Denver Journal of International Law & Policy

Volume 24
Number 1 *Fall*

Article 6

January 1995

Minimizing Reverse Engineering: Sample Language for Dual United States and European Union Software Licenses

John T. Soma

Sharon K. Black-Keefer

Alexander R. Smith

Minimizing Reverse Engineering: Sample Language for Dual United States and European Union Software Licenses

Keywords

European Union, Reverse Engineering, Software, States, Technology, Computer Programs, Computers, Copyright, Industrial and Intellectual Property, Intellectual Property Law

Minimizing Reverse Engineering: Sample Language For Dual United States and European Union Software Licenses

JOHN T. SOMA*
SHARON K. BLACK-KEEFER**
ALEXANDER R. SMITH***

I. INTRODUCTION

It is standard practice in the computer software industry for programmers to acquire a copyrighted copy of a competitor's code and to dissect it to discover its underlying ideas.[1] This process, known as "reverse engineering," has been defined by the United States Supreme Court as "starting with the known product and working backward to divine the process which aided in its development or manufacture."[2] In the European Union (EU), the analogous concept of "decompilation" is defined by the European Software Copyright Directive as "the reproduction of the code and translation of its form . . . to obtain the information necessary to achieve the interoperability of an independently created computer program with other programs."[3]

Reverse engineering has been held to be legal in both the U.S. and the EU. The U.S. Supreme Court, following the concept established by the framers of the Constitution of balancing the public availability of ideas with the need for protection of a creator's product, found reverse engineering to be legal under limited circumstances.[4] It is permitted where it provides access to the underlying ideas in unpatented or otherwise unprotected items in the public domain.[5] Lower courts have also

---

* Professor, University of Denver, College of Law. Denver, Colorado.

** Legal Consultant to the State of Colorado assigned to rewrite Colorado's telecommunications law; J.D., University of Denver 1995; M.S. in Telecommunications, University of Colorado 1972.

*** Research Associate, University of Denver, J.D. Candidate 1996.

1. Programmers do this by disassembling the program. That is "dumping" (or copying the software code into computer memory) and analyzing it with flow charts and line by line code comparisons. *See generally*, E.F. Johnson Co. v. Uniden Corp. of Am., 623 F. Supp. 1485, 1501-02 n.17 (D. Minn. 1985).

2. Kewanee Oil Co. v. Bicron Corp., 416 U.S. 470, 476 (1974).

3. The European Software Directive at art. 6, as implemented by each country [hereinafter *Software Directive*]. "The European Software Directive," as of September 24, 1993, compiled by Clifford Chance, London England.

4. Bonito Boats, Inc. v. Thunder Craft Boats, Inc., 489 U.S. 141 (1989).

5. Bonito Boats, 489 U.S. at 157, *citing* Sears Roebuck & Co. v. Stiffel Co., 376 U.S. 225 (1964); Compco Corp. v. Day-Brite Lighting, Inc., 376 U.S. 234 (1964).

held[6] that "based on the policies underlying the Copyright Act," reverse engineering of a copyrighted software program is, as a matter of law, a fair use of the work if the person seeking the understanding has a legitimate reason for doing so, and reverse engineering provides the only means of access to the unprotected elements of the work. In the EU, this "only means of access" language is pertinent to the decompilation concept specifically authorized in the European Software Directive, since the Directive only allows "indispensable" decompilation.[7]

What is neither clear from U.S. decisions nor from the European Software Directive is what happens if reverse engineering/decompilation is not the only means of access to the unprotected information desired and, therefore, not indispensable. What are the rules, for example, when a developer of original software voluntarily provides relevant portions or the entire source code to parties desiring it? Can reverse engineering be prohibited or limited by a restrictive clause in a license, sales contract, or shrinkwrap agreement?

For software developers, answers to these questions are of critical importance because of the significant economic costs associated with the possibility of unlimited reverse engineering — a concept which developers tend to view as "legal theft" of their work. Each program usually represents significant investment of personal, creative effort, time, and often millions of dollars in research and development. Obsolescence of their product by a competitor's reverse engineered product, before they can recover their investment, discourages further development.

Reverse engineering can have a world-wide impact, as nearly fifty percent of all copyrighted computer products created in the U.S. are exported.[8] Patent, copyright, trade secret, and other intellectual property protection afforded to those products in the United States do not necessarily follow the products beyond our borders.[9] Even if such protection is arranged in the recipient country, the protection afforded may not be deemed sufficient, since, it is believed, "reverse engineering" is legal to a greater extent in the EU and elsewhere.[10] Markets established by distributors and value-added resellers (VARs) can thus be undermined when ex-employees or other nationals reverse engineer an imported product and then develop a competitive "local" product that appeals to the nationalistic concerns of the recipient country and may be less expensive.

---

6. Sega Enterprises v. Accolade, Inc., 977 F.2d 1510, 1518 (9th Cir. 1992)

7. *Software Directive*, *supra* note 3, at art. 6.

8. Discerned from author's review of the annual reports of the top 50 computer companies in the U.S.

9. *See* specific provisions in the Paris Convention for the Protection of Industrial Property (1883, as amended until 1979), the Patent Cooperation Treaty of June 19, 1970, the Universal Copyright Convention (as revised at Paris on 24 July 1971), and the Berne Convention for the Protection of Literary and Artistic Works (Paris Act, 24 July 1971). *See also* Trade-Related Aspects of Intellectual Property Rights (TRIPs) in the General Agreement on Tariffs and Trade as implemented by Congress December 1, 1994.

10. *Software Directive*, *supra* note 3.

To protect themselves, software developers continuously seek ways to a) legally prohibit or limit the reverse engineering of their copyrighted product in the U.S. and abroad; b) protect their intellectual property claims; and c) place themselves in the best possible legal position to argue a claim if their product is reverse engineered and an infringing software product is created.

The purpose of this article is to identify the actions that software developers can take both within the United States and the European Union markets to accomplish these three goals. The article then provides suggestions for software licenses that, in an infringement case, will likely withstand both the tests of reverse engineering established by current U.S. statutes and case law and by the European Software Directive.

Section II of the paper gives an overview of the history of software protection and the significant current statutes and case law controlling reverse engineering in the United States. Section III considers the legality of limiting reverse engineering in contracts, licenses, and agreements in the United States. Section IV compares and contrasts this legal environment to that advanced by the European Software Directive. Section V provides suggestions to implement maximum legal protection for software programs that provides companies with as close as possible to a "single contract" for all sales, domestically and within the European Union. Section VI concludes the article.

II. U.S. SOFTWARE PROTECTION — PAST AND PRESENT

In the U.S., creative efforts are protected by intellectual property law,[11] contract law, and, to some extent, tort law. Each provides specific aspects or levels of protection that should be considered by software developers to achieve their three goals in protecting their computer programs.

A. *Pre-Copyright Software Protection — Contract and Trade Secret Law*

During the early years of the computer industry, from the first machines through the 1970's, computer systems were large, requiring full air-conditioned rooms, and the software necessary to run them usually accompanied the hardware. Software was thus acquired in face-to-face, individually negotiated transactions.[12] The contracts covering the sales and lease of the hardware customarily included confidentiality clauses that protected the ideas, logic, and engineering embodied in the software.[13]

---

11. This article will concentrate primarily on the applicability of patent, copyright, and trade secret law. Trademark and misappropriation law may also be applicable and should be explored.

