Vanderbilt Law Review

Volume 47
Issue 1 *Issue 1 - January 1994*

Article 4

1-1994

Reverse Engineering of Software for Interoperability and Analysis

S. Carran Daughtrey

NOTE

Reverse Engineering of Software for Interoperability and Analysis

I. INTRODUCTION

The rapid evolution of computer technology raises difficult questions about the scope of protection the law should afford computer programs. Computer programs are uniquely different from traditional literary works protected by the copyright laws[1] because they have machine-like properties, are primarily functional in nature, and frequently are distributed in a form that humans cannot read. Despite these differences, however, computer programs have received protection under the copyright paradigm along with literary and artistic works.[2] The United States historically has employed a highly

---

1. Copyright laws are grounded in the Constitution. See Patent and Copyright Clause, U.S. Const. Art. I, § 8, cl. 8 (empowering Congress "[t]o Promote the Progress of Science and useful Arts, by securing for limited Times te Authors and Inventors the exclusive Right to their respective Writings and Discoveries"). The laws are designed to balance the policies favoring protection with those favoring nonprotection. Brief Amicus Curiae of Eleven Copyright Law Professors 4-5, *Sega Enter. Ltd. v. Accolade, Inc.*, 977 F.2d 1510 (9th Cir. 1992) (No. 92-15655), reprinted in 33 Jurimetrics J. 147, 149-50 (1992) (stating that copyright law "represents a balance between policies favoring protection, such as basic fairness to authors and incontives to authorship, and policies favoring nonprotection, such as the free flow of knowledge and ideas and the social advantages resulting from one author's building on the work of another"). See also *Whelan Assoc., Inc. v. Jaslow Dental Lab., Inc.*, 797 F.2d 1222, 1235 (3d Cir. 1986).

The copyright laws ultimately strive to increase the public's wealth of culture and knowledge while simultaneously granting certain exclusive rights te authors for a limited time. United States Copyright Office, Report of the Register of Copyrights on the General Revision of the U.S. Copyright Law 5 (July 1961) ("Register Report"). See also Copyright Law Revision, H.R. Rep. No. 94-1476, 94th Cong., 2d Sess. 47-50 (1976). A secondary purpose of copyright law is to reward authors, which in turn stimulates the creation and dissemination of such works. Register Report at 5-6. See also *Mazer v. Stein*, 347 U.S. 201, 219 (1954) (stating that "[t]he economic philosophy behind the clause empowering Congress to grant patents and copyrights is the conviction that encouragement of individual effort by personal gain is the best way to advance public welfare through the talents of authors and inventers in 'Science and useful Arts'"); *Goldstein v. California*, 412 U.S. 546, 565 (1973); *United States v. Paramount Pictures, Inc.*, 334 U.S. 131, 158 (1948); Melville B. Nimmer, 1 *Nimmer on Copyright: A Treatise on the Law of Literature, Musical and Artistic Property, and the Protection of Ideas* § 1.03[A] at 1-31 to -33 (Matthew Bender, 2d ed. 1980) ("*Nimmer on Copyright*"). According to one commentater, the Copyright Act, 17 U.S.C. §§ 101 et seq. (1988 & Supp. 1992), seeks to promote authorship rather than publication of new works. Duncan M. Davidson, *Common Law, Uncommon Software*, 47 U. Pitt. L. Rev. 1037, 1057 (1986).

For many years the ceurts enforcing the copyright laws have struggled to establish a delicate equilibrium between (1) providing authors with incentives to create by affording protection and (2) limiting the extent of protection te avoid monopolistic effects. *Computer Assoc. Int'l, Inc. v. Altai, Inc.*, 982 F.2d 693, 696 (2d Cir. 1992). Thus, the copyright laws encourage competitors te copy the underlying, unprotected ideas of a work as long as they do not copy the protectable expression. *Feist Publications v. Rural Tel. Serv. Co.*, 111 S. Ct. 1282, 1290 (1991).

2. See 17 U.S.C. §§ 101, 102(a)(1).

protectionist approach to computer programs,[3] as evidenced by early software[4] infringement decisions in which courts slowly expanded protection by prohibiting copying of not only the literal or tangible aspects[5] of computer programs but also the nonliteral elements.[6] Recently, some courts have made an underlying shift in their interpretation of legal doctrine and policy from a broad standard of infringement that favors software copyright owners to a more narrow standard.[7]

Today's central controversial issue is whether the law should allow competitors to reverse engineer[8] a computer program to ascertain its underlying ideas, interface specifications, and protocols.[9] Many computer experts and legal scholars contend that the fair use doctrine[10] permits the reverse engineering of computer programs. Others disagree and instead believe that this type of copying always infringes the rights of the copyright owner.[11] This discrepancy highlights the basic tension between an author's right of control and the

---

3. J. H. Reichman, *Computer Programs as Applied Scientific Know-How: Implications of Copyright Protection for Commercialized University Research*, 42 Vand. L. Rev. 639, 699 n.312 (1989).

4. "Software" is simply another term for a computer program.

5. Literal elements of a computer program are the written words, including the source and object code. Source code is a human-readable code. See note 16 for a definition of object code. The nonliteral aspects include non-written elements, such as structure, sequence, organization, visual displays, and interfaces. That is, nonliteral aspects of computer programs are not reduced to code. See *Computer Assoc.*, 982 F.2d at 696; *Whelan*, 797 F.2d at 1248 (stating that nonliteral elements include "structure, sequence, and organization").

6. See, for example, *Johnson Controls, Inc. v. Phoenix Control Sys., Inc.*, 886 F.2d 1173, 1175 (9th Cir. 1989) (protecting nonliteral aspects of computer programs when those components are expression rather than ideas); *Whelan*, 797 F.2d at 1248 (providing protection that extended beyond the literal text of computer programs to the nonlitoral structure, sequence, and organization); *Broderbund Software, Inc. v. Unison World, Inc.*, 648 F. Supp. 1127, 1133 (N.D. Cal. 1986) (protecting structure, sequence, and organization).

7. This shift began with *Computer Assoc.*, 982 F.2d at 693. See also *Sega Enter. Ltd. v. Accolade, Inc.*, 977 F.2d 1510 (9th Cir. 1992); *Atari Games Corp. v. Nintendo of Am., Inc.*, 975 F.2d 832 (Fed. Cir. 1992).

8. "Reverse engineering" is a term of art that refers to the process of going backward from a finished product to the initial pieces to determine how a company made a product. The Court in *Kewanee Oil Co. v. Bicron Corp.*, 416 U.S. 470, 476 (1974), defined reverse engineering as a "fair and honest means [of] . . . starting with the known product and working backward to divine the process which aided in its development or manufacture." See notes 26-35 and accompanying text for a more detailed discussion of reverse engineering of computer programs.

9. See notes 27-28 for an explanation of interface specifications and protocals.

10. See notes 77-112 and accompanying text for a detailed discussion of the fair use doctrine.

11. See Anthony L. Clapes, Patrick Lynch, and Mark R. Stoinberg, *Silicon Epics and Binary Bards: Determining the Proper Seope of Copyright Protection for Computer Programs*, 34 U.C.L.A. L. Rev. 1493, 1546 (1987) (discussing the issue of whether copyright law protects against the copying of literal text only or extends to translation, paraphrasing, and nonliteral duplication).

public's right of access to the ideas and functions of a copyrighted work.[12]

The most recent developments in the area of reverse engineering of computer programs are *Sega Enterprises Ltd. v. Accolade, Inc.*[13] and *Atari Games Corp. v. Nintendo of America, Inc.*[14] In both cases, the courts radically departed from previous narrow definitions of infringement that afforded computer programs greater degrees of protection. The Federal Circuit in *Atari* rejected the plaintiff's claim that reverse engineering of a computer program constituted infringement per se.[15] In *Sega*, the Ninth Circuit held that, as a matter of law, disassembly of the object code[16] of a copyrighted computer program is fair use when disassembly is the only way to gain access te ideas and functions embodied in a computor program and when a legitimate reason exists for gaining access to those ideas and functions.[17]

This Note discusses whether it is appropriate to allow reverse engineering of computer programs for the purpose of analysis or to achieve interoperability[18] when the resulting product is created independently. Part II examines the characteristic features of computer programs that should influence the method of intellectual property protection. Part III briefly addresses the scope of copyright protection, reverse engineering, and the fair use doctrine. Next, this Note explains the case history of copyright law as applied to reverse engineering and intermediate copying of computer programs. It also addresses the fact that courts have had great difficulty distinguishing the nonliteral components from the literal components of a computer program. Part IV discusses the *Sega* and *Atari* holdings in light of the historical defenses to the necessary, intermediate copying required to reverse engineer computer programs. Part V examines the effects of these recent developments and the future of copyright law for reverse engineering cases with regard to the nonliteral components of

---

12. Copyright is a legal device that permits the author to disclose his work to others yet maintain the right to control the reproduction of the work. Register Report at 3 (citod in note 1). Copyright, an intangible, incorporeal property right of limited duration, gives the author an exclusive right to prevent others from reproducing the work. Stephen M. Stewart, *International Copyright and Neighboring Rights* § 1.06 at 4 (Butterworth, 1983). Copyright does not, however, prevent others from using their own expression of the same ideas and concepts. Register Report at 3.

13. 977 F.2d 1510 (9th Cir. 1992).

14. 975 F.2d 832 (Fed. Cir. 1992).

15. Id. at 844.

16. Object code is computer-readable code that originates as human-readable or source code and is translated to a format that the computer can use to perform specified tasks.

17. *Sega*, 977 F.2d at 1527-28.

18. See text accompanying notes 188-99 for a full explanation of interoperability.

computer programs. Finally, in Part V, this Note explores whether protection of computer programs should continue to be governed by copyright law or by a new sui generis law.

II. CHARACTERISTIC FEATURES OF SOFTWARE

Although computer programs currently are considered literary works and are protected under the copyright paradigm,[19] they have unique characteristics that make them very different from traditional literary works.[20] In particular, most programs distributed to the public are in a machine-readable format rather than a human-readable form. Additionally, the programs themselves have many machine-like attributes. The following subparts discuss the relevant features of computer programs and the method required to change the machine-readable format to a human-readable one.

A. General Attributes

The Copyright Act classifies computer programs as literary works.[21] A computor program is similar to a literary work because it consists of words, although these words actually are a list of written instructions. Unlike traditional literary works, however, these words or instructions accomplish a task and produce a result. To write a computor program, a programmer begins with a general design and eventually writes specific instructions in a higher-level computer language.[22] The set of instructions written in this human-readable higher-level language[23] is known as source code. Next, the programmer uses a compiler or interpreter to transform the source

---

19. See text accompanying notes 36-63 for a discussion of the scope of protection afforded computer programs.

20. For an excellent explanation of how authors write computer programs and how the programs work, see generally Randall Davis, *The Nature of Software and Its Consequences for Establishing and Evaluating Similarity*, 5 Software L. J. 299 (1992). See also Andrew Johnson-Laird, *Reverse Engineering of Software: Separating Legal Mythology from Actual Technology*, 5 Software L. J. 331, 336-38 (1992) (explaining software development and engineering); Pamela Samuelson, *CONTU Revisted: The Case Against Copyright Protection for Computer Programs in Machine-Readable Form*, 1984 Duke L. J. 663, 672-92 (discussing what computer programs are and how they work).

21. "Literary works' are works . . . expressed in words, numbers, or other verbal or numerical symbols or indicia, regardless of the nature of the material objects." 17 U.S.C. § 101. See also *Apple Computer, Inc. v. Franklin Computer Corp.*, 714 F.2d 1240, 1249 (3d Cir. 1983).

22. See *Whelan Assoc., Inc. v. Jaslow Dental Lab., Inc.*, 797 F.2d 1222, 1229 (3d Cir. 1986). See note 23 for examples of high-level languages.

23. Examples of high-level languages include Ada, Basic, C, Cobol, Fortran, and Pascal. See note 31 for a discussion of the difference between assembly language and high-level languages.

code into object code, a series of ones and zeroes that the computer can interpret and execute (also known as machine-readable code). Because humans cannot read the object code, this form of a computer program, unlike most literary works, does not disclose on its face the ideas underlying the work.

One can read the source code of a computer program from start to finish, just as one reads a book. Yet when the computer executes and performs the program, it almost never executes each line once in consecutive order. Instead, it performs quite differently, possibly skipping over some code and repeating other code multiple times.[24]

Additionally, software differs from other literary works because it is essentially functional and not creative. Creative, fanciful parts of a computer program only decrease the efficiency of the program, thus lowering its value. By contrast, creativity is highly valued in traditional literary works. Another distinction is that an engineer can create hardware[25] to perform any task that software performs on a general purpose computer.

B. Reverse Engineering

Reverse engineering of a computer program essentially is an analysis of an existing program.[26] Vendors usually supply ouly object code to customers who purchase their software; a competitor who wants to creato better software or software that is capable of interacting with the original program, however, must examine human-readable code to determine how the original software operates or must study the required interface specifications[27] and protocols to determine how to communicate with that system.[28] A programmer who only has access to object code must reverse engineer that object code to produce a rough version of the source code. This process,

---

24. For a discussion of program behavior and static versus dynamic structures, see Davis, 5 Software L. J. at 306-07, 312-13 (cited in note 20).

25. "Hardware" refers to the physical components of machines, including computers and other electronic and electrical devices.

26. For a non-technical explanation of reverse engineering, see Johnson-Laird, 5 Software L. J. at 331 (cited in note 20).

27. Interface specifications are the functional requirements that nust be present in a computer program for it to work or communicate in conjunction with other software or bardware.

28. Protocols define the formats, sequences, and timing of messages that are exchanged between two communicating systems. Michael A. Jacobs, *Copyright and Compatibility*, 30 Jurimetrics J. 91, 95 (1989). See also Johnson-Laird, 5 Software L. J. at 338-40 (cited in note 20). For a discussion of intoroperability, compatibility, and standardization, see text accompanying notes 188-99.

called decompiling[29] or disassembling,[30] transforms object code back into a terse, but human-readable, form. To accurately decipher or decompile a program of any size, the programmer must make an interim copy of the object code prior to decompilation. The focus of great debate in recent years is this intermediate copying that almost always is required for reverse engineering.

Reverse engineering of object code is not a straightforward, easy process. The source code for a computer program usually is written in a higher-level computer language that contains labels, comments, and mnemonic variable names that explain the program's instructions and identify functions of the code. These utilities are essential te the computer programmers who develop and maintain the program. When the source code is compiled or translated into object code, these labels, comments, and mnemonic variable names are stripped out of the machine-readable version of the program because they are useless to the computer. Analyzing this machine-readable or object code is unreasonably time-consuming, tedious, and error-prone. The only feasible option, therefore, is to decompile the object code. When a programmer decompiles or disassembles a computer program into assembly code,[31] however, the resulting assembly code lacks any of the programming utilities or human explanations. That is, the end product of decompilation is not the original source code but a bare-bones interpretation of how the computer program operates.[32] Reading decompiled code is comparable to reading a novel that has been stripped of all adjectives, adverbs, articles, and other explanatory words; reorganized te be completely chronological with no chapters or paragraphs; and changed so that the characters, places,

---

29. Decompilation is the opposite of compiling a program; the result of decompiling object code is a dense, hard-to-read series of cryptic instructions. For an explanation of computer program decompilation, see generally Johnson-Laird, 5 Software L. J. at 331 (cited in note 20). Decompilation is one of several ways to reverse engineer a computer program. Jonathan Band and Laura L.F.H. McDonald, *The Fair Use Bill: A Funny Thing Happened on the Way to Congress*, 10 Computor L. 9, 16 n.12 (March 1993).

