Hastings Communications and Entertainment Law Journal

Volume 15 | Number 3 Article 1

1-1-1993

The Games People Play: Sega v. Accolade and the Right to Reverse Engineer Software

William S. Coats

Heather D. Rafter

The Games People Play: *Sega v. Accolade* and the Right to Reverse Engineer Software

by
WILLIAM S. COATS
and
HEATHER D. RAFTER*

Table of Contents

* William S. Coats and Heather D. Rafter are attorneys with Gibson, Dunn & Crutcher in San Francisco. They represented defendant/appellant Accolade, Inc., in the *Sega v. Accolade* case. The case was recently settled.

Introduction

When the Ninth Circuit handed down its opinion on October 20, 1992 in *Sega Enterprises Ltd. v. Accolade, Inc.*,[1] the court resolved two issues of first impression that are of great importance to the U.S. software industry. The first issue is whether reverse engineering, creating a compatible program by studying a competitor's program, is permissible under the copyright laws. The legality of reverse engineering of software has been the subject of scholarly debate in the United States since Congress approved the Copyright Act of 1976. The second issue is whether a trademark can be used as part of a lock-out device to prevent competitors from making compatible products.

In addition, dicta in *Atari Games Corp. v. Nintendo of America Inc.*,[2] consistent with the decision in *Accolade,* reflect a consensus on the legality of reverse engineering. These two decisions taken together should make it exceedingly difficult to bring a copyright infringement action based solely on the acts undertaken by a competitor to reverse engineer a software program.

I
The Reverse Engineering Process

Programmers write software programs in high level "source code" languages that use alphanumeric characters understandable to humans, and routinely include labels and comments in the programs to identify parts of the program and to explain what the program is doing.[3] This source code then must be translated into object code to be utilized by the computer. The translation process, referred to as "assembly" or "compilation," generates a significantly altered object code version of the program.[4]

1. 977 F.2d 1510 (9th Cir. 1992).

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

3. *See* Andrew Johnson-Laird, *Technical Demonstration of "Decompilation"* 2-3, *in* Prentice Hall Law & Business, REVERSE ENGINEERING: LEGAL AND BUSINESS STRATEGIES FOR COMPETITIVE DESIGN IN THE 1990'S 102 (1992), which provides an excellent and understandable overview of the process.

4. Assembly involves more than just conversion of the alphanumeric source code into ones and zeroes. The source code labels and comments, which perform no useful function to the computer, are eliminated in the object code version. Additional program instructions that are functionally required for operation with a particular type of microprocessor may be added to the program automatically by the compiler. Further, the program may be "optimized"—a process by which the program instructions are placed in an order more usable by the computer. The *published* object code version thus is significantly different from the source code version. *Id.* at 4.

To reverse this process, the first step in "reading" the object code version of a computer program is to "disassemble" or decompile the object code. A program written in object code is one long string of "bits"—zeroes and ones. Just by looking at this string of zeroes and ones, a human reader has virtually no way of knowing what they represent. A person engaged in reverse engineering (a reverse engineer), therefore, must translate the ones and zeroes into assembly language—a low-level programming language that is comprehensible by humans.

To accomplish such translation by hand, a reverse engineer must group the bits into bytes, analyze the bytes to determine what they represent (e.g., data or instructions), and then translate the bytes into assembly language. Such a project would be impossible to accomplish solely from human memory, without making copies, by hand or by computer. Accordingly, disassembly is done by computers. The object code is copied into a computer's memory, and then a computer program translates it into assembly language that is intelligible to human readers. The assembly language is then printed out, i.e., "copied."

Because of the way programs are written and put into computer-usable form, the conversion into assembly language is only the first step in reading and understanding the object code version of a program. The assembly language still must be studied extensively and interpreted to understand how the program operates. The end product of this extensive study and analysis is not the original source code, but rather a reverse engineer's interpretation of how the program operates.