12. Ronald L. Johnston & Allen R. Grogan, *Trade Secret Protection for Mass Distributed Software*, 4 THE COMPUTER LAWYER (forthcoming 1994-95).

13. *Id.* at 4.

Contract and trade secret laws, therefore, were the principal forms of software protection during these years. No one really knew what protection patent, copyright, or other intellectual property law might provide for software.[14] As the industry has matured, so has the relevance of many areas of intellectual property, especially as they relate to reverse engineering.

B. *Patents*

For programmers and other inventors, patents offer the highest level of protection from reverse engineering or any other form of copying or use. Patents guarantee the holder the exclusive right to make, use, and sell the invention for 17 years, essentially providing a monopoly on the product during that time.

Section 101 of the U.S. Patent Act[15] made patents available to "[w]hoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, . . . subject to the conditions and requirements of this title."[16] As with all patents, the pivotal requirements were that the discovery be novel, non-obvious, and useful.[17]

To be "novel," all elements of the item must not have been previously patented, described in a printed publication, known by others, or used by others before it was invented by the patent applicant anywhere in the world.[18] Since early computer programs were mainly mathematical operations, they did not pass this statutory subject matter test. In addition, nearly all of the electronic and mathematical elements that caused the programs to work were "known or used by others in this country . . . before the invention"[19] of computers and were well described in printed publications prior to the development of any specific software program. Most computers and software were thus found to be unpatentable.

An invention must also be "non-obvious." This requires that the operation of an invention, product, or process not be readily deducible to persons skilled in the craft or significantly reflected in the "prior art" of the industry. Again, computers and their related software generally could not pass this test.

Finally, the invention must be "useful," to qualify for a patent. While computers and software passed this third test, they remained unpatentable since they could not pass all three tests. Additionally, the cost and time delays to receive a patent were found to be unrealistic for the dynamic, rapidly changing nature of the software industry where items

14. *Id.* at 5.
15. 35 U.S.C. §§ 1-376 (1984).
16. *Id.* at § 101.
17. *Id.*
18. *Id.* at §§ 102 and 103.
19. *Id.* at § 102(a).

risked becoming obsolete before a patent was granted.

Beginning in 1969, the Court of Customs and Patent Appeals (C.C.P.A.) began to reverse the Patent Office and to compel the issuance of software patents. A number of patents on computer programs were then readily issued from 1969 to 1972. But in 1972, the U.S. Supreme Court reversed the C.C.P.A. and blocked the issuance of a patent on a software program.[20] From 1972 to 1981, most computer programs were not patented, including the initial spreadsheets and other significant software developments.[21]

This began changing in the early eighties. In 1980, the C.C.P.A. was renamed the United States Court of Appeals for the Federal Circuit and was given exclusive appellate jurisdiction over all patent-related appeals and attached collateral issues. In 1981, the Supreme Court reconsidered a software patent in *Diamond v. Diehr*,[22] and authorized the grant of a software/hardware patent. The internal processes of many software programs are still not sufficiently "inventive" to be patentable, and the time/cost concerns remain; however, where a program can be patented, the significant protection provided by the Patent Act makes it clearly the protection of choice for computer software, both in the U.S. and in Europe.[23] It should, therefore, be the first avenue of protection considered by a software developer.

C. *Copyright*

Works that are original, but not necessarily novel or utilitarian, can be protected under copyright law. The Copyright Act of 1976[24] provides certain "exclusive rights" to authors for their unique expression of perhaps otherwise well-known ideas. Thus, while the electronics and mathematics behind software programs are well-known, the unique manner in which they are used by programmers has traditionally made copyright law the best means of protection for software.

Prior to 1980, computer software was not specifically covered by the copyright law. But in 1980, Congress amended the Copyright Act of 1976 to explicitly include computer programs in the definitions section of the copyright law,[25] making it officially available as the preeminent protection for software.[26] Computers had shrunk in size and software was broadly

---

20. Gottschalk v. Benson, 409 U.S. 63 (1972).

21. PETER B. MAGGS ET AL., COMPUTER LAW 185-186 (1992).

22. Diamond v. Diehr, 450 U.S. 175 (1981).

23. Johnston & Grogan, *supra* note 12, at 5.

24. 17 U.S.C. §§ 101-810 (1988 & Supp. IV 1992).

25. 17 U.S.C. § 101 (1988 & Supp. IV 1992).

26. In Apple Computer, Inc. v. Formula Int'l Inc., 725 F.2d 521, 525 (9th Cir. 1984), the Court stated that the statutory language, read together with the 1979 Final Report 1 of the National Commission on New Technological Uses of Copyrighted Works (CONTU REPORT), leads inexorably to the conclusion that the copyright in a computer program extends to the object code version of the program.

distributed on diskettes as part of the personal computer revolution.[27] Since software was no longer acquired in face-to-face transactions, "shrinkwrap" license agreements were the common form of software licensing.[28]

By the time of the 1980 Amendment, Congress had also abolished the "publication" limitation and provided that copyright protection attached the instant the work was fixed in a "tangible medium."[29] While Congress did not provide explicit protection or rights for computer software prior to 1980, it included such protection in the category of "literary works,"[30] one of the seven explicit categories of works of authorship granted copyright protection.[31] Thus, the 1976 Copyright Act granted software the same rights granted to books and other writings.[32] Courts also found some software generated displays to be protected as audiovisual works, another explicit category of the Copyright Act.[33]

With a copyright, a computer software developer obtains at least five exclusive rights related to the software program. The developer may do or authorize any of the following:

(1) make copies of the work;

(2) prepare derivative works based upon the copyrighted work;

(3) distribute copies of the copyrighted work to the public by sale or

---

27. Johnston & Grogan, *supra* note 12, at 5.

28. *Id.*

29. 17 U.S.C. § 102. Section 102 of the Copyright Act of 1976 states that: "Copyright protection subsists . . . in original works of authorship fixed in any tangible medium of expression, now known or later developed, from which they can be perceived, reproduced, or otherwise communicated, either directly or with the aid of a machine or device." *Id.* The "originality" requirement is met if the work required some degree of effort on the part of the author. No judgment about the artistic merit of the work is necessary. In all copyrighted works, what is protected is not the "idea," but rather the specific "expression" of the idea.

Publication, however, remains important today because it affects copyright holder's rights set by the "notice", "registration", and "deposit" requirements of copyrights set forth in Chapter 4 of the U.S. Copyright Act of 1976. *Id.* at §§ 401-412.

"Notice" of the copyright, usually a "c" within a circle, must appear on all copies of the work once it is published or the copyright protection may be lost. "Registration" of a copyright within the U.S. Copyright Office is optional, but is required before an author can sue for copyright infringement. While a copyright owner may register after learning of infringement and then file suit, certain remedies will be limited so that it is always in an author's best interest to register his/her copyright. This is easy to do since, unlike a patent which costs thousands of dollars to obtain, registration of a copyright costs $20.00. Once a work is published, the author must "deposit" two copies in the Library of Congress within three months after publication. *Id.*

30. *Id.*

31. *Id.* Section 102 lists the seven categories of "works of authorship" as: (1) literary works; (2) musical works; (3) dramatic works; (4) pantomimes and choreographic works; (5) pictorial, graphic and sculptural works; (6) motion pictures and other audiovisual works; and (7) sound recordings. *Id.*

32. John T. Soma et al., *Software Interoperability and Reverse Engineering*, 20 RUTGERS COMP. & TECH. L.J. 189, 203 (1994).