30. Disassembly and decompilation are essentially the same thing. Technically, dissassembly refers to the transfer of code from the machine-readable format to assembly code. See noto 31 for a description of assembly code.

31. Assembly code is a lower-level computer language that programmers can read. Often high-level languages first are translated to assembly code, and the assembly code is then translated to object code—that is, the assembly code is an intermediate format. Assembly code differs from computor to computor, whereas the high-level languages are machine independent. Because one line of a high-level language generally will translato into multiple lines of assembly code, the average programmer has more difficulty understanding or deciphering the assembly code. Therefore, some programmers actually specialize in assembly languages.

32. In fact, true decompilation is impossible because a decompiler never will produce the original source code.

and other nouns are represented by a single letter followed by a single digit.

After generating a reverse engineered version of a computer program, even an experienced and patient computer programmer will need a considerable amount of time to understand how the program works well enough to identify the interface specifications or how the program operates. Reverse engineering of software is not a routine conversion; rather, it is a very laborious, additive process that requires programmers to supply their own explanatory information because no higher-level information remains in the executable version of the program.[33] Although decompilation produces a list of human-readable instructions, generating higher levels of knowledge and understanding about the program requires time, skill, intellectual contribution, and experience.[34] This higher understanding is necessary te discern the interface specifications and protocols essential for inoperability with another system. Reverse engineering and analysis of a computer program essentially require the reinvention of parts of the wheel.[35] This process often is more difficult than writing a program from scratch.

III. INTELLECTUAL PROPERTY LAW FOR SOFTWARE

The Computer Software Copyright Act of 1980[36] now affords protection for software under the general copyright law that Congress originally designed to protect literary and artistic works.[37] The legislative history of the Copyright Act explicitly states that the

---

33. Johnson-Laird, 5 Software L. J. at 344 (cited in note 20).

34. Id. at 346.

35. See Reichman, 42 Vand. L. Rev. at 701 (cited in note 3); Davidson, 47 U. Pitt. L. Rev. at 1080-81, 1090-94 (cited in note 1).

36. In 1980, Congress adopted the recommendation of the National Commission of New Technological Uses of Copyrighted Works (CONTU). This amendment made significant changes te the Copyright Act of 1976, including the addition of the definition of computer programs in 17 U.S.C. § 101. The Copyright Act now defines a computer program as "a set of statements or instructions to be used directly or indirectly in a computer in order te bring about a certain result." 17 U.S.C. § 101. In addition, the Act defines literary works as "works, other than audiovisual works, expressed in words, numbers, or other verbal or numerical symbols or indicia, regardless of the nature of the material objects, such as books, periodicals, manuscripts, phonorecords, film tapes, disks, or cards, in which they are embodied." Id. See also H.R. Rep. No. 94-1476 at 54 (cited in note 1).

37. The law now includes computer programs in the broad category of protected works. See, for example, *Computer Assoc. Int'l, Inc. v. Altai, Inc.*, 982 F.2d 693, 702 (2d Cir. 1992); *Whelan Assoc., Inc. v. Jaslow Dental Lab., Inc.*, 797 F.2d 1222, 1234 (3d Cir. 1986); *Apple Computer, Inc. v. Franklin Computer Corp.*, 714 F.2d 1240, 1247 (3d Cir. 1983). For a history of the protection of computer programs under copyright law, see Samuelson, 1984 Duke L. J. at 692-703 (cited in note 20).

copyright laws protect the expression of a computer program to the extent that the expression of original ideas is distinguished from the ideas themselves.[38]

A. Scope of Protection

Although computer programs are copyrightable, defining and detecting infringement poses difficulties.[39] A fundamental principle of copyright law is that expressions, but not ideas, receive protection.[40] Distinguishing between ideas and expressions, however, is rarely a simple task.[41] Courts have struggled to delineate the distinction between ideas and expression in computer programs. Two courts have held that copyright protection extends beyond the litoral elements to structure, sequence, and organization.[42] More thoughtful courts and commentators squarely have rejected this type of rule because it creates ambiguity,[43] fails to account for independent creation, and does

---

38. H.R. Rep. No. 94-1476 at 54 (cited in note 1).

39. The ultimate defense te a copyright infringement action is independent creation. Reichman, 42 Vand. L. Rev. at 689 (cited in note 3).

40. This basic tenet of copyright law is codified in 17 U.S.C § 102(b), which states: "In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concopt, principle, or discevery, regardless of the form in which it is described, explained, illustrated, or embodied in such work." *Baker v. Selden*, 101 U.S. 99 (1879), the first case to address the limits of subject matter of copyrighted works, introduced this axiom. Id. at 102 (stating that "[t]o give te the author of the book an exclusive property in the art described therein, when no examination of its novelty has ever been officially made, would be a surprise and a fraud upon the public"). See also *Mazer v. Stein*, 347 U.S. 201, 217 (1954); *Apple Computer*, 714 F.2d at 1250.

41. Judge Learned Hand statod that "[n]obody has ever been able to fix that boundary, and nobody ever can." *Nichols v. Universal Pictures Corp.*, 45 F.2d 119, 121 (2d Cir. 1930). He later explained that the distinction between ideas and expression is necossarily ad hoc. See *Peter Pan Fabrics, Inc. v. Martin Weiner Corp.*, 274 F.2d 487, 489 (2d Cir. 1960). See also *Computer Assoc. Int'l*, 982 F.2d at 703-06 (discussing the idea-expression dichotemy and applying Judge Hand's abstraction analysis set forth in *Nichols*).

42. See *Whelan*, 797 F.2d at 1236 (holding that only the function or purpose of a utilitarian work would qualify as an idea and that everything else would be classified as expression); *SAS Inst., Inc. v. S & H Computer Sys., Inc.*, 605 F. Supp. 816, 830 (M.D. Tenn. 1985) (accepting evidence of literal and organizational similarities in detormining that infringment had occurred). One of the problems with the *Whelan* decision is that the court mistakenly implies that because the idea of an efficient organization of a dental laboratory can be expressed with a variety of program structures, the structure itself is not incident to the idea and, thus, is expression. *Whelan*, 797 F.2d at 1240. The fact that something can be done in multiple ways does not make a work autematically a protected expression.

43. See generally Pamela Samuelson, *Computer Programs, User Interfaces, and Section 102(b) of the Copyright Act of 1976: A Critique of* Lotus v. Paperback, 55 L. & Contemp. Probs. 311, 322-24 (Spring 1992); John P. Sumner, *The Copyright/Patent Interface: Patent Protection for the Structure of Program Code*, 30 Jurimetrics J. 107, 113-14 (1989); Pamela Samuelson, *Reflections on the State of American Software Copyright Law and the Perils of Teaching It*, 13 Colum.-V.L.A. J. L. & Arts 61, 62-65 (1988); J. Dianne Brinson, *Copyrighted Software: Separating the Protected Expression from Unprotected Ideas, A Starting Point*, 29 B.C. L. Rev. 803 (1988);

not recognize that software contains many distinct ideas.[44] Furthermore, the former antiquated rule implies that most of a computer program is expression,[45] an absurd proposition because software is essentially a utilitarian product in which creative embellishment only degrades performance.

The distinction between ideas and expression is complicated by the utilitarian nature[46] of software that is absent from traditional subject matter. Traditionally, copyright law has not protected utilitarian works[47] that have a function beyond merely portraying an appearance or conveying information.[48] Computer programs are utilitarian because they are capable of directly operating machines.[49] By including computer programs in the definition of literary works, however, Congress ignored this utilitarian aspect.[50]

As a result, computer programs generally are considered functional works in the copyright context.[51] Historically, courts have afforded functional works a lesser degree of protection[52] than creative

---

Steven R. Englund, Note, *Idea, Process, or Protected Expression?: Determining the Scope of Copyright Protection of the Structure of Computer Programs*, 88 Mich. L. Rev. 866 (1990).

44. See *Atari Games Corp. v. Nintendo of Am., Inc.*, 975 F.2d 832, 839 (Fed. Cir. 1992).

45. See *Syncrcom Technology, Inc. v. University Computing Co.*, 462 F. Supp. 1003, 1013 (N.D. Tex. 1978) (asking what separable idea is being expressed if sequencing and ordering are expression).

46. See *SAS Inst.*, 605 F. Supp. at 829.

47. See 17 U.S.C. §§ 101, 113; *Baker v. Scldcn*, 101 U.S. 99 (1879).

48. 17 U.S.C. § 101. Copyright extends te artistic features of utilitarian works only if a court can identify the artistic features separately from the utilitarian aspects of the work. Id. (defining pictorial, graphic, and sculptural works); *Mazer*, 347 U.S. at 218. See also Reichman, 42 Vand. L. Rev. at 693 n.288 (cited in note 3) (discussing the implications of *Baker v. Selden*).

49. See text accompanying notes 19-35 for a discussion of the characteristics of computer programs. See also Samuelson, 1984 Duke L. J. at 727-49 (cited in note 20) (discussing the utilitarian nature of computer programs).

50. J. H. Reichman, *Design Protection and the New Technologies: The United States Experience in a Transnational Perspective*, 19 U. Balt. L. Rev. 6, 12 (1989). Perhaps Congress saw computer programs as works that convey information. Id. at 12 n.23. In contrast, Congress may have extended copyright protection te the machine-readable form of computer programs because the CONTU Report failed to advise Congress of the utilitarian nature of computer programs and the serious changes te copyright law that could result from this action. Pamela Samuelson, *Creating a New Kind of Intellectual Property: Applying the Lessons of the Chip Law to Computer Programs*, 70 Minn. L. Rev. 471, 474-75 (1985).

51. Paul Goldstein, 2 *Copyright* § 8.5 at 116 (Little, Brown, 1989). For a general explanation of functional works and the differences between functional and nonfunctional works, see Goldstein, 1 *Copyright* § 2.15 at 195-98.

52. *LaST Frontier, Conference Report on Copyright Protection of Computer Software*, 30 Jurimetrics J. 15, 18-19 (1989) ("*LaST Frontier Report*"). The scope of copyright protection for functional works is "thin" or narrow. The Supreme Court most recently enunciated the doctrine of thin copyright for functional works in *Feist Publications, Inc. v. Rural Tel. Serv. Co.*, 111 S. Ct. 1282, 1289-91 (1991). See also Goldstein, 1 *Copyright* § 2.15 at 197; H.R. Rep. No. 94-1476 at 53-57 (cited in note 1); J.H. Reichman, *Goldetein on Copyright Law: A Realist's Approach to a Technological Age*, 43 Stan. L. Rev. 943, 970-73 (1991) (noting that the CONTU Report neglected to discuss the thin protection traditionally afforded te functional works).

works[53] based on Section 102(b) of the Copyright Act,[54] which states that systems, like abstract ideas, are not protectable.[55] Overprotection of functional works makes them look more like patentable than copyrightable works.[56] Although some courts and experts have adopted a highly protectionist standard for computer programs,[57] others have criticized this narrow view.[58]

Another way that computer programs are different from other copyrightable subject matter is that it usually is not feasible te examine the unprotected elements of a computer program in its

53. See Reichman, 42 Vand. L. Rev. at 693-95 n.288 (cited in note 3). Recently, one court failed to provide thin protection for a functional writing as recognized by the Copyright Act, *Baker v. Selden*, and other functional work cases. See *Lotus Dev. Corp. v. Paperback Software Int'l*, 740 F. Supp. 37 (D. Mass. 1990). See also Samuelson, 55 L. & Contemp. Probs. at 351 (cited in note 43).

54. 17 U.S.C. § 102(b).

55. See Samuelson, 55 L. & Contemp. Probs. at 324 n.61 (cited in note 43), for a comprehensive list of examples.

56. See Part V.A.2 for a more detailed explanation of how overprotection of functional works begins to resemble patent-like protection.

57. For example, the *Whelan* court held that the purpose or function of a computer program is the work's idea, and everything not necessary to that purpose or function is part of the expression. *Whelan*, 797 F.2d at 1236. The court also found that although computer programs are essentially utilitarian in nature, the law should not afford computer programs any less protection than traditional literary works. Id. at 1240. See also *Baker v. Selden*, 101 U.S. 99 (1879); *Bibbero Sys., Inc. v. Colwell Sys., Inc.*, 893 F.2d 1104 (9th Cir. 1990).

Arguably, a user interface is merely an idea, and only the cede implementing the interface is cepyrightable expression. Given this view, no cepyright infringement occurs if one only copies the functional aspects of an interface. Samuelson, 55 L. & Contemp. Probs. at 336 n. 112 (cited in note 43). "[T]he mere existence of alternatives does not demonstrate that a nonliteral aspect of a computer program is 'expressive.'" Id. at 323. Others feel that a user interface is a valuable nonliteral element of copyrighted computer programs and copying is, thus, an infringement. Samuelson takes this to mean that such reasoning does not allow the public to make products that have compatible user interfaces. Id. at 336 n.112. See also Jacobs, 30 Jurimetrics J. at 91 (cited in note 28); Dennis S. Karjala, *Copyright, Computer Software, and the New Protectionism*, 28 Jurimetrics J. 33 (1987); William T. Lake, John H. Harwood, II, and Thomas P. Olson, *Tampering with Fundamentals: A Critique of Proposed Changes in EC Software Protection*, 6 Computer L. 1 (Dec. 1989). One commentater claims that the copyright law allows Sega to select its own business strategy by choosing to license its interfaces or make them available for competitors to copy. Anthony L. Clapes, Sega v. Accolade *and the Intellectual Property/Antitrust Interface*, 15 Computer L. Rep. 270, 273 (1992).

58. Professor Reichman has suggested that because independently created functional works may not contain the author's personality, some courts require a stronger showing of creativity. Reichman, 42 Vand. L. Rev. at 684 (cited in note 3). In fact, Reichman advocates that courts should require quantitative creativity in computer programs to establish originality. Id. at 688.