There are a variety of ways to obtain object code from semiconductor chips in which it is embedded. In *Accolade*, Accolade obtained the object code by attaching a commercially available device to the chip that "read" the object code and downloaded it into a computer. In *Atari Games*, Atari "stripped" or "peeled" the chip to read the object code. In both cases, the object code was then disassembled through a commercially available program that translated the object code into assembly language. Human programmers then read the assembly language to understand the object code. This rather simplistic description of reverse engineering, however, belies the fact that reverse engineering is a difficult and time-consuming task—and one that often fails.

II
Prior Case Law

Before the recent decisions by the Court of Appeal for the Federal Circuit (CAFC) and the Ninth Circuit, reverse engineering opponents argued that the sparse case law did not support the legality of reverse engineering. They often cited, for example, *SAS Institute, Inc. v. S & H*

Computer Systems, Inc.[5] In *SAS*, the defendant violated its license agreement with SAS by copying the SAS source code, and using that source code as a basis for developing a competing program. The defendant was enjoined from distributing its infringing program. The implication of *SAS* was that copying of a protected work at any step in the development process of the allegedly infringing work is a sufficient act upon which to establish a claim for copyright infringement.

Reverse engineering opponents would also cite cases in which a court found infringement based upon defendant's possession of a nearly final version of a work that was substantially similar to plaintiff's copyrighted work.[6] For example, in *Walker v. University Books, Inc.*, the Ninth Circuit held that infringement could be based upon "camera ready mechanicals" from which the final product would be made, rather than the final product itself.[7] None of those cases, however, presented the issue of an intermediate copy made solely to ascertain the nonprotectible expression necessary to achieve compatibility.

Reverse engineering supporters, in turn, relied on *E.F. Johnson Co. v. Uniden Corp. of America*[8] and *NEC Corp. v. Intel Corp.*[9] In *Johnson*, the district court stated that "dumping" and analyzing copyrighted object code do not establish infringement. The court went on to find infringement, but only on the grounds of substantial similarity between the final products.

In *NEC*, Intel claimed that NEC infringed the copyright of the microcode contained in its microprocessors. NEC's programmer had disassembled Intel's microcode before writing the allegedly infringing code. The district court held that there was no infringement because the final version of the NEC code was substantially different from the initial version, and any similarities were due to functional constraints.[10]

Other cases cited by reverse engineering proponents include *See v. Durang*[11] and *Vault Corp. v. Quaid Software Ltd.*[12] In *See*, the Ninth Circuit suggested that intermediate copying is not an infringement if the

5. 605 F. Supp. 816, 828-29 (M.D. Tenn. 1985).

6. *See, e.g.*, Walker v. University Books, Inc., 602 F.2d 859, 861-64 (9th Cir. 1979); Johnson Controls, Inc. v. Phoenix Control Sys., Inc., 886 F.2d 1173, 1176-77 (9th Cir. 1989); Walt Disney Prods. v. Filmation Assocs., 628 F. Supp. 871, 876 (C.D. Cal. 1986).

7. 602 F.2d at 864. The Ninth Circuit in *Accolade* interpreted *Walker* to mean that intermediate copies could be the basis of copyright infringement irrespective of whether the end products were substantially similar. Sega Enters. Ltd. v. Accolade, Inc., 977 F.2d 1510, 1519 (9th Cir. 1992).

8. 623 F. Supp. 1485, 1501 n.17 (D.C. Minn. 1985).

9. 10 U.S.P.Q.2d (BNA) 1177 (N.D. Cal. Feb. 6, 1989).

10. *Id.* at 1186, 1189.

11. 711 F.2d 141 (9th Cir. 1983).

12. 847 F.2d 255 (5th Cir. 1988).

final products are not substantially similar. In *Vault*, the court broadly construed 17 U.S.C. § 117 to permit a person in lawful possession of a computer program to copy that program where copying is essential to its reasonable use, even if such incidental copying is unintended by the copyright holder. The court further held that a state statute that outlawed reverse engineering, decompiling, or disassembling was preempted because it was at odds with sections 117(1) and 117(2) of the Copyright Act.[13]

There is thus ample case law to support the argument that section 117 allows most of the disassembly process at issue in the *Accolade* case: the loading of Sega's program into computer memory for the purposes of disassembling it, the actual disassembly, and any modifications to the programs made in computer memory to discern how the programs worked.