33. *See, e.g.*, Williams Elecs., Inc. v. Artic Int'l, Inc., 685 F.2d 870, 874 (3d Cir. 1982).

other transfer of ownership, by rental, by lease, or by lending;

(4) publicly perform literary, musical, dramatic, and choreographic works, pantomimes, and pictorial, graphic, or sculptural works, including the individual images of a motion picture or other audiovisual work; and

(5) display the copyrighted work publicly. [34]

These rights exist for the lifetime of the software developer plus fifty years after death. During that time, the developer may transfer or license the copyright just like any other personal property. Licensing of a copyright allows others to use the work upon a payment of a specific amount of "royalty." To protect the software developers, transfer and license agreements include a statutory right of termination which may be exercised during a five year period beginning at the end of thirty-five years from the date of the grant.[35]

1. Exceptions to the "Rights" of Copyright Owners

Exceptions to these rights exist for certain uses of a work by educational or charitable institutions or libraries, public access, fair use,[36] and limited uses by the software owner.[37] It is not a copyright infringement, for example, for the owner of a copy of a computer program to make one archival copy of the program, or to make adaptions, enhancements, or modifications to a particular copy of the program for use by the owner.[38] The copier, however, may not transfer these adaptions without the authorization of the copyright holder.[39]

Of these exceptions, the "public access" and "fair use" exceptions play crucial parts in the legality and extent of reverse engineering.

a. The "Public Access Exception" to Copyright

In most copyright cases, beginning with *Baker v. Selden*[40] in 1879 and continuing through *Mazer v. Stein*,[41] *Sony Corp. v. Universal City Studies, Inc.*,[42] and *Feist Publications, Inc. v. Rural Tel. Serv. Co.*,[43] courts have followed federal public policy and favored free access by the public to the underlying ideas and functions of a work rather than tighter

34. 17 U.S.C. § 106 (1988 & Supp. IV 1992).

35. *Id.* at §§ 201(d), 202, 204(a), 205(a),(d), and 302(a).

36. *Id.* at § 106.

37. *Id.* at § 117(2).

38. *Id.*

39. *See* Foresight Resources Corp. v. Pfortmiller, 719 F. Supp. 1006, 1010 (D. Kan. 1989) (defendant enjoined from selling enhancements of plaintiff's products to other entities); *Sega*, 977 F.2d at 1520 ("Section 117 does not purport to protect a user who disassembles object code, converts it from assembly into source code, and makes printouts and photocopies of the refined source code version"). *Id.*

40. Baker v. Selden, 101 U.S. 99 (1879).

41. Mazer v. Stein, 347 U.S. 201 (1954).

42. Sony Corp. of Am. v. Universal City Studios, Inc., 464 U.S. 417 (1984).

43. Feist Publications, Inc. v. Rural Tel. Serv. Co., 499 U.S. 340 (1991).

protection of creative expression. This concept is a major rationale for approval of reverse engineering.

In 1992, two important "reverse engineering" cases from the Ninth Circuit, *Atari Games Corp. v. Nintendo of Am., Inc.*[44] and *Sega Enter. Ltd. v. Accolade, Inc.*,[45] were decided at the same time and addressed these four items. In both cases, the defendant companies reverse engineered the plaintiffs' software to develop non-infringing, compatible end products — computer game cartridges that worked on the hardware game consoles of the plaintiff companies.

In *Atari*, the Federal Circuit held that the Copyright Act permits an individual in rightful possession of a copy of a work "to undertake necessary efforts to understand the work's ideas, processes and methods of operation."[46] This permission "appears in the fair use exception"[47] to copyright exclusivity. Citing *Bonito Boats*, the *Atari* court stated that an author cannot restrict access and "acquire patent-like protection by putting an idea, process, or method of operation in an unintelligible format and asserting copyright infringement against those who try to understand that idea, process, or method of operation."[48] The Court explained that to protect "processes or methods of operation, a creator must look to patent laws."[49] The court also added that "the Copyright Act permits an individual in rightful possession of a copy of a work to undertake necessary efforts to understand the work's ideas, processes and methods of operation."[50] The Court thus held that interim copies for reverse engineering are a fair use exception to copyright infringement.[51]

Like *Atari*, the *Sega* Court concluded that "where disassembly is the only way to gain access to the ideas and functional elements embodied in a copyrighted computer program and where there is a legitimate reason for seeking such access, disassembly [reverse engineering] is a fair use of the copyrighted work as a matter of law."[52]

Two recent infringement cases indicate the extreme importance of the *Atari* and *Sega* decisions. In *MAI v. Peak*,[53] the Court of Appeals for the Ninth Circuit ruled that an independent service organization (ISO) made an infringing copy of a copyrighted computer program when the program was transferred to the computer's random access memory (RAM) for maintenance diagnosis. The Court of Appeals further held

---

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

45. Sega Enter. Ltd. v. Accolade, Inc., 785 F. Supp. 1392 (N.D. Cal. 1992), *aff'd in part, rev'd in part*, 977 F.2d 1510 (9th Cir. 1992).

46. *Atari*, 975 F.2d at 842 (stating that the purpose and policy of the copyright law is to encourage "authors to share their creative works with society.") *Id.*

47. *Id.*

48. *Id.*

49. *Id.*

50. *Id.*

51. *Atari*, 975 F.2d at 843.

52. *Sega*, 785 F. Supp. at 1527-28.

53. MAI Systems Corp. v. Peak Computer Inc., 991 F.2d 511 (9th Cir. 1993).

that this was not fair use based on the commercial nature of the intended use, the use of the entire copyrighted work, and the substantial market impact of the use. Finally inarguably the most controversial aspect of the entire case, the Court of Appeals, in a footnote, ruled that the licensee of the software was not an "owner" under the Copyright Act and could, therefore, be limited by license provisions as to third party access to the computer program, even for maintenance purposes. This reasoning was substantially followed in *Advanced v. MAI.*[54] Further, the District Court found no illegal tying arrangement between the computer program and maintenance services.

Commentary on these two cases has indicated that they may create an avenue by which the copyright owner of a computer program could create a monopoly for maintenance of that program without violating antitrust laws.[55] Thus, the only commercially feasible way an ISO could compete with the copyright owner may be to design a competing product by means of clean room reverse engineering based on the *Atari* and *Sega* fair use analysis.

b. The "Fair Use Exception" to Copyright

For certain socially-beneficial uses, such as criticism, comment, news reporting, teaching (including multiple copies for classroom use), scholarship, or research,[56] the copying of copyrighted works is considered legal so long as the use does not deprive the copyright owner of appropriate rights and economic rewards. These are known as "fair use" of the copyrighted works and are a major exception to an author's exclusive rights to reproduce a copyrighted work or to create a derivative work.[57]

Software users, in particular, are affected by this exception because they 1) must copy the object code of a copyrighted program into their computer's memory to run the program;[58] 2) may copy the program in order to access its underlying ideas, as permitted by public policy; and 3) may create an interim copy and modify parts of the original copyrighted

54. Advanced Computer Services of Michigan Inc. v. MAI Systems Corp., 845 F.Supp 356 (E.D. Va. 1994).

55. *See, e.g.*, Michael E. Johnson, *The Uncertain Future of Computer Software Users' Rights in the Aftermath of MAI Systems*, 44 DUKE L.J. 327 (1994).