Professor Samuelson has criticized *Whelan* for its narrow definition of what copyright law considers to be an "idea" in a computer program because the court presumed there was only one idea per computer program rather than many unprotectable elements. Samuelson, 55 L. & Contemp. Probs. at 322 (cited in note 43). In reality, a computer program consists of numerous subprograms that are programs themselves. The *Whelan* court also failed to recognize the § 102(b) prohibition against protection of elements such as processes, procedures, systems, and methods of operation, even though they are contained in the body of a copyrighted work. Id. at 322-23.

distributed executable form.[59] Someone who wants to discern the underlying ideas of a program must reverse engineer the program to create a human-readable form.[60] As indicated in Section 102(a), the Copyright Act provides copyright protection to original works "fixed in any tangible medium of expression" that can be perceived "directly or with the aid of a machine or device."[61] Although copyright protection does not depend on a human's ability to read or understand a copyrighted work,[62] the primary goal of copyright law is to give the public a right to access all ideas embodied in the copyrighted work.[63] To maintain consistency with the public access goal, interested persons in theory should have access to the underlying ideas of a computer program.

B. Reverse Engineering in Intellectual Property Law

The laws regarding reverse engineering derive from trade secret laws,[64] which traditionally allowed competitors to start with the known product and work backward to determine the process used to develop or manufacture the product.[65] Reverse engineering is standard industry practice in many fields, including the computer industry.[66] Under the patent paradigm, reverse engineering is permissible for unpatented technologies.[67] In addition, the

---

59. See text accompanying notes 21-25.

60. See text accompanying notes 26-35.

61. 17 U.S.C. § 102(a).

62. In *White-Smith Music Publishing Co. v. Apollo Co.*, 209 U.S. 1 (1907), the Supreme Court held that musical compositions embodied in player piano rolls were not copies. Id. at 18. Some courts have indicated that the 1976 Copyright Act eliminated this human perceptibility requirement. See, for example, *Apple Computer, Inc. v. Franklin Computer Corp.*, 714 F.2d 1240, 1248 (3d Cir. 1983); *SAS Inst., Inc. v. S & H Computer Sys., Inc.*, 605 F. Supp. 816, 829 (M.D. Tenn. 1985) (holding that using unauthorized human-readable versions of programs as a step in creating a competing program is a copyright infringement). See also 1 *Nimmer on Copyright* § 2.03[B][1] at 2-28 to -29 (cited in note 1).

63. See note 1.

64. See *Kewance Oil Co. v. Bicron Corp.*, 416 U.S. 470, 489-90 (1974).

65. The *Kewance* Court defined reverse engineering as a "fair and honest means [of] . . . starting with the known product and working backward to divine the procoss which aided in its development or manufacture." Id. at 476.

66. See, for oxample, *E.F. Johnson Co. v. Uniden Corp. of Am.*, 623 F. Supp. 1485, 1501-02 n. 17 (D. Minn. 1985) (identifying and endorsing reverse engineering of a computer program as standard practice in the industry). The court found infringement because the defendant copied the plaintiff's protected expression verbatim. Id. at 1503.

Note that when the Computer Software Copyright Act of 1980 was passed, software was less complex, compilers and decompilers were new, and reverse engineering was hard to do. Thus, reverse engineering was not an issue. See note 36 and accompanying text for discussion of the 1980 Copyright Act.

67. Competitors may reverse engineer works not protected by federal patent law. See, for example, *Bonito Boats, Inc. v. Thunder Craft Boats, Inc.*, 489 U.S. 141, 160-61 (1989) (holding that reverse engineering is beneficial because of its innovative character); *Sears, Roebuck & Co. v.*

Semiconductor Protection Act of 1984[68] expressly permits reverse engineering provided that the resulting product reflects a reasonable amount of investment and effort.[69]

The copyright paradigm permits reverse engineering in two cases. First, under Section 113(b) of the Copyright Act, intermediate copies of three-dimensional utilitarian articles may be made for the purpose of studying and analyzing the utilitarian features of the object.[70] Second, the exemption under Section 117 permits the owner of a copy of a computer program to make another copy or adapt the program for the purposes of archiving or using the program.[71]

Outside of the established exemptions, a programmer generally decompiles computer object code for one of three reasons: to analyze the underlying functionality of a program, te make one system compatible with another, or to pirate software. Some courts[72] and many commentaters[73] agree that reverse engmeering of computer programs

---

Stiffel Co., 376 U.S. 225, 232-33 (1964); *Compco Corp. v. Day-Brite Lighting, Inc.*, 376 U.S. 234, 237-38 (1964); *Secure Serv. Technology, Inc. v. Time & Space Processing, Inc.*, 722 F. Supp. 1354, 1361 (E.D. Va. 1989) (holding that trade secret law permits reverse engineering of a facsimile machine for the purpose of making an interoperable facsimile machine).

68. 17 U.S.C. §§ 901 et seq. (1988 & Supp. 1992).

69. 17 U.S.C. § 906. See generally Leo J. Raskind, *Reverse Engineering, Unfair Competition, and Fair Use*, 70 Minn. L. Rev. 385 (1985).

70. 17 U.S.C. § 113(b). See generally J. H. Reichman, *Design Protection in Domestic and Foreign Copyright Law: From the Berne Revision of 1948 to the Copyright Act of 1976*, 1983 Duke L. J. 1143, 1201-13 (explaining the evolution of § 113(b)).

71. 17 U.S.C. § 117 states:

> Notwithstanding the provisions of section 106, it is not an infringement for the owner of a copy of a computer program te make or authorize the making of another copy or adaptation of that computer program provided: (1) that such new a [sic] copy or adaptation is created as an essential step in the utilization of the computer program in conjunction with a machine and that it is used in no other manner, or (2) that such new copy or adaptation is for archival purposes only and that all archival copies are destroyed in the event that continued possession of the computer program should cease to be rightful. . . .

See also *Vault Corp. v. Quaid Software Ltd.*, 847 F.2d 255, 261 (5th Cir. 1988).

72. See, for example, *NEC Corp. v. Intel Corp.*, 10 U.S.P.Q.2d 1177, 1184 (N.D. Cal. 1989) (recognizing that one could disassemblo microcode and use the resulting information te develop separate microcode without infringement). Moreover, in its corrected brief to the district court, Accolade asserted that in the more than ninety published opinions addressing reverse engineering, no court has found reverse engineering of software to be inproper because computer programs must be copied and disassembled in order te read and understand them. Corrected Opposition of Accolade, Inc. te Sega Enterprises Ltd.'s Motion for Preliminary Injunction 13, *Sega Enter. Ltd. v. Accolade, Inc.*, 785 F. Supp. 1392 (N.D. Cal. 1992) (No. C91-03871 BAC), reprinted in 15 Computer L. Rep. 592, 601 (1992). See generally text accompanying notes 62-63 for a discussion of the public accoss goal. For the public to access the unprotectable aspects of a work, that work must exist in a human-readable form.

73. See, for example, Brief Amicus Curiae at 4-28, reprinted in 33 Jurimetrics J. at 149-62 (cited in note 1) (discussing decompilation of software for analysis purposes ouly); Samuelson, 70 Minn. L. Rev. at 524 (cited in note 50) (asserting that courts should allow users to copy a computer program for the purpose of reverse engineering, as in semiconductor chip law); Goldstein, *Copyright* § 5.2.1.4 at 116-23 (Supp. 1993) (cited in note 51) (utilizing the managed copying idea and the fair use doctrine to support reverse engineering of software); Rashind, 70 Minn. L. Rev. at 389 (cited in note 69) (justifying reverse engineering by distinguishing between piracy and

does not constitute copyright infringement if the final version is not substantially similar to the original copyrighted version.[74] Others believe the courts should use this substantial similarity test only if no direct evidence of copying exists.[75] Few courts have focused more

---

legitimate competition); Reichman, 42 Vand. L. Rev. at 702 (cited in note 3) (explaining that reverse engineering is not copyright infringement when a new product incorporates only the ideas of the original program).

In his discussion of nonliteral similarity, Nimmer implies that the Copyright Act does not permit reverse engineering of computer programs. 3 *Nimmer on Copyright* § 13.03[A][1] at 13-29 to -46 (cited in note 1). Clapes argues that disassembly is an attempt te discover information that one has withheld purposefully and lawfully from competiters. Clapes, 15 Computer L. Rep. at 272 (cited in note 57). Evidently, the maxi-protectionist view is that copyright law should protect translations, paraphrasing, and nonliteral copying. See Pamela Samuelson and Robert J. Glushko, *Comparing the Views of Lawyers and User Interface Designers on the Software Copyright "Look and Feel" Lawsuits*, 30 Jurimetrics J. 121, 135-36 (1989). Other commentators believe that reverse engineering is a low-cost, quick method of eliminating all lead time and that one easily can alter a computer program so that it will not resemble the original program evon though it is a copy. See Steering Committee, *Intellectual Property Issues in Software* 78 (Nat'l Academy, 1991). One scholar suggests that the large software producers provoke the reverse engineering debate te improve their competitive position, even though many early and current products contain information obtained through reverse engineering. See Johnson-Laird, 5 Software L. J. at 354 (cited in note 20).

74. *See v. Durang*, 711 F.2d 141, 142 (9th Cir. 1983) (stating that "[c]opying deleted or so distinguished as to be unrecognizable is not copying"); *NEC Corp.*, 10 U.S.P.Q.2d at 1184 (holding that there was no copyright infringement because the final version was not substantially similar to the original copyrighted program, even though the defendant had disassembled the plaintiff's code and made a direct copy); *Hubco Data Prod. Corp. v. Management Assistance, Inc.*, 219 U.S.P.Q. 450, 455-56 (D. Idaho 1983) (holding that the reverse engineering was itself legal, but that the wholesale copying and resale of the object code constituted copyright infringement); *Vault Corp.*, 847 F.2d at 261 (strengthening the making of copies for reverse ongineering by overruling a state statute that declared reverse engineering to contravene the copyright law, stating that the computer program owner was permitted to copy a program if it was "created as an essential step in the utilization of the computer program," and explaining that "Section 117(1) contains no language to suggest that the copy it permits must be employed for a use intended by the copyright owner, and, absent clear congressional guidance te the contrary, we refuse to read such limiting language inte this exception").

In *Johnson Controls, Inc. v. Phoenix Control Sys., Inc.*, 886 F.2d 1173 (9th Cir. 1989), the court held that the lower court did not abuse its discretion in a cepyrighted software infringoment case when it excluded evidenco of the final version of the code. Id. at 1177. This holding does not mean, however, that it is unnecessary to review the final version of a defendant's code in a copyright infringement case. See Richard H. Stern, Sega Enterprises, Ltd. v. Accolade, Inc.: *No Accolades for an Ill-Conceived Analysis of Software Copyright Infringement and Fair Use*, 15 Computer L. Rep. 263, 264 (1992). Furthermore, *Johnson Controls* did not involve reverse engineering of object code because the defendant had direct accoss te the source code. *Johnson Controls*, 886 F.2d at 1176.

For scholarly agreement that reverse engineering is permissible absent substantial similarity, see *LaST Frontier Report*, 30 Jurimetrics J. at 24-25 (cited in note 52); Goldstein, *Copyright* § 5.2.1.4 at 116-23 (Supp. 1993) (cited in note 51). In fact, one commentater believes that sufficient precedent exists te indicate that a work infringes a copyrighted work only if the subsequent work contains a substantial amount of the protected expression of the original, without regard to earlier versions of the subsequent work. See Stern, 15 Computer L. Rep. at 265.

75. Appellee's Brief 29-30, *Sega Enter. Ltd. v. Accolade, Inc.*, 977 F.2d 1510 (9th Cir. 1992) (No. 92-15655), reprinted in 15 Computer L. Rep. 1004, 1018-19 (1992). See also *Johnson Controls*, 886 F.2d at 1176 (stating that "[o]opying can be shown by circumstantial evidence of access to the copyrighted work, and substantial similarity between the copyrighted work and the infringer's work"); *Atari, Inc. v. North Am. Philips Consumer Elec. Corp.*, 672 F.2d 607, 614 (7th

intently on the intermediate copying than on the degree of similarity of the final product to the original copyrighted work; however, one court that has examined the intermediate copying issue found that it was fair use under certain circumstances.[76]

C. Fair Use Doctrine

The doctrine of fair use is a recognized defense to copyright infringement at common law. This judicial doctrine is a significant and well-established limitation on the exclusive right of copyright owners recognized by Congress[77] and is codified in Section 107 of the Copyright Act of 1976.[78] Congress permitted courts to adapt the fair

---

Cir. 1982); *Novelty Textile Mills, Inc. v. Joan Fabrics Corp.*, 558 F.2d 1090, 1092 (2d Cir. 1977); *Arnstein v. Porter*, 154 F.2d 464, 468 (2d Cir. 1946).

76. See *Atari Games Corp. v. Nintendo of Am., Inc.*, 975 F.2d 832, 843 (Fed. Cir. 1992) (holding that "[w]hen the nature of a work requires intermediato copying to understand the ideas and processes in a copyrighted work, that nature supports a fair use for intormediate copying. Thus reverse engineering object code te discern the unprotectable ideas in a computor program is a fair use"). For cases in which courts found infringement without analyzing the possible lawfulness of intermediate copying under certain circumstances, see *Johnson Controls*, 886 F.2d at 1176-77; *Walt Disney Prod. v. Filmation Assocs.*, 628 F. Supp. 871, 876-77 (C.D. Cal. 1986); *SAS Inst.*, 605 F. Supp. at 828-29; *Hubco Data*, 219 U.S.P.Q. at 455-56; *Walker v. University Books, Inc.*, 602 F.2d 859, 862-64 (9th Cir. 1979) (holding that infringement could be based on blueprints for I-Ching cards, but remanding on the substantial similarity issue). For cases in which the courts did not find infringement based not on the presence of intermediate copying as an initial step in software development but on the lack of substantial similarity, see *NEC Corp.*, 10 U.S.P.Q.2d at 1183-84 (finding that the common law permitted disassembly of cepyrighted microcode under cepyright law and that intermediate copying did not censtituto infringement because the final version and the original copyrighted version were not substantially similar); *E.F. Johnson Co. v. Uniden Corp. of America*, 623 F. Supp. 1485, 1501-02 n.17 (D. Minn. 1985); *See*, 711 F.2d at 143. Compare *Computer Assoc. Int'l, Inc. v. Altai, Inc.*, 982 F.2d 693, 706-12 (2d Cir. 1992) (refusing to treat earlier and later versions of a computer program as a single unit for purposes of determining cepyright liability, even though the earlier version was infringing and the later one revised the earlier version).