Ironically, both proponents and opponents of reverse engineering of software cited the cryptic case of *Hubco Data Product Corp. v. Management Assistance, Inc.*[14] The *Hubco* court referred to its earlier unpublished decision in which it found that reverse engineering by the alleged infringer, Hubco, was not infringement. The district court, however, ruled against Hubco because it copied wholesale the object code of Management Assistance, Inc. (MAI) and sold it to MAI computer owners.[15]

The leading commentators were divided on the issue of the legality of reverse engineering. Professors Goldstein and Samuelson,[16] for example, supported the legality of reverse engineering, while William Patry reached a contrary conclusion.[17]

13. *Id.* at 270. The sole issue not directly addressed by the court in *Vault* is whether *printouts* of the disassembled code constituted infringement. Other cases that have applied the section 117 exemption liberally to shelter research uses of copyrighted programs are Foresight Resource Corp. v. Pfortmiller, 719 F. Supp. 1006 (D. Kan. 1989), and RAV Communications, Inc. v. Phillipp Bros., 1988 Copyright L. Dec. (CCH) ¶ 26,263 (S.D.N.Y. Apr. 13, 1988).

14. 219 U.S.P.Q. (BNA) 450 (D. Idaho Feb. 3, 1983).

15. *Id.* at 455-56.

16. Respectively, P. GOLDSTEIN, COPYRIGHT § 5.2.1.4 at 85-91 (Supp. 1991) and Pamela Samuelson, *CONTU Revisited: The Case Against Copyright Protection for Computer Programs in Machine-Readable Form*, 1984 DUKE L.J. 663, 705-53. *See also, e.g.*, Donald S. Chisum et al., *LaST Frontier Conference Report on Copyright Protection of Computer Software*, 30 JURIMETRICS J. 15, 24-25 (1989) [hereinafter LaST Report] (panel of leading computer law scholars endorses intermediate copying of computer programs as being permitted by computer laws); Gervaise Davis, *Scope of Protection of Computer-Based Works: Reverse Engineering Clean Rooms and Decompilation* (1992), *reprinted in* Prentice Hall Law & Business, *supra* note 3, at 27. The European Economic Community (EEC) also recently adopted legislation allowing "decompilation" (another term for disassembly) to achieve compatibility. *See* EEC Software Directive, Article 5*bis*.

17. WILLIAM F. PATRY, THE FAIR USE PRIVILEGE IN COPYRIGHT LAW 401 (1985).

III
The *Accolade* Case

Accolade involved a copyright and trademark action brought by Sega Enterprises, Ltd. (Sega), a Japanese company that manufactures the Genesis console (Genesis) on which video games are played. Defendant Accolade is a developer and distributor of computer entertainment software.

Accolade developed its video game programs for the Genesis in two separate stages. In the first stage, in order to learn the requirements for creation of a Genesis-compatible game, Accolade reverse engineered the Genesis by disassembling object code stored in commercially available read only memory (ROM) chips in Sega's games. After disassembly, Accolade printed out the disassembled object code, studied and annotated the printouts, loaded the disassembled code back into computer memory, experimented to discover how the Genesis worked by modifying the programs, and studied the results of the modifications. Accolade undertook all of these steps—which resulted in the "intermediate copying" at issue—solely to understand how games functioned on the Genesis, with a view to independently creating games that would operate on the Genesis.

In the second stage, Accolade created its *own* games for the Genesis—a lengthy and laborious process wholly separate from reverse engineering. Moreover, Sega conceded for purposes of its preliminary injunction motion that Accolade's games were *not* "substantially similar" to its programs—a required element of infringement under prevailing copyright law.[18]

IV
The "Trademark" Security System

Accolade also raises a unique trademark issue which, although it has drawn less attention, is equally important to the right of software publishers to sell compatible software without the consent of hardware manufacturers. In September 1991, three years after Sega introduced the Genesis and after Accolade's games were in the marketplace, Sega released a modified Genesis. The modified Genesis contained a "lockout" feature, a so-called "trademark security system" (TMSS), which