56. 17 U.S.C. § 107 (1988 & Supp. IV 1992).

57. *Id.*

58. One unique aspect of computer software programs is that they are not readable by humans while they are in their "object code" format of 1's and 0's. To be read by humans, the software must be in "source code" or "word" format. This requires copying the targeted software and translating it from object code to source code. While this (act of) copying violates the copyright prohibition against copying, the courts have held that this form of interim copying is a "fair use" exception to copyright infringement. Atari, Sega, Galoob, Foresight, and NEC effectively overturn the interim copy analysis in Hubco and Walt Disney Prods. v. Filmation Assocs. (holding that the Copyright Act prohibits the creation of interim copies).

program to create compatible programs.[59]

Courts have generally ignored these "interim copies" in copyright infringement actions as a necessary part of the computer process. Instead, the courts have focused on a comparison of the end products, as a whole, to determine if the resulting end-product is so similar to the original software program that it "infringes" on the original copyright. The courts have similarly considered the use of the end-product to determine if it qualifies as a "fair use" under the "fair use exception."[60] The key information from these cases for software developers is that they should not focus on restricting reverse engineering of their products but rather should seek to understand the criteria the courts use in evaluating these two areas so that they may comply with them in their licenses and use them to place themselves in the best possible legal position to win if their software is infringed.

(1) Comparison of End-Products - A Two-Part Test

(a) Substantial Similarity

Since unauthorized copying is often difficult to detect or to prove and since "reverse engineered" software programs can result in unique programs that are independent expressions of an idea but can still perform the same tasks or functions as the original program, courts apply the "substantial similarity" standard to evaluate the end-products of reverse engineered programs. This standard is not clearly defined and often depends on the opinion of the trier of fact. Illegal copying, however, unquestionably occurs when 1) the entire structure of the original work is duplicated in some detail; 2) specific portions are copied verbatim; 3) both works contain common errors; or 4) both works contain the original programmer's "marks" — non-functional identifiers incorporated into the code by programmers specifically to discern later copying.

In all reverse engineering cases, the resulting end product must be sufficiently different from the original product so that it is clear that only the "unprotected underlying ideas," not the original programmer's specific expression, were used. Where the "expression" of the same "idea" is sufficiently dissimilar in the two programs, the courts will find no infringement.[61] Where the resulting end-product, however, is too similar to the original copyrighted work, it will be considered an infringement.

---

59. *Id.* at §§ 106, 117.

60. E.F. Johnson Co. v. Uniden Corp. of Am., 623 F. Supp. 1485 (D. Minn. 1985), Midway Mfg. Co. v. Artic Int'l, Inc., 547 F. Supp. 999 (N.D. Ill. 1982), Williams v. Arndt, 626 F. Supp. 571 (D. Mass. 1985), Hubco Data Prods. Corp. v. Management Assistance Inc., 219 U.S.P.Q. (BNA) 450 (D. Idaho 1983), and Telerate Sys., Inc. v. Caro, 689 F. Supp. 221 (S.D.N.Y. 1988). In all of these cases, the court found infringing end products and held against the defendant even though reverse engineering had occurred.

61. NEC Corp. v. Intel Corp., 10 U.S.P.Q.2d (BNA) 1177, 1184 (N.D. Cal. 1989).

(b) Access

Since it is also possible for two programmers to develop very similar software programs, even though they worked independently of one another and had no information from reverse engineering, the courts also require that a second criteria of "access" to the original copyrighted work be proven before copyright infringement is found. Substantially similar programs can exist even if no access is found, such as with "clean rooms,"[62] but if all or part of the works are identical or the degree of similarity is overwhelming, "access" will be presumed. "Intent" is not required for copyright infringement. Illustrations of this test can be seen in *E.F. Johnson,*[63] *Midway,*[64] and *Hubco.*[65]

It should be noted that even user manuals are scrutinized by this test. In *Williams,*[66] the defendant wrote a program for commodities market predictions that implemented a method described in the plaintiff's copyrighted manual.[67] The Court held that the defendant's resulting software was a mere translation of the manual's plain English into computer language and thus violated plaintiff's copyright as a derivative work, and, citing 17 U.S.C. § 101 (1988), held that a 'derivative work' is a work based upon one or more preexisting works, such as a translation. This decision is important for software developers because the manual that accompanies a vendor's software often describes many details of the vendor's software. The manual is thus a frequent source of compatibility information for those evaluating possible software copyright infringement.

(2) Fair Use Evaluation

To determine if a software program's "use" is "fair" as defined by the "fair use exception to copyright," courts use the following four-factor test:

> (a) the purpose and character of the defendant's use, ("intended use"), including whether such use is of a commercial nature or is for nonprofit educational purposes;

---

62. "Clean room" describes the procedure of developing software by a person or group of persons who have no access to the original, protected program but rather writes its own code using only functional specifications, or other unprotected elements, given to him by others from the original program. So long as that person can prove that its work was not contaminated by access to any of the protected elements in the original work, no infringement will be found. *See* Milton R. Wessel, *Introductory Comment on the Arizona State University Last Frontier Conference on Copyright Protection of Computer Software*, 30 JURIMETRICS J. 1, 23 (1989).

63. E.F. Johnson, 623 F. Supp. at 1485.

64. Midway Mfg. Co., 547 F. Supp. at 1014.

65. Hubco, 219 U.S.P.Q. (BNA) at 452. *But cf.* Foresight Resources Corp. v. Pfortmiller, 719 F. Supp. 1006 (D. Kan 1989).

66. Williams, 626 F. Supp. at 579.

67. Soma, *supra* note 32, at 207, *citing* Williams, 626 F. Supp. at 574.

(b) the nature of the copyrighted work;
(c) the amount and substantiality of the portion used in relation to the copyrighted work as a whole; and
(d) the effect the defendant's use will have on the potential market for or value of the copyrighted work.[68]

If, based on an analysis of these four factors, the "use" of the copy is determined to be "fair," then the "fair use" exception applies, and the copy does not infringe the copyright.[69] If this exception is not established, infringement of the copyrighted work is found. For this reason, "fair use" is a defense to claims of copyright infringement. The *Sega* court also ruled that all four factors, not just a few, must always be considered to determine whether a particular use constitutes a fair use.[70]

2. Remedies for Infringement

If infringement is found, possible remedies include 1) injunctive relief, in which the copyright owner may enjoin the infringing use or sales;[71] 2) monetary relief, in which a copyright owner may recover damages, attorneys fees plus the defendant's profits or "statutory damages" from between $500 and $20,000 as directed by the court;[72] 3) impoundment, seizure and destruction of all unauthorized copies;[73] and 4) criminal sanctions against the infringer, including imprisonment if the motive for the copying was commercial advantage or financial gain.[74]

3. Summary

In deciding copyright cases, the courts have established a set of parameters that define the legality of reverse engineering to 1) access; 2) intended use; 3) use only of the underlying ideas of the original product, not the unique expression provided by the author or programmer,[75] determined by "the amount and substantiality" of the original product used in

---

68. 17 U.S.C. § 107 (1988 & Supp. IV 1992).

