162] Indeed, only one court seems to have earnestly attempted to address the matter.[163] It might be due to a quirk of metaphysical expression that some courts have regarded source code as speech as if it were an apparent reality. For these courts, it is sufficient to substitute analysis for tautological conclusion. In doing so, these courts reason that since source code is literally expressed, it must have a communicative dimension. Hence, the courts' holdings are based upon unacknowledged assumptions that ultimately become conclusions.[164]

72. The Second Circuit in *Corley* upheld the district court's findings.[165] While the Second Circuit provided some thoughtful analysis on the question of what is speech, the court's answer to the "code as speech" question was less satisfying than the decision it was reviewing. In all, the court used a single paragraph to address the issue of whether source code is speech (although it used more length expressing an answer to what it believed was a suitably relevant question concerning the issue of whether computer programs are speech).[166] The court, citing the Sixth Circuit in *Junger v. Daley*, held that obscurity is not relevant to

[160] To be clear at the outset, this is not an attempt to argue that there is a single animating purpose that may form the basis for analyzing questions concerning free expression.
[161] *See generally* Robert F. Nagel, *How Useful Is Judicial Review in Free Speech Cases?*, 69 CORNELL L. REV. 302 (1984).
[162] It is true of many areas of the law that the route toward a unifying principle is aligned with dead-ends and circular trails; apparently, the jurisprudence of the First Amendment's conception of freedom of expression is no exception. Even so, abstract notions deserve rigorous coherent scrutiny. *See generally* Eugene Volokh, *Freedom of Speech and Appellate Review in Workplace Harassment Cases*, 90 NW. U. L. REV. 1009 (1996); F. Schauer, *Must Speech Be Special?,* 78 NW. U. L. REV. 1284, 1303-04 (1983); F. SCHAUER, FREE SPEECH: A PHILOSOPHICAL ENQUIRY 13-14 (Cambridge: Cambridge Univ. Press, 1982); Lawrence Alexander & Paul Horton, *The Impossibility of a Free Speech Principle,* 78 NW. U. L. REV. 1319 (1983); Daniel A. Farber & Philip P. Frickey, *Practical Reason and the First Amendment,* 34 UCLA L. REV. 1615 (1987).
[163] Notably, in *Corley*, the Second Circuit provided an interesting and useful guideline by positing an explanation that would support the conclusion that source code should be classified as artificial speech.
[164] *See* Junger v. Daley, 209 F.3d 481 (6th Cir. 2000); Bernstein v. United States Dep't of Justice, 176 F.3d 1132 (9th Cir. 1999); Karn v. United States Dep't of State, 925 F. Supp. 1, 9-10 (D.C. Cir. 1996).
[165] Universal City Studios, Inc. v. Reimerdes, 111 F.Supp.2d 294 (S.D.N.Y. 2000); Universal City Studios, Inc. v. Corley 273 F3d 429 (2nd Cir. 2001).
[166] Why the Court even addressed this rather inartfully phrased question is less than clear. It is one thing to claim source code is speech, and quite another to make the argument that a computer program is speech.

the constitutional inquiry.[167]

73. After telling us what was not relevant, the court, unfortunately, failed to tell us what *was* relevant. Instead, the court made an erroneous comparison of Sanskrit to source code.[168] The court did not refer to source code as "pure speech," which, in this case, would clarify nothing and would be quite like calling a plane a bird because it flies. For the reasons I present below, source code might be better classified as "artificial speech." By classifying the source code of technology and science as a unique type of "speech," courts might begin to more carefully assess how source code should fit within our First Amendment jurisprudence. Although I am urging that a new category of speech be used to meaningfully assess what it is about source code that might render it protectable speech, the use of the term artificial speech[169] is not meant to imply that source code is, in my view, outside the scope of the First Amendment. Indeed, it seems apparent that there are clear circumstances when source code should be subject to protection under the First Amendment; what is less apparent is whether courts have adopted a meaningful test to render their assessment.

74. What one can *say* necessarily affects how people view speaking or speech. Like natural language, speaking the vocabulary and syntax of computer language carries along with it more than the ends of computation – it facilitates communication about computation.[170] In this regard, to say that source code is speech is to say that there is a connection between the goal of computing or using

[167] Interestingly enough, like the binary code used to create text documents using Microsoft Word, HTML source code was intended to be hidden from the view of its user. By accident or oversight, HTML source code is viewable by Web users and its readability resulted in an early proliferation web site development and an unexpected interest in HTML. TIM BERNERS-LEE ET. AL., WEAVING THE WEB 42 (New York: Harper Collins ed., 1999).

[168] Universal City Studios, Inc. v. Reimerdes, 111 F.Supp.2d 294 (S.D.N.Y. 2000); Universal City Studios, Inc. v. Corley 273 F3d 429 (2nd Cir. 2001).

[169] The term "artificial speech" is used to describe source code, not directly as a result of its connectedness to machine language, but because of what appears to be, if the trend of the courts is an indicator, a natural tendency not to see source code as primarily speech to machines. The article's aim, however, is not to suggest that the arguments supporting the First Amendment protection of source code are wrong – though some undoubtedly are – but rather to give a brief illumination as to why the current analytic framework is unbounded and unanchored by classification or context that would allow for coherent repudiation of any intent to provide First Amendment treatment for intelligent agents or software implementations of artificial neural networks.

[170] Computer scientists might notably look askance at a lawyerly discussion over how best to classify source code since the lawyer's purpose is not in providing a full or complete analysis of source code. Consequently, it is noteworthy to acknowledge that the focus upon the communicative dimension and the functionality of source code is not intended to ignore the fact that computer scientists recognize other important qualities of source code as techniques in building software applications, including qualities affecting why a programmer may chose to write source code in one computer language over another when more than one language could do the same task. Indeed, the answer to such questions may reveal an entirely distinct dimension to source code that may shed interesting light upon the lawyer's inquiry. For example, many programmers write source code in programming languages that enable easy exploitation of programming techniques such as abstraction, inheritance, and polymorphism. How critical are these determinations? Do they support source code's communicative dimension or its functionality?

machines and software to master a task and natural language, which provides the means to carry out the goal.[171] Most courts addressing the question of whether source code is speech seem to quite correctly recognize the duality of the essence of source code: "it says what it does."[172] Yet, the recognition of the dual nature of source code seems to impede the development of a meaningful assessment of source code.[173] Source code exists to run machines more efficiently because of the vast benefits cross-platform use of software has over hardware.[174] The ability to use software to efficiently perform tasks once solely performed by expensive inefficient machines, along with the industry's recognition of the benefits of standardization, forms the primary basis for the extraordinary growth of the software industry.[175] Hence, the inherent connection between source code and machines cannot be discounted simply because source code serves an important secondary purpose.[176]

75. Of course, this conclusion reveals very little about what speech[177] really is since, in this manner, the task of characterizing speech has become an extension of whether natural language is itself tied closely to thought. This characterization offers no distinctions and shades the fact that source code is anchored to machine language and the task of instructing machines directly, which is a distinction that ought to be critically important rather than overlooked or consumed by the analytical framework used to address the question.[178] Although source code simulates speech, the speech is directed toward a machine that responds in the

[171] *See* Orin S. Kerr, *Are We Overprotecting Code? Thoughts on First-Generation Internet Law*, 57 WASH & LEE L. REV. 1287 (2000) (providing an interesting insight that the proponents of the 'source code is speech' claim mistakenly argue that speech should be regulated for what it is rather than what it does).
[172] LAWRENCE LESSIG, THE FUTURE OF IDEAS: THE FATE OF THE COMMONS IN A CONNECTED WORLD, 57 (New York: Random House ed., 2001) (noting that source code is essentially performative).
[173] As explained in note 182, *infra*, we can derive from one of the bedrock principles of Speech Act theory that any human language has the same potential dual essence viewed as significant in source code, but rarely acknowledged as notable when the act of "speaking" is the target of our First Amendment jurisprudence. An example of the duality of human language would include praying or reciting an oath, wherein the "speech" says what it does.
[174] *Id.* at 51.
[175] *See* Dennis Karjala & Keiji Sugiyama, *Fundamental Concepts in Japanese and American Copyright Law*, 36 AM. J. COMP. L. 613 (1988).
[176] As noted *infra*, machines can generate source code. Indeed, the day may come when machines primarily generate source code. This will not alter its utility, but it should impact our intuitive sense of whether the regulation of the use of source code in most contexts should be strictly reviewed. On the web, for example, an increasing number of web pages are dynamically generated by scripting programs that respond to user input, which contrasts with the static web pages authored by humans producing HTML source code.
[177] Underlying the process of speech is comprehension and the production of more speech.
[178] One issue that will eventually be of deep concern is the certain success of artificial neural networks, particularly in the area of speech recognition. If source code *is* speech because it is potentially or often mediated by human interaction – programmers communicate to each other through source code – the success of speech recognition and character recognition software will soon enable programmers and anyone else to instruct machines directly through a type of verbal source code, or list of commands closely approximating natural language. The problem is, of course, that verbalization, like the source code that precedes it, is not human "speech" when it is not mediated by human interaction.

way it is instructed.[179] There is no doubt that computers do not respond to source code with questions to the programmer about the implications of what the programmer's source code has said. The point here is more nuanced than the often-repeated refrain that source code has a dual purpose and retains the concomitant quality of sustaining communication of content and executing the functionality of software. At bottom, computers follow a set of operations or an instruction set performed whenever certain binary strings or properties of binary code cause certain events to occur.

76. Source code resembles natural language because it is written in computer language,[180] some of which closely approximates natural language.[181] Even so this resemblance can be carried forth too far in the direction of absurdity.[182] Unlike natural language, on the level of abstractions from high level computer language

[179] *See* Junger v. Daley, 8 F. Supp. 2d 708, 716 (N.D. Ohio 1998) ("'Speech' is not protected simply because we write it in a language … what determines whether the First Amendment protects something is whether it expresses ideas.").