18. *See, e.g.*, Atari Games Corp. v. Nintendo of America Inc., 975 F.2d 832 (Fed. Cir. 1992); Computer Assocs. v. Altai, 982 F.2d 693 (2d Cir. 1992). In any case, to state that Accolade's game programs are substantially similar to Sega's is analogous to saying that two 300-page novels are substantially similar because they both began with the phrase, "It was a dark and stormy night." A 300-page novel would consist of approximately 1,000,000 bytes; "it was a dark and stormy night" would consist of 30 ASCII characters (each of the letters plus each of the spaces between the words being one byte).

"searches" a game's program to "find" a security code, in this case four bytes of data, consisting of the ASCII representation of the letters "S-E-G-A," in a specified location. If the modified Genesis finds the four bytes of data in the correct location, it allows the game to operate, but produces an initial screen display that lasts approximately three seconds and states in plain text "Produced By or Under License From Sega Enterprises Ltd." (Screen Display). If the modified Genesis does not find the four bytes of data in the correct location in the game program, the modified Genesis "locks out" the game.

In October 1991, Sega sued Accolade claiming trademark infringement, unfair competition, and, after amending its complaint, copyright infringement. Sega sought injunctive relief based on two theories. First, Sega claimed the intermediate copies made during Accolade's reverse engineering process were infringing copies, and that it was entitled to an injunction against use of the copies or any ideas derived from them. More broadly, Sega argued that Accolade's games, even though not similar to Sega's games, were unavoidably tainted by the knowledge that Accolade gained from the intermediate copies. Distribution of those games, it contended, should be enjoined as well. Sega also claimed that the appearance of the Sega message when an Accolade game was played on the Genesis constituted trademark infringement and false designation of origin under section 43(a) of the Lanham Act.[19]

V
The District Court's Ruling in *Accolade*

The district court agreed with Sega.[20] In a short opinion, the district court first rejected Accolade's contention that no court had ever found infringement based upon intermediate copying absent a finding of misappropriation or some other wrongful act and/or substantial similarity in the final product. The district court also rejected Accolade's argument that the intermediate copies it made were a fair use. In addition, the district court found that Accolade's acts constituted trademark infringement. Based on its finding that Accolade had committed both copyright and trademark infringement, the district court enjoined Accolade from any further disassembling of Sega's object code, using the disassembled object code, or developing, manufacturing, shipping, distributing, or selling any game compatible with the Genesis or which otherwise "prompts" the Screen Display.

19. 15 U.S.C. § 1101 (1988).

20. Sega Enters., Ltd. v. Accolade, Inc., 785 F. Supp. 1392 (N.D. Cal. 1992).

Shortly after issuing its opinion and order, the district court modified its order, requiring Accolade to demand the return of all its games in the marketplace. Upon Accolade's motion, the Ninth Circuit expedited the appeal and granted the stay in part, staying the modified order but requiring Accolade to provide disclaimers on its games.

On August 28, 1992, in a "minute order," the Ninth Circuit dissolved the injunction in its entirety without requiring disclaimers, and on October 20, 1992, the Ninth Circuit issued its written decision. Meanwhile, two weeks after the issuance of the minute order, the CAFC in its *Atari Games* opinion stated—albeit in dicta—that reverse engineering is a fair use under 17 U.S.C. § 107.

VI
The *Atari Games* Decision

At issue in *Atari Games* was whether the game cartridges created by Atari for the Nintendo Entertainment System (NES) infringed the "10NES" computer program developed by Nintendo. 10NES was a security system designed by Nintendo to prevent the NES from accepting unauthorized cartridges.