77. H.R. Rep. No. 94-1476 at 65 (cited in note 1). Courts have used the Report of the House Committee on the Judiciary as authority for interpreting Congressional intent for enacting the 1976 Copyright Act. See, for example, *Feist Publications, Ine. v. Rural Tel. Serv. Co.*, 111 S. Ct. 1282, 1293-95 (1991); *Community for Creative Non-Violence v. Reid*, 490 U.S. 730, 747, 749 n.15 (1988); *Bonito Boats, Inc. v. Thunder Craft Boats, Inc.*, 489 U.S. 141, 167-68 (1988); *Harper & Row, Publishers, Ine. v. National Enter.*, 471 U.S. 539, 549, 552-55 (1984); *National Car Rental, Inc. v. Computer Assoc. Int'l, Inc.*, 991 F.2d 426, 433 (8th Cir. 1993); *Forward v. Thorogood*, 985 F.2d 604, 607 (1st Cir. 1993); *Atari Games*, 975 F.2d at 840; *Broadcast Music, Inc. v. Claire's Boutiques, Inc.*, 949 F.2d 1482, 1488 (7th Cir. 1991); *Miss America Org. v. Mattel, Inc.*, 945 F.2d at 536, 548 (2d Cir. 1991); *National Broadeasting Co., Inc. v. Satellite Broadeast Networks*, 940 F.2d 1467, 1469 n.3 (11th Cir. 1991); *Ford Motor Co. v. Summit Motor Prod., Inc.*, 930 F.2d 277, 299 (3d Cir. 1991). By incerporating the fair use doctrine into the Copyright Act of 1976, Congress recognized "one of the most important and well-established limitations on the exclusive right of copyright owners." H.R. Rep. No. 94-1476 at 65.

78. 17 U.S.C. § 107. The doctrine allows federal courts to "provide relief from 'copyright abuse' by copyright owners toward non-owners of a cepy." Stephen Kyle Tapp and Daniel E. Wanat, *Computer Software Copyright Issues: Section 117 and Fair Use*, 22 Memphis St. U. L. Rev. 197, 272 (1992). The doctrine is a defense to an otherwise valid claim of copyright infringement. *Sega*, 977 F.2d at 1521. The fair use doctrine is a mixed question of law and fact:

use doctrine to particular situations on a case-by-case basis during periods of rapid technological change.[79] This judicial freedom to adapt the doctrine to technological change also applies to infringing computer programs.[80]

Courts generally have taken two approaches to the fair use doctrine. The narrow view is that fair use is "an equitable rule of reason"[81]—a privilege used to excuse a technical violation of the exclusive right of an author.[82] The broader view is that the fair use doctrine embodies the public policy of ensuring access to information that actually exists outside the realm of copyright.[83] These two views demonstrate the basic doctrinal tension between control and accessibility.

Section 107 of the Copyright Act enumerates four factors for a court to consider when determining whether a use of copyrighted material is a fair one:

"[W]here the district court has found facts sufficient to evaluate each of the statutory factors," the appellate court may determine whether fair use has occurred as a matter of law. *Elsmere Music, Inc. v. Nat'l Broadcasting Co., Inc.*, 482 F. Supp. 741, 747 (S.D.N.Y. 1980), aff'd, 623 F.2d 252 (2d Cir. 1980).

The first enunciation of the principles of this doctrine in United States copyright law was in *Folsom v. Marsh*, 9 F. Cas. 342, 348 (C.C.D. Mass. 1841) (No. 4901) (stating that "[i]f so much is taken, that the value of the original is sensibly diminished, or the labors of the original author are substantially to an injurious extent appropriated by another, that is sufficient, in point of law, to constitute a piracy pro tanto"). Subsequent fair use decisions developed the theory. See, for example, *Rosemont Enter., Inc. v. Random House, Inc.*, 366 F.2d 303, 307 (2d Cir. 1966) (stating that a fundamental justification for fair use existed in the constitutional purpose for copyright: "To Promote the Progress of Science and the Useful Arts"). The courts have varied widely in recent years with regard to the fair use doctrine. See, for example, *New Era Publications, Int'l v. Henry Holt & Co., Inc.*, 873 F.2d 576, 583-85 (2d Cir. 1989) (holding it was not fair use to reproduce unpublished writings of the scientology founder); *Salinger v. Random House, Inc.*, 811 F.2d 90, 100 (2d Cir. 1987) (holding it was not fair use to use unpublished letters in a biography); *Hustler Magazine, Inc. v. Moral Majority, Inc.*, 796 F.2d 1148, 1151-56 (9th Cir. 1986) (holding that it was fair use when a minister used, without authority, a copy of a parody for fundraising and rebuttal); *Harper & Row*, 471 U.S. at 560-69 (holding that substantial copying of an unpublished article was not fair use); *Sony Corp. v. Universal City Studios, Inc.*, 464 U.S. 417, 447-50 (1984) (holding that home videotaping is fair use).

79. "The bill endorses the purpose and general scope of the judicial doctrine of fair use, but there is no disposition to freeze the doctrine in the statute, especially during a period of rapid technological change. Beyond a very broad statutory explanation of what fair use is and some of the criteria applicable to it, the courts must be free to adapt the doctrine to particular situations on a case-by-case basis." H.R. Rep. No. 94-1476 (cited in note 1).

80. Tapp and Wanat, 22 Memphis St. U. L. Rev. at 259-60 (cited in note 78).

81. Id. at 259. See also *Harper & Row*, 471 U.S. at 560 (quoting H.R. Rep. No. 94-1476 at 65).

82. 3 *Nimmer on Copyright* § 13.05 at 13-79 to -82 (cited in note 1). See also *Benny v. Loew's, Inc.*, 239 F.2d 532, 536-37 (9th Cir. 1956) (holding that a television show parody of a successful movie and play was not fair use because the original was copied practically verbatim, although the actors created burlesque).

83. See *Rosemont Enter.*, 366 F.2d at 306-11 (holding that publication of an unauthorized biography was outside the gambit of the "commercial use" contemplated by the "right of publicity").

> (1) the purpose and character of the use, including whether such use is of a commercial nature or is for nonprofit educational purposes; (2) the nature of the copyrighted work; (3) the amount and substantiality of the portion used in relation to the copyrighted work as a whole; and (4) the effect of the use upon the potential market for or value of the copyrighted work.[84]

These factors, however, function only as guides, not exclusive requirements.[85]

1. Purpose and Character of Use

The first criterion, known as the purpose test, requires a court to consider the purpose and character of the use, particularly whether the questioned usage is of a commercial nature or is for nonprofit educational purposes.[86] In *Sony Corp. of America v. Universal City Studios, Inc.*,[87] the Supreme Court established two rebuttable presumptions under the purpose test: commercial use is treated differently from noncommercial use and is presumed unfair, and every commercial use results in harm to the potential market for the copyrighted work.[88] The Supreme Court later rejected the implication in *Sony* that noncommercial use was presumptively fair,[89] but consistently has held that commercial use tends te weigh against a finding of fair use.[90] The Court stated that the distinction between profit and non-profit usage is not whether the sole purpose is for monetary gain, but whether the copier stands to profit by exploiting the copyrighted material without paying the customary price to the copyright owner.[91]

---

84. 17 U.S.C. § 107.
85. H.R. Rep. No. 94-1476 at 65-66 (cited in note 1).
86. 17 U.S.C. § 107(1).
87 464 U.S. 417 (1984).
88. Id. at 449, 451. The Court specifically stated: "If the Betamax were used to make copies for a commercial or profit-making purpose, such uso would presumptively be unfair. . . . Thus, although every commercial use of copyrighted material is presumptively an unfair exploitation of tho monopoly privilege that belongs to the owner of the copyright, noncommercial uses are a different matter." Id.
89. *Harper & Row*, 471 U.S. at 561.
90. Id. at 562. See also *Sony*, 464 U.S. at 451 (stating that "every commercial use ef copyrighted material is presumptively an unfair exploitation of the monopoly privilege that belongs to the owner of the copyright").
91. *Harper & Row*, 471 U.S. at 562. See also *Roy Export Co. Establishment v. Columbia Broadcasting Sys., Inc.*, 503 F. Supp. 1137, 1144 (S.D.N.Y. 1980); 3 *Nimmer on Copyright* § 13.05[A][1] at 13-88.1 & n.25.3 (cited in note 1).

2. Nature of Copyrighted Work

The nature test requires the court to examine the nature of the copyrighted work.[92] Generally, courts uphold the fair use defense more often with factual works than with entertainment works.[93] The fair use defense, however, fails when the use interferes with the author's right to control the first publication of the work.[94]

3. Amount and Substantiality of Portion of Work Used

To apply the substantiality test, a court compares the amount and substantiality of the copied material with the entire copyrighted work.[95] Two prongs comprise this test. First, the court must examine the qualitative substantiality, which includes the content of the use and the commercially valuable parts of the infringing work.[96] Next, the court must examine the extent of use, or quantitative substantiality.[97] One court has stated that the wording of Section 107(3) supports the suggestion that wholesale copying is not fair use.[98] Other courts have adopted a broad quantitative substantiality test, holding that a substantial percentage of copying constituted fair use.[99]

In the computer program context, this quantitative factor almost always will weigh against the copier, who usually will have to make an intermediate copy of all the machine-readable object code in the process of reverse engineering it into a human-readable format. With regard to computer programs, this factor alone gives courts the discretion to deny the fair use defense: a court may deny use of the defense by pointing to the intermediate copying of the entire program. Other courts are savvy enough to realize that this step is necessary to access the unprotected elements of a computer program and will give little weight to this factor.[100]

---

92. 17 U.S.C. § 107(2).
93. *Harper & Row*, 471 U.S. at 563.
94. Id. at 564.
95. 17 U.S.C. § 107(3).
96. See id.
97. See id.
98. See *Marcus v. Rowley*, 695 F.2d 1171, 1176 (9th Cir. 1983) (stating that "wholesale copying of copyrighted material precludes application of the fair use doctrine").
99. See, for example, *Rosemont Enter.*, 366 F.2d at 306 (holding that the fair use doctrine applied even though the defendant used approximately 14% of the plaintiff's articles that appeared in a magazine); *Williams & Wilkins Co. v. United States*, 487 F.2d 1345, 1353 (Ct. Cl. 1973) (holding that it was fair use to copy an entire journal article under certain narrowly defined circumstances and noting that the extent of copying, though important, is only one factor to be taken inte account).
100. See *Sega*, 977 F.2d at 1521.

4. Effect of Use on Potential Market

The market demand test set out in the fair use doctrine requires a court te consider the effect of the use of the subsequent work on the potential market for or value of the copyrighted work.[101] Some courts consider this inquiry the most important test of the fair use doctrine.[102] One authority has suggested that the fair use doctrine only applies to copying that does not materially impair the market value of the copyrighted work.[103] Some courts have examined whether widespread use of the copied work would affect the potential market for the copyrighted work.[104] Other courts have defined adverse market effects to include diminishing potential sales, interference with marketability, and usurping the market.[105] Legitimate competition in the same field, however, does not preclude a fair use defense automatically,[106] even though the competition undoubtedly affects the market for the copyrighted work.

When a subsequent work is used commercially, courts presume potential harm on the market for the copyrighted work.[107] For noncommercial uses, the copyright holder must establish that a causal connection between the infringement and a loss of revenue exists and that widespread use would affect adversely the copyrighted work's potential market.[108] Consistent with the purpose and character of use tests, courts have distinguished between the exploitation of a copyrighted work and copying the work to create an independently creative expression.[109]

5. Public Policy Interests

Although public policy interests are not included explicitly in the fair use factors, some courts have held that certain public interests, such as access to information, outweigh the author's proprietary

---

101. 17 U.S.C. § 107(4).

102. *Harper & Row*, 471 U.S. at 566. See also 3 *Nimmer on Copyright* § 13.05[A][4] at 13-88.12 (cited in note 1).

103. See 1 *Nimmer on Copyright* § 1.10[D] at 1-87.

104. See, for example, *Sony*, 464 U.S. at 451.

105. See, for example, *Hustler Magazine, Inc. v. Moral Majority, Inc.*, 796 U.S. 1148, 1155-56 (9th Cir. 1986). The *Harper & Row* Court found that a use was not fair when it usurped the market for the copyrighted work. *Harper & Row*, 471 U.S. at 567-69.

106. See *Sega*, 977 F.2d at 1523.

107. *Sony*, 464 U.S. at 451.

108. *Harper & Row*, 471 U.S. at 566-67.

109. See, for example, *Sega*, 977 F.2d at 1523.

interest in the original work.[110] At least one court has rejected the access-to-information argument because it recognized that preserving the incentive for authors to create works generally serves the public interest; the court explained that the idea-expression dichotomy provides sufficient protection for the public interest in disseminating information.[111] Similarly, another court has stated that the fair use doctrine is not a license for corporate theft and that a court cannot ignore a copyright whenever it determines that the original work was of interest to the public.[112]

IV. RECENT COPYRIGHT CASES ADDRESSING REVERSE ENGINEERING OF SOFTWARE

The courts uniformly hold that the literal elements of a computer program receive copyright protection.[113] The central, current question is what scope of copyright protection courts should give to the nonliteral elements of computer programs. The result of this inquiry currently revolves around this central question: what elements of software will the reviewing court consider nonliteral? Until recently,

---

110. In *Berlin v. E.C. Publications, Inc.*, 329 F.2d 541 (2d Cir. 1964), the court held that the overriding public interest in parody permitted the defendant to conjure up images of the plaintiff's work because there was no claim that the defendant's parodies of the plaintiff's songs would partially or fully satisfy the demand for the originals. Id. at 545.

Another court found that it was fair use for two government libraries to photocopy thousands of journal articles in their entirety because the public interest in medicine and medical research outweighed the authors' interests. *Williams & Wilkins Co.*, 487 F.2d at 1362. Note that there was no evidence that the original works were substantially harmed by the photocopying. See id. at 1353-54.

The Supreme Court has held that private, noncommercial time-shifting of free broadcast television programming is fair use because of the public interest in accessing information through television. *Sony*, 464 U.S. at 454-55. Note, however, that the *Sony* Court ignored the nature and substantiality tests in examining the fair use of a subsequent nonprofit work.

111. *Harper & Row*, 471 U.S. at 609.

112. *Iowa State Univ. Research Found., Inc. v. American Broadcasting Co., Inc.*, 621 F.2d 57, 61 (2d Cir. 1980).

113. See, for example, *Computer Assoc. Int'l, Inc. v. Altai, Inc.*, 982 F.2d 693, 702 (2d Cir. 1992); *Whelan Assoc., Inc. v. Jaslow Dental Lab., Inc.*, 797 F.2d 1222, 1233 (3d Cir. 1986); *Apple Computer, Inc. v. Franklin Computer Corp.*, 714 F.2d 1240, 1247-48 (3d Cir. 1983). Much support exists for the proposition that copyright protection is only available for literal elements of computer programs. See Samuelson, 55 L. & Contemp. Probs. at 319 (cited in note 43). First, the Copyright Act defines computer programs. Id. Next, the legislative history does not indicate that nonliteral elements are protected by copyright law. Id. Finally, case law denies protection to nonliteral elements of computer programs. See, for example, *Plains Cotton Coop. Ass'n of Lubbock, Tex. v. Goodpasture Computer Serv., Inc.*, 807 F.2d 1256, 1262 (5th Cir. 1987); *Syncrcom Technology, Inc. v. University Computing Co.*, 462 F. Supp. 1003, 1012-14 (N.D. Tex. 1978). Some courts, however, have ruled that nonliteral elements of computer programs receive protection under copyright law. See, for example, *Whelan*, 797 F.2d at 1237-38; *Apple Computer*, 714 F.2d at 1249 (holding that a computer program is a literary work whether it is in the form of source code or object code and thus is protected from unauthorized copying).

the leading case involving the copyright protection of nonliteral aspects of computer programs was *Whelan Associates, Inc. v. Jaslow Dental Laboratory, Inc.*..[114] In that case, the plaintiff alleged that the defendant used the nonliteral structure of the original program te develop a competitive version.[115] In distinguishing between the ideas and the expressions in the program, the court held that a utilitarian work's purpose or function is the work's idea and that everything not necessary to the purpose or function is expression.[116] The Third Circuit concluded that copyright protection extended beyond the code itself to the structure, organization, and sequence of the program,[117] thus including nontangible aspects of the program in the protectable elements.

