UIC John Marshall Journal of Information Technology & Privacy Law

Volume 8
Issue 4 *Computer/Law Journal - Fall 1988*

Article 3

Fall 1988

Breaking the Mold: Forging a New and Comprehensive Standard of Protection for Computer Software, 8 Computer L.J. 389 (1988)

Jack Sholkoff

Follow this and additional works at: https://repository.law.uic.edu/jitpl

 Part of the Computer Law Commons, Internet Law Commons, Privacy Law Commons, and the Science and Technology Law Commons

Recommended Citation
Jack Sholkoff, Breaking the Mold: Forging a New and Comprehensive Standard of Protection for Computer Software, 8 Computer L.J. 389 (1988)

https://repository.law.uic.edu/jitpl/vol8/iss4/3

This Comments is brought to you for free and open access by UIC Law Open Access Repository. It has been accepted for inclusion in UIC John Marshall Journal of Information Technology & Privacy Law by an authorized administrator of UIC Law Open Access Repository. For more information, please contact repository@jmls.edu.

NOTES

BREAKING THE MOLD: FORGING A NEW AND COMPREHENSIVE STANDARD OF PROTECTION FOR COMPUTER SOFTWARE

TABLE OF CONTENTS

I. INTRODUCTION

As the science community develops new technologies, and the business community finds easier ways to market them, the legal community must respond to these new events with an attitude of flexibility and adaptability or risk being overwhelmed by them.[1] Recognizing this, the Congress in 1975 created the National Commission on New Technological Uses of Copyrighted Works (CONTU)[2], and following the Commission's recommendations,[3] amended the 1976 Copyright Act in 1980 to include special provisions for computer software.[4] The new laws, by defining a computer program as a "set of statements. . ." and categorizing it as a "literary work" focused solely upon the code element of a computer program. The laws did not reflect, and implicitly rejected, the potential copyrightability of the other elements contained in a computer program. These elements include the program's internal design and structure, the program's display output, and the program's user interface.[5] Because the laws do not explicitly address these elements of a computer program, because judicial decisions have been inconsistent

1. National Commission on New Technological Uses of Copyrighted Works, Final Report 3, [hereinafter *CONTU Report*]

2. *Id.* at 4.

3. The Congress noted that its amendments embodied the recommendations of CONTU, and consequently, several courts have relied on CONTU as legislative history for the Software Protection Act of 1980. H.R. Rep. No. 1307, 96th Cong., 2d Sess. 4, *reprinted in* 1978 U.S. CODE CONG. & ADMIN. NEWS 6460, 6482. *See* Apple Computer, Inc. v. Franklin Computer Corp. 714 F.2d 1240, 1247 (3rd Cir. 1983), *cert. dismissed*, 464 U.S. 1033 (1984); Whelan Assoc., Inc. v. Jaslow Dental Laboratory, Inc. 797 F. 2d 1222, 1241 (3rd Cir. 1986), *cert. denied*, 107 S.Ct. 877 (1987).

4. The Copyright Act is 17 U.S.C. §§ 101-810 and the particular software amendments of 1980 are in 17 U.S.C §§ 101:117. (Software Protection Act of 1980, Pub. L. No. 96-517, § 10(a), 94 Stat. 3028 (1980). These sections defined a computer program as "a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result." 17 U.S.C. § 101. In addition, the Amendments allowed buyers of computer programs to make archival and utility copies of programs without infringing a copyright. Without this latter provision, such a practice, while common in computer use, would be technically illegal. 17 U.S.C. § 117. It should also be noted that CONTU believed that the Copyright Act of 1976 already covered computer programs; the Commission's role was to evaluate the scope and desirability of copyright protection of software. *CONTU Report*, *supra* note 1, at 9, 16.

5. The definitions and actual copyrightability of all the elements of a computer program are discussed in Part III of this Note. Still, simply because the copyright law explicitly protects the code elements of a computer program, and the code produces output, it does not necessarily follow that the output is also copyrightable. This issue, the extent of

and inadequate, and because of the vast increase in the market for computer software, the current copyright laws relating to software no longer respond to the needs of either software developers or the software marketplace.

Since "the design and presentation, or in short, the 'look and feel' of a computer software product often involves much more creativity and often is of much greater commercial value than the program code which implements the product,"[6] the current struggle within the legal and software communities does not concern the protection of computer code, which is clearly protectible.[7] Instead, the debate focuses upon the extent that the copyright law will protect a computer program against *non-literal* copying.[8] Essentially the question is whether current copyright protection of software is underprotective, or whether protecting the "look and feel" of software would be overprotective.[9]

Although it remains the subject of much controversy, "look and feel" is not a legal term.[10] Instead, "look and feel" is a term of the com-

---

copyright protection over a computer program, is the crux of the current debate over the scope of software copyright protection.

6. Ranney, *'Look and Feel' Discussed as Major Copyright Issue* INFOWORLD, Nov. 11, 1985, at 13. *See* Note, *A Thousand Clones: The Scope of Copyright Protection in the "Look and Feel" of Computer Programs*, 63 WASH. L. REV. 195, (1988) [hereinafter *A Thousand Clones*]. It is the program code, however, that receives the most protection under current standards.

7. *See* Apple Computer Inc. v. Franklin Computer Corp., 714 F.2d 1240, 1249 (3d Cir. 1983) (holding that program code in object or source form is protected by the program's copyright); *A Thousand Clones*, *supra* note 6, at 200; *see infra* notes 107-134 and accompanying text.

8. Comment, *Protecting the 'Look and Feel' of Computer Software*, 1 HIGH TECH. L. J. 411, (1987) [hereinafter *Protecting the 'Look and Feel'*]. While several lawsuits have been filed, only two have been decided. *See* Broderbund Software, Inc. v. Unison World, 648 F.Supp. 1127 (N.D. Cal. 1986) and Digital Communications v. Softklone Distributing Corp., 659 F.Supp. 449 (N.D. Ga. 1987). In addition, Lotus Corporation, the maker of the popular *1-2-3* spreadsheet application program, has sued two other software developers for copyright infringement. Lotus claims that the defendants illegally copied the "look and feel" of *1-2-3*. Lotus Dev. Corp. v. Mosaic Software, Inc. No. 87-0074-k, and Lotus Dev. Corp. v. Paperback Software, Inc. No. 87-0076-k (D. Ma. January 12, 1987). *See* Warner, *Lotus Says It May File for Injunction to Halt Sales of 1-2-3 Work Alikes*, INFOWORLD, Jan. 19, 1987, at 1, col. 3. [hereinafter *Lotus*] *See also* Rosch, *The Copyright Law on Trial*, PC MAGAZINE, May 26, 1987, at 157-161 (an article explaining the various arguments made by Lotus in its suits). *See also* Warner, *Lotus Suit Stirs Debate Among Users*, INFOWORLD, Jan. 26, 1987, at 1, col. 3. [hereinafter *Debate*] (explaining the opinions of the "look and feel" controversy within the legal and computer communities). Most recently, Apple Computer, Inc. filed suit against Microsoft Corporation and Hewlett-Packard Corporation (Apple Computer, Inc. v. Microsoft Corp. and Hewlett-Packard Corp. No. 88-20149 (N.D. Cal. Mar. 17, 1988)) for alleged infringement of Apple's Macintosh user interface.

9. Warner, *Debate*, *supra* note 8, at 1; *A Thousand Clones*, *supra* note 6, at 195-96.

10. *Protecting the Look and Feel*, *supra* note 8, at 416. However, such a concept is not foreign to copyright law. It was considered in *Roth Greeting Cards v. United Card Co.* 429

puter industry, that refers generally to "the way the product appears on the screen and the way it works".[11] This Note will define "look and feel" as the program's user interface (the way the user communicates with the computer, i.e. "the way the program works") and the program's visual display output (what the user sees or how the program communicates with the user, i.e. the "way the program looks").[12] Furthermore, this Note will, because the controversy concerns the marketability of software, refer to "look and feel" as the aspects of software to which the consumer most readily responds.[13]

Copyright doctrine is based upon the principle that allowing a creator to obtain personal pecuniary gain from his creations is the best way to promote further creativity and innovation for the *public good.* Consequently, an effective copyright law for software will protect the financially most valuable aspects of a computer program.[14] This Note, however, argues that because current copyright law is not specifically tailored to meet the peculiar nature of software, it cannot adequately protect software, and cannot achieve its goal of promoting intellectual progress. This Note will demonstrate that while current copyright standards protect some aspects of software, they remain, because of the nature of computer software, at best unclear and inconsistent, and at worst inadequate to protect software developers and promote progress in software development. In fact, under these standards, some courts have been forced to stretch the bounds of current copyright law to provide even limited protection to computer software. Consequently, without a legislative standard, there is little *guidance, predictability,* or *consistency* in software copyright protection, and this may prevent some future developers from creating new software.[15] Hence, this Note recommends that Congress create a new work of authorship, "computer software". The new category should explicitly recognize the individual

F.2d 1106, 1110 (9th Cir. 1970). In that case, the court held that the defendant's greeting cards infringed the plaintiff's greeting cards by having a similar "total concept and feel."

11. *Taking the Stand: The Look and Feel Issue Examined* PC MAGAZINE, May 26, 1987, at 155.

12. *See* Pilarski, "User Interfaces and the Idea-Expression Dichotomy, or Are the Copyright Laws User Friendly?" 15 A. I. P. L. A. Q. J. 325, 326 (1987); *Protecting the 'Look and Feel,' supra* note 8, at 416. (the author describes "look and feel" as the program's "design, presentation and output as experienced by the user"). *But see A Thousand Clones, supra* note 6, at 197, 207; Siegel & Derwin, "Copyright Infringement of the 'Look and Feel' of an Operating System by its own Applications Programs" 4 COMP. LAW. 1, 2 (Jan. 1987) (both authors do not differentiate between user interfaces and screen displays).

13. *See supra* note 6 and accompanying text.

14. M. NIMMER, NIMMER ON COPYRIGHT, 1-32 (1987); Arnstein v. Porter, 154 F.2d 464, 469 (2d Cir. 1946). *See infra* section IIIA.