[180] For example, the following is an excerpt from source code written in the programming language C:

```
for (; ;)

(

uch = gtchr();
if (!(n & 31))

(

for (i = 0; i64; i++)
l [ ctr[i] = k[i] + h[n - 64 + i]
Hash512 (wm, wl, level, 8);

)
```

[181] For example:

```
sub printluckynum {
 $n +=1; # Global variable $n
 print "Hello World, your lucky number is $n!\n";
}
```

[182] Some have argued that speech act theory might provide a basis for setting forth an analytic tool to assist with the constitutional classification of source code since source code has both expressive and functional elements. Unfortunately, it may be difficult to borrow from speech act theory without doing a significant disservice to First Amendment jurisprudence, which, unfortunately, seems to have benefited too little from the helpful analysis of speech and speaking developed within other disciplines. At any rate, speech act theory divides speech acts into a hierarchy of acts ranging from locutionary and perlocutionary speech acts through illocutionary speech acts depending upon the meaning inferred from an utterance. *See* JOHN R. SEARLE, EXPRESSION AND MEANING: STUDIES IN THE THEORY OF SPEECH ACTS (Cambridge University Press ed., 1979). Since, under Speech Act theory, any act of speaking where the speaker intends his spoken or written utterance to be heard or read by someone at a given point in time is a speech act, "speech" may have a too broad and a too narrow view of "speech" to fit within American constitutional jurisprudence (neither intent of the speaker, nor locutionary value are essential requirements of the First Amendment). Generally, our First Amendment jurisprudence treats these speech acts as the same, notwithstanding that some illocutionary speech acts (e.g., fighting words, defamatory statements, and shouting fire) have special significance under First Amendment jurisprudence. Hence, Speech Act theory is too nuanced and perhaps parochial to assist in the more generalized approaches relied upon by legal analysis of speech.

toward machine code, source code seemingly becomes fundamentally different from natural language by assuming a direct relationship with the object it represents rather than the meaning upon which is imposed for the purpose of conveying content to humans. In this regard, this state of moving from symbolic representation to object sufficiently supports viewing source code as artificial speech.[183]

77. Perhaps the most puzzling aspect of the Second Circuit's opinion in *Corley* is its content-neutral analysis. The court erred here by indicating that the district court's injunction "targets" only the nonspeech of the component, or its source code functionality of DeCSS. The Second Circuit's conclusion was not based on what the injunction does, but rather on what is its purpose.[184] Under the circumstances, the court twisted the test of content-neutrality into a disingenuous analysis of whether an injunction directed to the posting of DeCSS should be called content-neutral simply because source code in digital form has potential functionality. If the Second Circuit was correct, of which I remain unconvinced, what would restrict the government from banning the posting of any or all source code on websites because of potential functionality? As long as the reason for the prohibition was alleged to be based on a "nonspeech" component of source code, the Second Circuit would apparently approve.

78. If one was to reinterpret the Second Circuit's guideline on whether computer programs are speech,[185] a useful conception of how one might go about setting a floor in the debate concerning what is speech might be forthcoming. What is important is not to define speech in a verbal sense,[186] but to establish a construct that provides meaningful content to one's conception of speech.

79. In this regard, as a starting point, I am suggesting that one's understanding of what might be deemed speech leads to the conclusion that *speech is "speech,"* if it is capable and intended to be capable of conveying a certain message or comprehensible information – where "information" means thought, feeling, idea,

[183] For the purpose of drawing a distinction between the communicative content of source code and object code, the difference is unavailing. Even binary code such as a series of off and on switches where 1 represents on and 0 off, an instruction concerning the memory location, 1010, can convey as much content as its hexadecimal representation, 0Xa, or decimal representation 10, but in binary form its symbolism is reduced and it becomes the object for which it stands; machine code instruction to an electrical circuit.
[184] This is the purpose of the DMCA, which authorizes the injunction. Universal City Studios, Inc. v. Reimerdes, 111 F.Supp.2d 294 (S.D.N.Y. 2000), *aff'd*, Universal City Studios, Inc. v. Corley, 273 F.3d 429 (2nd Cir. 2001).
[185] Why the Second Circuit conceived that the matter before it involved determining whether computer programs are speech is unclear from its opinion. More troubling, as noted below, the analysis of that question seems to have little relationship to the question and relies on less than thoughtful analogies. Even so, the court frames general principles useful for answering a different question quite well.
[186] Indeed, people already can conceive of speech in the verbal/nonverbal and expressive activity/pure speech senses, and those tools have not aided them in the endeavor for a meaningful framework to address what it means to say source code, or, for that matter anything like source code, is or is not speech. In other words, to say that the musician, Prince, has the freedom of expression to identify himself in a glyphic manner, or by a symbol having no verbal content, does not tell us much about what "speech" is or is not.

or any other generic use of the term – to a human where an intercession of mind or thought is required to trigger a response or understanding by the receiver(s). A momentary intercession of human action or response, however, would not without more render nonspeech, "speech." Hence, noise from an alarm clock that causes waking – albeit a momentary human response – is not speech. Sweating from the raise of the sun is not speech. Shouting "ouch!" from a flame is not speech. And machine language – whatever we agree that is – is not speech.[187]

80. Of course, the question of what constitutes speech is not novel to the Supreme Court. The Court often answers the question by tautological reference to the observation that speech is "speech" if it is communicative or if it conveys a message. In this regard, the difficulty that source code presents is its relation to the machine. Unlike the typical expressive activity cases[188] that courts have resolved involving draft card or flag burning, armband wearing, or street protesting, source code does not fit neatly within that framework. Those cases did not resolve the question of whether a flag, a piece of clothing or sleeping on the street was itself speech but rather whether in the context of the activity engaged in, the activity was sufficiently expressive to come within the scope of the First Amendment.[189] Some distinguish this question by wisely pointing out that although the output of machines – such as a piano roll or a compact disc – might be speech protected under the First Amendment, that fact does not require the conclusion that the means producing the output is itself speech.

81. At first blush, it might seem odd to urge on the one hand that source code is a cauldron of ideas often hacked together by programmers in a manner that strips the programming of originality while urging, on the other hand, that courts are

[187] For those who view source code as "pure speech" without distinction from any form of speech strictly protected under current First Amendment jurisprudence, the arguments are sometimes bolstered by citation to cases which conflate into one the two distinct questions that are presented; namely, whether source code is "speech" and whether the speaker's use of source code is "speech." In other words, the question what is speech is analytically distinct from whether the speaker is speaking. The latter question is largely a fact-based inquiry that should not be short-circuited by assuming that one is addressing the former question, which is a far more complex question for a number of reasons not the least of which includes the recognition that source code may be produced with or without a human agent. *Cf.* Hurley v. Irish American Gay, Lesbian and Bisexual Group of Boston, 515 U.S. 557, 573 (1995) (although noting that the "fundamental rule of protection under the First Amendment [is] that a speaker has the autonomy to choose the content of his own message," the court did not need to address the issue of human agent); Riley v. National Federation of the Blind of N.C., Inc., 487 U.S. 781, 790-91 (1988) (again, by noting that the "First Amendment mandates that we presume that speakers, not the government, know best both what they want to say and how to say it" the court is addressing whether the speaker is speaking rather than what is speech).

[188] *See, e.g.,* Tinker v. Des Moines Indep. Cmty. Sch. Dist., 393 U.S. 503, 508 (1969) (holding that a black armband worn during the Vietnam era was "a silent, passive expression of opinion"). Admittedly, the Court must have been either reasoning analogically or using rhetorical language for its persuasive value when it said that the expressive activity of wearing an armband was "akin" to or like "pure speech," *Id.* at 505-06, rather than mean that the activity was *pure* speech in any normative sense; not viewing the Court's intent in this manner would likely drain meaning from the term, pure speech.

[189] Nor is the question *should* source code be speech, which is what was at issue in the Courts' early child pornography cases.

misguided when they conclude that "source code is speech."[190] Of course, the point, precisely stated, is that in a given context the use of source code may serve a communicative basis, but that determination neither requires that source code be viewed as literal expression for all times or in all manner under copyright law nor warrants subscription to the refrain that source code is fundamentally free expression under the First Amendment. In the context of copyright law, courts have long understood that some categories of literal expression should warrant thin copyright protection.[191] Similarly, regarding freedom of expression, courts have embraced the utility of viewing constitutionally protected speech as including a shorter range than the absolute universe of communication.

82. The source code inquiry is difficult because there have never been cases substantially similar to provide a pertinent analogy. Clearly, a person writing a computer program by using source code is intentionally capable of conveying a certain message to a human by using source code to convey that message directly.[192] What is apparent from First Amendment jurisprudence is that speech or expression denotes human reality and the intermediation of machine alters the constitutional context. Certainly, the time will come when source code will instruct computers to process speech that humans use naturally. Even before natural language processing by machines is perfected, however, computer users will be able to address computers as though they were addressing another person. Sony Corporation's effort to introduce robotics to a mass audience through

[190] The argument advanced is not that the government should be able to place greater restrictions on speech than judges who favor the idea that "source code is speech" would permit. Instead, the argument is that there should be intellectual coherence in First Amendment classifications of speech. Americans should be mindful that as the nation moves steadily into an information economy, it will ill-serve many of those who value the First Amendment for its protections of individual rights to assume that "information" will not be regulated if it is called "speech"; the path of that argument will surely lead to incoherent exceptions to protectable speech rebutted only by the power and might of those who control the information economy. Hence, the future of incoherence will see the augmentation of speech protections for commercial interests concomitant with a decline in the rights of those who challenge them. For an impressive example of the early groundwork in this direction, *See, e.g.,* Martin H. Redish, *The First Amendment in the Marketplace's Commercial Speech and the Values of Free Expression*, 39 GEO. WASH. L. REV. 429 (1971). "By providing the consuming public with information, commercial speech aids in the attainment of society's goal of intellectual self-fulfillment and, more importantly, helps the individual to rationally plan his life to achieve the maximum satisfaction possible within the reach of his resources. In so doing it serves an important function as a catalyst in the achievement of personal self-realization." *Id.* at 472. In this regard, compare how the presumed intellectual property rights of the motion picture industry outweigh the First Amendment free speech rights of individuals in the *Corley* case with how the First Amendment interests of the telecom industry outweigh individual privacy rights of consumers in US West v. FCC, 182 F.3d 1224 (10th Cir. 1999), *cert. denied*, 530 U.S. 1213 (2000).