Courts have had mixed responses to the *Whelan* decision.[118] Recently, courts have been more careful to examine the second computer program in detail and grant less protection to copyrighted programs. In *Computer Associates International, Inc. v. Altai, Inc.*,[119] the Second Circuit rejected the *Whelan* holding outright and established a three-step procedure to identify the nonliteral elements and evaluate the substantial similarity of these elements.[120] Under this tost, a court must break down the allegedly infringed program into its constituent structural parts,[121] strip away nonprotectable material,[122] and compare the creative expression that remains with the structure of the

---

114. 797 F.2d 1222 (3d Cir. 1986).

115. Id. at 1225-27.

116. Id. at 1236.

117. Id. at 1248.

118. A few courts have adopted the reasoning in *Whelan*. See *Bull HN Info. Sys., Inc. v. American Express Bank Ltd.*, 1990 U.S. Dist. LEXIS 3819, *8 (S.D.N.Y. 1990); *Broderbund Software, Inc. v. Unison World, Inc.*, 648 F. Supp. 1127, 1133 (N.D.Cal. 1986). Other courts soundly have rejected it. See *Computer Assoc.*, 982 F.2d at 705-06; *Plains Cotton Coop.*, 807 F.2d at 1262; *Syncrcom Technology*, 462 F. Supp. at 1014. The academic community consistently has criticized the overly broad conception established in *Whelan*. See note 43. See also J.H Reichman, *Electronic Information Tools—The Outer Edge of World Intellectual Property Law*, 17 U. Dayton L. Rev. 797, 814-16 (1992); Reichman, 42 Vand. L. Rev. at 696-98 (cited in note 3); Peter S. Menell, *An Analysis of the Scope of Copyright Protection for Application Programs*, 41 Stan. L. Rev. 1045, 1074, 1082 (1989); Marc T. Kretschmer, Note, *Copyright Protection for Software Architecture: Just Say No!*, 1988 Colum. Bus. L. Rev. 823, 837-39; Peter G. Spivack, Comment, *Does Form Follow Function? The Idea/Exprcssion Dichotomy in Copyright Protection of Computer Software*, 35 U.C.L.A. L. Rev. 723, 747-55 (1988); Thomas M. Gage, Note, Whelan Associates v. Jaslow Dental Laboratories: *Copyright Protection for Computer Software Structure—What's the Purpose?*, 1987 Wis. L. Rev. 859, 860-61.

119. 982 F.2d 693 (2d Cir. 1992).

120. Id. at 706. See also *Brown Bag Software v. Symatec Corp.*, 960 F.2d 1465, 1475-77 (9th Cir. 1992) (holding that analytic dissection of software is necessary when determining substantial similarity).

121. *Computer Assoc.*, 932 F.2d at 706-07.

122. Id. at 707-10.

allegedly infringing program.[123] Furthermore, two additional courts recently rejected *Whelan* outright. These cases emphasize the growing need to re-evaluate the appropriate paradigm for protecting computer software.

A. Sega v. Accolade

Sega Enterprises, Ltd. ("Sega") manufactures and markets a video entertainment system (the Genesis console) and video game cartridges.[124] 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.[125] Although Accolade is not a licensee of Sega, Accolade pursued the possibility of entering into a Sega licensing agreement but later abandoned the effort because the license required that Sega be the exclusive manufacturer of all Accolade's games although Accolade was already selling its games for many other systems.[126]

To make its software compatible with Sega hardware, Accolade used a two-step process.[127] First, it reverse engineered the software in Sega's video cartridges to identify the interface specifications.[128] 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.[129] 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[130] of code that were necessary for Accolade's own code to work properly in a Genesis console.[131] Accolade's engineers

---

123. Id. at 710-11.

124. *Sega Enter. Ltd. v. Accolade, Inc.*, 977 F.2d 1510, 1514 (9th Cir. 1992).

125. Id.

126. Id. Accolade had been selling its games for several different hardware systems for years.

127. Id.

123. Id. Accolade purchased a Genesis console and several game cartridgos and used a decompiler, which was wired into the console, to electrically generato printouts of the sourco code. Id. at 1514-15. Accolade software engineers then experimented with similar portions of the source code from the different games to find the needed intorface specifications. Id. at 1515. Next, they wrote a manual that described the functional requirements of Genesis-compatible software. Id.

129. Id.

130. A byte generally consists of eight ones and zeros that represent a single keyboard character.

131. *Sega*, 977 F.2d at 1515-16. In fact, Accolade created a standard header file (a file containing general information used by many different parts of the program) to be included in all software created for the Genesis console. Id. at 1516. This header file contained 20 to 25 bytes in comparison to the total code size, which ranges from 500,000 to 1,500,000 bytes. Id. Although the

testified that they copied only these several bytes from Sega's code into the final version of their program.[132] 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.[133] 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."[134] Most games that Accolade developed for the Genesis console previously had been developed for other computer systems and had been created independently.[135] In packaging these cartridges, Accolade clearly stated that it was not associated with Sega, indicated that the cartridges were compatible with the Genesis console, and separately identified the Sega and Accolade trademarks.[136]

Sega filed suit in the fall of 1991 against Accolade, alleging trademark infringement, false designation,[137] and copyright infringement.[138] Accolade filed a counterclaim alleging false designation.[139] Both parties sought preliminary injunctions on their respective claims;[140] the district court granted Sega's motion.[141] That court held that Accolade could not assert a functionality defense to the alleged trademark infringement because the TMSS initialization code used by Accolade was not functional.[142] In addition, the court rejected Accolade's fair use defense to the copyright infringement claim because the decompilation was for commercial purposes and negatively affected the market for Sega's game cartridges.[143]

---

case does not reflect this fact, it is safe to assume that Sega's games were approximately the same length. Thus, at most, Accolade used 0.005% of Sega's code when it developed games for the Genesis cousole.

132. Id. at 1516.

133. Id. at 1515.

134. Id.

135. Id. at 1515-16.

136. Id. at 1516. The front of the Accolade box stated that the game cartridge was "for use with Sega Genesis and Mega Drive Systems." Id. The back of the box stated that "Sega and Genesis are registered trademarks of Sega Enterprises, Ltd. Game 1991 Accolade, Inc. All rights reserved. Ballistic is a trademark of Accolade, Inc. Accolade, Inc. is not associated with Sega Enterprises, Ltd. All product and corporate names are trademarks and registered trademarks of their respective owners." Id.

137. Id. The console display indicated that Sega was the game's producer when a player inserted Accolade's game cartridges inte the Genesis console. Id.

138. Id.

139. Id.

140. Id.

141. Id.

142. Id. at 1517. The court based this decision on the testimony of a Sega employee who stated that the programs could be modified in such a way that the Sega message would not appear on the console at startup. Id. at 1516-17.

143. Id. at 1517. Accolade then filed a motion for a stay of preliminary injunction pending appeal and subsequently filed a motion for an emergency stay in the United States Court of

The Ninth Circuit reversed the district court and held that when someone lacks any alternative access to unprotected elements of an original work and has a legitimate reason for accessing those elements, that person's disassembly of a copyrighted work is fair use as a matter of law.[144] The court further held that when no other access to a computer is available to competitors, use of initialization code that triggers a display of the hardware manufacturer's trademark by the competitor does not violate the Lanham Trademark Act.[145]

The Ninth Circuit addressed Accolade's four arguments regarding copyright issues and quickly dismissed the first three.[146] The court, however, accepted Accolade's final argument that disassembling the programs to gain access to the unprotected functional ideas of the program was necessary and, thus, was fair use as embodied in Section 107 of the Copyright Act.[147] The court agreed that Accolade had a legitimate reason for gaining access to the unprotected elements of Sega's software: to determine how te make its software compatible with the Genesis console.[148] The court allowed disassembly for this purpose.[149]

Appeals for the Ninth Circuit. Id. The Ninth Circuit temporarily stayed the injunction and later dissolved it. Id.

144. Id. at 1514.

145. Id.

146. First, the court held that intermediate copying of computer programs infringed Sega's exclusive rights granted by § 106 of the Copyright Act even when the final version of Accolade's software was not substantially similar te the original work. Id. at 1519. The court based its holding on its decision in *Walker v. University Books, Inc.*, 602 F.2d 859 (9th Cir. 1979), stating that "the fact that an allegedly infringing copy of a protected work may itself be only an inchoate representation of some final product te be marketed commercially does not in itself negate the possibility of infringement." *Sega*, 977 F.2d at 1518 (quoting *Walker*, 602 F.2d at 864). The court explained that the plain language of the Copyright Act that grants the copyright holder exclusive rights to "prepare derivative works based upon the copyrighted work" and to "authorize the preparation of copies" dictated its decision in both cases. *Sega*, 977 F.2d at 1518 (citing 17 U.S.C. § 106(1)-(2)).

The court next rejected Accolade's argument that it was lawful, under § 102(b) of the Copyright Act, to disassemble the object code in order te examine and understand the functional aspects of the programs because functional elements are exempted from copyright protection. *Sega*, 977 F.2d at 1519-20.

The court also held that Accolade's assertion that § 117 of the Copyright Act permitted it to disassemble the object code exceeded the drafters' contemplation of that section. Id. at 1520. The court noted that Congress enactsd § 117 on the recommendation of the National Commissien on New Technological Uses of Copyrighted Works (CONTU), which stated that because "the placoment of any copyrighted work into a computer is the preparatien of a copy [since the program is loaded into the computer's memory], the law should provide that persons in rightful possession of copies of programs be able to use them freely without fear of exposure te copyright liability." Id. (queting CONTU, Final Report 13 (1979)) (brackets in original).

147. *Sega*, 977 F.2d at 1520. The court reasoned that because the need to disassemble code generally arises only when the functions are not visible to the user and no alternative means are available to access these ideas, courts should examine computer object code on a case-by-case basis under the fair use analysis. Id.

148. Id.

149. Id.

In determining that Accolade's copying of Sega's object code was fair use, the court considered the Section 107 factors.[150] First, the court found that although Accolade used the copied work for commercial purposes, Accolade overcame the presumption of unfairness because the use was legitimate and essentially nonexploitative, with minimal commercial significance.[151] Next, the court found that Sega would suffer only minor economic loss because potential customers probably would buy multiple Genesis-compatible video games, and Accolade's games differed from Sega's.[152] Applying the nature test, the court emphasized that computer programs are utilitarian articles that contain many functional elements in addition to protected expression.[153] Because humans cannot read object code and because of the code's large size, however, copies of the object code are necessary to translate it into source code.[154] Of the four factors to be considered in determining whether Accolade's copying of Sega's object code was fair use, the court found that only the third statutory factor, the amount of copying, weighed in Sega's favor.[155] The court gave this factor little weight, however, and stated that the amount of copying did not preclude a fair use finding.[156] Thus, the court found that the factors weighed heavily in favor of Accolade and concluded that, as a matter of law, disassembly in this situation was a fair use of a copyrighted work.[157]

B. Atari v. Nintendo[158]

The facts of *Atari* are similar to those in *Sega*, except that Atari lacked clean hands.[159] Nintendo makes a home video system

---

150. Id. at 1521-22.
151. Id. at 1522-23.
152. Id. at 1523-24.
153. Id. at 1524-26.
154. Id. at 1525-26.
155. Id. at 1526.
156. Id. The court cited *Sony Corp. v. Universal City Studios, Inc.*, 464 U.S. 417, 449-50 (1984), and *Hustler Magazine, Inc. v. Moral Majority, Inc.*, 796 F.2d 1148, 1155 (9th Cir. 1986), for support.
157. *Sega*, 977 F.2d at 1527-28.
158. *Atari Games Corp. v. Nintendo of Am., Inc.*, 975 F.2d 832 (Fed. Cir. 1992). For articles discussing this case, see Harold C. Moore, Comment, Atari v. Nintendo: *Super Mario Uses "Expressive" Security Feature to "Lock" Out the Competition*, 18 Rutgers Computer & Tech. L. J. 919 (1992); Susan Kostal, *Federal Circuit Backs Reverse Engineering*, Los Angeles Daily J. 1 (Sept. 16, 1992). See also Arthur R. Miller, *Copyright Protection for Computer Programs, Databases, and Computer-Generated Works: Is Anything New Since CONTU?*, 106 Harv. L. Rev. 977, 1015-22 (1993).
159. *Atari*, 975 F.2d at 836.

that consists of a monitor, a console, and controls.[160] To start the game, a user inserts a game cartridge into the console.[161] The system contains a software lock that controls access to the console by rejecting cartridges that do not contain the "key."[162] After unsuccessful attempts to access Nintendo's security system, Atari became a licensee.[163] This agreement strictly limited Atari's access to Nintendo's technology as well as the number of different games that Atari could produce each year.[164] Atari, under false pretenses, then acquired a reproduction of the security program from the Copyright Office.[165] Atari used this information to correct errors in object code obtained through microscopic examination of Nintendo's microprocessors.[166] Using a different programming language and microprocessor, Atari subsequently developed a program that enabled access to the console by generating siguals that functionally were indistinguishable from Nintendo's security system.[167] The district court found that Atari infringed Nintendo's copyright.[168] Atari appealed and asserted copyright misuse as a defense to the infringement.[169] The Federal Circuit, which had jurisdiction because the action included a patont infringement claim, heard the appeal.[170]

The Federal Circuit affirmed the district court by applying the law of the Ninth Circuit[171] and held that Atari was not permittod to raise the fair use defense because it had obtained an unauthorized copy of the work.[172] Additionally, the court rejected the fair use argument because Atari copied, into its final version of software, parts of the expression that were unnecessary for functionality and compatibility.[173] The Federal Circuit, however, went further than the Ninth Circuit did in *Sega*. The Federal Circuit judges, who hear all the appealed patent cases, probably realized that to reject reverse engiueering of computer programs or to deny access in all cases would

160. Id. at 835.
161. Id.
162. Id. at 836.
163. Id.
164. Id.
165. Id. Atari's lawyer requested a reproduction of Nintendo's program from the Copyright Office by falsely alleging that his client was a defendant in a current case. The attorney agreed to use the program only in connection with litigation. Id.
166. Id.
167. Id.
168. Id. at 837.
169. Id.
170. Id.
171. Id.
172. Id. at 843.
173. Id. at 845.

be to grant disguised patents to software.[174] Thus, they held that reverse engineering to discern the unprotectable ideas of a computer program is fair use when a programmer legitimately obtains the intermediate copy.[175] The court further explained that reproduction of protectable expression is necessary to determine the bounds of the work's protectod information.[176] The tost that the court used for copyright infringement purposes, however, was not fair use of reverse engineering but substantial similarity.[177] In addition, the court recognized that the copyright paradigm cannot be used to exclude competiters from the market.[178]

V. REVERSE ENGINEERING OF SOFTWARE FOR THE PURPOSES OF INTEROPERABILITY AND ANALYSIS

For years the courts and commentators in the field of intollectual property law have struggled to determine the appropriate scope of protection for computer programs. The circuits are split on this issue, and commentators are at odds with each other. Nonetheless, a clear trend has developed recently: the courts are moving from a maxi-protectionist view to a radical, nonprotectionist one.