69. *Id.* at § 501(a). "Infringement" is defined as a violation of any of the exclusive rights of the copyright owner.

70. Sega, 977 F.2d at 1521-27. The Sega analysis of these four factors should be noted by software developers. Further, the market impact analysis of Harper & Row, 471 U.S. at 562, *Sony*, 464 U.S. at 451, and Lewis Galoob Toys, Inc. v. Nintendo of Am., Inc., 964 F.2d 965, 969 (9th Cir. 1992), *cert. denied*, 113 S.Ct. 1582 (1993), should be explored.

71. 17 U.S.C. § 502(a) (1988).

72. *Id.* at § 504, 505.

73. *Id.* at § 503.

74. 17 U.S.C.A. § 506 (1984).

75. The Copyright Act does not extend to the ideas underlying a work or to the functional or factual aspects of the work. 17 U.S.C. § 102(b). "In determining whether a product feature is functional, a court may consider a number of factors, including — but not limited to — 'the availability of alternative designs: and whether a particular design results from comparatively simple or cheap method of manufacture.' " Sega, 785 F.Supp at 1531, *citing* Clamp Mfg. Co. v. Enco Mfg. Co., Inc., 870 F.2d 512, 516 (9th Cir.), *cert. denied*, 493 U.S. 872 (1989).

the reverse engineered product; 4) interim copies are, generally, considered legal "fair use" so long as they are not distributed or sold[76], and the resulting end product is not "substantially similar" to the original product;[77] and 5) the resulting software does not negatively impact the potential market of the original product. These are the key elements that software developers, interested in protecting their work product and positioning themselves for the strongest possible claim in an infringement cases, must include in their licensing agreements.

D. *Trade Secret*

A product such as a computer program may not be sufficiently unique or inventive enough to qualify for a patent. It may, however, still have economic value, so long as its underlying secrets are not generally known by others. If so, the product may qualify for protection under the Uniform Trade Secrets Act (UTSA).[78] The UTSA defines a trade secret as "any information that derives economic value from not being generally known by others" and "the subject of efforts that are reasonable under the circumstances to maintain its secrecy."[79] Generally, a trade secret has value because it provides its owner with a competitive advantage.

A trade secret may, however, be a poor option for software developers. Unlike a patent which protects an invention for 20 years, a trade secret is valid only so long as the secret remains a secret. In this manner, trade secret law provides "far weaker protection in many respects than the patent law."[80]

Second, and most important in the reverse engineering context, the public at large remains "free to discover and exploit the trade secret through reverse engineering of products in the public domain or by independent creation"[81] during the time the trade secret is still a secret. Trade secret protection does not cover any unpatented information that is in the public domain because "secrets" are not considered to be "public" or in the public domain. State and federal trade secret laws protect only material that is confidential and does not conflict with the *Policy of Free Copyability* that applies to material in the public domain.

---

76. Bly v. Banbury Books, Inc., 638 F. Supp. 983 (E.D. Pa. 1986) (holding that the defendant used the interim copy for commercial gain).

77. This standard was set in See v. Durang, a non-software, basic copyright case where the plaintiff author claimed that the defendant's play was based on a draft script written by the plaintiff. Rather than looking at the interim copies or process used to write the play, the court compared the two plays (end products) as a whole and found no infringement of expression. Where the interim copies do not meet the two-prong standard, and are held to be technically illegal, they are subject to standard royalty payments or statutory damages. *See* Soma, *supra* note 32, at 212. The non-infringing end products do not need to be licensed from the plaintiff.

78. The Uniform Trade Secrets Act with 1985 Amendments, §§ 1-12.

79. *Id.* at §§ 1(4)(i), (ii).

80. Kewanee Oil Co., 416 U.S. at 489-90.

81. *Id.* at 490.

Third, trade secrets also appear to be contrary to the public policy of making information widely available, since it encourages people to keep their inventions secret. Nonetheless, the Supreme Court held in *Aronson v. Quick Point Pencil Co.*[82] that trade secret law is not inconsistent with or pre-empted by public policy or patent law because, by definition, trade secrets are not in the public domain.

Fourth, trade secrets are governed by state law, and thus the states are free to regulate it "in any manner not inconsistent with federal law."[83] Thus, trade secret laws vary throughout the states.

Fifth, trade secrets can be "stolen" by "friends" of those who learn of the secrets legitimately, with the consent of the owner, but then use the secret to compete with the owner. Trade secrets can also be stolen by "strangers" who learn of the secret in an illegal or improper manner that successfully overwhelms the owner's reasonable efforts to keep the matter confidential.[84] To be protected under trade secret law, the owner of an alleged trade secret must take "reasonable steps" to preserve the secrecy of the item.[85] Absolute secrecy is not required.

Sixth, trade secret protection does not cover the innocent use of stolen secrets. This occurs when a party uses the secret without knowing that the secret has been stolen or misappropriated. In those cases, the "innocent user" is not liable to the owner until the innocent user becomes aware that the material is a trade secret.[86] This may have special significance for software released on networks such as the Internet or on electronic bulletin boards.

Despite these apparent shortcomings, it is widely recognized today that "probably the single most important 'product' eligible for trade se-

---

82. Aronson v. Quick Point Pencil Co., 440 U.S. 257 (1979).

83. *Id.* at 262.

84. Uniform Trade Secrets Act with 1985 Amendments, §1(2). Note, for theft of trade secret by "strangers", no federal criminal law exists at this time, but approximately 20 states have statutes prohibiting such activity. To protect secrets filed with the Federal Government, an exception to the Freedom of Information Act (FOIA) permits the government to refuse to reveal the trade secrets. If a government agency inadvertently releases the trade secret, the owner of the secret, with some statutory basis for a claim of confidentiality, may file suit under the Administrative Procedure Act to enjoin the disclosure.

85. *Id.* at §1(4)(ii). "Reasonable steps" may include: 1) contractually imposing an "explicit duty of confidentiality" on any person with whom the secret is shared through the use of "non-disclosure" and "non-compete" agreements with business investors, product partners, and employees; or 2) by an "implied duty of confidentiality" if the relationship between the parties warrants. Of particular importance to software developers, is that if a party in such a relationship uses the secret to his own advantage or reveals it to another, the trade secret owner will be able to secure judicial relief — usually through tort law.