[191] Benkler, *supra* note 22 at 356.

[192] The use of source code to convey a message through the mediation of a computer or a computer program is an entirely different First Amendment question. In that case, the nexus between source code and the computer program embodying the source code would vitiate the functionality distinction that is invoked in most debates over the regulation of source code. Notably, it is not entirely implausible that the content within an executable computer program could be protected by a legal theory. Hence, binary versions of computer software are protectable by copyright law and trade secret law. Even so, such facts are not the basis of the discussion here.

products like its AIBO robot entertainment line represents one example.[193] Software developed to make personal computers more user-friendly for the disabled computer users represent another.[194] In this regard, fundamental notions about what constitutes "speech" may be under severe strain, if First Amendment jurisprudence continues to blend questions of text with those of context when matters of dispute involve computers or computing devices.

83. The very process of reverse engineering object code into some form of source code dislodges any argument that human action is necessary to produce some forms of source code.[195] Hence, there may be no way to ensure that source code in any particular endeavor was produced by an individual, much less to show that the source code at issue was produced to convey information to a human rather than merely to instruct a machine. Consequently, it seems hardly noteworthy to conclude, as a general matter, that source code is speech.[196] Instead, the logically coherent argument might be to say that source code is *artificial speech*[197] that may be protectable under the First Amendment as an expressive activity.[198]

[193] *See* http://www.us.aibo.com/ (last visited Mar. 31, 2003).

[194] Although there has been remarkable advances in speech and voice recognition software, computers cannot process human language with the native language competence of a typical human.

[195] Unfortunately, the use of labels often shields more than it reveals; hence, in the context of this discussion, the distinction between object code and source code seems to miss the point. There is no valid argument that source code is inherently more like "speech" than object code, machine code or any other label. Rather, the point is that the code used by the programmer to develop software *and* to express something about the code's content for the purpose of communicating that expression warrants some degree of protection under the First Amendment regardless of the label that is affixed to it. Hence, this article's author is less enamored by Professor Lessig's conclusion that object code is the "lifeblood of the computer" because "humans write 'source code'; computers run 'object code.'" LAWRENCE LESSIG, THE FUTURE OF IDEAS: THE FATE OF THE COMMONS IN A CONNECTED WORLD 50 (New York: Random House 2001).

[196] *But see* R. Polk Wagner, Note, *The Medium is the Mistake: The Law of Software for the First Amendment*, 51 STAN. L. REV. 387, 407 (1999) ("[S]oftware, to the extent it can be considered a set of instructions, communicates only to machines"). Perhaps, it might be more precise to say that source code exists along a continuum that at some point ceases to serve any communicative purpose of humans. Where this point is and whether it should be called source code, object code, machine code, or some other oxymoron depends upon the context of use and the programming language involved. There is probably neither a general answer nor one that will stand the test of time.

[197] Of course, assuming that computers or machines do not begin autonomously speaking today or tomorrow, what is at stake is analytical imprecision more than the classification of speech. Undoubtedly, the regulation of source code should trigger intermediate scrutiny, or no First Amendment scrutiny if courts push aside the notion that what is at issue is whether source code is speech, and acknowledge that the more precise inquiry is whether the use of source code in a given context should be strictly scrutinized. *See* Cohen v. Cowles Media, 501 U.S. 663, 669 (1991) (stating that "generally applicable laws do not offend the First Amendment simply because their enforcement against the press has incidental effects on its ability to gather and report the news" and allowing breach of contract claim against newspaper for identifying confidential source).

[198] Of course, this is not to say that such a classification would disallow plaintiffs who seek an equitable remedy against a prior restraint from doing so. Rather, the point is that speech is intimately tied to human action, and the production of source code occasionally (perhaps, often) is not. In this light, source code cannot be pure speech, and the ultimate outcome of its First Amendment protection should, as is true of other expressive activity cases, depend upon the particular facts of the litigated case.

B. Recent case law

1. Universal City Studios, Inc., v. Corley

84. The first court to apply the DMCA to computer source code, *Universal City Studios, Inc., v. Corley*,[199] rejected the defendants' argument that under the circumstances of the case the practice of reverse engineering copyright-protected works like a software program was supported by doctrinal as well as statutory privilege, which allows defendants to avoid the liability that would otherwise apply under the DMCA.[200] According to the district court (whose opinion was upheld by the Second Circuit), if a software user successfully obtained access to source code by circumventing a technological "lock" or barrier to access without authorization from the copyright holder of the computer program, the circumvention would not be excused by the doctrine of reverse engineering under the DMCA or by relevant case law defining the same doctrine unless the defendant could persuade the court that her purpose for breaking the access control was to determine how two programs interoperate.[201]

85. In *Corley*, the plaintiffs, eight motion picture studios, brought suit under the DMCA, *inter alia*, seeking an injunction[202] against the magazine and website, known as 2600 and 2600.com, (and several other defendants) to prohibit the defendants from publishing, disseminating, trafficking in, and posting on or linking to a website with the source code to a decryption program that allowed computer users of the Linux operating system (OS) to play lawfully acquired movies digitized by the copyright holder onto digital video discs (DVDs).[203] The source code was alleged to provide a way of circumventing a technological access barrier created by the CSS. This was accomplished by use of DeCSS.[204] Ostensibly, at the time DeCSS was developed, DVDs were sold with CSS to allow users to play the content on DVD-ROM drives connected to computers running Windows.

86. *Corley* both narrowed the scope of the defendants' conduct that could come within the statutory exception permitting reverse engineering and limited the range of protection allowing public access to a work for the purpose of reverse engineering.[205] In this regard, the court took the unprecedented step of restricting

[199] 111 F. Supp. 2d 294 (S.D.N.Y. 2000).
[200] *Id.*
[201] *See id.* at 324 (S.D.N.Y. 2000), *aff'd*, Universal City Studios, Inc. v. Corley, 273 F.3d 429 (2nd Cir. 2001).
[202] *See* Universal City Studios, Inc. v. Reimerdes, 82 F. Supp. 2d 211, 219 (S.D.N.Y. 2000) ("Defendants, however, are not here sued for copyright infringement. They are sued for offering to the public and providing technology primarily designed to circumvent technological measures that control access to copyrighted works and otherwise violating Section 1201(a)(2) of the Act. If Congress had meant the fair use defense to apply to such actions, it would have said so.").
[203] 111 F. Supp. 2d at 315-19.
[204] *Id.*
[205] 111 F. Supp. 2d at 315.

the reverse engineering exception to circumstances involving the dissemination of information, solely for the purpose of achieving interoperability among disparate computer programs – which the court summarily, but mistakenly, determined was unlikely to be among the defendants' actual purposes – given that, in the court's view, interoperability is not likely to have been the goal of the reverse engineer, if the immediate fruits of reverse engineering lead to the development of a decryption program designed to operate on the same OS, rather than immediately interoperate on a different OS.[206] In addition, the court, rather conspicuously, failed to consider whether obtaining access to unprotectable ideas had been the basis of defendants' efforts of reverse engineering.[207] Instead, the district court's analysis followed a lock-step argument presented by the plaintiffs concerning the copyright in the motion picture content on the DVD, which had nothing at all to do with reverse engineering the CSS.[208] *Corley* so far has had the perverse effect of invalidating reverse engineering, when occurring in the context of the public dissemination of source code that contains information concerning how reverse engineering might be undertaken.[209]

2. DVDCCA v. Bunner

87. Sony Corporation's[210] robotics division temporarily shut down the website at www.aibopet.com when it invoked the DMCA as the basis for claiming that the website owner could not freely distribute software that AIBO robot dog owners could use to train the robot to do tricks.[211] AIBO is the result of Sony's view of the future direction of computer programming and software development. That software will largely serve the purpose of creating artificial intelligence agents and robots that function with artificial neural networks[212] as brains.[213]

[206] *Id.*
[207] *Id.*
[208] *Id.* at 303-24.
[209] Eric Corley, *Analysis of the Decision Against 2600.com, The Hacker Q., at* http://www.2600.com/news/view/article/301 (Aug. 21, 2000) (providing a first-hand and pointed account of the impact of the district court's decision against 2600.com).
[210] Sony Electronics Corporation's Los Angeles-based Entertainment Robot America division was established and headquartered in the United Sates to maintain sales, engineering, and marketing operations for AIBO's sold in the United States, and that division's lawyers contacted the owner of the www.aibopet.com web site.
[211] Lisa Guernsey, *ROBOTS; A Programmable Dog That Will Bite Back*, NEW YORK TIMES, Nov. 8, 2001 at 3.
[212] There is no universally accepted definition of an artificial neural network or what might be called in the popular media, artificial intelligence. In this case, intelligence means an ability to learn. In the case of AIBO, it means the capability to respond to input. (Sony tries to enhance the appearance of learning by selling software that changes the behavior of the AIBO over time. Although the appearance of growth is not an example of learning, the AIBO can do a great more than these preprogrammed responses.) The goal of much of the research on artificial intelligence is to further our understanding of the human brain and, perhaps, develop practical applications run by networked processors that, with sufficient input, can manipulate computations similar to those the human brain routinely performs. One idea is that computers running the appropriate software can be trained in much the same manner as children learn to recognize dogs and cats – from sensory input from examples of dogs and cats – and also exhibit some capability for generalization beyond the training data, hence, learning. Grammar-checking software, voice recognition

88. In an attempt to introduce robotics into homes, Sony created entertainment robots with limited functionality, but vastly more advanced as an artificial intelligent system than consumers had ever experienced.[214] AIBO (pronounced eye-bo) is Sony's first mass-market produced entertainment robot and its name, AIBO symbolizes that the system is a robotic artificial intelligence system programmable by software.[215]

89. Sony claimed that the free software interoperated with Sony's proprietary software and that the interoperability was achieved by breaking the encryption protecting its software, which the DMCA prohibited.[216] Breaking the access/copy control protecting Sony's AIBO software could make it feasible to install additional software that could interoperate with the robot's firmware to create different "behaviors" in the robot.[217] For example, one free program distributed on the aibopet web site was "DiscoAibo," which enabled AIBO owners to train AIBO to dance to music.[218]

---

programs and handwriting recognition software might be considered the most closely associated examples of artificial intelligence software available on the mass-market software. AIBO might be considered a notable example of the use of artificial neural networks for similar reasons including, its voice recognition system and the capability of its sensors to decrease or increase random responses based on input. *See* ALBERT NIGRIN, NEURAL NETWORKS FOR PATTERN RECOGNITION, (Cambridge, MA: MIT Press 1993).