15. *See infra* note 316 and accompanying text.

elements of a computer program that combine to form a work of software and mandate protection of the software work as a whole. It should not protect *only* the software's literary *or* visual aspects.[16]

A new standard that is related to the developer's financial interests would necessarily protect a program's "look and feel" because of its centrality to the software market.[17] In addition, to avoid overprotecting software, the new standard would specify that the new work of authorship, "computer software" consists of four protectible elements: program code, program design, program display output, and program user interface. Although courts currently protect program code, program design, and, to a limited extent, program displays, no court has yet directly protected a program's user interface.[18] This Note's proposal incorporates current judicial standards of protection for software, but also protects, in a limited way, the program's user interface. Most important, the standard directs courts examining computer software to an-

16. Computer software, as noted, is currently considered a literary work under the copyright law. 17 U.S.C § 101. Some courts have held that the display is protectible by a separate copyright on the display as an audiovisual work or compilation. *See* Atari v. North American Philips Consumer Elecs. Corp. 672 F.2d 607 (7th Cir. 1982) (concerning the copyrightability of the video game Pac-Man); Stern Elecs., Inc. v. Kaufman 669 F.2d 852 (2d Cir. 1982) (holding that the display of a video game is an audiovisual work and subject to copyright protection); M. Kramer Mfg. Co. v. Andrews 783 F.2d 421 (4th Cir. 1986) (holding that the audiovisual display protects the underlying program); Digital Communications v. Softklone Distributing, Inc. 659 F.Supp. 449 (N.D. Ga. 1987) (holding that the screen of a communications program can be copyrighted separately and in addition to the code as a compilation and not an audiovisual work). However, in June, 1988, the Copyright Office decided to allow only one copyright registration per computer program. This decision casts doubt upon those cases which hold that a computer program's display and user interface may be copyrightable by a separate copyright in addition to the copyright protecting the program code. *See* "Copyright Office Notice on Computer Screen Registration", 36 BNA'S PATENT, TRADEMARK & COPYRIGHT J. 152, 152-55 (1988) [hereinafter *Copyright Ruling*]. Unlike this Note, the Copyright Office did not recommend the creation of new work of authorship for computer software. Instead, despite the many varying elements contained in a work of software, the Office decided that the authorship which predominates in the work will determine the work's registration class. *Id.* at 155.

17. *But see* Note, *Broderbund Software Inc. v. Unison Inc: "Look and Feel" Copyright Protection for the Display Screens of An Application Microcomputer Program*, 13 RUTGERS COMPUTER & TECH. L. J. 105, 132-33 (1987) [hereinafter *Look and Feel*] (protecting look and feel in programs will overly restrict innovation and competition. The Note's author fails to consider that the purpose of copyright is to promote innovation, not promote cheaper versions of existing works).

18. This Note is separating, for the purpose of analysis, the screen displays from the user interfaces of a computer program. It is a fine distinction to make, since in some sense, any reaction the user has when even just looking at the screen display could be defined as part of the user interface. However, this Note defines the user interface as the particular method the program utilizes to enable the programmer to use the program, not just view it. *But see A Thousand Clones, supra* note 6, at 197, 207 (the author does not distinguish between displays and interfaces).

alyze each of the four ennumerated elements of software *separately* to determine the copyrightable material within each element. By separately analyzing each element, this will enable the court to avoid copyrighting broad ideas, which are not copyrightable. Instead, the court will be able to determine what the copyrightable expression is within each element. After completing this analysis, the proposal then directs the court to cumulate the individual differences and similarities of all copyrightable material *as a whole* to determine whether there was copyright infringement of the entire work.[19] This cumulative standard of infringement serves the public by promoting further innovation and software developement while continuing to protect the creator's investment.[20]

This Note will analyze the current software problem in three sections. Section II will briefly define and explain some of the computer terminology that will be used in this paper. Section III will consider the purposes of the copyright law, the broad scope of copyright law, and the current statutory and judicial formulations concerning software protection. It will focus its analysis on judicial attitudes toward each of the four areas of software: the program's code, the program's design, the program's display, and the program's user interface. This section will demonstrate how some courts, relying on policy considerations, have misconstrued traditional copyright doctrine in order to grant protection to non-literal elements of a computer program. However, these decisions are rare and offer little security or encouragement to developers. Instead, the decisions tend to confuse the whole area, chilling potential developers from creating new software because of their fear that they will be unable to protect their creations against unauthorized copying.[21] Finally, section IV will present in more detail this author's proposal for a new copyright standard for software, and will demonstrate how the proposal works to fulfill the goals of copyright law.

II. COMPUTER BACKGROUND AND TERMINOLOGY[22]

A computer program is a set of instructions that tells a computer to perform a mathematical function, or algorithm.[23] The computer re-

19. Statutorily directing courts as to the factors they should consider in a particular type of case has precedent in 17 U.S.C. § 107, where the statute lists four non-exclusive factors a court should examine when determining whether an adequate defense of fair use exists. Similarly, this Note's proposal provides a *framework* for determining the scope of copyright protection for computer software.

20. *See infra* notes 332-338 and accompanying text.

21. *See infra* notes 330-338 and accompanying text.

22. The information contained in this section about computer technology comes from *Protecting the 'Look and Feel'*, *supra* note 8, at 411-12, n. 4-6; and *Apple Computer, Inc. v. Franklin Computer Corp.*, 714 F.2d 1240, 1243 (3d Cir. 1983).

23. *See infra* text accompanying note 95.

ceives its instructions in the form of binary or object code, which is a series of ones and zeros in various combinations. However, it is difficult for programmers to write computer programs using solely object code. Instead, programmers must write their programs in a higher level language, or source code, such as BASIC, Pascal, or Assembly Language. These languages, once understood, are fairly easy to use, because they are logical and use symbols that bear some connection to their function. A device called a compiler, located inside the computer, translates the source code into object code. *Apple Computer, Inc. v. Franklin Computer Corp.* was the first case to specifically acknowledge that a program was copyrightable in either object code or source code.[24]

Programs themselves can either be contained in Random Access Memory (RAM), a memory device in which different programs can be read in and read over (working like a tape recorder), or stored in Read Only Memory (ROM), in which the program is copied on to a circuit within the computer. The placing of a program in ROM is permanent and cannot be erased; in order to change the program, the circuit must be replaced. In addition, there are two kinds of programs: operating programs and application programs. Operating programs control the internal operations of the computer, such as input/output control, data management and, most importantly, any application programs the computer is running. Application programs are programs that perform a specific function, such as playing a game or performing word processing.[25]

Furthermore, in order to use a program, the user must be able to interface, or communicate, with the computer. The user must be able to input data into the computer in a way that is compatible with the computer program. The particular method in which the user controls and uses the program is called the user interface. There are three basic types of user interfaces: menu systems, command systems, and direct manipulation techniques.[26] Within these three basic interface designs, a programmer can create many variations. Generally, the interface is contained in the program's display output because this is the only aspect of the program the user actually sees. Unlike the other aspects of the display, which communicate results of internal functions from the computer to the user, the interface is that part of the display which enables the user to communicate with the computer. Because the user can only use the program if he can master the interface, and because the only part of the program he sees is the screen display, proper development

24. *Apple*, 714 F.2d at 1249.

25. *Protecting the Look and Feel*, *supra* note 8, at 411-12, n. 4.

26. *See A Thousand Clones*, *supra* note 6, at 197-98. The author explains in more detail the basic types of user interfaces.

and protection for both the interface and the screen display remain critical to the market success of the software.[27]

Unfortunately, the unique nature of computer software poses a difficult problem for copyright law. Identical displays and interfaces can be created with varying forms of program code. Thus, a programmer can steal an interface and display design from another program without copying the other program's protected source code. This process, called "knocking off," means that a scheme of legal protection, which only protects against the illegal copying of program code, will not prevent a subsequent programmer from knocking off an originator's successful user interface and display for use in his own program. If this occurs, the party creating the knock off program will be able to sell its version of the program cheaper, since it was able to avoid display and interface development costs. The lower price of its identical program will clearly reduce the innovator's income stream. For this reason, developers are often more concerned with protecting display elements directly than they are about protecting the program code.[28]

III. CURRENT COPYRIGHT STANDARDS FOR SOFTWARE

A. PURPOSES OF COPYRIGHT

In order to create an appropriate copyright standard for software, it is necessary to determine the basic goals of copyright doctrine. The purpose of American copyright doctrine, which originates in Article I, section 8 of the Constitution,[29] is to *foster the intellectual progress of society*. Copyright law achieves this goal by granting the creator of an "art" a monopoly over the use of the "art." The theory is that this financial reward will encourage people to create and innovate, and society as a whole will benefit from their innovations. As one court has said, "the immediate effect of our copyright law is to secure a fair return for an 'author's' creative labor. *But the ultimate aim is, by this incentive, to stimulate artistic creativity for the general public good.*"[30] Indeed, as Professor Nimmer has noted:

---

27. *See* Ranney, *supra* note 6, at 13.

28. Note, *Defining the Scope of Copyright Protection for Computer Software*, 38 STAN. L. REV. 497, 521 (1986) [hereinafter *Defining the Scope*]; Ranney, *supra* note 6, at 13.

29. "The Congress shall have Power . . . to promote the progress of Science and useful Arts, by securing for limited Times to Authors and Inventors the exclusive Right to their respective Writings and Discoveries."

30. M. NIMMER, NIMMER ON COPYRIGHT, § 1.03[A], at 1-32 (1988), quoting *American Int'l Pictures v. Foreman*, 400 F.Supp. 928 (S.D.Al. 1975) (emphasis added). The Supreme Court has noted that the reward given to the author is a "secondary" purpose to that of furthering the intellectual progress of society as a whole. *U.S. v. Paramount Pictures*, 334 U.S. 131, 158 (1947).

> The primary purpose of copyright is not to reward the author, but is rather to secure 'the general benefits derived by the public from the labors of authors'. . . . [T]he public benefits from the creative activities of authors, and . . . the copyright monopoly is a necessary condition to the full realization of such creative activities.[31]

A copyright standard that fulfills the goals of the doctrine will be one that "create[s] the most efficient and productive balance between protection (incentive) and dissemination of information, [in order] to promote learning, culture and development."[32]

Clearly, tension exists between these two conflicting goals.[33] If the law is too restrictive, and grants creators too many rewards and monopolies, it will curtail competition and innovation because competitive variations of existing products would be forbidden by law.[34] Conversely, if the law is too permissive, and fails to grant creators enough rewards and monopolies, it will also inhibit innovation, since developers would have no incentive to innovate, because their works would not be protected from copying by competitors.[35] Because of these dangers, an *effective* copyright system "must pay particular attention to the *pragmatic* considerations that underlie . . . copyright generally."[36] Consequently, the standard for computer programs should reflect the realities of the computer industry and marketplace and of computer program development.

B. SCOPE OF COPYRIGHT

Because copyright possesses a history of precedent and provides an established framework for protecting creative works, this Note does not suggest that Congress create a *sui generis* form of intellectual property protection for computer software. A *sui generis* solution, while creative, often leads to unforeseen and unmanageable problems and can create more problems than it solves. Because this Note does not seek a *sui generis* solution, its proposal must fall within the scope of traditional copyright protection. This subsection will discuss the scope of current copyright law into which this Note's proposal must fit.