86. *See, e.g.*, Computer Print Systems, Inc. v. Lewis, 422 A.2d 148, 155-56 (Pa. Supp. 1980) (innocent recipient became aware, in course of dealing with owner, that plaintiff's former officer had not been authorized to make the duplicate software program); Components for Research, Inc. v. Isolation Products, Inc., 241 Cal. App.2d 726 (1966) (defendant directors put on notice, thus entitling plaintiff to injunctive relief against both the directors and the corporation on whose board they sat).

cret protection is computer software."[87] The unpublished holding in *Stac Electronics v. Microsoft Corp*[88] "placed this issue on the agenda of companies throughout the computer industry."[89] *Stac* represents the first determination that the source code of a widely distributed computer program is protectable under the trade secret laws, winning one of the largest verdicts ever in a trade secret case.[90] Attorneys reviewing the case believe that "what some in the computer industry have advocated should be permissible reverse engineering is precisely what one jury concluded constitutes "willful and malicious misappropriation."[91] The reviewers concluded that:

> . . . the combination of practical and legal restrictions on access to the ideas and engineering reflected in the source code . . . provides a basis for trade secret protection. As long as the owner of the software takes industry standard measures . . . and is diligent in the prosecution of its rights, trade secret protection may be available unless or until such information in fact becomes generally known . . . . At a minimum, trade secret law may afford software developers the valuable head start against competitors that the trade secret laws were designed to protect.[92]

III. THE LEGALITY OF CONTRACT OR LICENSE CLAUSES RESTRICTING REVERSE ENGINEERING

Most reverse engineering cases focus on the acts of the defendants. However, defendants may also raise defenses that focus on the acts of the plaintiff.[93] One such defense, if the technology is legally protected by trade secret and copyright, is copyright misuse. Under misuse, a court will consider whether a copyright holder is illegally extending copyright protection through a contract or license which contains a clause that restricts reverse engineering of the copyrighted material. The Fourth Circuit recently decided a series of cases on this issue and held that such clauses are illegal extensions. But because of the extreme facts in each case, as discussed below, these rulings may or may not be adopted elsewhere.

In *Lasercomb Am., Inc. v. Reynolds*, the plaintiff produced software for designing dyes to make boxes.[94] The company then distributed the

---

87. Johnston & Grogan, *supra* note 12, at 7, *citing* Computer Print Systems, Inc. v. Lewis, 422 A.2d 148 (Pa. Sup. 1980); Components for Research, Inc. v. Isolation Products, Inc., 241 Cal. App.2d 726 (1966).

88. Stac Electronics v. Microsoft Corporation, 1994 U.S. App., Lexis 18042 (1994).

89. Johnston and Grolan, *supra* note 12, at 1.

90. *Id.* at 2 (noting that pursuant to the settlement agreement between the parties, all orders, verdicts and judgments in the case have been vacated).

91. *See* Uniform Trade Secrets Act with 1985 Amendments, § 3(b). "If willful and malicious misappropriation exists, the court may award exemplary damages in an amount not exceeding twice any award made under subsection (a)." *Id.*

92. Johnston & Grolan, *supra* note 12, at 3-4.

93. Soma, *supra* note 32, at 223.

94. Lasercomb Am., Inc. v. Reynolds, 911 F.2d 930, 971 (4th Cir. 1990).

software under a license that contained a noncompetition clause which restricted the licensee from developing his own dye-making software or assisting others in developing such software for a period of ninety-nine years.[95] The defendant, Reynolds, did not sign the license agreement, but obtained a copy of the software which he reverse engineered to remove certain safeguards and then sold infringing copies of the software.[96] Lasercomb sued for copyright infringement and Reynolds pled copyright misuse as a defense.[97] The District Court rejected the copyright misuse defense, but the Fourth Circuit reversed, citing public policy[98] to uphold the copyright use defense. The Fourth Circuit further refused to enforce *Lasercomb's* copyright,[99] viewing the ninety-nine year noncompetition clause as an "anticompetitive restraint" that sought to "control competition" beyond the level granted by copyright law.[100] The court was unswayed by the fact that Reynolds was not harmed by the clause.

In *PRC Realty Systems Inc. v. National Ass'n of Realtors*,[101] an unpublished opinion, the plaintiff licensed software that allowed access to real estate multiple listing information. PRC's license included a clause requiring the licensee to exert its best efforts to also promote the multilisting publishing business.[102] The National Association of Realtors licensed the PRC software and then independently developed a desktop publishing system that allowed licensees of PRC's software to publish in-house multiple listings on a laser printer.[103] PRC sued for breach of contract and copyright infringement.[104] The district court held for PRC, but the Fourth Circuit reversed, emphasizing the public policy concerns articulated in *Lasercomb*.The Fourth Circuit stated in its "best efforts" clause, PRC attempted "to use its copyright as a hammer to crush all future development of an independent idea by [the defendant], or any other licensee."[105] It thus refused to continue an injunction enforcing the contract, but did uphold one count for breach of contract based on the fact that the defendant made non-exclusive license arrangements with parties other than PRC.[106]

Courts have upheld such clauses, however, where the license clause is not overreaching. In *Telerate Sys., Inc. v. Caro*,[107] for example, Telerate

---

95. *Id.* at 972.
96. *Id.* at 971.
97. *Id.*
98. The Court adopted the equitable public policy approach of Morton Salt Co. v. G.S. Suppiger Co., 314 U.S. 488, 494 (1942), to uphold the copyright use defense.
99. Lasercomb, 911 F.2d at 979.
100. *Id.* at 978.
101. PRC Realty Systems Inc. v. National Ass'n of Realtors, 766 F. Supp. 453, 456 (E.D. Va. 1991), *aff'd*, No. 91-1125, 91-1143, 1992 WL 183682 (4th Cir. 1992) (per curiam).
102. *Id.*
103. *Id.*
104. *Id.* at 458.
105. *Id.*
106. *Id.*
107. Telerate Sys., Inc. v. Caro, 689 F. Supp. 221 (S.D.N.Y. 1988).

developed a financial database, marketed as both a higher priced, standard personal computer-based software, and a lease-based special terminal which carried the right to access the database at a lower fee. Defendant Caro developed an alternative access software package that ran on any PC and allowed a Telerate customer to purchase Telerate's database-access rights at the lower leased-terminal rate, disconnect the terminal and still access the database.[108] Caro's software provided several desirable access improvements,[109] but also caused some performance problems for a number of Telerate's customers.[110] The Court found that Caro's reverse engineering of Telerate's software to develop the package likely violated various contract provisions[111] and granted Telerate's motion for a preliminary injunction finding a likelihood that Caro's software infringed Telerate's database copyrights when it copied data from the database.[112]

If a court finds one or more terms of a contract impermissible, the court will normally eliminate only the offending portions of the contract and enforce the rest.[113] In a court of equity, however, the court could void the entire contract under the "doctrine of unclean hands."[114] In *Atari*, the Court specifically required that a party seeking an injunction seek equity with clean hands. This was a significant factor in the outcome of the case, which permitted the reverse engineering of Atari's product because Atari was found to have had "unclean hands" after it made false statements to the Patent, Trademark, and Copyright Office.[115]

The equitable defense of unclean hands requires misconduct by the plaintiff, relevant to the issues before the court, that directly harms the defendant.[116] However, no injury to the defendant is required in cases of fraud, misrepresentation, or other unconscionable conduct since that conduct is illegal, unfair, and against public policy.[117]

Thus, a copyright or patent holder, in a contract that attempts to impose restrictions beyond the protection provided by copyright and patent law or in conflict with antitrust laws, may be considered to have unclean hands and lose all protection.[118] This can include any attempt to withhold "ideas" from the public because doing so imposes patent-like protection on software without meeting the rigorous requirements of pat-

---

108. *Id.* at 224.