Atari reverse-engineered chips by chemically removing layers from Nintendo's chips. Through microscopic examination of the "peeled" chips, Atari's engineers transcribed the 10NES object code into a handwritten list of ones and zeroes. Those ones and zeroes were then keyed into a computer that "disassembled" the object code in order to make the code understandable.[21]

Atari, however, was not able to obtain sufficient information to replicate the NES security system through this process.[22] Instead, the trial court found that Atari's former counsel obtained the source code for the 10NES program from the U.S. Copyright Office under false pretenses, by alleging that the code was necessary for litigation when no case was pending. Atari then copied the "purloined" code, and that code was used to facilitate its reverse engineering efforts. After successfully deciphering the 10NES program, Atari developed its own program—the Rabbit program—to unlock the NES.[23]

Nintendo moved for a preliminary injunction. It contended that Atari committed copyright infringement by: (1) copying the source code obtained from the Copyright Office; (2) making an intermediate copy

21. *Atari Games*, 975 F.2d at 844.
22. *Id.* at 836.
23. *Id.* at 836, 843-44.

during the reverse engineering process; and (3) creating programs substantially similar to its own.[24]

The district court found that Nintendo was likely to prevail on its copyright claim at trial and granted a preliminary injunction.[25] The CAFC upheld the injunction but disagreed with the district court's reasoning. The CAFC stated that intermediate copying, as a step in the reverse engineering process, was not in and of itself actionable: "Reverse engineering, untainted by the purloined copy of the 10NES program and necessary to understand 10NES, is a fair use. An individual cannot even observe, let alone understand, the object code on Nintendo's chips without reverse engineering."[26]

In support of its opinion that reverse engineering is a fair use,[27] the CAFC relied on the U.S. Supreme Court's decision in *Feist Publications, Inc. v. Rural Telephone Corp.*,[28] and the language of 17 U.S.C. § 102(b), for the proposition that copyright is not intended to afford patent-like protection over the use of ideas, processes, or methods of operation.[29] In addition, the CAFC cited to the legislative history of section 107 which suggested that courts should adapt the fair use doctrine to accommodate technological innovations.[30]

The CAFC also viewed its finding that reverse engineering was a fair use consistent with the "nature of the work" factor set forth in section 107(2). The court reasoned that "[w]hen the nature of a work requires intermediate copying to understand the ideas and processes in a copyrighted work, that nature supports a fair use for intermediate copying."[31]

The CAFC, however, held that Atari could not avail itself of the fair use defense due to its use of a "purloined" copy of the source code in the reverse engineering process.[32] Moreover, the court found that Nintendo was likely to prove infringement because there was substantial similarity between Atari's final programs and the 10NES.

24. *Id.* at 840.

25. 18 U.S.P.Q.2d (BNA) 1935 (N.D. Cal. Apr. 11, 1991).

26. *Atari Games*, 975 F.2d at 843.

27. It is interesting that the CAFC *sua sponte* declared that reverse engineering was a fair use. The fair use defense was not raised by Atari Games in any of its briefs.

28. 111 S. Ct. 1282 (1991).

29. *Atari Games*, 975 F.2d at 842.

30. *Id.* at 843, *citing* H.R. REP. NO. 1476, 94th Cong., 2d Sess. 66 (1976), *reprinted in* 1976 U.S.C.C.A.N. 5659.

31. *Atari Games*, 975 F.2d at 843, *citing Feist*, 111 S. Ct. at 1290.

32. *Atari Games*, 975 F.2d at 843, *citing* Harper & Row Publishers, Inc. v. Nation Enters., 471 U.S. 539, 562-63 (1985) (knowing exploitation of purloined manuscript not compatible with "good faith" and "fair dealings" underpinning of fair use doctrine).

Thus, although the CAFC explicitly stated that reverse engineering of software is a fair use—as is the necessary step of making an intermediate copy—it denied this defense to Atari.