1. *The Differences Between Copyright Law and Patent Law*

Copyright is only one form of intellectual property protection in the United States. It works concurrently with patent law to achieve the

---

31. M. NIMMER, NIMMER ON COPYRIGHT, § 1.03[A], at 1-31, 1-32 (1988).

32. Whelan Associates, Inc. v. Jaslow Dental Laboratory, Inc. 797 F.2d 1222, 1235 (3rd Cir. 1986).

33. *Protecting the Look and Feel*, *supra* note 8, at 419, n. 48.

34. *See infra* note 338.

35. *Defining the Scope*, *supra* note 28, at 498.

36. *Whelan*, 797 F.2d, at 1235 (emphasis added).

optimum balance between protection and the free dissemination of ideas[37]. The differences between a patent and a copyright are significant, and the two work together as a system to promote the intellectual progress mandated by the Constitution.[38]

A patent is a significant monopoly over ideas underlying an invention.[39] The patent owner has complete control over all uses of his invention. However, obtaining a patent is an arduous and expensive process.[40] The individual must demonstrate that the invention is new, useful, and non-obvious.[41] The individual cannot obtain a patent until the Patent Office has determined, by administrative inquiry, that the invention meets patent requirements. If it does, the patent owner will receive a monopoly over the uses of the invention for a period of seventeen years (fourteen years for a design patent).[42]

Unlike a patent, a copyright only protects originality of expression; it does not protect novelty, invention, or an underlying idea of a work.[43] Section 102 of the Copyright Act notes that "in no case does copyright protection . . . extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery. . . ."[44] The copyright applicant need only demonstrate that his work is an "original work of authorship fixed in a tangible medium of expression. . ."[45] Furthermore, no administrative hearing is required, and under the 1976 Act, a copy-

37. *Look and Feel*, *supra* note 17, at 106. Note that in 1984, Congress created a third form of intellectual property especially for semiconductor chips. *See* Semiconductor Chip Protection Act of 1984, Pub. L. No. 98-620, 98 Stat. 3347 (1984) (establishing new chapter 9 in 17 U.S.C.). Because the SCPA only applies to semiconductor chips, copyright and patent remain the primary forms of intellectual property protection in the United States.

38. *Look and Feel*, *supra* note 17 at 106.

39. *Id.*; 35 U.S.C. § 101.

40. *Id.*; 35 U.S.C. § 154.

41. *Id.*; 35 U.S.C. §§ 101-103.

42. *Id.*; 35 U.S.C. § 173; 35 U.S.C. § 154.

43. *Apple*, 714 F. 2d at 1253; NIMMER, *supra* note 31, at 1-31, 1-32.2; *Look and Feel*, *supra* note 17, at 106-07.

44. 17 U.S.C. § 102(b). This language has been used extensively to support the argument that software, and particularly the user interface aspect of it, should not receive copyright protection. *See Apple*, 714 F.2d at 1250-52; *Whelan*, 797 F.2d at 1235; *Look and Feel*, *supra* note 17, at 130-32. *See also*, *CONTU Report*, *supra* note 1, Dissent of Commissioner Hersey, at 36-37, Concurring Opinion of Commissioner Nimmer, at 26-27 (both arguing that no copyright should be given to those elements of a computer program which control actual computer functions).

45. 17 U.S.C. § 102(a). The Copyright Act enumerates seven categories of authorship: literary works, musical works, dramatic works, choreographic works, pictorial works, audiovisual works, and sound recordings. However, as the House Report accompanying the act noted, "[t]he listing [of categories of authorship] is 'illustrative and not limitative,' and that the seven categories do not necessarily exhaust the scope of 'original works of authorship' that the bill is intended to protect. Rather, the list sets out the general area of copyrightable subject matter. . . ." H.R. Rep. No.1476, 94th Cong., 2d Sess. 53 (1976). The

right can be obtained simply by publishing a copyrightable work with a copyright notice.[46] Finally, a copyright monopoly is generally extended for the life of the author plus fifty years.[47] The purpose is to protect the *expression* of the idea, but not to foreclose other possible expressions or variations of the same idea.[48]

Thus, "[u]nlike a patent, a copyright gives no exclusive right to the art disclosed; protection is given only to the expression of the idea—not the idea itself."[49] In addition, a work is not copyrightable if it is purely utilitarian, that is, if it has no expressionable elements separable from any utilitarian functions it may possess.[50] Reconciling these problems, which is essential to establishing software's copyrightability, is not an easy task when the work is a computer program.[51]

2. *Separating the Copyrightable Expression From the Non-Copyrightable Idea in Computer Software*

Because copyright "does not protect ideas, but only expressions of ideas,"[52] the problem that arises, which is especially difficult in the computer program arena, "is defining the underlying 'idea' [and by implication, the protectible expression] of the copyrighted work."[53] Indeed, although this distinction may be the most important element in determining the scope of a copyright on a particular work, "[t]here is no litmus paper test by which to apply the idea-expression distinction; the determination is necessarily subjective."[54] As Judge Learned Hand

---

categorization of a work is important, as different categories of authorship possess different rights.

46. 17 U.S.C § 401 (1986).

47. 17 U.S.C. § 302(a). If the author is anonymous or pseudononymous, or the work qualifies as a work made for hire, then the copyright extends 75 years from date of first publication, or 100 years from creation, which ever expires first. 17 U.S.C. § 302(c).

48. *Look and Feel*, *supra* note 17, at 107; *Apple*, 714 F.2d at 1253.

49. Mazer v. Stein 347 U.S. 201, 217 (1954).

50. NIMMER, *supra* note 31, § 2.08[B], at 2-94. Put another way, if there is only one, or only a limited number of ways of expressing a certain idea, then that expression will not be copyrightable, since a copyright upon that necessary expression would be tantamount to copyrighting the underlying idea.

51. While it might be argued that a program is utilitarian because it performs functions within the machine, it is clear that Congress intended that despite this, computer programs should be copyrightable. *CONTU Report*, *supra* note 1, at 16; *See A Thousand Clones*, *supra* note 6, at 200; *See supra* note 44. The issues raised by the "look and feel" controversy stem from disagreements over the scope of the protection not over the *per se* copyrightability of software.

52. *Whelan*, 797 F.2d at 1234. In other words, a copyright will not protect a discovery, but will protect a writing about the discovery. *See Apple*, 714 F.2d at 1250.

53. Digital Communications Inc. v. Softklone Distributing, Inc., 659 F.Supp. 449, 458 (N.D. Ga. 1987).

54. Atari Inc. v. North American Phillips Consumer Elecs. Corp., 672 F.2d 607, 615 (7th Cir. 1982).

noted, "Nobody has ever been able to ever been able to fix that boundary, [between ideas and expression] and nobody can."[55] Furthermore, Judge Hand added, "Obviously, no principle can be stated as to when an imitator has gone beyond copying the 'idea', and has borrowed its 'expression.' Decisions must therefore inevitably be ad hoc."[56] Still, despite the difficulties in separating ideas and expressions, courts have generally recognized that this process is the best possible method for determining copyrightability. They also note that, if applied correctly, it "accurately conceptualizes the fundamental elements in an artistic creation and balances the competing interests inherent in the copyright law."[57] The determination of what constitutes a work's idea and what constitutes a work's expression, while *crucial* for defining the scope of copyright protection,[58] is a most difficult process. Courts tend to form their decisions based upon policy judgments that try to balance freedom and protection in a way that will best further intellectual progress.[59]

Although courts have articulated several tests for making this determination,[60] courts in computer cases have essentially relied upon a "plurality of expressions" test.[61] In the computer arena, this test was first articulated by the court in *Apple Computer Inc. v. Franklin Computer Corp.*:

> [I]f the same idea can be expressed in a plurality of totally different manners, a plurality of copyrights may result and no infringement will exist. . . . If other methods of expressing that idea [the operating sys-

---

55. Nichols v. Universal Pictures Corp. 45 F.2d 119, 121 (2d Cir. 1930).

56. Peter Pan Fabrics, Inc. v. Martin Weiner Corp., 274 F.2d 487, 489 (2d Cir. 1960).

57. Sid & Marty Krofft Television Prod., Inc. v. McDonald's Corp., 562 F.2d 1157, 1163 n. 6 (9th Cir. 1977).

58. The distinction is crucial because it determines the scope of copyright protection. If the idea and expression are inseparable, then copying of the expression will be allowed, since copyright will not, in any instance, protect ideas. If copying of the expression were not allowed in such a case, the copyright owner would also gain a monopoly over the work's idea, which is antithetical to the doctrine of copyright. *See* Herbert Rosenthal Jewelry Corp. v. Kalapakian, 446 F.2d 738, 742 (9th Cir. 1971); *accord*, *North American*, 672 F.2d at 616.

59. *Protecting the Look and Feel*, *supra* note 8, at 420.

60. *See Protecting the Look and Feel*, *supra* note 8, at 420-30. The author outlines the various tests used by courts to separate ideas from expressions. However, the computer program cases demonstrate that the plurality of expressions described in the text is the test accepted for use on computer software. *See also Apple*, 714 F.2d at 1253 (court determines that there are several different ways to make an operating system); M.Kramer Mfg. Co. Inc. v. Andrews, 783 F.2d 421, 436 (4th Cir. 1986) (court determines that a particular computer game is not an idea); *Whelan*, 797 F.2d at 1236 (court finds that the underlying idea is what the program's purpose is, in this case the management of a dental program since there are several ways of designing a program to serve this purpose). *But see North American*, 672 F.2d at 615-17 (the court uses the abstractions test rather than the plurality of expressions test to determine the expressionable elements of a video game).

61. The name comes from *Protecting the Look and Feel*, *supra* note 8, at 427.

> tem at issue in *Apple*] are not foreclosed as a practical matter, then there is no merger.[62]

In other words, *if there are a variety of ways to program a computer to perform a certain function*, then the function is the non-copyrightable idea, and the particular program performing the function is the copyrightable expression.[63]

For example, in *Apple*, the defendant had sought to make a line of computers compatible with the plaintiff's Apple II line of computers. In order to do this, the defendant copied, literally, the source code and object code of the Apple II's operating system programs. The defendant admitted that it could have written an operating system, one that carried out the exact same functions as the Apple operating system, through a variety of different programs. It had only copied the Apple code because it sought to make its computers compatible with the Apple line of computers. Because the court determined that compatibility was not an "idea,"[64] and found that Apple's particular source code and object code programs were only one method of developing an operating system, the court held that Apple's particular source code and object code programs were copyrightable expression. Apple's programs formed various expressions of the idea of a disk operating system.[65] As the court noted: "If other programs can be written or created which perform the same function as [the programmer's] operating system program, then *that program* is an expression of the idea [of a disk operating system] and hence copyrightable."[66] Similarly, in *Whelan Associates v. Jaslow Dental Laboratory*[67] the court, using analysis similar to the court's analysis in *Apple*, also found that copyrightable expression exists in a computer program. In *Whelan*, the plaintiffs alleged that the defendants had copied their program for the computerized management of a dental laboratory.[68] The court used the "plurality of expressions" test, finding that "the purpose or function of a utilitarian work would be the work's idea, and everything that is not neces-

---

62. *Apple*, 714 F.2d at 1240; *accord M. Kramer Mfg.*, 783 F.2d at 436.