109. *Id.*

110. *Id.* at 225.

111. *Id.* at 226.

112. *Id.* at 240.

113. Soma, *supra* note 32, at 224, *citing* Somerset Importers, Ltd. v. Continental Vintners, 790 F.2d 775, 781-82 (9th Cir. 1986); Quiller v. Barclays American/Credit, Inc., 764 F.2d 1400, 1403 (11th Cir. 1985) (dissenting opinion), *cert. denied*, 476 U.S. 1124 (1986).

114. Atari, 975 F.2d at 846.

115. *Id.*

116. *Id.* at 846-47.

117. *See, e.g.*, Campbell Soup Co. v. Wentz, 172 F.2d 80, 83 (3d Cir. 1948).

118. *See* Jere M. Webb & Lawrence A. Locke, *Intellectual Property Misuse: Developments in the Misuse Doctrine*, 4 HARV. J. L. & TECH. 257, 257-58 (1991).

entability. It is therefore unfair and inconsistent with the purpose and policy of copyright law. A copyright may also imply market power and thus open antitrust issues.[119] Similarly, the practice of "tying," or the requirement that a purchaser of one patented or copyrighted product also purchase a non-protected product or service, would be considered to cause "unclean hands" since the practice violates antitrust law.[120]

However, a license that does not impose restrictions beyond the protection provided by copyright and patent law may be considered legal. Software developers should therefore utilize these protections to preserve important legal coverage of their products.

IV. THE EUROPEAN SOFTWARE DIRECTIVE

A. *Similarities of European Software Copyright Law to U.S. Copyright Law*

The objectives, rights, and restrictions of the European Union regarding software copyrights parallel most of those in the U.S.. In the preamble to its original proposal for a European Directive on the Legal Protection of Computer Programs, submitted to the European Council on January 5, 1989, the European Commission noted that "the size and growth of the computer industry is such that its importance in the economy of the Community cannot be overemphasized" and that "unless a legal environment is created which affords a degree of protection against the unauthorized reproduction of computer programs . . . comparable to that given to works such as books and films, research and investment in that vital industry will be stifled."[121] The objectives of the Commission's proposal for the Software Directive therefore were:

> a. to promote the free circulation of computer software within the Community and allow industry to take advantage of the single market by harmonizing the national laws of the Member States relating to the use and reproduction of computer software; and
> b. to prevent the unlawful copying of computer software, or 'computer piracy,' within the Community by ensuring an adequate level of protection for those who create computer software.[122]

The final version of the Software Directive,[123] adopted by the European Council on May 14, 1991, reads as follows:

---

119. *See* William M. Landes & Richard A. Posner, *Market Power in Antitrust Cases*, 94 HARV. L. REV. 937 (1981).

120. In 1988, Congress passed the Patent Misuse Reform Act, 35 U.S.C. § 271(d)(5), which requires that before misuse will be found for tying, a patent owner must have market power in the relevant market for the patented product on which a license or sale is conditioned. If so, a "rule of reason" analysis must be presented for tying activity. Copyright misuse analysis follows that of patent misuse. Lasercomb, 911 F.2d at 977.

121. Preamble, "European Directive on the Legal Protection of Computer Programs," submitted by the European Commission to the European Council on January 5, 1989.

122. *Id.*

123. *Software Directive*, *supra* note 3.

1. "A logical and, where appropriate, physical interconnection and interaction is required to permit all elements of software and hardware to work with other software and hardware" to enable full use of computers' intended function.[124]

2. "This functional interconnection and interaction is generally known as 'interoperability'; whereas such interoperability can be defined as the ability to exchange information and mutually to use the information which has been exchanged."[125]

3. Further, "in accordance with this principle of copyright, to the extent that logic, algorithms and programming languages comprise ideas and principles, those ideas and principles are not protected under this Directive."[126]

4. Copyright law provides authors the exclusive right to do or to authorize the making of copies or derivative works, and/or the distribution of their copyrighted work[127] for the life of the author plus 50 years after his death or the death of the last surviving author in cases of multiple authors.[128]

5. "In the absence of specific contractual provisions," lawful acquirers of computer programs may make [interim] copies of the program, where necessary a) to run the program in accordance with its intended purpose, including error correction;[129] b) to make a back-up copy;[130] and c) to "observe, study or test the functioning of the program in order to determine the ideas and principles which underlie any element of the program,"[131] "provided that these acts do not infringe the copyright in the program."[132]

6. Decompilation or "reverse engineering" of software code and the translation of its form are permitted by authorized persons, with certain restrictions.[133]

B. *Differences Between European Software Copyright Law and U.S. Copyright Law*

The main differences between European and U.S. copyright laws are as follows:

1) The EU limits the need for access to the underlying ideas as those needed for "interoperability," while access is unclear under *Sega* and

---

124. *Id.* at Recital 10.
125. *Id.* at Recital 12.
126. *Id.* at Recital 14.
127. *Id.* at art. 4.
128. *Id.* at art. 8 and Recital 25.
129. *Id.* at art. 5.1 and Recitals 17 and 18.
130. *Id.* at art. 5.2.
131. *Id.* at. art. 5.3.
132. *Id.* at Recital 19.
133. *Id.* at art. 6.1(a) and Recitals 21 and 22.

Nintendo.

2) The Directive limits the "fair practice exceptions" to an author's exclusive rights[134] to those which "make it possible to connect all components of a computer system, including those of different manufacturers, so that they can work together."[135] It further states that "this exception may not be used in a way which prejudices the legitimate interests of the rightholder or the normal exploitation of the program."[136]

3) The Directive limits decompilation or "reverse engineering" of software code and the translation of its form to those circumstances where reverse engineering is indispensable to obtain the information necessary to achieve the interoperability of an independently created computer program with other programs.[137]

4) An authorized user may "observe, study or test the functioning of the program in order to determine the ideas and principles which underlie any element of the program" only if he does so while performing any of the acts of loading, displaying, running, transmitting or storing the program which he is entitled to do.[138] This effectively limits access to the underlying ideas only through the "black box" of normal use of the program — not the more in-depth dumping, flow-charting, and rigorous analysis allowed in the U.S.

5) This process also requires adequate disclosure by the user.[139]

6) The Directive also recognizes the "dominate reseller" provisions of articles 85 and 86 of the European Treaty.[140]

7) Infringement of the exclusive rights of the author are defined as the "unauthorized reproduction, translation, adaptation or transformation of the form of the code *in which a copy of a computer program has been made available.*"[141]

The publication of this proposal resulted in one of the largest controversies ever experienced in the EU over a Directive. The concern focused mainly on the conditions under which a computer program could be reverse engineered and copied for profit. The Europeans, as did software-knowledgeable persons in the U.S., recognized that some copying is necessary to run a program, but they also sought to protect the creative efforts of the programmers to encourage further development.