VII
The *Accolade* Decision

In contrast to the result in (but not the reasoning of) *Atari Games*, the Ninth Circuit in *Accolade* held that Accolade was entitled to the protection of the fair use defense. The panel held that reverse engineering of software in order to make compatible products is not a violation of the copyright law where there is "no alternative means of gaining an understanding" of the ideas and functional concepts necessary to design a compatible product.[33] In reaching its decision, the Ninth Circuit specifically noted that its analysis was consistent with that of the Federal Circuit in *Atari Games*.[34]

A. The Fair Use Analysis

The Ninth Circuit analyzed the fair use factors contained in 17 U.S.C. § 107 and concluded after a detailed examination that those factors weighed in Accolade's favor. The four enumerated factors are:

> (1) the purpose and character of the use, including whether such use is of a commercial nature or is for a nonprofit educational purpose;
> (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.[35]

The statutory factors are not exclusive. Rather, the doctrine of fair use is in essence "an equitable rule of reason."[36]

In analyzing the first factor, the court noted that the evidence established that Accolade reverse engineered Sega's programs solely to discover the functional requirements for compatibility with the Genesis console. Therefore, although Accolade's ultimate purpose was the release of Genesis-compatible games for sale, its direct purpose in copying Sega's code was for study. The court rejected as "too simplistic" the view that any commercial use of the work precluded the availability of the fair use defense. Indeed, implicit in the Ninth Circuit's opinion is its

33. 977 F.2d 1510, 1520 (9th Cir. 1992).

34. *Id.* at 1513 n.1. The Ninth Circuit amended its opinion on January 4, 1993, to add a footnote to stress this point.

35. 17 U.S.C. § 107 (1988).

36. *Harper & Row*, 471 U.S. at 560 (quoting H.R. REP. NO. 1476, 94th Cong., 2d Sess. 65 (1976), *reprinted in* 1976 U.S.C.C.A.N. 5659, 5679).

view that the making of compatible products for the Genesis was a legitimate and "non-exploitive" purpose.[37]

The court next considered the fourth factor—effect on the potential market for the copyrighted work—and concluded that it weighed in favor of Accolade. The court stated that reverse engineering is beneficial because there is an "increase in the number of independently designed video game programs offered for use with the Genesis console."[38] The court further noted that there was no basis for assuming that the market for Sega's products would be adversely affected. In any case, an economic loss due to the failure of an attempt to monopolize the market is not a strong equitable basis to defeat fair use.[39]

In considering the second factor, the court acknowledged that "computer programs pose unique problems" for the application of traditional copyright rules.[40] Nonetheless, it ultimately relied on the well-established principle that the copyright laws protect only the expression of ideas, and not ideas themselves. Section 102(b) provides that "[i]n no case does copyright protection . . . extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work."[41] This principle, first enunciated by the Supreme Court in *Baker v. Selden*[42] and codified at 17 U.S.C. § 102(b), is commonly referred to as the "idea/expression dichotomy" or "idea/expression distinction."

Applying this idea/expression distinction to the realm of computer software, proponents of reverse engineering argue that copyright law does not protect mere functional elements of computer programs necessary for the operation of a program, because those elements do not constitute protected expression. Computer object code cannot be read by humans. If disassembly of object code were always an unfair use, the owner of a copyright could obtain a virtual monopoly over functional

37. *Accolade*, 977 F.2d at 1523.

38. *Id.*

39. *Id.* at 1523-24.

40. *Id.* at 1524.

41. 17 U.S.C. § 102(b).

42. 101 U.S. 99, 102-04 (1879). *Accord*, Mazer v. Stein, 347 U.S. 201 (1954) (copyright, unlike patent, gives no exclusive right to the art disclosed; protection extends only to expression of idea); Landsberg v. Scrabble Crossword Game Players, Inc., 736 F.2d 485, 489 (9th Cir.) (*scenes a faire*, incidents, characters, or settings that, as a practical matter, are indispensable, are not copyrightable because copyright "would give the first author a monopoly on the commonplace ideas behind the *scenes a faire*"), *cert. denied*, 469 U.S. 1037 (1984); Sid & Marty Krofft Television Prods., Inc. v. McDonald's Corp., 562 F.2d 1157, 1168 (9th Cir. 1977) ("idea and the expression will coincide when the expression provides nothing new or additional over the idea").