63. *Apple*, 714 F.2d at 1253.

64. *Id.* at 1240. If compatibility were an "idea", then the Apple codes would not constitute copyrightable expressions, since they represent the *only* method of gaining compatibility with the Apple computers. In such a case, copyrighting the programs would be tantamount to copyrighting the idea of compatibility, and this would violate the principles of copyright.

65. *Apple*, 714 F.2d at 1253. In actuality, the court remanded the case in order to let the district court determine if there were other ways to write an operating system program. However, the court voiced its belief that it there were other such programs, so the thrust of the holding is as is stated in the text.

66. *Id.*

67. 797 F.2d 1222 (3rd Cir. 1986).

68. *Id.* at 1238.

sary to that purpose or function would be part of the expression of the idea."[69] Thus, *Whelan* and *Apple* demonstrate that under current copyright doctrine anything not necessary to the purpose or function of the program forms the program's protectible expression.[70]

Both *Apple* and *Whelan* exemplify how courts determine the copyrightable aspects of computer programs under current law. Courts generally use the "plurality of expressions" test on the entire program as a whole; that is, they define the program's idea globally, as an idea that underlies all aspects of the program. This Note argues that a more sophisticated, accurate and effective method would be to conduct the "plurality of expressions" test upon each of the four basic elements of a program.[71] This would prevent a court from defining the program's underlying idea too broadly, and prevent overbroad and inefficient protection of computer programs.[72] Furthermore, the cumulative

69. *Id.* at 1236.

70. *Id.* As *Whelan* and *Apple* demonstrate, the key determination is what the court decides is the underlying idea of the program, because anything not necessary to create that idea will be copyrightable. This Note's proposal suggests that this "idea/expression" inquiry be focused upon each of the four elements of the program independently to avoid overbroad copyright protection of computer programs. *See infra* notes 330 338 and accompanying text. Several courts have endorsed the *Whelan* and *Apple* approach to this key issue. *See M. Kramer Mfg. Co. v. Andrews* 783 F.2d 421, 436 (4th Cir. 1986) and *Broderbund Software, Inc. v. Unison World, Inc.* 648 F.Supp. 1127, 1132-33 (N.D. Cal. 1986). However, the Copyright Office's recent ruling allowing only one copyright per computer program, by asserting that a computer program is a unitary and integrated work, seems to contradict this Note's proposal to independently review the ideas and expressions of each program element. *Copyright Ruling, supra* note 16, at 153. To the extent that the Copyright Office's ruling can be read to prevent this type of inquiry, this Note disagrees with the Copyright Office's ruling.

71. A recent court did not view the work as an entirety when determining what aspects of the program were copyrightable. In *Frybarger v. International Business Machines Corp.*, 812 F.2d 525 (9th Cir. 1987), the court dissected the various features of a video game and determined that each of the features was necessary to the game's idea. Consequently, the court found the game non-copyrightable. *Id.* at 529-30. *See A Thousand Clones, supra* note 6, at 206. This Note only urges that the four basic elements of a program (code, design, display, and interface) be subject to separate idea/expression analysis. This note relies on *Frybarger* only to point out that courts have separated elements of a program when conducting this kind of analysis. It does not endorse the court's particular method or result.

72. *See Protecting the Look and Feel, supra* note 8, at 429. An extreme example will serve to demonstrate the danger of defining the underlying program too broadly. For instance, a court, examining a word processing program, could determine the program's purpose as simply a "word processing program." This might lead to protection of basic interface designs, such as a command interface, since a word processing program does not require a command interface, but could use another kind of interface. This would mean that the copyright holder would possess a monopoly over all command based interfaces. No other program could use a command based interface without that copyright owner's permission. This would be far too broad control and inhibit the development of other programs. *But cf. supra* note 70.

infringement standard proposed in this Note permits a court to determine infringement[73] only *after* considering all elements of the program. This ensures that a court's examination of a work of software will always include an examination of the work as a whole.[74] This Note's proposal also would require that a developer register only one copyright, and not separate copyrights on each element of the program to protect the entirety of his creation.[75] In addition to providing better protection over the software product, the single copyright for computer software proposed by this Note, and recently endorsed by the Copyright Office,[76] also eliminates the risk that an author will fail to protect his work because he did not register each element of the program sepa-

---

73. The method of determining infringement is of equal importance to that of determining protectible expression. Because a court will only issue an injunction against an infringing program, the ease or difficulty of establishing infringement will determine the extent of actual protection a programmer will receive over his work.

74. In this respect, this Note's proposal reflects the recent Copyright Office decision to register all aspects of a computer program under a single copyright registration. *See Copyright Ruling, supra* note 16 at 153.

75. As has been mentioned, on June 3, 1988, the Copyright Office decided "generally to require that all copyrightable expression embodied in a computer program, including computer screen displays . . . be registered on a single application form. . . . The office finds that in the interest of a clear, consistent public record, our registration practices should discourage piecemeal registration of parts of works." *Copyright Ruling, supra* note 16, at 153. In the sense that the Copyright Office's ruling permits recognition of all of the elements of computer program as a complete whole, this Note believes that the Office's decision wisely increases the protection given to software developers over their creations. However, the Office's ruling does not appear to permit protection for textually based screen displays. *Id.* at 152. This Note disagrees with this part of the Office's decision, and instead agrees with the court's decision in *Digital Communications Assoc. v. Softklone Distributing, Inc.* 659 F.Supp. 449, 455-56 (N.D. Ga. 1987) which held that a textual display was copyrightable as a compilation.

In essence, this Note agrees with some aspects of the Copyright Office's ruling, but also agrees with some of the previous judicial formulations on the issue. It agrees with the Copyright Office that "all copyrightable elements embodied in the work [should be] covered by [a] single registration," *Copyright Ruling, supra* note 16, at 153; however, it also agrees with the *Softklone* court that it is necessary, to adequately protect software works, to examine non-literal elements of software separately from the literal elements. This Note suggests, therefore, that one copyright cover all elements of the program, but each of the four elements of software be analyzed separately for ideas and expressions. Because after the Office's ruling current copyright formulations do not appear to permit this kind of analysis, this Note suggests that the Copyright Law be amended with this Note's proposal. While this proposal would alter some of the formalities of copyright law to fit the peculiar nature of software, this Note's proposal still remains well within the scope and purposes of copyright doctrine and does not form a *sui generis* form of intellectual property protection.

76. The Copyright Office has only held that it will register all aspects of a computer program under one copyright. It has not followed the other, more significant, proposals this Note makes.

rately.[77] The point to reiterate, however, is this Note's proposal still maintains the traditional copyright formulations established in *Apple* and *Whelan*, and would only protect the expressionable elements of computer programs, not the ideas underlying the programs.

3. *Non-copyrightability of Processes and Systems*

Section 102(b) of the Copyright Act, in addition to preventing copyright protection of ideas, also prevents copyright protection of processes and systems. This is a traditional tenet of copyright, dating from the Supreme Court's decision in *Baker v. Selden*.[78] Although the case is old, it still remains relevant to copyright cases that involve potentially utilitarian works.[79] *Baker v. Selden* concerned the publication of a book that contained a system for accounting and also included forms for this system. The defendant's work made use of the same accounting system as the plaintiff's, but used forms that had a different style of columns and headings. The plaintiff charged that the defendant had infringed upon his copyright by *using* the same *accounting system*; he argued that the defendant's use of different looking forms was immaterial.[80] The Court rejected this argument, noting that the accounting system itself was not copyrightable, although it might have been patentable.[81] Copyrightability, the court asserted, would not apply to a system which was *used* and then described in a copyrighted work. As the Court noted: "[T]he teachings of science and the rules and methods of useful art have their final end in application and use . . . [b]ut as embodied and taught in a literary composition or book, their essence consists only in their statement. This alone is what is secured by the copyright."[82] When applied to computer software, this means that computer *functions* are not copyrightable.[83] As the *Apple* court held, "Apple does not seek to copyright the *method which instructs the computer to perform its operating functions but only the instructions themselves*."[84] Thus,

77. *Defining the Scope*, *supra* note 28, at 530. The Copyright Office's recent ruling also eliminates this risk.

78. 101 U.S. 99 (1879).

79. *Whelan*, 797 F.2d at 1235. "Utilitarian" refers to a work that is functional. For example, an accounting system is not copyrightable because it helps organize financial records. Novels, on the other hand, are purely aesthetic.

80. *Baker*, 101 U.S. at 100-01.

81. *Baker*, 101 U.S. at 101-03. *See supra* notes 37-51 and accompanying text.

82. *Baker*, 101 U.S. at 104.

83. In other words, the mathematical functions, which enable a computer to achieve the purposes the programmer seeks to achieve, are not copyrightable. The program itself, however, which instructs the computer to perform certain functions, is copyrightable.

84. *Apple*, 714 F.2d at 1251 (emphasis added). More specifically, Apple Computer, Inc. did not try to protect the functions and tasks of an operating system program. For example, it did not attempt to protect the way the operating system controlled the organization

the operation or function that the program instructs the computer to execute is not copyrightable, but the particular program instructions are copyrightable. In other words, expression describing the function is copyrightable, whereas the function itself is not copyrightable.[85] In this sense, utilitarian concerns reflect the same policy concerns as the idea and expression analysis.

Unfortunately, *Baker v. Selden* has been given an expansive interpretation to mean that if a particular work is merely *used*, then the entire work is not copyrightable.[86] This was Franklin's argument in the *Apple* case. Franklin asserted that because Apple's programs were *used to perform* disk operating system functions, Apple could not copyright its programs.[87] This approach, however, despite the *Baker* decision was specifically rejected by CONTU,[88] and the *Apple* court.[89] The court said that although the programs were "used ultimately in the implementation of a process [this] should in no way affect their copyrightability."[90] Hence, the terms of a computer program remain copyrightable despite the fact that they are ultimately functional or used in a process (in the sense that they work to operate a computer). A program's operationality does not preclude its copyrightability.[91] This Note's proposal is in accordance with this traditional copyright analysis; it does not extend protection to purely functional processes or systems (i.e., the actual operating system in *Apple*) which may exist in a software work.