C. *Importance of the Similarities and Differences Between European*

---

134. *Id.* at Recital 22.
135. *Id.* at Recital 24.
136. *Id.* at Recital 24.
137. *Id.* at art. 6(1), (2) and Recitals 21 and 22.
138. *Id.* at art. 5.3.
139. *Id.* at art. 5.2.
140. *Id.* at Recital 27.
141. *Id.* at Recital 20 (emphasis added).

Software Law and U.S. Law to Software Developers

Software developers have legal protection in the EU that is similar to the protection in the U.S. If carefully drafted, the same or similar license contracts can often be used for customers in both continents. If anything, the European Directive appears to provide greater protection for software developers than the U.S. since the Directive seems to limit a users "access to the underlying ideas" to the "black box" approach to reverse engineering. That is, reverse engineering using decompilation for interoperability is only permitted if indispensable to obtain the underlying ideas. It can thus be argued that if a software developer makes the source code available to the user, infringement may be found if the user reverse engineers the code since such action would not be indispensable. This is significant for companies with unique software products, which for marketing purposes may be better off releasing the source code and encouraging compatibility and licensing in this manner. A product with major yearly revisions would be an example of such a strategy.[142]

The only major limitations to these European protections are that the Directive recognizes the "dominate reseller" provisions of articles 85 and 86 of the European Treaty, and these protections have not been widely tested in the courts. The various national statutes are quite new and almost no case law exists in this area. Thus, while the statutes provide critical protection and guidance to software developers in licensing their software, the developers should be aware that this is still a new and unsettled area of the law.

V. SAMPLE LANGUAGE FOR SOFTWARE LICENSES

Software developers seeking to protect their product from unlimited reverse engineering and to strengthen their legal position in a product infringement case should implement the following action items:[143]

1. Software developers cannot completely prohibit reverse engineering of their programs, since public policy requires access to all unprotected information in the public domain. Software developers can, however, proactively provide access to the information under a license that both limits certain activities and contractually obligates the user/licensee to honor the information accessed as a trade secret.

2. Case law indicates that contract or license clauses which limit re-

142. The Directive also makes clear that patents, trade secret, trademark and other intellectual property and contract laws are also available to the software developer. The full scope of patents issued in the U.S. and Europe for software, however, is still undecided, and a somewhat guarded attitude about the issuance of software patents remains in both constituencies.

143. Software developers should apply for a patent on the product since patents provide the strongest intellectual property protection. Historically, patents have not been granted in the U.S. to software, but since Diehr, they are becoming increasingly more of an option. The situation in Europe remains guarded and uncertain.

verse engineering are legal so long as the language of the clause does not exceed the authority granted under copyright and other laws. Software developers, therefore, should include a clause prohibiting reverse engineering in their license. Sega did not do so and appears to have lost certain claims due to this omission. In Europe, however, caution should be observed in including a reverse engineering prohibition without a thorough Article 85 and 86 analysis.

3. The license requires the user to state its intent on reverse engineering, and appropriate notice. In *Sega*, the Court required that the defendant have a "legitimate interest" in or "reason for" gaining access to the reverse engineered information.

> a. If the user's interest is to develop a compatible, interoperable product that benefits software developers by providing added market recognition and product sales, reverse engineering should be encouraged or appropriate source code should be provided.
> b. If the user's interest is to develop a competing product, such an interest is legal and cannot be restricted. In *Sega*, the Court stated that "public policy plus the authorization of section 117(1) [is] not limited [only] to [the] use intended by [the] copyright owner."[144] The information gained from this statement in *Sega*, however, provides the software developer with important "notice" and establishes the "access" element for a possible later infringement case. The plaintiff, then, need only show that the reverse engineered product is "substantially similar" to the original product and that the resulting product negatively impacts the potential market of the original product.
> c. If the user states one intent but carries out another, the software developer has stronger breach of contract, misappropriation, trade secret theft, and/or "unclean hands" arguments.

4. Write the license to convey that everything needed for the user's "legitimate," intended purpose is provided. In *Sega*, the Court stated that reverse engineering of a copyrighted software program was "fair use" because it was the only means of gaining access to the unprotected aspects of the program; the European Software Directive allows decompilation only when indispensable to achieving interoperability. However, if the developer of the original software voluntarily provides the object and source code to the party desiring it, this leaves no opportunity for the user to later claim that reverse engineering was necessary or indispensable to gain access or that the copyright owner was using the code or the contract to "hide information as in a patent." In drafting the license, software developers should be certain to avoid words that indicate ambiguity, such as "sample programs" or "all needed to do a specific task." The wording of the clause must satisfy U.S. public policy which requires that all unprotected ideas in the public domain are available. To convey compliance with public policy, developers should offer to provide any additional information that the users discover as lacking for their intended purpose.

---

144. Sega, 977 F.2d at 1520 n.6.

This offer, of course, must be accompanied by a *caveat* that allows the software developers to use commercial discretion in providing the additional information so as not to overwhelm their staff. The users may still access the information through reverse engineering, but the time and expense required to do so may discourage some users from doing so.

5. Clearly define "interoperability" in the definitions section of the contract. Doing so will avoid misunderstandings and establish "mutual intent" between two skilled persons in the field.

6. Note in the license that while verbatim interim copies of copyrighted material are permitted as a "fair use exception," they are not to be sold or distributed.

7. Take all necessary and "reasonable" steps to protect your trade secret rights, including provisions in the license to protect trade secret rights. These may include restricted use and/or confidentiality agreements. If the user then reveals the secret, a breach of contract and/or misappropriation claim can be made by the software developer. Sega did not do this, which contributed to its loss.

8. Be certain to maintain "clean hands."

9. Track the market for the product involved and maintain an accurate, detailed historical record. This will facilitate clear documentation of the impact of competitive products and of the value of trade secrets.

10. In negotiating and signing the contract, use persons skilled in reverse engineering who understand the contractual restrictions. This will strengthen contractual evidence if problems arise later.

VI. CONCLUSION

It is evident from the U.S. Constitution, subsequent U.S. statutes and case law, and the European Software Directive, that the U.S. and European Union are seeking to provide a balance between providing access to unprotected ideas to benefit the public and providing sufficient protection to software developers to encourage them to continue producing innovative products. Reverse engineering is permitted under certain circumstances in the U.S. and within the European Union, but software developers still maintain a number of well-supported options to achieve their three goals of a) legally prohibiting or limiting the reverse engineering of their product, b) protecting their state law trade secret and other intellectual property claims, and c) placing themselves in the best possible legal position to argue a claim if their product is reverse engineered and an infringing software product is created.

The initial step in claiming these options is the protection of the product under a license that should be able to withstand a test based on current U.S. and European law. Concepts for such a license are provided in the above sample license.

The defendant in a U.S. infringement case challenging this approach, would likely argue the Fourth Circuit holding that the limiting clauses in

the license are illegal. The plaintiff, on the other hand, would likely argue the Ninth Circuit "fair use" parameters, contract law, and intellectual property defenses of misappropriation, trade secret, and clean hands. If the plaintiff can write the license to win even just one of these claims, that is enough to prevail for infringement protection. This is also known as the "one strike and you're out!" aspect of intellectual property law. The final determination of the validity of these concepts, both in the U.S. and the European Union, however, must wait for a test case addressing these issues.