[213] The robot runs on firmware that has minimal functionality. Since the robot looks like a pet – although Sony no longer considers AIBO a dog – AIBO owners are encouraged to train the robot to do certain tasks or to play tricks. Many of these tasks require use of additional software, called AIBO-ware by Sony, sold on proprietary media made by Sony called memory sticks. A memory stick is like a diskette, except its size and shape resembles a stick of gum, and can be used to store software programs or data files of various type. The memory sticks used with AIBO contains an encryption program that controls access to the contents on the memory stick, but in a manner that seems to be directly connected to the interoperability of the software and the AIBO robot. Consequently, an end-user can access and copy contents to and from the memory stick and so can the robot, itself. But, if an end-user alters the software on the memory stick in certain instances, AIBO can overwrite or ignore the change or malfunction. In this manner, Sony seems to have had two minds on the use of access/copy controls, but there are reasons why the company did not want to completely block user access, including concerns for technical support and debugging an advanced consumer electronic product that happens to be, essentially, a mobile computer. *See generally* Sidney Luk, *Sony finds itself in dog house over Aibo site ban*, SOUTH CHINA MORNING POST, November 1, 2001 (Business Post) at 10.

[214] *Id.*

[215] *Id.*

[216] Dave Wilson & Alex Pham, *Sony Dogs Aibo Enthusiast's Site; Courts: The company uses a controversial law to stop owners from altering the robotic pet. Some consumers balk*, L.A. TIMES, Nov. 1, 2001, § 3, at 1.

[217] Some of the "behaviors" are the result of purely preprogrammed responses to software that may be installed in AIBO. For instance, AIBO will read e-mail messages or the content on a web page, if the proper software and hardware is installed. Or, an end-user may train AIBO to respond to its name through voice recognition or un-train some of its preset behaviors with the aid of installed software.

[218] Sony's response is mystifying in the face of its considerable experience with robotics and consumers. In particular, since consumer robotics is targeted toward hobbyists, it is not unusual for robotics end-users to desire to alter or reprogram the robot. Indeed, Sony must have had this in mind when it developed one of its software packages that enabled AIBO owners to interrupt the firmware's control over the robot and hand off that control to the end-user. More to the point, that the DMCA applies to this type of software use without recognition of the custom or practice of consumer robotics provides greater support for why the DMCA's anti-circumvention and trafficking provisions need revision. The adverse impact of how these

90. AIBO is an expensive robotic entertainment system.[219] Generally, the consumers likely to purchase AIBO are the target of most luxury goods advertisers. Consequently, it is not surprising that in this limited circumstance Sony withdrew its threat of litigation and abandoned its demands on the aibopet.com ISP to permanently remove the website when AIBO owners organized a raucous response in favor of the free distribution of software on the www.aibopet.com website.[220] Even so, other owners of intellectual property have proven to be substantially less responsive to consumer preferences or demands and have used the DMCA as if it were a sword rather than a shield. Indeed, some copyright owners have shown little or no reluctance to disguise the fact that the DMCA is being used to maintain a monopoly over software for particular hardware architecture, rather than to protect copyright.[221] A notable example is the battle for market share in the burgeoning electronic book (e-book) publishing industry.

91. Although e-books can be read on desktop computers, the growth and success of e-books is closely tied to the industry's ability to mimic – at least in the mind – the reading of printed books by allowing end-users to download e-books to mobile devices that simulate the look and feel of printed or paperback books. Since e-books on the desktop computer or on a mobile device are considered easily susceptible to copyright infringement, e-books are "secured" from end-users by use of access or copy controls that tether the e-book to the device that stores it.[222] Unfortunately, a large segment of the e-book publishing industry[223] has been overcome with so much fear from the sellers of e-book readers[224] that copyright infringement would destroy their business, which the access and copy controls that accompany e-book readers have begun to impose significant impediments to a reader's enjoyment of e-books, may be harmful to this nascent industry.[225]

---

provisions could affect an open and competitive software development market where software and hardware interoperate was not sufficiently thwarted by the narrowly crafted exception found in Section 1201(f).

[219] Luk, *supra* note 213.

[220] Wilson, *supra* note 216.

[221] This is a devastating claim since its impact upon the progress of software development could be deep and far-reaching. According to Linus Torvalds, "[p]ortability has long been a holy grail of the computer industry." Linus Torvalds, *The Linux Edge, in* OPEN SOURCES: VOICES FROM THE OPEN SOURCE REVOLUTION 101, 101 (Chris DiBona et al. eds., 1999). Ideally, a perfectly portable program is "write once, run anywhere" software. In other words, mating other programs to the first requires little, if any, customization. *Id.*

[222] The e-book and the device that stores it are mediated by software called a "reader," which provides the end-user with a sophisticated interface for viewing the e-book in a manner that simulates a printed book. The reader may also provide additional features such as the ability to use digital bookmarks or digital highlights. It has also become industry practice to install the access or copy control in the reader software. Hence, the reader software has become a critical element in the distribution of e-books.

[223] *See* Mark Stefik & Alex Silverman, *The Bit and the Pendulum: Balancing the Interests of Stakeholders in Digital Publishing*, THE COMPUTER LAWYER, Jan. 1999, at 1 (discussing the use of technological barriers on copying for effective control of copyrighted material).

[224] Adobe and Microsoft are developing the two most popular e-book readers today.

[225] A decision as innocuous as switching from a Palm PDA to a Pocket PC PDA to download e-books will lockout an e-book reader from access to all e-books purchased and downloaded to the old device using Microsoft Reader, which has not developed a transparent method to avoid these lockouts. Not surprisingly,

92. In the summer of 2001, a 26-year-old Russian programmer, Dmitri Sklyarov, was arrested in Los Angeles, California as a result of an allegation that he and his employer in Russia, ElcomSoft, a software development firm, had been selling a $99 computer program that could decrypt – or defeat technological controls protecting – the Adobe Acrobat E-reader, a software for electronic books, in violation of the DMCA.[226] Sklyarov's program, it was alleged, permitted e-book readers using the software made by Adobe to copy e-books without authorization and in violation of e-book software licenses.[227]

93. Similarly, on December 27, 1999, the DVDCCA (the Association) initiated an action under the Uniform Trade Secrets Act (UTSA or "Act") against Andrew Bunner and "numerous other named and unnamed individuals" who had allegedly republished or "linked" to DeCSS. The Association alleged that DeCSS embodies or is a derivative work based on its confidential proprietary information.[228] According to the Association, the proprietary information contained in DeCSS had been obtained by willful hacking and/or improperly reverse engineering the CSS software created by the Association's licensee Xing Technology Corporation (Xing).[229] Xing had allegedly licensed its software to users exclusively under a license agreement that prohibited reverse engineering. The Association argued that Bunner "knew or should have known" that by posting DeCSS or providing "links" to the program, he was "misusing proprietary confidential information gained through improper means." An injunction was sought to prevent any future disclosures of DeCSS.[230]

94. The Association conceded that "computer code is speech," but argued that even if Bunner had not initially known that DeCSS contained a trade secret that had been acquired by improper means, he clearly found out once the Association initiated its action, and therefore, he was required to refrain from disclosing the trade secret.[231] In response, Bunner argued that injunctive relief would violate his First Amendment rights.[232]

---

rather than the threat of copyright infringement, these unexpected lockouts as well as other copyright squabbles are having a pernicious effect on the growth of the e-book industry.

[226] *See, e.g.,* Konstantinos Kargiannis, *Are You An On-Screen Reader Too?*, PC MAGAZINE, Jan. 15, 2002, at 48.

[227] Sklyarov spent three weeks in prison before he was released on bail.

[228] DVD Copy Control Ass'n, Inc., v. Bunner, 93 Cal. App. 4th 648, 653 (Cal. Ct. App. 2001) (not citable); *see also* DVD Copy Control Association v. Bunner, (Santa Clara, CA 6th App. Dist. Nov. 1, 2001) (unreported).

[229] *Id.*

[230] The Association sought injunctive relief in the form of an order "restraining Defendants…from making any further use or otherwise disclosing or distributing … or 'linking' to other web sites which disclose, distribute or 'link' to any proprietary property or trade secrets relating to the CSS technology and specifically enjoining Defendants … from copying … distributing, publishing … or otherwise marketing the DeCSS computer program and all other products containing, using, and/or substantially derived from CSS proprietary property or trade secrets." *Id.*

[231] *Id.* at 654.