C. STATUTORY FORMALITIES

All creative works, in order to obtain copyright protection, must meet certain statutory formalities. Any new standard of protection for computer software, unless it is a *sui generis* standard, must work within the existing copyright framework. The Copyright Act of 1976 defines the subject matter of copyright as "original works of authorship fixed in any tangible medium of expression."[92] A work of authorship is defined

---

and manipulation of data files. It only sought to protect its particular data file management program, that is, the actual instructions that instructed the computer to perform particular data file management functions. 714 F.2d at 1244, n. 4. Franklin Computer Corp. was free to design a program that performed the same functions, in the same way, but it was not free to use Apple's computer program. It had to design its own instructions to make the computer perform the desired functions.

85. *Defining the Scope*, *supra* note 28, at 508-09.

86. *See Baker*, 101 U.S. at 103.

87. *Apple*, 714 F.2d at 1251-52.

88. *CONTU Report*, *supra* note 1, at 21.

89. *Apple*, 714 F.2d at 1252.

90. *Id.* (quoting *CONTU Report*, *supra* note 1, at 21).

91. *Apple*, 714 F.2d at 1152.

92. 17 U.S.C. § 102(a) (1982). The Copyright Act does not require an author to register his work with the Copyright Office. Instead, an author automatically gains a copyright

by, but not limited to, one of the seven categories enumerated in the Act.[93] The fixation requirement requires that the work be "sufficiently permanent or stable to permit it to be perceived, reproduced, or otherwise communicated for a period of more than transitory duration."[94]

Currently, the Act defines a computer program as "a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result."[95] Congress avoided creating a specific work of authorship for computer programs, because it believed that computer programs fell within the realm of literary works.[96] However, categorizing software as a literary work remains problematic because it reinforces the misconception that a work of software consists only of the program code. Such a misconception encourages protection of *only* the code element of the program.[97] Although the literary work definition encompasses "statements or instructions,"[98] and thus includes program code (since program code is essentially a set of instructions that directs the computer to perform a certain function), this definition does not encompass displays and interfaces. This is because displays and interfaces "radically differ from the programming code. They bear little resemblance to literary works, thus the proper analysis treats the elements of look and feel as non-literary works of authorship."[99] The Copyright Office, apparently recognizing this, recently decided to register certain programs, in which non-literal elements predominate, as audiovisual works.[100] Consequently, in an effort to protect both the non-literary together with the literary elements of software, this Note pro-

---

as soon as his original work is fixed in a tangible medium of expression. *Id.* However, in reality, a registration is required for protection of the copyrighted material. Section 411 requires that in order to bring an action for infringement, the work must be registered with the Copyright Office. 17 U.S.C. § 411 (1982).

93. *See supra* note 45 for a listing of the seven categories of works of authorship.

94. 17 U.S.C. § 101 (1982).

95. *Id.*

96. *CONTU Report supra* note 1, at 16.

97. *Protecting the Look and Feel, supra* note 8, at 430-32.

98. 17 U.S.C. § 101 (1982). The full definition of a literary work is: "works, other than audiovisual works, expressed in words, numbers, or other verbal or numerical symbols or indicia, regardless of the nature of the material objects, such as books, periodicals, manuscripts, phonorecords, film, tapes, disks or cards, in which they are embodied."

99. *Protecting the Look and Feel, supra* note 8, at 432. *But see Softklone*, 659 F.Supp. at 462 (the court notes that a textual screen display is essentially a literary work on the screen). However, the court concluded that the work was copyrightable as a "compilation" of material, and not merely as a literary work. *Id.* at 463. The court also noted that because a display screen and interface can be created with various forms of code, the display could not qualify as a copy of the program code. *Id.* at 456.

100. *Copyright Ruling, supra* note 16, at 153. Unfortunately, the Copyright Office's solution goes too far; registering works as either literal *or* non-literal works necessarily excludes certain aspects of the program.

poses establishing a new work of authorship specifically tailored for computer software.[101]

Furthermore, current law does not define the scope of copyright protection given to computer software. Congress intentionally did not include any provision concerning the scope of protection for software in the copyright law because CONTU, citing the rapid pace of changing technology, recommended leaving this question open for the courts to decide.[102]

Unfortunately, Congress's failure to establish or define the scope of copyright protection for computer software, while perhaps appropriate when software development was in its infancy, now may be quite limiting to the software industry. Because Congress's standard merely legitimized copyright as the appropriate mechanism for protecting software, the statutory definition of a computer program and its categorization as a literary work of authorship has become less viable as software development has become more popular and more complex.[103] The Congressional definition fails to address issues concerning the design, presentation, and marketability of software. The current statutory formulation, therefore, has become an insufficient tool in which to protect computer software developers and to stimulate intellectual innovation.[104]

D. CURRENT JUDICIALLY FORMULATED PROTECTION STANDARDS FOR COMPUTER SOFTWARE

Before proposing a change in current law, it is necessary to understand the current standards. While statutes provide only a framework for judicial interpretation, judicial decisions determine current law. This section will show how courts have dealt with the four aspects of a work of software: the program's literal code, the program's design, the program's display, and the program's user interface.

1. *The Initial Protection: Protecting the Code of a Computer Program*

In the Software Protection Act of 1980,[105] Congress, following the

101. *See infra* notes 330-338 and accompanying text.

102. *CONTU Report*, *supra* note 1, at 22-23. "[T]he many ways in which programs are now used and the new applications which advancing technology will supply may make drawing the line of demarcation more and more difficult. To attempt to establish such a line in this report written in 1978 would be futile. . . . [The] line should be drawn on a case-by-case basis by the institution designed to make fine distinctions—the federal judiciary."

103. *Protecting the Look and Feel*, *supra* note 8, at 445.

104. *Protecting the Look and Feel*, *supra* note 8, at 430-32.

105. Pub. L. No. 96-517, § 10(a), 94 Stat. 3028 (1980) (codified as amended at 17 U.S.C. §§ 101:117 (1983)).

specific recommendations of CONTU, amended the Copyright Act to specifically include computer programs within the subject matter of copyright.[106] However, it was not until *Apple Computer, Inc. v. Franklin Computer Corp.*[107] that the *courts* protected both the source code and the object code versions of a computer program from unauthorized copying.[108] Because the court held that Apple's operating system programs, despite possessing potential utilitarian applications, were copyrightable,[109] *Apple* remains a landmark case in the struggle for the copyright protection of computer software,[110] and serves as the starting point for any analysis of current judicial attitudes regarding the copyright of computer software.

Although the facts of *Apple* were noted briefly above, it is necessary to restate them here in greater detail.[111] The plaintiff in *Apple*, Apple Computer, Inc., sought a preliminary injunction against the defendant, Franklin Computer Corp., seeking to enjoin it from infringing upon copyrights Apple held on various computer programs.[112] The programs at issue were operating system programs used on Apple's popular Apple II line of computers.[113] Apple Computer had successfully built and marketed the Apple II; and, like all personal computers, it required an operating system to run the various application programs designed for it. Following Apple's marketing success, Franklin Computer designed its own computer to be Apple compatible. By making its own computer (the Franklin ACE 1000) compatible with the Apple II, Franklin's computer could then use the same application programs and peripheral devices designed for the Apple II.[114] Franklin believed that the ACE 1000 would thus attract the large amounts of consumers then purchasing the Apple II. However, in order to make the ACE 1000 compatible with the Apple II, the ACE 1000 had to use the same type of operating system as the Apple II. Unfortunately for Franklin, Apple owned copyrights on the operating system programs necessary to achieve Apple II compatibility.[115] Despite knowing this, and recogniz-

106. *CONTU Report*, *supra* note 1, at 1; 17 U.S.C §§ 101:117.

107. 714 F.2d 1240 (3rd Cir. 1983).

108. *Id.* at 1249.

109. *Id.* at 1249-54. *See infra* text accompanying notes 62-66.

110. *Protecting the Look and Feel*, *supra* note 8, at 411. The court itself noted that because the ruling dealt specifically with the copyrightability of software, it would have "considerable significance to the computer services industry." *Apple*, 714 F.2d at 1242.

111. *See supra* text accompanying notes 62-66.

112. *Apple*, 714 F.2d at 1242.

113. *Id.* at 1243-44 n. 4.

114. *Id.* at 1242-43. A peripheral device is a mechanism that a user can add to the computer (usually by plugging into the main board of the computer an additional circuit board) to install a printer, modem, or other similar device.

115. *Id.* at 1243, 1244 n.4, 1245.

ing the possibility of a copyright infringement action against it, Franklin copied the Apple programs.[116] Franklin believed that since the programs were *operating system programs*, they were not copyrightable because they were a functional or utilitarian work.[117] The court rejected this argument. Instead, the court held that computer programs, either in object or in source code were copyrightable,[118] that a computer program was still copyrightable even if imbedded in the functional ROM of a computer,[119] and finally, that operating system programs were also copyrightable works.[120]

The heart of the court's opinion focused upon the copyrightability of operating systems.[121] Franklin's argument against copyrightability of operating systems rested primarily on two points. First, Franklin argued that an operating system, because it controls the internal processes of a computer, is a process or function under the copyright law and is *per se* not copyrightable.[122] The court rejected this argument, holding that rather than protecting the operating system *functions*, Apple's copyright only protected the actual *code instructions* themselves. The court compared the program instructions to instructions included in a manual describing "the necessary steps to activate an intricate complicated machine."[123] In such an instance, the instructions, which *describe* the process are copyrightable; the process itself is not copyrightable. Others are still free to write different instructions describing and *accomplishing* the same process. Apple's copyrights did not prevent Franklin from writing new instructions (a new program code) that would accomplish the same process as the Apple programs. It could not, however, simply copy Apple's programs.[124] The court held that the fact that "the words of program are used ultimately in the implementation of a process should in no way affect their copyrightability . . . . [Copyrightability] is unaffected by the fact that [the program] direct[s] the actions of those . . . who carry out the process."[125] Hence, the court

116. *Id.* at 1245.

117. *Id.* at 1245, 1249.

118. *Apple*, 714 F.2d at 1249. The court relied upon the specific language of the Copyright Act and also upon the CONTU report's findings calling for protection of computer programs. *Apple*, 714 F.2d at 1248. *See infra* text accompanying notes 62-66 and notes 72-91.

119. *Apple*, 714 F.2d at 1249. The court held that simply because a ROM was a part of a machine did not mean that a program fixed in the ROM was also utilitarian and thus not copyrightable.

120. *Id.* at 1253.

121. *Id.* at 1249-54.

122. *Id.* at 1250.