The majority of courts and commentators currently agree that computer programs should receive only thin protection.[179] The question is whether thin protection does or should allow reverse engineering for the purpose of interoperability or to create better programs when the final software product is not substantially similar to the original program. Fearing widespread piracy, many who advocate only thin protection for software are slow to accept reverse engineering of computer programs even under these circumstances.[180]

---

174. See id. at 842.

175. Id. at 843. This holding is much stronger and clearer than the Ninth Circuit's holding in *Sega.*

176. Id.

177. Id. at 844-45.

178. Id. at 845-46. The misuse defense is well recognized in the patent paradigm and generally arises when a patent owner attempts to secure exclusive rights beyond the scope of patent law and contrary to public policy. Thomas F. Smegal, Jr., *Misuse Defense Gains in Federal Court*, Nat'l L. J. 18, 18 (Feb. 15, 1993). See also *Atari*, 975 F.2d at 845-46.

179. See *Feist Publications, Inc. v. Rural Tel. Serv. Co.*, 111 S. Ct. 1282, 1289-91 (1991); *Kepncr-Tregoe, Inc. v. Carabio*, 203 U.S.P.Q. 124, 130 (E.D. Mich. 1979). See generally note 52 and text accompanying notes 39-76.

180. See notes 205-06 and accompanying toxt.

A. Reverse Engineering: Why Bother?

Over the last three decades, the number of computers in use has increased exponentially. Literally hundreds of computer hardware and software manufacturers operate in the United States. Many computers are expected to communicate, or interoperate, with other computers. In addition, computer users need and expect to use a variety of different software programs on their hardware. These capabilities require compatibility among different brands of hardware and software, and implementation of these expectations creates the need for standardization.[181] Hardware and software vendors who want to discourage competitors often are unwilling to provide specific information about their product's interface specifications, protecols, and underlying ideas. This information, however, is necessary te achieve the compatibility that is required for two systems te communicate. When this necessary information is incomplete, inaccurate, or otherwise unavailable, reverse engineering, which almost always requires intermediate copying, often is the only way to access the unprotected information.[182] Thus, reverse engineering has become an industry norm of competition in software production because of the need for standardization and compatibility.[183] Although reverse engineering enjoys widespread use as a fundamental step in normal software development,[184] it often is the last resort because it is so costly, time-consuming, and tedious.[185]

Courts usually assess the legitimacy of reverse engineering by examining the purpose of the activity. For example, the Section 117 exemption of the Copyright Act permits execution or copying of the

---

181. For articles recognizing the need for compatibility and interoperability, see generally Peter A. Wald, Michael K. Plimack, and Harold M. Freiman, *Standards for Interoperability and the Copyright Protection of Computer Programs*, in 365 *Intellectual Property/Antitrust 1993* 891 (P.L.I., 1993); Joseph Farrell, *Standardization and Intellectual Property*, 30 Jurimetrics J. 35, 43-44 (1989); Jacobs, 30 Jurimetrics J. at 92-100 (cited in note 28) (discussing different situations requiring compatibility and the need for development of independently compatible software). See also Timothy S. Teter, Note, *Merger and the Machines: An Analysis of the Pro-Compatibility Trend in Computer Software Copyright Cases*, 45 Stan. L. Rev. 1061 (1993).

182. Johnson-Laird, 5 Software L. J. at 345 (cited in note 20).

183. See Farrell, 30 Jurimetrics J. at 35 (cited in note 181).

184. Johnson-Laird, 5 Software L. J. at 345-46 (cited in note 20); Raskind, 70 Minn. L. Rev. at 387 n.10 (cited in note 69).

185. See text accompanying notes 31-32 for an explanation of why reverse engineering has these attributes. Decompilation of software usually does not save money or time in software development. In fact, reverse engineering an entire program often is more costly and time-consuming than designing a program from scratch. Johnson-Laird, 5 Software L. J. at 348 (cited in note 20). The fact that reverse engineering usually is a last resort is evidenced by the difficulty Atari had in deciphering Nintendo's original copyrighted program. See *Atari*, 975 F.2d at 836.

program for archival purposes or to use the program.[186] In contrast, a competitor who copies a program in the process of decompilation and subsequently markets a product that is substantially similar to the original program is infringing. The borderline case occurs when a programmer copies a computer program for the purpose of analyzing the program's unprotoctod ideas or to determine its interface specifications in an effort to produce a competitive, compatible, noninfringing product. Fortunately, courts are beginning to distinguish between the unfair exploitation of an author's copyrighted work and copying that work to make independent creative expression possible.[187]

B. Interoperability

1. The Need for Interoperable Systems

Interface specifications are essential to interoperability and compatibility. For two computors to communicate, they must speak the same language by using exactly the same interface specifications and protocols.[188] Any inaccuracy will cause the interaction between the two computers to fail, probably at the most inopportune time.[189] In other words, nothing short of total compatibility will be effective; the point of connection to the operating systom (the low-level software that makes a computer operato) must be totally compatible.[190]

Compatibility benefits consumers in numerous ways. Users avoid waste because they need not purchase and learn new software, and they may access additional computing power via modems and networks. Compatibility also provides users with more options for their systems, thus broadening or adding to the competitive market.[191] The use of networks and expensive computer systoms and consumer

186. 17 U.S.C. § 117. See note 71 for the text of § 117.

187. See *Sega*, 977 F.2d at 1522.

188. See notes 27 and 28 for an explanation of interfaces and protocols.

189. Johnson-Laird, 5 Software L. J. at 340 (cited in note 20).

190. Even if a program is 99% compatible, at some point it will fail. Such a failure can be a disastor depending on the context.

191. Jacebs, 30 Jurimetrics J. at 92 (cited in note 28). See also Steering Committee, *Intellectual Property Issues in Software* at 66-67, 71-72 (cited in note 73) (discussing public demand and need for compatibility and interoperability of software). Compatibility of software increases its value because of network externalities, or benefits that accrue as a result of being part of a large system. This cempatibility usually helps small cempanies. Id. at 71-72.

market pressure demand compatibility.[192] Often, manufacturers achieve compatibility through either a formal or a de facto standardization.[193] Standardization has a number of benefits. It facilitates the creation and use of networks,[194] permits the use of multi-user and multi-application files,[195] saves money,[196] and encourages competition and innovation.[197] Compatibility is more economical and much easier once industry standards are established.[198] The difficulty arises when no generally known standard exists, forcing competitors in the software industry to study existing programs' interface specifications and communication protocols. The ultimato goal of this examination is not to free ride but to produce a unique and marketable product.

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.[199]

---

192. Standardization means making products similar enough so that they are compatible. Farrell, 30 Jurimetrics J. at 36 (cited in note 181). The industry values standardization and efficiency. Formal standardization usually is more prevalent when compatibility is important. Id. at 45. De facto standards are inevitable and reflect consumers' desires for compatibility. For a discussion of standards, see Steering Committee, *Intellectual Property Issues in Software* at 67-68, 70-72 (cited in note 73).

193. De facto standards are those standards that are defined by computer hardware and software manufacturers. See Jacobs, 30 Jurimetrics J. at 99 (cited in note 28). De facto standards often result from consumer demands and preferences.

194. See Farrell, 30 Jurimetrics J. at 36 (cited in note 181).

195. See id.

196. See id. (stating that standardization saves costs of retraining); Jacobs, 30 Jurimetrics J. at 100 (cited in note 28) (asserting that standardization benefits consumers by not requiring another investment in new software, training, and data input and storage).

197. See Farrell, 30 Jurimetrics J. at 36 (cited in note 181). Standardization and compatibility encourage the development of new products and promote innovation without reinvention of the wheel. See id.; Jacobs, 30 Jurimetrics J. at 100 (cited in note 28). Standardization alse can discourage iimovation, however, and can result in a loss of variety. Farrell, 30 Jurimetrics J. at 36.

198. See Farrell, 30 Jurimetrics J. at 36.

199. See text accompanying notes 26-35.

2. Legal Acceptance of the Need for Interoperability

Despite the compelling justifications for adopting a doctrine that permits reverse engineering of computer programs for the purpose of interoperability, an immense amount of controversy surrounds the need to make an intermediate copy in the process of reverse engineering. To make a program interoperate with another, a programmer only needs to use some of the functional elements of the original computer program, yet no United States court directly or explicitly has embraced the disassembly of computer programs for the purpose of interoperability. Several courts have rejected the argument that program developers make intermediate copies for the sole purpose of determining interface specifications of a program in order to achieve compatibility with another system.[200] Other courts have disagreed.[201] Although the Federal Circuit rejected the

---

200. For example, the district court in *Sega* rejected the interoperability argument: "The grant of a copyright is intended to motivato creative activity by the provision of a special reward, and eventually allows the public total access te the products of their genius after the limited period of exclusive control has expired. . . . Without the economic incentive to create which copyright protection provides, this incentive and the advantages it creatos for society may well be lost." *Sega Enter. Ltd. v. Accolade, Inc.*, 785 F. Supp. 1392, 1400 (N.D. Cal.), aff'd, 977 F.2d 1510 (9th Cir. 1992) (citing *West Publishing Co. v. Mead Data Central, Inc.*, 616 F. Supp. 1571, 1582 (D. Minn. 1985)). In *Lotus Dev. Corp. v. Paperback Software Int'l*, 740 F. Supp. 37 (D. Mass. 1990), the court stated that "[t]he desire to achieve 'compatibility' . . . cannot override the rights of authors to a limited monopoly in the expression in their intellectual 'work.'" Id. at 69. The *Apple Computer* court held that compatibility does not limit the copyright protection embodied in § 102(b) of the Copyright Act. *Apple Computer, Inc. v. Franklin Computer Corp.*, 714 F.2d 1240, 1253 (3d Cir. 1983) (stating that "Franklin may wish to achieve total compatibility . . . but this is a commercial and competitive objective which does not enter inte the somewhat metaphysical issue of whether particular ideas and expressions have merged."). Another court found that the idea-expression dichotemy could be invoked only if it was "the only and essential means of accomplishing" compatibility. *Apple Computer, Inc. v. Formula Int'l, Inc.*, 725 F.2d 521, 525 (9th Cir. 1984) (quoting CONTU, Final Report 20 (1979)).

Several cases that feund copyright infringement of computer programs are distinguishable from cases dealing with tho interoperability issues addressed in this Noto. In *Atari Games Corp. v. Nintendo of Am., Inc.*, the infringer copiod more than necessary to achieve compatibility by going "beyond any legitimate understanding of the 'functionality' of the program." 18 U.S.P.Q. 1935, 1937 (N.D. Cal. 1991), aff'd, 975 F.2d 832 (Fed. Cir. 1992). In *SAS Inst.*, SAS obtained S & H's source code under false pretenses and wrongfully used the information after the licensing agreement terminated. *SAS Inst., Inc. v. S & H Computer Sys., Inc.*, 605 F. Supp. 816, 820, 828-29 (M.D. Tenn. 1985). In addition, the defendant's final code cepied the original copyrighted code 44 times. Id. at 822, 830.

201. See, for example, *Plains Cotton Coop. v. Goodpasture Computer Serv., Inc.*, 807 F.2d 1256, 1262 (5th Cir. 1987) (declining te follow *Whelan* because the similarities in the computer pregrams were dictated by externalities of the cotton market); *NEC Corp. v. Intel Corp.*, 10 U.S.P.Q.2d 1177, 1183-89 (N.D. Cal. 1989) (recognizing that it was permissible to disassemble and use information in an effort to develop a separato microcode). Although one conceivably could distinguish *Plains Cotton* and *NEC* by external censtraints (for example, the cotton market or IBM personal computer architecture), user expectations are external restraints on many other software programs such as the hardware constraint—the Genesis console—in the *Sega* case. See also *E.F. Johnson Co. v. Uniden Corp. of Am.*, 623 F. Supp. 1485, 1502 (D. Minn. 1985) (holding

contention that Atari merely incorporated unnecessary instructions to guarantee future compatibility,[202] the court approved the necessary reproduction of protectable expression.[203] With its holding, the Federal Circuit indirectly agreed that reverse engineering is permissible for the purpose of immediate interoperability. Thus, the trend appears to be in the more progressive direction of allowing reverse engineering for the purpose of independently developing interoperable software.[204]

Some commentaters fear that a doctrine allowing reverse engineering to achieve interoperabilty will be abused and result in industry-wide piracy.[205] Reverse engineering of software for piracy purposes clearly is unfair and inappropriate, and courts should continue to prohibit it. When a programmer both independently creates a computer program and reverse engineers another system solely to discern the interface specifications and protocols required to make the two programs compatible, however, the free-rider problem does not arise.[206]

Rejecting the maxi-protectionist views adhered to by United States courts, the European Economic Community recently adoptod the Council Directive on the Legal Protection of Computer Programs ("the Directive") to protect computer programs under the copyright paradigm.[207] This legislation allows decompilation, but only when it is absolutely necessary to achieve interoperability of an independently

---

that it was permissible for the defendant essentially to reverse engineer the plaintiff's software to determine specifications necessary to write functionally compatible software, but not to incorporate parts of the plaintiff's software that were not essential to functional capability).

202. *Atari*, 975 F.2d at 845.

203. Id. at 843, 844.

204. The Ninth Circuit came the closest te this trend in *Sega*. The court accepted the need for interoperability in ene fell swoop and said little more about the issue. See *Sega*, 977 F.2d at 1524-26. See also *Atari*, 975 F.2d at 842-43. Leading copyright scholars believe that studying and copying software for the purpose of developing a compatible computer program is fair use even if the resulting programs compete. *LaST Frontier Report*, 30 Jurimetrics J. at 25 (cited in note 52). See generally Brief Amicus Curiae 1, reprinted in 33 Jurimetrics J. 147 (cited in note 1).

205. Clapes, Lynch, and Steinberg, 34 U.C.L.A. L. Rev. at 1499-1510, 1575-76 (cited in note 11). Software piracy is widespread because it is quick and easy, and the cost of creating software is much greater than the cost of copying it. Samuelson, 1984 Duke L. J. at 689-91 (cited in note 20). The prevalence of piracy and the great cost to the producers demonstrate the difficulties of using cepyright law to protect software. Id. at 692. See also Dennis S. Karjala, *Copyright, Computer Software, and the New Protectionism*, 28 Jurimetrics J. 33 (1987) (advocating only as much protection of software as is necessary to prevent piracy).