[232] *Id.* at 652. "Bunner asked the court to take judicial notice of a Norwegian law that permitted reverse engineering of computer software for the purpose of achieving "interoperability" and prohibited any

95. The trial court issued a preliminary injunction. The order enjoined defendants from "[p]osting or otherwise disclosing or distributing, on their web sites or elsewhere, the DeCSS program, the master keys or algorithms of the [CSS], or any other information derived from this proprietary information."[233] The court expressly refused to enjoin the defendants from linking to other web sites that contained protected information because the links were indispensable to Internet access and a web-site owner could not be held responsible for the content of other web sites. [234]

96. The appellate court began its analysis by stating its assumption that the trial court correctly concluded that the Association had established a "reasonable probability" that it could prove these allegations and had shown that the relative burden of harms favored issuance of injunctive relief. Turning to the question of whether DeCSS is "speech" within the scope of the First Amendment, the court concluded:

> "DeCSS is a writing composed of computer source code which describes an alternative method of decrypting CSS-encrypted DVDs. Regardless of who authored the program, DeCSS is a written expression of the author's ideas and information about decryption of DVDs without CSS. If the source code were "compiled" to create object code, we would agree that the resulting composition of zeroes and ones would not convey ideas."[235]

97. On the basis of that reasoning the court determined that merely because source code is capable of compilation, it does not destroy the expressive nature of that code. Hence, according to the court, the trial court's preliminary injunction barring Bunner from disclosing DeCSS was ostensibly a restriction on Bunner's freedom of expression.[236] To determine whether the trial court's injunction was an impermissible restriction on speech, the court noted that the trial court's prohibition of future disclosures of DeCSS was a prior restraint on Bunner's First Amendment right to publish the DeCSS program. Since prior restraints[237] on pure speech are highly disfavored and presumptively unconstitutional, the court concluded that a preliminary injunction cannot be used to restrict Bunner from

---

agreement to the contrary. According to Bunner, Johansen had reverse-engineered Xing's software to create DeCSS so that CSS-encrypted DVDs could be played on computers that run under a computer operating system known as Linux. Even if Johansen had agreed not to reverse-engineer Xing's software, the Norwegian law invalidated that term of the license agreement. Hence, Johansen's reverse engineering was not "improper means" within the meaning of the UTSA." The trial court declined to decide whether Norwegian law prohibited Johansen's alleged reverse engineering.

[233] *Id.* at 656.

[234] *Id.* at 657. There was no evidence that Bunner himself had ever contributed any of these writings indicating disrespect for the law.

[235] *Id.* at 661.

[236] *Id.* at 662. Strangely enough, the court felt it noteworthy that DeCSS was unquestionably questionable "pure speech" as it conspicuously characterized the "social value" of DeCSS as "questionable."

[237] The court expressed no opinion as to whether permanent injunctive relief may be obtained after a full trial on the merits, although in this case it would seem that the distinction is without a difference.

disclosing DeCSS.[238]

3. Sega v. Accolade

98. Sega Enterprises, Ltd. (Sega) manufactures and markets a video entertainment system (the Genesis console) and video game cartridges. Accolade, Inc. (Accolade) is an independent developer, manufacturer, and marketer of entertainment software for computers, including game cartridges that are compatible with the Genesis console and other computer systems. Sega filed suit in the fall of 1991 against Accolade alleging copyright infringement.[239]

99. To make its software compatible with Sega hardware, Accolade used a two-step process. First, it reverse engineered the software in Sega's video cartridges to identify the interface specifications. Next, Accolade wrote its own games for the Genesis console using only that portion of Sega's code that was necessary to interface with the Genesis console.[240] When Accolade discovered Sega's plan to introduce a newer version of the Genesis console on which Accolade games would not work, software engineers at Accolade did more research and found several more bytes of code that were necessary for Accolade's own code to work properly in a Genesis console.[241] Accolade's engineers testified that they copied only these several bytes from Sega's code into the final version of their program. This code, known as a TMSS initialization code, is essentially a software lock designed to prevent an unauthorized code from working on the Genesis console.[242] This initialization code prompts a visual display after the insertion of the game cartridge into the console that reads "PRODUCED BY OR UNDER LICENSE FROM SEGA ENTERPRISES LTD."[243]

100. Sega argued that Accolade illegally wrote input/output routines, based upon what was extracted from the enhanced assembly programs derived from Sega's copyrighted object code, that were used by Accolade in developing its Genesis compatible games. Accolade stated that its disassembly of Sega's object code was only for the purpose of achieving compatibility with the Genesis console and that it only replicated a minimum amount of functional code in its video game cartridges essential to the operation of cartridges on the modified Genesis console. Sega conceded that Accolade's use of disassembly for reverse engineering Sega's games was for the purpose of achieving compatibility with the Sega Genesis game console. But Accolade denied that it incorporated "input/output routines" acquired in the reverse engineering process into its own game cartridges or based its own programs on Sega's game cartridges.[244]

[238] *Id.* at 665.
[239] Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510 (9th Cir. 1992).
[240] *Id.*
[241] *Id.*
[242] *Id.*
[243] *Id.* at 1526-1527.
[244] *Id.* at 1526-27.

101. Not all writings are the subject of copyright. According to the court, Sega wrongly attempted to assert, by its copyright in its game programs, an exclusive right in the entire video game system and the method by which video game cartridges operate with the Genesis game console. This holding reinforces the fundamental principle of copyright that the grant of copyright exists to foster the creation of works that ultimately provide for the enrichment of the public domain.[245] Under *Sega*, the Copyright Act does not prohibit reverse engineering by disassembling copyrighted object code into functional assembly language for the purpose of achieving compatibility or interoperability with computers or game consoles which are available to the public. Indeed, *Sega* instructs that reverse engineering is fundamentally a fair use activity privileged because the public should have access to unprotectable aspects of computer software. This conclusion follows logically from the principle that copyright cannot protect ideas.[246]

102. In the source code context, this means that when specific instructions, even though previously copyrighted, are the only and essential means of accomplishing a given task, their later use by another will not constitute copyright infringement. Therefore, the question arises whether such conduct, nonetheless, could violate the DMCA because of that statute's narrow scope of what may constitute permissible reverse engineering.

103. Since it is clear that source code should receive only thin copyright protection, it also follows that the doctrine of reverse engineering should broadly protect the practice of reverse engineering. Two of these uses would include broad protection for purposes of ensuring interoperability and to create new works when those works are transforming or not substantially similar to the original software. In terms of the DMCA, Congress should revise Section 1202(f) by adopting a broad exception for reverse engineering that fits the direction of software development and is anchored more closely with how reverse engineering is actually used, rather than its potential abuses.

V. Open Source Removes Independence and Originality from Copyright

104. Recent changes in computer software development – largely the result of a

[245] In this instance one may view this argument with a jaundiced eye since the works at issue are computer gaming cartridges. It is noteworthy, however, to be mindful that the purpose of copyright is fulfilled regardless of the value one might place upon a given work, if the work comes within the scope of copyrightable subject matter. Hence, the question of whether Congress has made a prudent choice in its selection of copyrightable subject matter is an entirely distinct issue from the question of whether a court has applied a specific limiting doctrine in accordance with copyright principles.

[246] Alfred C. Yen, *A First Amendment Perspective on the Idea/Expression Dichotomy and Copyright in a Work's "Total Concept and Feel,"* 38 EMORY L.J. 393, 396-97 (1989); Lionel S. Sobel, *Copyright and the First Amendment: A Gathering Storm?*, 19 COPYRIGHT L. SYMP. 43, 63 (1971); Melville B. Nimmer, *Does Copyright Abridge the First Amendment Guarantees of Free Speech and Press?*, 17 UCLA L. REV. 1180, 1180-86 (1970).

paradigm shift in programming initiated by "Copyleftists"[247] and the open source code community in Cyberspace – and the recent approval of the argument that the nature of software development often involves the free expression of ideas should sufficiently set the groundwork to advance copyright jurisprudence by freeing courts from reliance on inconsistent and incoherent distinctions between copyrightable and uncopyrightable aspects of computer programs. Most importantly, viewing computer source code as an artifact of the public domain suitably reinforces an important goal of copyright. This goal illustrates that the government should grant copyrights in works to meaningfully motivate the creativity of authors in a manner that ultimately ensures public access to authors' products.[248] In this regard, copyright law should permit the unfettered access to public domain material by protecting source code authors from copyright infringement when the elements of a work at issue in an infringement action are the artifacts of the public domain.[249] Thus, courts adjudicating copyright infringement actions involving computer software should undertake a thorough reassessment of the limiting principles of copyright law, recalibrate the boundaries and the scope of copyright protection for software, and rarely regard source code as a category of expression created as a result of independent and original authorship.[250]

105. At bottom, a given slice of computer source code cannot be protectable as both copyrightable expression and as expression belonging to the marketplace of ideas.[251] Source code should be viewed as a resource for other software authors to draw upon when writing source code for their own programs. Subsisting within the purposes of copyright allows the public unfettered access to the uncopyrightable aspects of a work.[252] Copyright law supports the progress of science and the useful arts by withholding the grant of copyright, in the context of infringement actions, from aspects of works that constitute ideas, merge ideas with expressions, and constitute scenes-a-faire or singular modes of expression. In this regard, it would be consistent with the objectives of copyright that source

[247]As noted below, the movement includes a range of viewpoints and alternative labels. The label "open source" accurately captures the salient conceptual basis of the movement without risking unnecessary confusion presented by the use of other terms. "Copyleftist" is a short hand reference to the members of the faction of the open source code movement whose participants do not oppose proprietary use of open source code projects as long as the software applications are governed by copyleft. As noted more fully below, to copyleft a software application also involves distributing source code with a so-called public license that essentially dislodges the exclusive rights granted to a work by copyright. The terms of the public license prematurely pushes the source code into a public commons. In other words, copyright is turned on its head, hence, the term, copyleft.

[248] Rod Dixon, *When Efforts to Conceal May Actually Reveal: Whether First Amendment Protection Of Encryption Source Code and the Open Source Movement Support Re-Drawing The Constitutional Line Between the First Amendment and Copyright*, 1 COLUM. SCI. & TECH. L. REV. 1, (Sep. 28, 2000).