123. *Id.* at 1251.

124. *Id.* at 1253.

125. *Id.* at 1252 (quoting *CONTU Report*, *supra* note 1, at 21).

held that even though an operating system program enables the computer to run application programs, which, at least theoretically is a function, the program remains copyrightable. The court opined that Congress had intended to spur innovation by protecting, through the copyright law, *all* types of computer programs. A court cannot vitiate that Congressional intent because a program ultimately directs the computer to carry out a function.[126]

Franklin also argued that the program's idea and expression were inseparable. It asserted that there were only a limited number of ways to create an operating system for the Apple II. Allowing Apple to protect its operating system, therefore, would be allowing Apple to copyright the *idea* of an operating system for the Apple II which would violate established copyright principles.[127] Indeed, "copyright protection will not be given to a form of expression necessarily dictated by the underlying subject matter."[128] Thus, the court had to determine whether an operating system could be written without the use of the Apple code, for "if other methods of expressing that idea [the functions of a disk operating system] are not foreclosed as a practical matter, then there is no merger [and the program codes are copyrightable].[129] Although the court remanded this determination to the district court, it noted that Franklin had conceded that some of the programs at issue could be rewritten in different ways than the Apple programs, so presumably the programs would be copyrightable.[130]

Furthermore, the court rejected Franklin's argument that *compatibility* was an issue when determining the copyrightability of a computer program. Franklin had argued that there were only a limited number of ways to "arrange operating systems to enable a computer to run the vast body of Apple-compatible software."[131] The court, in rejecting this contention, noted that

> [t]his [Franklin's compatibility] claim has no pertinence to either the idea/expression dichotomy or the merger . . . The idea of one of the operating system programs is, for example, how to translate source code into object code. If other methods of expressing that idea are not foreclosed as a *practical matter*, then there is no merger. Franklin may wish to achieve total compatibility with independently developed application programs written for the Apple II, but *that is a commercial and competitive objective which does not enter into the somewhat metaphys-*

126. *Id.* at 1252-54.

127. *Id.* at 1252-53.

128. *Id.* at 1253 (quoting *Freedman v. Grolier Enterprises, Inc.*, 179 U.S.P.Q. 476, 478 (S.D.N.Y. 1973).

129. *Apple*, 714 F.2d 1253.

130. *Id.*

131. *Id.*

ical issue of whether particular ideas and expressions have merged.[132] This element of the *Apple* holding remains highly significant to the computer software industry. Many members of the software community argue against protection of displays and interfaces because this would endanger clone programs and would end standardization of interfaces.[133] However, the *Apple* holding clearly indicates that *compatibility is not a relevant part of copyright analysis.*

Apple Computer, Inc. v. Franklin Computer Corp. remains the seminal case in the copyright/software area for several reasons:[134] First, because it establishes that a program's eventual use in a utilitarian process does not preclude its copyrightability; second, because it rejects compatibility as an issue in determining copyrightability; and third, because it suggests that courts use the "plurality of expressions"[135] test when analyzing the copyrightability of a computer program. In view of the success of this test, this Note proposes *that courts subject each element of the program (the code, the internal design, the display, and the user interface) to the plurality of expressions test.* Those portions of each element which could be expressed in a variety of ways would be protectible. After completing this analysis, the court would then cumulate the various differences and similarities between each element of the two programs and determine if, as a "whole", the programs were substantially similar to each other and a finding of infringement was appropriate.[136]

2. *Beyond Literalism: Protecting the Structure and Internal Design of a Program*

Although *Apple* involved a situation of identical code copying, most infringement cases do not take this form.[137] In *Whelan Associates v.*

---

132. *Id.* (emphasis added). *See A Thousand Clones, supra* note 6, at 214-16.

133. *Look and Feel, supra* note 17, at 134; *See Common Law, Uncommon Software*, 47 U. PITTSBURGH L. REV. 1037, 1103 n. 167.

134. However, this Note recognizes that if read narrowly, the case only holds that the copyright on a computer program only prevents literal copying of program code. *Apple*, 714 F.2d at 1245 ("[t]he variations [between the programs] that did exist were minor, consisting merely of such things as deletion of reference to Apple or its copyright notice.").

135. *See supra* notes 61-71 and accompanying text.

136. *See infra* notes 330-338 and accompanying text.

137. It is precisely because of this that the "look and feel" issue has arisen. Most infringement comes from "knock offs" of existing programs written in different code. *See also* Stern Electronics, Inc. v. Kaufman 669 F.2d 852, 855 (2d Cir. 1982) (in a case involving video games, the court noted that a copyright on the code alone "would not have prevented a determined competitor from manufacturing a 'knock off'. . .") *But cf. Copyright Ruling, supra* note 16 at 153-55. The Copyright Office's recent ruling providing for one copyright per computer program means that a separate copyright on the screen display is no longer necessary or available to protect the display element of the program.

Jaslow Dental Laboratory, Inc.,[138] the court examined programs that had entirely different program codes. *Whelan* concerned two programs developed to aid in the management of a dental laboratory.[139] The plaintiff's program "Dentalab" was designed primarily for mainframe computers, while the defendant's program "Dentcom" was designed for smaller, personal computer systems.[140] The defendant, the plaintiff's former business partner, obtained "surreptitiously and without consent"[141] a copy of the plaintiff's program code. Using this copy as a guide, but utilizing different program code, the defendant designed his dental laboratory management program. Not surprisingly, his program was similar to the plaintiff's program.[142] Recognizing this, the plaintiff sought to enjoin the distribution of the defendant's program.[143]

Initially, the plaintiff had a serious problem. The defendant's program, while clearly based upon the plaintiff's program, was not a literal copy, nor a translation of it.[144] Consequently, there was no *literal* infringement of the plaintiff's program, and the plaintiff needed an alternative theory of copying (other than literal copying, which had been the case in *Apple*) in order to stop the defendant from distributing its version of the plaintiff's program. Fortunately, the court found such a theory. First, the court recognized that under traditional copyright law, literary works can be infringed without substantial similarities existing between the *literary elements* of the original and infringing works.[145] Second, and more important, the court found that creation of a computer program's literal elements (the encoding of the program code) formed "a comparatively small part of programming. By far the larger portion of the expense and difficulty in creating computer programs [was] attributable to the development of the structure and logic of the program . . . "[146] Because encoding the program takes the least effort,

---

138. 797 F.2d 1222 (3d Cir. 1986).

139. *Id.* at 1225.

140. *Id.* at 1226-27.

141. Whelan Associates v. Jaslow Dental Laboratory, Inc., 609 F. Supp. 1307, 1314 (E.D. Pa. 1985).

142. Surprisingly, the defendant advertised his program as being similar to the plaintiff's program. *Whelan*, 797 F.2d at 1227. Interestingly enough, most "clone" developers stress the similarity of their products to the original developer's program.

143. *Id.* at 1229.

144. *Id.* at 1228. If it had been found to be a translation, then the defendant would have been enjoined from distributing it because the 1976 Act gives the copyright owner all rights to any derivative works and a translation is a derivative work. 17 U.S.C §§ 101:106 (2). Similarly, if the codes of the two programs had been identical, the *Apple* case would have provided precedent in which to find infringement. *See supra* notes 107-134 and accompanying text.

145. *Whelan*, 797 F.2d at 1234. Again, a computer program is a literary work under current copyright law.

146. *Id.* at 1231. In deciding that encoding a computer program was the *least* expensive

protecting the program solely against literal code copying provides the least amount of the protection for the developer of the original program. Instead, by protecting the program code's *design*, or the "*manner* [or design] in which the program operates, controls and regulates the computer in receiving, assembling, calculating, retaining, correlating, and producing useful information either on a screen, print-out or by audio communication,"[147] the court could protect the financially most valuable and intellectually most innovative aspects of the program. As the court noted, "[t]he rule proposed here, which allows copyright protection beyond the literal computer code, would provide the proper incentive for programmers by protecting their most valuable efforts [which are program designs]. . . "[148] The court understood that every computer program aims to solve a certain problem; in this case, the programs sought to efficiently manage a dental laboratory.[149] The court tried to protect *how* the programmer had designed his computer program to solve this problem. The actual problem or purpose underlying the pro-

---

part of programming, the *Whelan* court outlined the process of program creation into several steps. First, the programmer must decide what the program will try to accomplish or what problem the program will solve. Second, the programmer must design a program to solve that particular program. Generally this is accomplished through flow charts or similar logical structures. Third, the programmer must determine how the program will work, and what kinds of programming techniques he will use to follow the logic of the flow chart; more particularly, what kinds of subroutines or program modules he will use in the program, and how exactly the program will input data and arrange the data within the program. As the court noted, "each solution may have particular characteristics— efficiencies or inefficiencies, conveniences or quirks— that differentiate it from other solutions and make the overall program more or less desirable." Finally, after the program is designed, the programmer transcribes it into source code so it will run on the computer. *Id.* at 1229-31.

147. *Id.* at 1239 (emphasis added). This clearly includes the program's particular arrangement of subroutines and code modules. *See Defining the Scope*, *supra* note 28, at 526. The author argues that "copyright law should also recognize that organizing and connecting the modules and subroutines is a protectible act of authorship. . . .[Copyright protection] which considers the overall design of a computer program and the interrelationship of its best parts meets the needs of the both program developer and society."

148. *Whelan*, 797 F.2d at 1237.

149. The court held that "it is clear that the purpose of the utilitarian Dentalab program was to aid in the business operations of a dental laboratory." *Id.* at 1238. *But see Protecting the Look and Feel*, *supra* note 8, at 425, n. 85. (arguing that anything could be classified as the program's purpose, and the purpose will determine the extent of the protection given to the software. However, the author fails to consider that CONTU noted that courts would best be able to define the amount of protection required for a particular work precisely upon such an ad hoc basis. *CONTU Report*, *supra*, note 1, at 22-23). The court also noted, however, that the program's design would not always be protected, if the purpose of the design to perform a *particular* function in a *particular* way, and the program design was the *only method* for performing the function in that particular way. *Whelan*, 797 F.2d at 1238 n.34.

gram can not be protectible; it is an idea.[150] Finally, the court, noting that there was neither a doctrinal, nor a policy reason to limit the copyright protection of computer programs to the literal copying of program code, held that a computer program copyright protects not only the program code, but also the program's internal arrangement or logical structure, sequence, and organization.[151] In short, *Whelan* holds that a copyright on a computer program protects the "design" of the computer program.