206. See Jacobs, 30 Jurimetrics J. at 102 (cited in note 28). Nete that the *Whelan* decision does not address independent development of software. See *Whelan Assoc., Inc. v. Jaslow Dental Lab., Inc.*, 797 F.2d 1222 (3d Cir. 1986).

207. Council Directive 91/250, 1991 O.J. (L 122) 42. The EEC adopted the final versien on May 14, 1991. For a description of the EC Directive and its history, see generally Bridget Czarnota and Rebert J. Hart, *Legal Protection of Computer Programs in Europe—A Guide to the EC Directive* (Butterworth, 1991). The Directive generally seeks to award broad protection to computer programs. Id. at 8.

created computer program.[208] Although the Directive is a step in the right direction, even a broad interpretation may hinder competition because of the narrow circumstances under which reverse engineering is permitted.[209] Thus, only the larger software producers who already dominate the market benefit. The closest thing to the Directive's interoperability exception in the Copyright Act is the fair use doctrine of Section 107, which United States courts have not interpreted as directly including reverse engineering for interoperability.[210] The United States, however, would benefit nationally and internationally by adopting a clear and definite doctrine that generally permits reverse engineering for the purpose of independently developing interoperable software that is not substantially similar to the original copyrighted program.

C. Permitting Analysis Versus Granting Patent-Like Protection

When a programmer wants to identify the unprotectable, underlying ideas and functions of a computor program, that programmer must reverse engineer the program for analysis purposes. If reverse engineering of computer programs is not allowed for this purpose, then, under the rubric of copyright, the rights in a work's underlying ideas will exceed those rights conferred by copyright law.[211]

---

208. Council Directive 91/250, Art. 6(1), 1991 O.J. (L 122) at 45.

209. The three prerequisites to qualify for the decompilation exception are that (1) the copier must have a right to use the program under analysis, (2) the needed information must be unavailable, and (3) one must limit decempilation to the part required for interoperability. Id. The first requisito is reasonable, but the other two are overly restrictive because they prohibit access to underlying ideas if interface specifications and protocols are available. Furthermore, it usually is impractical to limit reverse engineering to the intoroperable parts because the competiter typically cannot distinguish the parts of the program from each other. See generally Linda G. Morrison, Noto, *The EC Directive on the Legal Protection of Computer Programs: Does It Leave Room for Reverse Engineering Beyond the Need for Interoperability?*, 25 Vand. J. Transnat'l L. 293 (1992).

The debates that arose prior to the final adoption of the Directive focused on the reverse engineering exception to the exclusive right of reproduction of computer pregrams. Thus, the Directive's position on reverse engineering actually is a compromise that only allows decompilation for compatibility purposes.

210. The Directive does not have an exception resembling the fair use doctrine. In fact, Article Six of the Directive likely will be more strict than the fair use doctrine in some cases, although the Directive can be varied by contract. Czarneta and Hart, *Legal Protection of Computer Programs* at 126 (cited in note 207).

211. Corrected Opposition of Accolade at 13, reprinted in 15 Computer L. Rep. at 601 (cited in note 72). In its brief for the Ninth Circuit, Accolade explained how the District Court ruling misuses patent law: "[W]hat Sega is seeking here is patont law protection with a patont [by] trying to use the copyright and trademark laws to monopolize the idea of how the Genesis works." Appellant's Opening Brief 9, *Sega Enter. Ltd. v. Accolade, Inc.*, 977 F.2d 1510 (9th Cir. 1992) (No. 92-15655), reprinted in 15 Computer L. Rep. 976, 984 (1992). See also text accompanying note 174. Patent law grants the right to prohibit reverse engineering to patent holders, hut this right exists for less time than a work is protected under cepyright.

The codification of the idea-expression dichotomy in Section 102 of the Copyright Act indicates that Congress intended that only a patent can or should prohibit the use of an idea.[212] Patented products receive more protection than copyrighted works, but the patent requirements of novelty, usefulness, and non-obviousness[213] are significantly harder to meet than the copyright requirements. In addition, patents are granted for a much shorter duration.[214]

Restrictions on the reverse engineering of computer programs create a virtual long-term patent under the Copyright Act but without the strict requirements of patents. If no aspect of the program may be copied, then no one will be able to access its nonvisible, underlying functions and ideas. Essentially, the program will receive the same protection as a patent, but for much longer than the patent laws provide. Furthermore, it generally is inappropriate to afford computer programs patent-like protection because the development of computer programs does not require the same level of investment as industrial products.[215] In summary, patent-like protection for software provides too much protection. This kind of system circumvents the policies behind the patent and copyright laws that encourage public access to information.

The courts prior to *Sega* and *Atari* essentially afforded computer programs patent-like protection under copyright law. Recently, some courts have approved of copying and decompiling computer programs to determine and analyze the underlying unprotected ideas when these processes are the only practical means of obtaining that information.[216] By holding that reverse engineering of a computer

---

212. *Bonito Boats, Inc. v. Thunder Craft Boats, Inc.*, 489 U.S. 141, 168 (1989) (quoting *Lear, Inc. v. Adkins*, 395 U.S. 653, 656 (1969)) (stating that "[b]y offering patent-like protection for ideas deemed unprotected under the present federal scheme, the Florida statute conflicts with the 'strong federal policy favoring free competition in ideas which do not merit patent protection'").

213. 35 U.S.C. §§ 101-103 (1988).

214. Patents exist for 17 years. 35 U.S.C. § 154 (1988). A copyright extends for the life of the owner plus 28, 50, or 56 years. 17 U.S.C. §§ 301-305 (1988 & Supp. 1992).

215. Although the development of software can be very expensive and time-consuming, the development of a patentable industrial product often requires additional time, money, and intellect: someone must generate an idea for a product, make and test prototypes, purchase and alter the machinery and equipment required to mass produce the product, and purchase the raw material to make the product.

216. See, for example, *NEC Corp. v. Intel Corp.*, 10 U.S.P.Q.2d 1177, 1189 (N.D. Cal. 1989); *E.F. Johnson Co. v. Uniden Corp. of Am.*, 623 F. Supp. 1485, 1501 n.17 (D. Minn. 1985). The *E.F. Johnson* court specifically stated:

> The mere fact that defendant's engineers dumped, flow charted, and analyzed plaintiff's code does not, in and of itself, establish pirating. As both parties' witnesses admitted, dumping and analyzing competitors' codes is a standard practice in the industry. Had Uniden contented itself with surveying the general outline of the EFJ program, thereafter converting the scheme into detailed code through its own imagination, creativity, and independent thought, a claim of infringement would not have arisen.

program for the purpose of determining its underlying ideas is fair use, both the Federal Circuit and the Ninth Circuit implicitly have agreed that reverse engineering for the purpose of analysis is permissible under certain circumstances.[217] These courts have provided an alternative analysis that other courts should follow in the future.

D. Public Policy Concerns

The United States copyright laws encourage people to build on the ideas and information contained in a work.[218] It is not the copyright system but the patent laws that protect processes and methods of operation.[219] An author may not enjoy patent protection by putting a method, process, or idea into an unintelligible form, seeking a copyright in the work, and then asserting infringement when someone else attempts to analyze those unprotected elements.[220] Disallowing reverse engineering of computer programs for legitimate purposes circumvents the main public policy or goal of the copyright laws by hindering growth and access to unprotectable ideas.[221] Instead, this overprotection clearly gives much higher priority to the copyright laws' secondary purpose of rewarding authors.[222]

If reverse engineering is disallowed, then software developers who wish to create legitimate, competitive, compatible products will face huge obstacles. For example, although a computer program may be written in a number of different ways, software often is dictated functionally such that most or all other versions are less efficient.[223]

E.F. Johnson, 623 F. Supp. at 1501 n.17. But see *Walt Disney Prod. v. Filmation Assoc.*, 628 F. Supp. 871, 875-77 (C.D. Cal. 1986) (holding that copies made in motion picture production could infringe the reproduction rights of the copyright owner of the original work). The district court in *Atari* rejected the argument that analysis of a computer program to accomplish compatibility with a competiter's hardware is legal. *Atari Games Corp. v. Nintendo of Am., Inc.*, 18 U.S.P.Q.2d 1935, 1939 (N.D. Cal. 1991), aff'd, 975 F.2d 832 (Fed. Cir. 1992) (holding that intermediate copying of copyrighted programs was an infringement). Even though the Federal Circuit affirmed the district court holding, the circuit court disagreed with the district court on this particular point. See *Atari*, 975 F.2d at 842-44.

217. "This fair use did not give Atari more than the right to understand the [security] program and to distinguish the protected from the unprotected elements.... Any copying beyond that necessary te understand the [security] program was infringement." *Atari*, 975 F.2d at 844.

218. See *Feist Publications v. Rural Tel. Serv. Co., Inc.*, 111 S. Ct. 1282, 1290 (1991).

219. See *Bonito Boats, Inc. v. Thunder Craft Boats, Inc.*, 489 U.S. 141 (1989).

220. *Atari*, 975 F.2d at 842.

221. Register Report at 5 (cited in note 1). See also H.R. Rep. No. 94-1476 at 47-50 (cited in note 1).

222. See text accompanying note 1.

223. External technological constraints rather than personal expression frequently dictate the independent creation aspects and market value of functional works. Reichman, 43 Stan. L. Rev. at 953-54 (cited in note 52).

Yet the market demands fast, efficient software. In addition, one efficient method may become the standard. Thus, if the copyright given to computer programs is maxi-protectionist, then the first copyright owner effectively receives a monopoly for that particular system. This scheme produces two negative results. First, it leaves no room for competition because an inefficient product will have little market value and could lead to a proliferation of inefficient systems. In the end, consumers will suffer the most. Second, if courts disallow disassembly of computer programs, competitors probably will be unable to achieve full compatibility with other software and hardware. Those who copyright their software will defeat others' attempts to create computer programs that are compatible with their system. This system eventually would lead to mini-monopolies.

This possibility introduces another important issue: whether hardware manufacturers may control the software products that end users may utilize with their particular hardware.[224] By forbidding reverse engineering for the purpose of interoperability, consumers and competitors alike will not have legal access to the ideas and functions contained in most software. If a hardware or software vendor successfully restricts access to its interface specifications, then competiters will be unable to offer compatible software. Consumers will have to purchase all software for that vendor's system at that vendor's price. The Copyright Act does not envision this kind of monopoly.

Some commentators have argued that permitting decompilation would leave the software copyright owner with no lead time.[225] This concern is valid only if reverse engineering is allowed for piracy. Furthermore, although reverse engineering may shorten lead time, it does not eliminate it altogether because the process of decompilation itself is so time-consuming.[226] In many cases, the underlying ideas and functions rarely can be discerned absent either disclosure by the copyright owner or the ability to decompile programs for legitimate purposes because the guesswork required to obtain this information would take too long to be cost effective.

Unfortunately, some traditional features of copyright inhibit the achievement of compatibility in software.[227] Additionally, strong intellectual property protection tends to retard formal

224. See Stern, 15 Computer L. Rep. at 266 (cited in note 74). Unfortunately, neither the district court nor the circuit court in *Sega* directly addressed this issue.

225. See generally Miller, 106 Harv. L. Rev. at 978 (cited in note 158).

226. See Part II.B.

227. See Farrell, 30 Jurimetrics J. at 35 (cited in note 181).

standardization,[228] which otherwise could provide many benefits to consumers.

E. Identifying the Appropriate Paradigm for Software

Another important question is whether computer programs should continue to receive protection under copyright law or under a new sui generis law like the one enacted for semiconductor chip technology.[229] Clearly, patent law is not the best paradigm in which to protect computer programs.[230] Although the Patent Office grants

---

228. Strong intellectual property protection tends te increase a company's vested interest. Id. at 44. The effects on informal standardization are unknown. Id. But see Clapes, 15 Computer L. Rep. at 271-72 (cited in note 57).

229. Semiconductor Chip Protection Act of 1984, 17 U.S.C. §§ 901 et seq. (1988 & Supp. 1992) (SCPA).

230. See text accompanying notes 211-14. Patent law requires extensive disclosure in return for a 17-year monopoly. Samuelson, 70 Minn. L. Rev. at 512-13 (cited in note 50). Neither the Constitution nor the Copyright Act indicates that disclosure to the public is a goal or prerequisito for federal copyright protection. In fact, disclosure is not even an integral element of copyright protection policies. Leo J. Raskind, *The Uncertain Case for Special Legislation Protecting Computer Software*, 47 U. Pitt. L. Rev. 1131, 1140 (1986). See also Samuelson, 1984 Duke L. J. at 711 (cited in note 20). Disclosure of ideas embodied in a copyrighted work is not a quid pro quo for copyright protection. Brief for Amici Curiae of Computer and Business Equipment Manufacturers Association 15, *Sega Enter. Ltd. v. Accolade, Inc.*, 977 F.2d 1510 (9th Cir. 1992) (No 92-15655), reprinted in 15 Computer L. Rep. 1043, 1054 (1992). Disclosure of computer programs is required only in an infringement action.

Registration of a computer program only requires the deposit of the first and last 25 pages of source code and less than that if trade secrets are involved. 37 C.F.R. § 202.20(c)(2)(vii) (1992). The statute exempts registration of programs published in only machine-readable form from mandatery deposit requirements. 17 U.S.C. § 407 (1988). As a resnlt of the Copyright Act of 1976, the concept of a public right of access is at odds with the autematic copyright protection that is provided for unpublished worlis from the moment of fixation. See 17 U.S.C. § 302(a).

One commentater has argued that use of the computer program by the public satisfies the accoss need. See William F. Patry, *The Fair Use Privilege in Copyright Law* 401 (B.N.A., 1985). Patry argues that "[t]he public's need for access to the copyrighted work is fully satisfied by the cepyright owner's marketing of the original. A cempetiter who reverse engineers a copyrighted computer program is not at all interested in increasing that access; te the contrary, his only purpose is to get the public te purchase his work rather than the original thereby eliminating the market for the original." Id. However, the denial of accoss to ideas in cemputer programs by prohibiting reverse engineering, which by definition involves intermediato copying in the software arena and is necessary to read and understand those ideas, violates the federal policy of copyright law to protect expression but not ideas.

As previously discussed, the copyright owner of a computer program effectively recoives a patent to the object code because no one can accoss the underlying ideas embedded in the program. Furthermore, this refusal to disclose contravenes the purpose of the law: "[T]o promote the Progress of Science and useful Arts." U.S. Const. Art. I, § 8, cl. 8. See also *Feist*, 111 S. Ct. at 1290:

> To this end, cepyright assures authors the right to their original expression, but encourages others to build freely upon the ideas and information conveyed by a work. This principle . . . applies to all works of authorship. As applied to a factual compilation, assuming the absenco of original written expression, only the compiler's selection and arrangement may be protected; the raw facts may be cepied at will. This resnlt is neither unfair nor unfortunate. It is the means by which copyright advances the progress of science and art.

patents for a limited number of computer programs, most programs do not meet the requisite novelty, non-obviousness, or other standards that are required to get a patent, because most programs usually contain only incremental differences from previous programs. Several scholars have explained that the danger of patent protection for software is that it provides too much protection for too few computer programs.[231] Until recently, however, the copyright paradigm has given too much protection for too many software designs.[232] Certainly, the recent developments of *Sega, Atari,* and *Computer Associates* support the need for at least some change in the law regarding the protection of computer programs.