[249] *Id.*

[250] *Id.*

[251] *Id.*

[252] *Id.*

code be freely copied, distributed, or used in the creation of derivative works.[253] Stated simply, the law of copyright would provide sufficient incentive for software developers to create works from a vast public domain. The public domain would provide access to source code for future authors and, in turn, those authors would create works that could promote the progress of science, thereby further enriching the public domain.[254]

106. The phrase open source[255] represents a paradigm shift[256] in computer programming.[257] Generally, the words refer to the "source code," or programming

[253] There may be no better example of how open sourced works result in extraordinary proliferation of works than the rapid growth of content and innovation on the Web, where HTML source code was open and viewable by Internet users who learned to write in HTML source by reading the source code. *See* LAWRENCE LESSIG, THE FUTURE OF IDEAS: THE FATE OF THE COMMONS IN A CONNECTED WORLD 57 (Random House 2001) (noting that the fact that web browsers allowed HTML source code to be revealed to any Internet user made it extremely easy for users to build upon the teachings of others and to innovate); TIM BERNERS-LEE, WEAVING THE WEB, *supra,* note 40 (noting that although he did not design the web so that "HTML source code [could] be seen by users," the "human readability of HTML was an unexpected boon" to the explosive growth of the web). It is noteworthy that Berners-Lee had intended to promote open development of web page authorship, but probably had underestimated the excitement for web page development of the early entrants to the World Wide Web. *Id.* For these 'early adopters,' HTML coding was not unappealing or difficult. Today, web pages are often authored in the manner envisioned by Berners-Lee; users frequently use a software program that requires less knowledge of HTML source code. Unfortunately, these programs also produce source code that is increasingly difficult to understand, and likely to result in fewer users learning to build web pages.
[254] *Id.*
[255] Open source software has essentially three important features distinguishing it from other forms of software distribution like shareware, freeware, and shrink-wrap or general-off-the-shelf consumer software. Open source software is distributed with the source code open to the public for free use and carries a so-called public license precluding a potential software developer from capturing the source code by "closing" the source code. The general public license (known as a "GPL") allows users to sell, copy, and change "copylefted" software programs–which can also be copyright protected–but the author must pass along the same freedom to sell or copy her modifications and change them further. The author must also make the source code of her modifications freely available. In other words, open source software removes the usual restrictions on what a user may do with the program imposed by copyright by (1) requiring that the products developed as open source code software be distributed with a GPL, (2) requiring that derivative or any product developed by modifying the original software product be distributed with access to the source code, hence, open source, and (3) requiring that the derived program be distributed with a provision in the GPL offering some degree of copyleft protection. To date, the GPL is not known to have been subject to legal challenge.
[256] In some ways, this paradigm shift could be predicated on the last shift; namely, the adoption of object-oriented programming (OOP) that re-oriented programming away from procedures and toward objects. OOP saved programmers time by increasing a programmer's ability to create multiple uses of pre-written code. *See, e.g.,* JOSEPH WEBER, USING JAVA 1.1 74 (3d ed. 1997). Regardless of a software author's programming philosophy, the interplay among economic, cultural, and technological forces of Cyberspace is reshaping the course of how one does computer programming.
[257] Some examples of successfully launched open source software include well-regarded applications used in Cyberspace such as: sendmail, the program that routes over 80% of all emails on the Internet; Perl, the programming language that is used to write most of the common gateway interfaces (also called "cgi") or applications that enable most electronic commerce features on many web sites; Apache, the most popular web server software run on web servers connected to the Internet; BIND (or "Berkeley Internet Name Daemon"), the de facto software used to run the entire DNS (the "Domain Name System") server on the Internet; and perhaps the most popular, Mozilla, the open code software used in the well known and widely

of various pieces of software, wherein the end user is guaranteed[258] free[259] and open access to the software code. Many off-the-shelf software developers try to keep their source code secret, mistakenly assuming that copyright and secrecy were coordinate. But a growing number of organizations are bucking the trend, especially in Cyberspace.[260] Source code creation is now, largely, an open and shared process in Cyberspace.[261] Programmers engage in code sharing efforts on

---

used Netscape browser. *See generally* ERIC S. RAYMOND, THE CATHEDRAL & THE BAZAAR: MUSINGS ON LINUX AND OPEN SOURCE BY AN ACCIDENTAL REVOLUTIONARY 21-24 (O'Reilly 1999). Although all of the aforementioned programs are examples of open source projects, there is considerable debate as to which programs are in substantial compliance with the terms of the GNU GPL, which could be described as the constitution of open source. The greater a program's public license departs from the terms of the GNU GPL, the more likely that the program's license will restrict rather than broaden the freedoms associated with open source code distribution and copyleft.

[258] This "guarantee" is supported by the use of a license. The license is called a general public license or GPL, and it binds anyone who consents to its provisions by downloading or purchasing an open source program. GPLs vary widely, but most have their genesis in the GNU GPL drafted by the Free Software Foundation, which, in many respects, is notably at the forefront of the open source movement. Generally, a GPL achieves three goals: it designates ownership of copyright in the open source project; it grants everyone the right to modify, copy, and distribute the source code and the derivative program; and it sets distribution terms. The distribution terms require software developers, who produce programs using the original source code, to re-distribute all source code along with the GPL, and to distribute the source code in a form that is open to others or made available to others. In this regard, the phrase "open source code" is an appropriate reference to the GPL.

[259] "Free" is not a reference to the cost of the software. Instead, it refers to the free(dom) to change the source code and re-distribute it as a derivative program. It is entirely permissible to charge a price for use of software produced by the open source movement. *See* Berkman Center for Internet and Society, *The Power of Openness: Why Citizens, Education, Government and Business Should Care About the Coming Revolution in Open Source Code Software (A Critique and a Proposal for the H2O Project)*, 1999, *available at* http://h2oproject.law.harvard.edu/opencode/h2o/ (visited Feb. 8, 2000). Since free software may cost money, it is confusing to use the label "free software movement" to denote what is going on. Hence, "open source movement" is preferred.

[260] An influential paper subsequently published as a book by an open source software advocate – Eric Raymond – was first published in May 1997. The Cathedral and the Bazaar, *at* http://www.tuxedo.org/~esr/writings/cathedral-bazaar/ (visited Nov. 13, 1999). Raymond's paper was reportedly expressly cited by Netscape management as a motivation for their decision to release browser source code. The new programming paradigm acknowledges that "good programmers know what to write, and great programmers know what to rewrite" (and reuse). *Id.* In this regard, open source code programmers are more likely to efficiently and openly reuse code than traditional programmers not simply because they are always guaranteed access to the entire source code, but also because they need not waste resources keeping their code secret to either protect an intellectual property interest or avoid apparent notice that their software creation efforts violate the intellectual property interests of others.

[261] Although the copyleft movement is often associated with other open source code movements – such as the Open Source Initiative ("OSI") – these movements are distinguishable by their goals. *See, e.g.,* Thomas Scoville, *Whence the Source: Untangling the Open Source/Free Software Debate*, *available at* http://opensource.oreilly.com/news/scoville0399.html (visited Mar. 7, 1999) (stating that OSI "is galvanized by the rather more utilitarian revelation that users are much less clueless than previously imagined, and that enlightenment of their cooperation…results in much more useful, well-designed, robust creations"). The copyleft movement seeks to do away with intellectual property altogether, whereas OSI seeks to use collaborative style projects to loosen the software industry's grip on intellectual property. Ira V. Heffen, *Note*, *Copyleft: Licensing Collaborative Works in the Digital Age*, 49 STAN. L. REV. 1487, 1490 (1997) (noting that "under the proprietary software model, most software developers withhold their source code from users"); David Betz & Jon Edwards, *GNU's Not UNIX, Richard Stallman Discusses His Public-Domain Unix-Compatible Software System with BYTE Editors*, July 1986, *available at*

web sites, on Internet bulletin boards and newsgroups, and in e-mail exchanges. [262] For some, they will not be able to put the pieces together meaningfully. For others, the pieces will fit neatly together again and again.[263] At it is, jigsaw pieces, like subroutines in a given section of source code, build upon each other and are only valuable inside of their respective framework.[264]

---

http://prep.ai.mit.edu/pub/gnu/GNUinfo/INTERVIEW (visited Apr. 8, 1997); GNU was a name chosen by Stallman for the free operating system he created that is compatible to UNIX, a leading academic operating system. The name "was chosen following a hacker tradition, as a recursive acronym for 'GNU's Not Unix.'" Richard Stallman, *The GNU Project*, *available at* http://prep.ai.mit.edu/gnu/thegnuproject.html (visited Mar. 6, 1999); *What is Copyleft?, available at* http://prep.ai.mit.edu/copyleft/copyleft.html (visited Mar. 6, 1999) (describing goal and structure of copyleft); Richard Stallman, *Reevaluating Copyright: The Public Must Prevail*, 75 OR. L. REV. 291, 292-93 (1996) (criticizing traditional notions of copyright law because the copyright bargain necessary when "only a publisher could copy a book economically" is "no longer a good deal for the public") [hereinafter Stallman, Reevaluating Copyright]; Richard Stallman, *What is Free Software?*, *available at* http://prep.ai.mit.edu/philosophy/free-sw.html (visited Mar. 6, 1999) (describing three levels of freedom as: "the freedom to study how the program works and adapt it to your needs; the freedom to redistribute copies so you can share with your neighbor [; and] the freedom to improve the program, and release your improvements to the public, so that the whole community benefits"); P.J. Connolly, *Enterprise Toolbox: The Sharing Debate: When Source Code is Outlawed, Only Outlaws Will Have the Source Code*, INFOWORLD, Sept. 4, 2000, *also available at* 2000 WL 20918043.

[262] Dixon, *When Efforts to Conceal May Actually Reveal*, *supra* note 248.