Although some commentators have expressed concern that the *Whelan* decision might be extended to include all aspects of a computer program,[152] the court's application of its standard, and particularly the court's use of evidence in determining infringement, demonstrate that by protecting the "design" of a program, the court only meant to protect the *internal design of a program*.[153] The court used three elements of the program to prove infringement.[154] The court examined file structures, screen displays,[155] and significant subroutines contained in the

150. The *Whelan* court applied a version of the *Apple* court's "plurality of expressions" test and concluded that "everything that is not necessary to that purpose [or the solving of that problem] would be part of the expression of that idea [and thus protectible]." *Id.* at 1236. Because the court found that "[t]here is evidence in the record that there are other software programs for the business management of dental laboratories in competition with plaintiff's program [and there was] no contention that any of them infringe although they may incorporate many of the same ideas and functions," *Id.* at 1229-31. it concluded that the program's individual design was copyrightable and protected by the plaintiff's copyright on the program code. *But cf. supra* note 149.

151. *Whelan*, 797 F.2d at 1224-25.

152. *Protecting the Look and Feel*, *supra* note 8, at 426; *Common Law, Uncommon Software*, *supra* note 133, at 1103; *Case Comment*, 8 COMP. L. J. 535, 541-42 (1987). While this Note argues that protecting the internal and external aspects of a computer program through a single, though limited, copyright would be a good goal for copyright, it does not, however, argue that *Whelan* stands for this proposition. Still, the commentator's concern that *Whelan* could be misinterpreted to create virtually unlimited copyright protection over a computer program is clearly warranted, as one court has erroneously interpreted *Whelan* in order to justify overbroad protection of a computer program. *See* Broderbund Software v. Unison World 648 F.Supp. 1127, 1133. *See also Case Comment supra* note 152 at 540-42.

153. "*Whelan Associates* should not be extended beyond its facts. Instead, the proper interpretation of *Whelan Associates* is that . . . copyright protects more than the literal code; [and] the expression of a program includes the structure, sequence and organization of the *internal* components of the program . . ." *Common Law, Uncommon Software, supra* note 133, at 1103.

154. The court, in considering the file structures, screen displays, and subroutines sought to "make a qualitative, *not quantitative*, judgment about the character of the work as a *whole* and the importance of the substantially similar portions of the work." *Whelan*, 797 F.2d at 1245 (emphasis added).

155. The defendants made an argument that because screens can be created with a variety of code formulations, they are completely irrelevant to a determination concerning infringement of the underlying program, and they should not be part of the infringement

program code[156] to determine whether the programs were similarly designed. The court was careful when admitting the screens into evidence, and took pains to make clear that evidence of the screens's similarities alone might be insufficient to find copyright infringement.[157] It did not examine any similarities in user interfaces or other external functions, save the limited usage of the display screens. Because under then current law, these external aspects were protectible, if at all, by copyrights separate from the computer program.[158] The court's focus on the program's internal elements demonstrates that the court sought to protect only the *internal* "design" of the computer program and the creativity associated with it. *Whelan*, therefore, should not be interpreted as meaning any more than this limited, though extremely significant, holding.[159]

Despite the controversy and misinterpretation stemming from it, the *Whelan* decision remains important to the computer industry, and more specifically, helps to justify this Note's proposal to protect all the elements of software through one copyright. The case legitimizes the idea that the creation of software involves far more than writing a code of instructions; it involves the creation of a tangible entity that solves a problem. As long as there are other ways of solving that problem, the copyright law should protect the programmer's particular method of solving the problem.[160] Such protection will encourage other develop-

---

analysis. The court, however, rejected this assertion, holding that while under then current law the screens were considered separate elements from the underlying programs, they were still part of the program, *bearing some relation to the "whole" of the program*, and could be considered as evidence of substantial similarity of the underlying programs. *Id.* at 1243-46. The court's inclusion of the screens as evidence provides another example of the benefits of a *holistic approach to computer software as proposes in this note. See cf. Copyright Ruling, supra* note 16 at 152-53 (finding that displays are not separate elements from their underlying programs).

156. Subroutines are mini-programs contained in a computer program that enable the programmer to make the computer perform certain functions repeatedly. In this way, the programmer does not have to rewrite the same instructions over and over again in the program code; he can simply instruct the computer to go to a certain subroutine, perform the routine, and then continue executing the main program.

157. *Id.* at 1244 n.45.

158. Note that the Copyright Office's recent decision on computer software makes the existence of separate copyrights on the same work of software unlikely. *See Copyright Ruling, supra* note 16, at 152-55. Although this Note proposes that external and internal elements should be protected by one copyright, it also provides that each element should be subjected to an independent idea/expression analysis. Without this separate analysis, a broad copyright leads to the danger that utilitarian or functional aspects of the program would be copyrightable. These kinds of elements, of course, are beyond the scope of copyright law.

159. *Contra Broderbund*, 648 F.Supp. at 1133. *See Case Comment, supra* note 152, at 540-42.

160. Clearly, the copyright law should only protect those aspects of the program that

ers to create *other* and possibly better ways of solving the same problem. This will enable society as a whole to progress and discover better and more efficient solutions to its problems. Protection of software, then should in no instance be limited to literal code copying.

3. *Pragmatic Visions: Protecting the Output of Computer Programs Through the Copyrightability of Display Outputs*

Whelan did not extend the scope of copyright protection for a computer program to the program's screen or audiovisual display.[161] However, many courts, within the context of video games, and most recently in a case of a communications application program, have been willing to protect a program's display screens with either an audiovisual copyright on the entire work, or with a separate copyright from that of the program code.[162] It is important to reiterate, for the sake of clarity, that the screen display (what the user sees) *is not* the same as the user interface (how the user uses the program). This distinction is significant, but can get blurred easily.[163] A good way to think of the difference might

---

are not necessary to solving the particular problem at issue. This Note's proposal addresses this concern by requiring a separate idea/expression analysis for each of the program's elements. This analysis would, because each element is part of the copyright for the whole program, be conducted within the scope of the entire program. For example, the court might inquire, "What elements are necessary to a menu system user interface operating in a spreadsheet program?"

161. *Case Comment*, *supra* note 152, at 538-39; *Softklone*, 659 F.Supp. at 455. *Contra Broderbund*, 648 F.Supp. at 1133.

162. M. Kramer Mfg. Co. v. Andrews, 783 F.2d 421, 436 (4th Cir. 1986); Midway Mfg. Co. v. Arctic Int'l, 704 F.2d 1009 (7th Cir. 1983); Atari, Inc. v. North American Philips Consumer Electronics Corp., 672 F.2d 607 (7th Cir. 1982); Stern Electronics, Inc. v. Kaufman, 669 F.2d 852, 856 (2d Cir. 1982); Digital Communications Assoc. v. Softklone Distributing, Inc., 659 F.Supp. 449, 456 (N.D. Ga. 1987). *Midway*, *North American*, *M. Kramer Mfg.* and *Stern* involved video games and the respective courts protected the displays as audiovisual works. *Softklone* involved a status screen from a communications screen and that court protected it as a compilation and specifically not as an audiovisual work. It should be noted, however, that because of the Copyright Office's recent computer software decision, these cases, because they involved programs protected by separate copyrights over their screen displays, remain of questionable value.

The use of video game precedents in arguing for broad protection of computer program screens of non-video games has been criticized. One commentator has objected to the use of these precedents noting that while the analogy is interesting, it is of "little help" because a video game possesses no utilitarian considerations, unlike other application programs. *Protecting the Look and Feel*, *supra* note 8, at 436-39. However, courts have rejected this interpretation. Note that *Softklone* used the video game precedents to protect the status screen of a communications program. *Softklone*, 659 F.Supp. at 462-63.

163. It is often difficult to separate what the user sees from the method of use. Both, however, remain crucial to the financial success of the program. An attractive display with an incomprehensible user interface will not likely be financially successful. For a good example of one court separating these two elements, but relying on a separate screen copyright to do so, *see Softklone*, 659 F.Supp. at 457-63. This Note's proposal does not advo-

be that the display allows the computer to communicate with the user, whereas the user interface allows the user to communicate with the computer. In any case, the importance of the screen displays to the financial incentive of the program developer must not be underestimated. The screen displays are what the consumer sees and a significant portion of his decision whether to purchase the product will be based upon his perception of the screen display. It is not an exaggeration to argue that consumer opinion of the screen display (along with the user interface) can determine the success or failure of a program.[164]

Because another programmer can "knock off" the screen (and corresponding user interface) of an original program, a copyright which extends solely to the program code and its design will not prevent a developer from suffering pecuniary loss resulting from unauthorized copying of his program's display screen or user interface.[165] Since the potential severability between the code copyright and the screen copyright meant that many software developers were unaware that they needed the additional copyright to protect their program's screen display and failed to adequately guard their programs against unauthorized copying,[166] the Copyright Office, in June 1988, ruled that it would register only one copyright per computer program.[167] Despite this ruling, it remains unclear whether a textually based screen display, as opposed to a graphically based screen display, is copyrightable at all. To date, only the court in *Softklone* has protected such a screen, but the Copyright Office explicitly rejected the *Softklone* decision, "reasoning that there is no authorship in ideas, or the format, layout or arrangement of text on the screen."[168] However, the screen display cases demonstrate that a new copyright standard which explicitly protects "look and feel," or all of the elements of software (including screen displays), such as is proposed in this Note, would not be a radical departure from traditional

---

cate complete separation of the two elements when determining whether infringement has occurred. Instead, it argues that a court should use all protectible elements of the program when deciding the infringement aspect of a copyright case. *See infra* notes 330-38 and accompanying text.

164. *See infra* notes 320-329 and accompanying text; *The Difficult Path to the Easy to Use Computer*, BUS. WK., Feb. 27, 1984, at 93.

165. The *Stern* court noted that protection of the code alone "would not have prevented a determined competitor from manufacturing a "knock off" of "Scramble" [the game at issue] that replicates precisely the sights and sounds of the game's audiovisual display." *Stern*, 669 F.2d at 855. Furthermore, since *Whelan* does not protect the external aspects of a computer program, a developer should not rely on that case to protect against knock off programs. Once again, the Copyright Office's recent ruling may mean that the copyright on program code will also protect the program's screen display and user interface. *Copyright Ruling, supra* note 16, at 152.

166. *Defining the Scope, supra* note 28, at 530.

167. *Copyright Ruling, supra* note 16, at 153.

168. *Copyright Ruling, supra* note 16, at 152.

copyright standards.[169] Indeed, the screen display cases show that current copyright law will protect at least certain types of computer program displays.