1. Copyright Protoction

Many courts have justified copying those parts of a computor program that contain only the unprotectod elements.[233] The holdings in *Atari* and *Sega* indicate that at least some modern courts are willing to use the fair use defense for reverse engineering under certain circumstances, such as for the purpose of developing a separate but noninfringing program. Although the Supreme Court could affirm these decisions and make them law throughout the circuits, a Congressional mandate would provide more definite and permanent guidelines. To clarify the requirements and application of the doctrine under copyright law, therefore, Congress should amend the Copyright Act as it did in 1980, if computer programs are to remain in the copyright regime. Until recently, most scholars have adhered to this view.[234] Given all the backlog and gridlock in Congress, modification of the current law may pass with greater ease than the sui generis laws that commentators have proposed te date.[235]

Courts and commentators have set forth additional reasons for not creating sui generis protection for computor programs.[236] First, Congress has expanded copyright law numerous times to include new

---

231. Reichman, 42 Vand. L. Rev. at 698 (cited in note 3); John A. Kidwell, *Software and Semiconductors: Why Are We Confused?*, 70 Minn. L. Rev. 533, 544-49 (1985).

232. Reichman, 42 Vand. L. Rev. at 698 (cited in note 3).

233. See generally notes 73, 78, 99, and 110.

234. See, for example, Raskind, 47 U. Pitt. L. Rev. at 1175-82 (cited in note 230) (finding that in 1986, prior to the *Whelan* decision, basic issues of software protection were being decided effectively by the courts). See also note 242 for commentaters who have rejected the sui generis approach to protecting software.

235. See note 242; Raskind, 47 U. Pitt. L. Rev. at 1175-82 (cited in note 230).

236. The House Report for the SCPA presents similar reasons for protecting semiconducter chips under copyright rather than a sui generis law. Semiconductor Chip Protection Act of 1984, H.R. Rep. No. 98-781, 98th Cong., 2d Sess. 5-11 (1984).

forms of expression and already has integrated computer programs into its fold. Next, because Congress should promote certainty and stability, the enactment of a new law would create an initial upheaval. Copyright law is well-established and gives guidance to copyright owners and users regarding their respective rights. Additionally, computer programs are more certain te obtain international protection if they are protected in the United States under the established copyright law. Furthermore, protecting software under sui generis laws would introduce more complexity into intellectual property law. Therefore, retaining protection of computer programs in copyright law also is more economical. However, although maintaining protection for computer programs under the copyright paradigm appears to be more expedient, it may cause more serious problems in the future.

2. Sui Generis Protection

The *Sega*, *Atari*, and *Altai* decisions represent a shift from the traditional application of full protection for computer programs toward utilization of the fair use defense under certain circumstances. To accomplish this change, the courts in each of these cases relaxed or applied the fair use factors to computer programs. Perhaps the lack of uniformity in applying the fair use doctrine is as strong an indicator as any that computer programs do not fit comfortably under the patent or copyright laws.[237] For example, in the software context, the third fair use factor, which asks the courts to examine the amount and substantiality of the portion of the copyrighted work used, generally will weigh against the subsequent developer who almost always must make an intermediate copy of the computer program to extract unprotected ideas.[238] Courts have altered the fair use defense in an effort to apply it to software; these alterations may be inappropriate for other works protected under the Copyright Act. If this trend continues, courts may erode the fair use doctrine further.

The multiple forms of protection sought for computer programs indicate that traditional copyright protection is inadequate for software.[239] Software producers often attempt to obtain additional, differ-

---

237. See generally Reichman, 17 U. Dayton L. Rev. at 797 (cited in note 118).

238. See Part III.C.3.

239. Samuelson, 70 Minn. L. Rev. at 514-19 (cited in note 50). Patent laws in combination with copyright laws, trade secret laws in combination with copyright laws, and patent, copyright, and trade secret laws all in combination work to protect software. Id. See also *Hubco Data Prod. Corp. v. Management Assistance, Inc.*, 219 U.S.P.Q. 450, 455-56 (D. Idaho 1983) (holding that

ent sets of exclusive rights through licensing agreements.[240] Although Section 117 of the Copyright Act allows computer program users to adapt a computer program if necessary to use or archive it, this narrow exception does not permit reverse engineering for the purpose of analysis or achieving interoperability.

Computer programs are essentially utilitarian works in which creative expression only decreases efficiency. In this respect, they closely resemble semiconductor chips, which have been removed from the copyright law with a sui generis scheme, the Semiconductor Chip Protection Act of 1984 (SCPA).[241] Several scholars have argued that computer programs are actually a distinctive kind of intellectual property that warrant a new law that would acknowledge these differences from traditional literary works.[242] Computer programs are too machine-like to fit within copyright law and too much like a writing to fit within patent law.[243] Giving copyright protection to computer programs disrupts the social bargain equilibrium of both patent and copyright laws.[244] A sui generis law for computer programs would balance successfully public and private interests to reflect the economics of software protection instead of the general economic principles underlying copyright and patent laws.[245] In addition, a sui generis law would eliminate the need to distort copyright law further to accommodate computer programs and similar technologies.

---

reverse engineering is permissible under trade secret law but that copyright infringement occurred as a result of copying a program in the process of studying its content).

240. Samuelson, 70 Minn. L. Rev. at 519 (cited in note 50).

241. 17 U.S.C. §§ 901 et seq. (1988 & Supp. 1992).

242. See generally Pamela Samuelson, *Some New Kinds of Authorship Made Possible by Computers and Some Intellectual Property Questions They Raise*, 53 U. Pitt. L. Rev. 685 (1992); Davidson, 47 U. Pitt. L. Rev. at 1080-100 (cited in note 1) (advocating a sui generis system of protection for all types of software that resembles the Uniform Commercial Code with general principles and default provisions rather than complex laws and regulations); Samuelson, 70 Minn. L. Rev. at 507-28 (cited in note 50); Samuelson, 1984 Duke L. J. at 762-69 (cited in note 20); Pamela Samuelson and Rebert J. Glushko, *Intellectual Property Rights for Digital Library and Hypertext Publishing Systems*, 6 Harv. J. L. & Tech. 237 (1993). For a list of other authorities and commentators who support a sui generis scheme for software, see Samuelson, 70 Minn. L. Rev. at 507 n.184. But see Morton David Goldberg and John F. Burleigh, *Copyright Protection for Computer Programe: Is the Sky Falling?*, 17 AIPLA Q. J. 294 (1989) (rejecting the sui generis approach and explaining that case law has applied traditional copyright principles te computer programs adequately); Clapes, Lynch, and Steinberg, 34 U.C.L.A. L. Rev. at 1575-77 (cited in note 11) (rejecting the need for a sui generis law).

243. Pamela Samuelson, *Survey on the Patent/Copyright Interface for Computer Programs*, 17 AIPLA Q. J. 173, 283 (1989). Computer programs are not well suited to traditional copyright protection because the important part of the program is the structure rather than the details that garnish the work.

244. Samuelson, 70 Minn. L. Rev. at 513 (cited in note 50).

245. Id. at 501-31.

The strongest argument for adopting a sui generis law for computer programs is the success of the SCPA.[246] The SCPA demonstrates the need for a sui generis law when a newly developed technology does not fit into patent or copyright law.[247] Legislative history indicates that Congress passed the SCPA to protect semiconductor chips because they are a utilitarian product that the Copyright Act could not protect appropriately.[248]

The SCPA designates reverse engineering as noninfringing copying of protected semiconductor chip products in certain situations.[249] This statutory provision indicates that Congress has accepted reverse engineering as a standard industry practice for the purpose of developing new products.[250] Apparently, the Copyright Office also hesitated to advocato the fair use doctrine to the semiconductor industry for several reasons: the Copyright Act does not extend to utilitarian works; copyright protection is not effective because the fair use defense cannot accommodate reverse engineering, an industry practice; and the Copyright Office feared the distortion of the fair use doctrine through efforts to accommodato reverse engineering.[251] The SCPA and its history, however, do not address the fair use defense for disassembly of object code.[252]

Several advocates have proposed that Congress reconsider the copyright protection of machine-readable computer programs.[253] Instead of extending copyright protection te software, Congress could adopt a sui generis approach for protecting computor programs in machine-readable form.[254] This new law should distinguish between piracy and legitimate reverse engineering for the purpose of analysis

---

246. 17 U.S.C. §§ 901 et seq. The SCPA is a marked departure from traditional copyright law based on the nature of the products involved. Congress did indicate its intent not te increase or decrease the scope of copyright law with the passage of SCPA. H.R. Rep. No. 98-781 at 28 (cited in note 236).

247. Samuelson, 70 Minn. L. Rev. at 514 (cited in note 50). Congress created a sui generis law combining elements of patent, copyright, unfair competition, and trade secret laws to protect a new form of technology that did not fit neatly into any existing laws.

248. H. R. Rep. No. 98-781 at 8-10 (cited in note 236).

249. 17 U.S.C. § 906.

250. Raskind, 70 Minn. L. Rev. at 385 (cited in noto 69).

251. Id. at 392-93.

252. Professor Raskind analogizes the reverse engineering provision of the SCPA te the fair use doctrine of the Copyright Act. Id. at 389. By contrast, Professor Stern believes that the legislative history of the SCPA indicates that Congress did not intend reverse engineering to be considered fair use under copyright law. Stern, 15 Computer L. Rep. at 266 (cited in note 74). Professor Storn refers te a version of the bill that was not enacted, howover, and fails to recognize that the SCPA does not immunize all copying against liability for infringement.

253. Samuelson, 70 Minn. L. Rev. at 471 (citod in noto 50).

254. Id. at 507-31. See also Samuelson, 1984 Duko L. J. at 692-705 (citod in noto 20) (explaining and criticizing the CONTU Report).

or achieving interoperability.[255] It also should adopt a slightly different test for infringement because the traditional test[256] does not adequately identify infringement of software.[257] Additionally, if Congress adopted sui generis protection for computer programs, it could establish a different set of exclusive rights that are tailored to this subject matter in order to balance more equitably the rights of software producers, software users, and the public.[258] Indeed, it would be easier to grant different exclusive rights through a sui generis law than to modify existing legal standards.[259] It also would be appropriate to grant the exclusive rights for a shorter duration because the commercial lifetime of a computer program usually is shorter than other literary works,[260] therefore making earlier entry into the public domain desirable.[261]

VI. CONCLUSION

By upholding reverse engineering of computer programs under the defense of the fair use doctrine for the purpose of analysis of nonprotected expression of ideas, two federal circuits have departed from

---

255. For example, it may be appropriate to allow reverse engineering of object code only if the object code is unavailable. In addition, courts could permit reverse engineering only for the purpose of analysis or to achieve interoperability requirements (determining interface specifications and protocols) with competing hardware or software systems. In addition to requiring a legitimate purpose, the law could require a substantial showing of effort and investment.

256. The traditional test for infringement is whether a lay observer would regard the two works as substantially similar enough to conclude that expression has been appropriated. 3 *Nimmer on Copyright* § 13.03[E][1][a] at 13-75 to -76 (cited in note 1). The problem with this approach is that most lay observers are computer illiterate and thus may find substantial similarity where it does not exist or fail to find it where it does exist. Thus, Nimmer advocates a design or pattern theory to determine infringement of computer programs: a "comprehensive nonliteral similarity" constitutes copyright infringement. Id. § 13.03[A][1] at 13-25 to -26. Recently he has suggested a "successive filtering" test, which is more along the lines of the *Computer Assoc. Int'l* test. Id. § 13.03[F] at 13-78.26 to -78.30.

257. Samuelson, 70 Minn. L. Rev. at 525 (cited in note 50) (stating that the substantial similarity portion of the test is appropriate).

258. Id. at 519-20. Professor Samuelson recommends the following exclusive rights: the right for software developers to make and distribute copies of the protected program (currently done in copyright law, patent law, and the SCPA) and a limited right to produce derivative works that do not include all material generated through the use of software (currently only the copyright paradigm provides this right in 17 U.S.C. § 103). Samuelson, 70 Minn. L. Rev. at 520-23. She also recommends limits on patent-type exclusive rights by allowing the use of protected programs as tools to create additional works and allowing the modification of protected programs to the extent necessary to fulfill the intended purpose of the program. Id. at 524.

259. This observation is especially true if Congress decides not to include a provision giving exclusive rights to prepare derivative works.

260. Hardware and software innovations occur so quickly that software rapidly becomes outdated. Even a shorter duration of protection would give the owner enough opportunity to recover investment expenses.

261. If the duration of protection is too long, the software likely will be useless to the public.

the broad infringement requirement for software set forth by the Third Circuit in *Whelan*, which provided a high degree of protection to the copyright owners of computer programs. This shift to a less protectionist view requires courts to examine not only the general purpose and structure of a computer program but also the details of the software itself. This approach places a higher burden of proof on the plaintiff to show infringement, which arguably affects the author's right of control. Nonetheless, it certainly upholds the primary purpose of copyright law: to provide additional and higher quality products to the public.

Because the circuits are split on whether reverse engineering of software for analysis or interoperability purposes constitutes infringement, the Supreme Court may grant certiorari in the near future. The Court's guidance is necessary given the national market of most hardware and software vendors. It is unlikely that the Supreme Court would permit reverse engineering when the final product is substantially similar to the original work. Should the Supreme Court grant certiori to one of these cases, the Justices should address carefully both the analysis and interoperability issues of reverse engineering.

With the rapidly growing technological changes in the computer industry, the law surrounding these issues will continue to develop. If these laws remain inconsistent, they will cause great confusion to consumers and likely will have a negative effect on the market for such products.[262] Thus, Congress either should amend the Copyright Act by adding a clear provision regarding the permissibility of reverse engineering for the purposes of analysis and achieving interoperability or create a tailored sui generis law that would be useful for many years to come. Not only is there a need for uniformity within the United States, but the United States also needs to consider carefully the movements of other industrialized countries with regard to the limits on exclusive copyright protection for computer software.

*S. Carran Daughtrey**

262. It is not unusual to find articles in computor journals written by legally trained experts explaining the rights that competitors have. See, for example, Pamela Samuelson, *Computer Programs and Copyright's Fair Use Doctrine*, 36 Communications of the ACM 19 (Sept. 1993).

* The Author wishes to thank Kathryn Spann for her assistance and support, Professor J. H. Reichman for his excellent classes, and the editors of this Note for their eternal patience and good nature!