[263] Interestingly enough, the specifications for the software protocol that controls the flow of information on the World Wide Web, HTTP (Hypertext Transfer Protocol) and the specifications that allow website authors to create websites, HTML (Hypertext Markup Language), were developed as open source technologies by the father of the World Wide Web, Tim Berners-Lee. Berners-Lee, who is often wrongly referred to as a physicist, released the specifications for web browsers and the web page programming language, HTML, so that others could adapt or improve the specifications for uses beyond the needs of his employer, CERN, an international research center in Geneva. E-mail correspondence between Rod Dixon and Tim Berners-Lee, (Apr. 9-21, 1999) (on file with author).

[264] Some adherents to the open source community discourage the use of "open" as the appropriate label to describe the goals and purposes of the movement. For some, the significance of using the term "open" rather than "free" or perhaps "free(dom)" highlights two diverging views of what open source is really about. There is no dispute that open source challenges the proprietary framework of software development and asserts that the current intellectual property regime is misapplied with regard to software; those conjoining goals notwithstanding, some open source supporters prefer to emphasize their objective of developing software that grants users and other developers freedom to use the works as they wish. These views are most commonly associated with the Free Software Foundation, which is managed by Richard Stallman. To some extent, the free software movement, unlike the open source community, is attempting to do more than shift the economic model of proprietary software development toward a more open framework. According to Stallman, the concept of "copyleft" grew out of the Free Software Movement. *See* E-mail correspondence between Rod Dixon and Richard Stallman, (Feb. 4-8, 2000) (on file with author). The distinction is important because "Free Software is a political stand; Open Source is a development methodology. That gives a clear idea of the difference." *Id.* Even so, the movement's factionalism represents differences of degree, rather than kind. Semantics aside, although Stallman's group rightly emphasizes that the new software paradigm is about a great deal more than deconstructing inaccurate and archaic views on software development, no one in the movement would suggest that software should be free, not sold. In this respect, it does seem more useful to label the movement as an open source community rather than a "free(dom)" software movement. In reaction to this shift from a cooperative, free-software environment to a proprietary market, Stallman resigned from his position at Massachusetts Institute of Technology so that they would have no legal claim to his software. Stallman then began the FSF and the copyleft movement. FSF is a nonprofit organization "dedicated to eliminating restrictions on copying, redistribution, understanding, and modification of computer programs" by

107. The new trend is to reveal the code to the world's programming community and let everyone try their hands at improving it.[265] Organizations that follow this trend are so numerous on the Internet that the open source community represents a global movement that is successfully challenging the contemporary proprietary model of software development with a model in which "openness" is considered a virtue. [266] The open source/free software model produces a superior product from input from potentially hundreds of programmers and reinforces market competition by precluding "lock in" to proprietary technology.[267] In other words, open source programming eschews the use of software development[268] to strategically control markets without regard to the production of superior software applications.[269]

108. The most famous open source project is probably Linux, a version of the Unix operating system.[270] Its proponents distributed a version of the code on the Internet several years ago and hundreds of programmers have added their own refinements. The result is claimed to be a much faster, less crash-prone operating system than Microsoft's Windows operating systems.

109. The link between open source and public domain should not be overstated. Although open source projects have the attributes of the public domain, they are not public domain works. One open source project can rarely begat another. The individual or organization managing the open source project retains the exclusive

promoting development and use of free software, specifically a free replacement for the UNIX operating system.

[265] The computer programming paradigm sustains such momentum today that it is not unlikely that the entire computing infrastructure may change course and become some version of an open source model. TECHNOLOGY REVIEW, Software Frontier, Jan./Feb. 2000, at 97.

[266] Indeed, source code written in Cyberspace-based programming languages like Perl and JavaScript is, generally, easily accessible by others, and can be read in any text editor. This open "feature" has enhanced code sharing even outside of the open source code movement. On the other hand, some programmers may have unknowingly sacrificed enforcement of their copyright interests as a result of using programming tools unsuitable for proprietary source code development. *See* EDWARD A. CAVAZOS & GAVINO MORIN, CYBERSPACE AND THE LAW 47-48 (1994); James Gleick, *I'll Take the Money, Thanks*, N.Y. TIMES MAGAZINE, Aug. 4, 1996, at 16; Thomas K. Landry, *Roundtable on Electronic Rights*, 20 COLUM.-VLA J.L. & ARTS 605, 658 (1996); Pamela Samuelson, *The Copyright Grab*, WIRED, Jan. 1996.

[267] This movement unquestionably deviates from the once prevailing view of computer source code as a trade secret. *See* Delta Filter Corp. v. Morin, 108 A.D.2d 991, 992, 485 N.Y.S.2d 143, 144 (App. Div. 1985); Support Sys. Assocs. v. Tavolacci, 135 A.D.2d 704, 706, 522 N.Y.S.2d 604, 606 (App. Div. 1987).

[268] "Ask a software company what it regards as its most valuable asset and the answer will probably be 'our source code.'" Stephen H. Wildstrom, *Freeware? What's Not To Like?* BUS. WK., Jan. 11, 1999, at 26.

[269] At first blush, some may find this conception of copyright to embrace a perverse notion of incentive. Why would an author, one might say, create a work without compensation for each copy? The short answer is that authors create such works frequently; notably, employees, under the work-for-hire doctrine, do not retain copyright interests in the works they create. In addition, the open source code community challenges prior assumptions as to what establishes sufficient incentive for authors to create works. More directly, software programs contain other aspects that may be suitable to copyright protection, including output, screen interface, program design, and graphical images. *See, e.g.,* Breyer, *supra* note 5.

[270] Linux is a free software operating system developed by thousands of volunteer programmers and hackers. Some commentators have argued that Linux is the only existing viable competitor to Microsoft's Windows-based operating systems. *See, e.g.,* Moglen, *supra* note 10.

copyrights to the works created or derived from the original code. The GNU General Public License (GPL), on which open source projects relying upon copyleft[271] provisions are based, grants non-exclusive rights to distribute or copy the original source code or to make derivative works based on the original source code. In this regard, works created from an open source do not provide the same benefit to authors that works created from public domain could provide. Of course, the point of the open source community is that copyright[272] is not necessary to promote the progress of science – at least, in so far as computer programs are concerned. Authors create works due to reputational benefits and other economic advantages.[273]

110. The most significant constraint on an open source code project may involve finding enough programmers available and interested in contributing their time jointly authoring freely available software projects.[274] In this respect, Cyberspace has provided the tools necessary to bring together enough people to harness the intellectual efforts required to create serious software programs sufficient to support the paradigm shift in programming. It is quite possible that the growth of the Internet will complete the programming paradigm shift since enough Cyberspace-based programmers will be available to make both large scale projects and small programming alliances viable and routine.[275] Open source code collaborative programming efforts may become standard. In this regard, the existence of the open source code community amply supports a conception of copyright that provides sufficient incentive for software developers to create works from a vast public domain.[276] The public domain would provide access to source code for authors and in turn, would encourage them to create works that could promote the progress of science and thereby, further enrich the public domain.[277]

[271] To copyleft a program, the author first must hold the lawful copyright to the source code. Then, distribution terms are attached to the source code, which usually grants anyone the right to use, modify, and redistribute the program's code or any program derived from it, but only if the distribution terms are unchanged. Thus, as noted, the code and the freedom to decide how one may use the code are viewed as legally inseparable.
[272] Although open source/free software is a frontal attack upon prevailing views of the inter-relationship between the grant of copyright and the promotion or advancement of software works, the GNU GPL is, of course, a copyright license. More to the point, many members would only cautiously urge the disentanglement of copyright and computer software for fear that patent law would replace the current preference of software authors for copyright, but would ill-serve software development as well.
[273] ROD DIXON, OPEN SOURCE SOFTWARE LAW (forthcoming 2003) (ch. 4, on file with author).
[274] Some have predicted the imminent death of copyleft, rather than copyright; these commentators refer to the internal debates among members of the open/free software community that have marginalized the community and its importance in the eyes of some developers. Although the discussions among free software and open source members are frequently vituperative, the range or nature of disagreement seems to have had little or no adverse effect on the community's continued expansion.
[275] DIXON, *supra* note 273.
[276] *See* Mark A. Haynes, *Commentary: Black Holes of Innovation in the Software Arts*, 14 BERKELEY TECH. L.J. 567, 568-69 (1999) (arguing that software innovation is impeded because developers use copyright to protect their software, forcing other developers to constantly reinvent the wheel).
[277] This is consistent with the Supreme Court's analysis in Feist Publications Inc., v. Rural Tel. Serv. Co., where the Court required "some minimal degree of creativity," or a "minimal creative spark" before finding

VI. Re-Calibrating the Scope of Copyright protection for Software

111. Although interoperability is one purpose for which previous law permits reverse engineering, other purposes might also find shelter under that line of cases – such as to extract from the public domain algorithms or other elements of style. Literal application of Section 1201 would not permit circumvention of technological measures to gain access to copyrighted works in order to perform reverse engineering of that latter type. As note above, this is a particularly vexing failure of the Act since access to public domain material itself embodies a constitutional limitation upon the scope of copyright.

112. Although a software key could itself be viewed as a copyrightable composition, the courts have not been friendly to an argument that copying such a software key constitutes infringement. For current purposes, therefore, let us assume, along with the Senate, that it too is uncopyrightable.

113. In this respect, the First Amendment should cut off the implacable drive to diminish, if only "temporarily," the stock of raw materials available to other authors.[278] The Copyright Act does not protect efficiency per se. It protects "original expression," a term of art that refers to the particular instantiation given to an idea or theme. That instantiation may be efficient, but efficiency is not synonymous with originality and the expression as a whole may be efficient and original for different reasons. An effective surgeon may be considered an artist within the medical profession, but the procedures the surgeon has developed and perfected are not protected by copyright. If the procedures are novel enough, they may qualify for patent protection. Otherwise, they belong in the public domain – even if the surgeon writes a book about them that details how they are to be performed. The fact that other surgeons may describe the procedures as "creative" or "elegant" is beside the point. In the copyright context, those words are terms of art and Section 102(b) of the Act makes clear that they do not apply to systems, procedures, or routines.