aspects of a work, because no one could read the program to discover those functional aspects.[43] Such a result would effectively provide protection for copyrighted software greater than that provided by the patent laws without meeting the stringent standards of patentability. Moreover, copyright requirements—originality and fixation—are far more lenient than the pre-conditions imposed under patent law, which demands a more rigorous showing of novelty, usefulness, nonobviousness, and disclosure of the best mode of invention.[44] Only after demonstrating all these elements can a patentholder exclude nonlicensees from using its ideas.[45] Even after meeting the requirements for a patent, the patentholder is entitled only to a seventeen-year monopoly. Extending patent-like protection under the copyright laws would grant a monopoly for the author's life plus fifty years.[46]

In addition, as the *Accolade* court recognized, this result would be directly contrary to the mandate of 17 U.S.C. § 102(b). The court in *Accolade* concluded: "Because Sega's programs contain unprotected aspects that cannot be examined without copying, we afford them a lower degree of protection than more traditional literary works."[47]

The only fair use factor that weighed in Sega's favor is the third—the amount and substantiality of the portion used in relation to the copyrighted work as a whole—because Accolade copied the entire program as part of the disassembly process. Nevertheless, the ultimate use by Accolade of the copied material was very limited, and this factor was given little weight by the court.[48]

In sum, the court concluded that three of the four fair use factors weighed in Accolade's favor, and that it had "by far the better case on the fair use issue."[49]

B. The Trademark Analysis

The Ninth Circuit also rejected Sega's efforts to use its trademark as part of the TMSS. The court held that Sega was responsible for the confusion caused by its TMSS because the TMSS has the effect of regulating access to the Genesis, and there was no evidence of public or industry awareness of any feasible alternate method of gaining access.[50] Sega made an affirmative decision to utilize its trademark as an element of a

43. *Accolade*, 977 F.2d at 1526.
44. 35 U.S.C. §§ 101-102, 111-112 (1988).
45. *Id.* §§ 101-03, 154.
46. 17 U.S.C. § 302(a); 35 U.S.C. § 154.
47. *Accolade*, 977 F.2d at 1524, 1526.
48. *Id.* at 1526.
49. *Id.* at 1527.
50. *Id.* at 1528.

functional device. Accordingly, the court held that the consumer confusion caused by the device is Sega's responsibility.[51]

The panel found unpersuasive Sega's claim that the purpose of the TMSS was to prevent pirates in Taiwan from stealing its products. It held that Sega should have foreseen that a legitimate competitor might discover how to utilize the TMSS to gain access to the Genesis, and that legitimate competitors might be discouraged or harmed by the TMSS.[52] Indeed, the court noted that Sega had attempted to use the TMSS for that very purpose. Such a use of a trademark, the court insisted, is "inconsistent with the Lanham Act."[53] As such, the Ninth Circuit reversed the injunction issued by the district court, and remanded the matter for a determination of whether Accolade was entitled to an injunction against Sega because of the TMSS.[54]

VIII
Conclusion

The Ninth Circuit's decision in *Accolade* is consistent not only with the CAFC's *Atari Games* decision but also with a recent trend among appellate courts to protect access to ideas and other nonprotectible expression contained in computer programs.[55]

That trend is broadened further by the approach of the CAFC and the Ninth Circuit toward reverse engineering of computer software. Both circuits, applying Ninth Circuit law, have demonstrated their support for reverse engineering of computer software where it is necessary to develop compatible programs and foster robust competition in what is essentially the aftermarket for compatible games. Rather than permitting copyright holders to use the copyright laws to gain a monopoly over both the protectible and nonprotectible elements of their computer programs, the Ninth Circuit in *Accolade* came down on the side of competition and the consumer in construing the copyright laws to permit the free flow of functional ideas contained in computer programs.

The Ninth Circuit manifested a similar reluctance to construe trademark law to protect the type of lock-out device present in *Accolade*. The prohibition of the use of trademarks to obtain patent-like protection of functional ideas is consistent with the trend toward rejecting overexpansive use of intellectual property rights to stifle competition.

51. *Id.* at 1529.
52. *Id.*
53. *Id.* at 1530.
54. *Id.* at 1533.
55. *See* Lewis Galoob Toys, Inc. v. Nintendo of America Inc., 964 F.2d 965 (9th Cir. 1992), and Computer Assocs. v. Altai, 982 F.2d 693 (2d Cir. 1992).