---

copyrightability in a compilation of a telephone book's white pages. 499 U.S. at 362, 363 (1991). Notably, excluding copyright protection in source code is not tantamount to eliminating software programs from the scope of copyright protection. Computer software, like a book or a screenplay, may contain both copyrightable and uncopyrightable aspects. A computer program's screen output may be copyrightable, although the source code would not be. *See, e.g.,* Gates Rubber Co., v. Bando Chem. Indus., Ltd., 9 F.3d 823 (10th Cir. 1993). Since courts have long held that software programs contain both literal and nonliteral elements subject to copyright protection, it is highly doubtful that removing source code from the copyrightable aspect of a computer program would have a perceptible adverse impact on Congress' ability to promote the progress of computer science, should such congressional action be considered necessary.

[278] Some have argued that the incentives the law of copyright provides are solely those of the author, not the public. *See generally* Mitel, Inc. v. Iqtel, Inc., 896 F. Supp. 1050 (D. Colo. 1995). Nonetheless, there are more than sufficient instances demonstrating that in the context of technology, and perhaps beyond, authors would create works without the protection of copyright. Indeed, scientific works, although subject to patent protection, may be outside the scope of copyright entirely. Funk Bros. Seed Co. v. Kalo Inoculant Co., 333 U.S. 127, 130 (1948) (copyright law recognizes no claims for scientific inventions). Perhaps the best proof lies within the Copyright Act, itself, wherein it excludes expression such as business forms and type fonts from copyrightability, yet, authors continue to create such works for compensation.

114. In other cases, a programmer may be efficient in the sense of getting a task done, without being at all creative, if the most efficient way to perform the various steps needed to complete the task is well known and standard within the industry. The fact that the programmer accomplishes the task by writing should not occasion a quantum leap in the level of protection afforded to the work. Here again, the treatment of computer programs as literary works obscures the issue. To the extent that industry-standard efficient routines are comparable to literary works at all, they are comparable to the alphabetical arrangement of entries in a telephone directory or dictionary, a convention so commonplace that it has been held uncopyrightable as a matter of law. If the programmer incorporates some idiosyncratic features into the efficient routine, copyright may protect those features, but not the routine itself. Even if the efficient routine is what gives the program its commercial value, copyright does not permit the programmer to complain when the routine is duplicated.

115. Notwithstanding that presently our circuit courts do not agree on the exact contours of the scope of copyright protection for software, some courts seem willing to concede that the better the view is that the very nature of computer programs – as utilitarian devices formed using technological expression that is dictated by the functions to be performed by the program, considerations of efficiency in the programs execution, and external factors requiring compatibility and standardization – reduces the software author's claim of protectable expression to a limited or narrow range of protectable elements in a program.[279]

116. The distinction between ideas and expression is supposed to provide a way of reconciling two competing interests – the interest in rewarding ingenuity and the interest in allowing the public to benefit from new works by other authors on the same subject. Since the function of copyright is to promote creativity so that the public may benefit from the labor of authors, the Federal government provides authors with an incentive to create by granting them the exclusive right to profit from and control specified uses of their works.

117. As noted, copyright also has the powerful capacity to diminish the potential for creativity. The exclusive rights granted by copyright may hinder the efforts of new authors who seek to build on the creativity of the past. It is in this regard that the idea/expression dichotomy helps copyright strike a balance between providing incentives to create and maintaining the store of raw materials needed for new creations. However, under the dichotomy, the boundary between unprotectable ideas and protectable expression is often difficult to discern.[280]

[279] *See, e.g., Apple Computer, Inc. v. Microsoft Corp.*, 35 F3d 1435, 1439 (9th Cir. 1994) ("When the range of protectable expression is narrow, the appropriate standard for illicit copying is virtual identity.")
[280] Indeed, the notion that a court (or anyone else, for that matter) can separate an idea from its expression seems to beg for judicial invention. It is a fundamental linguistic principle that people grasp ideas through expression; an idea cannot exist apart from expression. Although, in some metaphysical sense, expressions refer to ideas outside themselves, there is an intimate tie between expressions and ideas that the act of untying substantially disturbs. In other words, expressions are coefficients of ideas that are not easily subjected to the anachronistic tools of the idea/expression dichotomy. *See, e.g.,* GEORGES GUSDORF,

118. Presumably, when copying is literal, an idea can easily be isolated from its expression. Perhaps, the most important part of the public domain constitutes those works comprising copyrighted material and material that copyright does not protect. In other words, the public domain includes works that contain both copyrightable and uncopyrightable aspects. The concept that portions of works protected by copyright are owned by no one and are available for any member of the public to use is such a fundamental one that it receives attention only when something seems to have gone awry. Although the public domain is implicit in all commentary on intellectual property, it rarely takes center stage. But a vigorous public domain is a crucial buttress to the copyright system. Without the public domain, it might be impossible to tolerate copyright at all.

VII. Conclusion

119. Courts can no longer rely on a variety of flawed exceptions to copyright law to ensure the free flow of information in today's technologically-oriented world. Although more precise analysis by courts could aid in addressing this problem, it is likely that accommodating the interests of the public at this stage in the life of copyright protection for computer software calls for the potential decisiveness of a legislative solution.

120. By granting authors the exclusive right to reproduce and distribute their original expression, the Copyright Act allows some authors and copyright holders the right to use copyright as a means to suppress facts as well as expression. The limiting doctrines of copyright should not be distorted to permit government-issued monopolies on what is supposed to be original expression to implacably continue to define out of or remove from the public domain and marketplace of ideas common methods of expressing computer instructions in source code.

121. Despite its literal expression, computer source code rarely should be regarded as a category of expression created as a result of independent and hence, original authorship. Instead, source code is better viewed as constituting an artifact of the public domain.

122. The open source code community represents a global community that is successfully challenging the contemporary proprietary model of software development with a model in which "openness" is considered a virtue in software development.

123. Open source code programming both may produce superior products (as a result of input from potentially hundreds of programmers) as well as reinforce market competition by precluding "lock in" to proprietary technology. In this respect,

---

SPEAKING (LA PAROLE) (Studies in Phenomenology and Existential Philosophy) (Paul T. Brockelman trans., Northwestern Univ. Press 1965) (for an interesting view on how existential phenomenologists consider the conjunction of expressions and ideas as a constitutive element of human reality that cannot be meaningfully separated in the context of the human experience).

open source programming eschews the development of proprietary source code products in favor of software products that contain freely available source code. Software development is becoming largely an open and shared process in Cyberspace.

124. This new programming paradigm acknowledges the increasingly popular refrain: good programmers know what to write and great programmers know what to rewrite (or reuse). Open source code programmers are more likely to efficiently and openly reuse code than traditional programmers, not simply because they are always guaranteed access to the entire source code, but also because they need not waste resources keeping their code secret either to protect an intellectual property interest or worse, to avoid apparent notice that their software creation efforts violate the intellectual property interests of others. As such, source code is suitably recognized as an artifact of the public domain.

125. The digital age has brought along a notably critical challenge for copyright: how to continue the vitality of copyright protection in an environment where violations of copyright are not only rampant and disgorging, but also undermining of the very basis of the copyright regime. To date, the short answer to this perplexing question has been the support of an implacably expanding reach of copyright. Like a nine-headed Hydra, the reach of copyright is growing as is the presumed threat that digital technologies and Cyberspace seem to place upon the legal regime. Since digital works must be copied to be used, these technologies will inevitably require courts and Congress to confront the conflict between copyright and the First Amendment in a straightforward manner. To date, no clearer example of this confrontation has arisen than in the context of the encryption debates.

126. Viewing computer source code as an artifact of the public domain suitably reinforces an important goal of copyright; namely, that the government grants copyrights in works to meaningfully motivate the creative activity of authors in a manner that ultimately ensures public access to the products of an author's creativity. In this regard, copyright law should permit the unfettered access to public domain material by protecting source code authors from copyright infringement when the elements of a work at issue in an infringement action are the artifacts of the public domain. Thus, courts adjudicating copyright infringement actions involving computer software should undertake a thoroughgoing reassessment of the limiting principles of copyright law, recalibrate the boundaries and the scope of copyright protection for software, and rarely regard source code as a category of expression created as a result of independent and hence, original authorship.

127. The doctrine of reverse engineering should be grounded in the context of a doctrine of fair use that is vital in supporting public access to the ideas embedded in software.

128. Divorcing the reverse engineering doctrine from its context in fair use and free expression removes the public's only access to the ideas and functional elements embodied in software. That this privilege of access is being replaced by a technological barrier sanctioned by Congress with the force of civil and criminal liability is demonstrative of how close we have come to an ill-fated future for the public domain of information and information products. This does not simply promise a future where the user of digital information will pay an "owner" for its use in each and every instance, but includes a potential guarantee that some owners will be more equal than others. For some, copyright law rightfully will favor the dissemination of works like CSS over DeCSS; for others, copyright law will become an increasing patchwork of legislative favoritism dislodged from its constitutional purpose.

129. *Corley*, in particular, so far has had the effect of invalidating reverse engineering when occurring in the context of the public dissemination of source code that contains information concerning how reverse engineering might be undertaken. Oddly enough, this trend provides an apparent cover for content-based speech restrictions directed toward the activities of some libertarian-oriented Internet-based computer hackers and the open source community, whose activities are focused upon unleashing the locked up ideas of software, not hiding them. Although some courts have recently grasped the importance of supporting the recognition that software contains ideas within its source code, these courts' popular refrain that "source code is speech" is imprecise and too analytically debilitated to sufficiently sustain the proper balance between copyright and free expression. Source code is not speech anymore than an airplane is a bird. In the First Amendment context, judicial support is important for protecting source code when it is used for expressive purposes.

130. The DMCA's ostensible approval of locking up access to source code regardless of whether the source code meets the originality requirement may violate copyright's constitutional mandate under circumstances where the technological barrier protects an unoriginal work.