Graphically based computer screens traditionally have been protected by an audiovisual copyright.[170] The Copyright Act defines audiovisual works as "works that consist of a series of related images which are intrinsically intended to be shown by the use of machines, or devices such as projectors, viewers, or electronic equipment, together with accompanying sounds, if any. . . ."[171] The court in *Stern Electronics v. Kaufman*.[172] was the first to protect a screen display. This case involved the video game "Scramble," which the plaintiff had registered as an audiovisual work.[173] The defendant "knocked off" the visual display, using different computer code, but producing a game that was "virtually identical in sight and sound" to the plaintiff's work. With a virtually identical game, but drastically lower development costs, the defendant sold its game for several hundred dollars less than the plaintiff's program.[174] Even though the plaintiff possessed a valid audiovisual copyright registration over the program, the defendant argued that the display screens were *per se* not copyrightable. Instead, the defend-

169. Clearly, what would be different than current law is this Note's proposal that the screen displays be protected by the same copyright as the literal code, the program design and user interface. This might require an exception to current copyright formulations concerning software, particularly aspects concerning statutory definitions of copies and fixation.

170. *Stern*, 669 F.2d 852 (2d Cir. 1982); *North American* 672 F.2d 607 (7th Cir. 1982); Williams Electronics, Inc. v. Arctic International, Inc. 685 F.2d 870 (3d Cir. 1982); *M. Kramer Mfg.*, 783 F.2d at 434-37. *See also Softklone*, 659 F.Supp. at 462 (holding that a screen in that case was protectible separately from the program code as a compilation because it was textually based). *But cf. Copyright Ruling, supra* note 16, at 152-53.

171. 17 U.S.C § 101 (1986).

172. 669 F.2d 852 (2d Cir. 1982).

173. *Id.* at 855. The court found that the plaintiff intentionally "eschewed registration of its program as a literary work and chose instead to register the sights and sounds of "Scramble" as an audiovisual work." *Id.* It is clear that the plaintiff did this because it knew that its product could be easily "knocked off" and consequently, a copyright on the code itself would provide little financial protection against unauthorized copying. *Id.* It is important to note that the plaintiff only held one copyright; the audiovisual copyright designed to protect only the screen display. *But cf. Copyright Ruling, supra* note 16, 152-53.

174. *Id.* at 855. The price undercutting that occurs from a "knock off" is the primary reason to protect software against such copying. The copier is not providing any innovative leap, only taking advantage of a prior work and selling it for a lower price. Such a system only encourages programmers to wait for others to invest the time and money to develop new programs, and then simply knock them off and sell them cheaper. *See also A Thousand Clones, supra* note 6 at 215, n.154. (noting that the defendant's program in *Softklone* was sold at approximately $100 less than the plaintiff's similar program). *See infra* notes 320-329 and accompanying text.

ant asserted that only the program code was protectible. The defendant claimed that the screen displays of a computer program were neither "fixed" nor "original" for the purposes of copyright law because the display does not look the same every time the program runs, since the user's inputs always cause variations in the display. Alternatively, it claimed the screen possessed no originality because it was "determined by the previously created computer program."[175] The court rejected both of these arguments.[176]

While acknowledging that copyright law requires every copyrightable work to be an original work of authorship fixed in a tangible medium of expression,[177] the court noted that the program code which creates the screen display is a copy of the screen display.[178] The Copyright Act defines a "copy" as "a work fixed . . . and from which the work can be perceived, reproduced, or otherwise communicated, either directly or with the aid of machine or device."[179] The court noted that when a program is run through the computer the same display always results; hence, a program is a copy of its display.[180] Since the program is a copy of its display, and the program is fixed in the memory chip of the computer, the screen display is also fixed for copyright purposes.[181]

However, even if the display is fixed in the program code, this still does not address the defendant's charge that the screen is not fixed since the user's inputs vary the screen each time the game is played.

---

175. *Id.* at 856. For example, each "game" of "Scramble" would look different, depending upon how the player plays the game. On the other hand, the court pointed out that a static or unchanging audiovisual display would clearly meet the fixation requirement because the program is fixed in the computer memory devices and is reproduced through the aid of the computer. *Id.* at 855-56. *See also* 17 U.S.C. § 101 (defining fixation). *See supra* text accompanying note 94.

176. *Stern*, 669 F.2d at 855-57.

177. 17 U.S.C § 102(a).

178. *Stern*, 669 F.2d at 855-56.

179. 17 U.S.C. § 101.

180. *Stern*, 669 F.2d at 855-56. *But cf. infra* notes 210-223 and accompanying text, noting that the screen is not a copy of the program.

181. *Stern*, 669 F.2d at 856. The statutory definition of fixation notes that "[a] work is 'fixed' in a tangible medium of expression when its embodiment in a copy . . . is *sufficiently permanent or stable to permit it to be perceived, reproduced, or otherwise communicated for a period of more than transitory duration.*" 17 U.S.C § 101 (emphasis added). As the court found, since the program (a copy of the screen display) is permanently etched upon the memory chip of the computer, this satisfies the fixation requirement. This defeats the argument that the display is not fixed because it disappears when the computer is turned off.

Many of the current application programs are not contained in ROM, however, but are loaded into RAM. Still, the *Softklone* holding, in protecting the status screen of a communication program, shows that a program need not be fixed inside the computer to meet the fixation requirement. Instead, it need only be fixed on a disk or some other element and capable of being shown through the use of the computer.

The court responded to this argument by noting that the fact that the display changes because of user influence does not change the fact that all aspects of the display remain within the program chip and are reproducible by the computer.[182] Although it is possible that not all of the various elements of the program will be seen in any given game, "many aspects of the sights and the sequence of their appearance remain constant during each play of the game. . . . The repetitive sequence of a substantial portion of the sights and sounds of the game qualifies for copyright protection as an audiovisual work."[183] In other words, as long as the program is fixed and the screen has constancy to it,[184] a computer display may be copyrightable.[185]

The defendant's second argument against copyrightability of the display was that the display was not an "original" work of authorship. The defendant's argument proceeded along two opposite lines. First, the defendant asserted that a new and original work resulted *every time* the program ran, since the display varied each time a user played the game. If the court accepted this argument, then the plaintiffs would have possessed a copyright only on the display actually deposited with the Copyright Office. The plaintiff would have held a copyright only on that particular screen display, and would have had to obtain a new copyright for each variation of the screen display in order to protect the every possible program screen display from unauthorized copying.[186] The court also rejected this argument, holding that the variations of the display screens were not different enough to warrant a new copyright on each different version.[187]

Alternatively, the defendant argued that the displays were not "original" at all. It asserted that the features on the screen were predetermined by their inclusion in the program code. In other words, the defendant claimed that the display was merely part of the code's originality, and possessed no originality of its own. Again, the court rejected the defendant's argument. This time the court held that "[t]he visual and aural features of the audiovisual display are plainly original variations sufficient to render the display copyrightable even though the underlying written program has an independent existence and is itself

182. *Stern*, 669 F.2d at 856.

183. *Id.* at 856. *But cf. Softklone*, 659 F.Supp. at 462 (classifying a text based screen display as a compilation and not as an audiovisual work).

184. The court expressly refused to determine the exact amount of repetitiveness a program would need to have to attain copyright status. *Stern*, 669 F.2d at 857. However, it is likely that most programs, by nature, would have enough repetitive sequences to qualify for a copyright.

185. An analogy might be to a crossword puzzle, which although changing form when people write in the answers, remains copyrightable.

186. *Stern*, 669 F.2d at 856.

187. *Id.*

eligible for copyright."[188]

Stern demonstrates that under current law the display output of a computer program may be copyrightable. In other words, a screen display of a computer program meets the fixation and originality requirements of copyright. However, some courts have held that the display is only copyrightable separately from the underlying program code.[189] Fortunately, the Copyright Office's recent decision to register only one copyright per computer may prevent potential policy problems that separability creates. Still, because the Copyright Office's new standard has not yet been tested in court, it is unclear whether the ruling will enable courts to consider the similarities between program elements within the context of programs as a whole. If it does not do this, and courts fail to examine each program element individually *within the broad context of the entire program*, then a potential copier conceivably could escape an infringement judgment against it if its program possessed only small similarities in each individual program element. The similarities in each individual program element, when viewed in isolation, might not constitute enough similarity to warrant a finding of copyright infringement. In such a case, the court's holding would enable the copier to continue producing a broadly similar work, and the innovator's time and effort spent in creating the original program would be wasted. However, if the similarities and differences within each program element are *cumulated* and *compared within the context of the entire program*, they might then show enough similarity to constitute infringement and the court could stop the unfair copying of the originator's software. For this reason, this Note proposes that there be only one copyright available for computer software, a specifically formulated "computer software" copyright.[190] Through the suggestion of this new

188. *Id.*

189. *Stern* is not the only case to have held that the screen display is copyrightable separately from the computer program. *Whelan*, while expanding the protection of programs beyond mere code to structure, and allowing screens as evidence of substantial similarity, made clear that the displays were copyrightable as separate elements from their programs. 797 F.2d at 1244. Furthermore, both *Williams Elecs. Corp v. Artic Int'l.*, 685 F.2d 870, 874 (3d Cir. 1982) and *Midway Mfg. Co. v. Strohon*, 564 F.Supp. 741 (N.D. Ill. 1983) also support the conclusion that current law, until the Copyright Office's recent ruling to the contrary, allowed displays to be separately copyrightable as audiovisual works. Furthermore, *Softklone* 659 F.Supp. at 463, holds that a textual screen is also copyrightable separately from its underlying program display. *See A Thousand Clones*, *supra* note 6, at 201-202. *But see Copyright Ruling*, *supra* note 16, at 152-53.

190. This Note's proposal would differ from the Copyright Office's current standard because it would be specifically tailored to encompass all elements of the program. The proposal would still, however, require individual idea/expression analysis upon each element of the program. It would maintain the benefits of separability (more precise idea/expression analysis) while still providing complete protection to the developer's work.

work of authorship, this Note hopes to create a better and more comprehensive standard of copyright protection for software.

The court's decision in *M. Kramer Mfg. Co. v. Andrews*[191] exemplifies how copyright law should protect both the display and the underlying code from illegal copying. More important, it also demonstrates the advantages of *using more than just one program element to determine copying*. *M. Kramer Mfg.* involved a poker video game used in arcades.[192] The plaintiff had developed several versions of a video poker game which the defendant distributed for him. The two parties had discussed becoming partners, but the negotiations had collapsed. Soon thereafter, the defendant, Andrews, hired a software engineer to develop a similar version of the plaintiff's poker game. The engineer copied non-copyright noticed circuit boards from the plaintiff's game, changed the ROM board configuration, and varied words in the video display of the program, but ultimately created a game similar to the plaintiff's game.[193] After the game was completed, Andrews distributed his own game instead of the plaintiff's game.[194] The plaintiff charged copyright infringement of both his program's screen display (for which it possessed a valid copyright registration) and his program code (for which it did not initially possess a valid copyright registration over but which was later awarded one).[195]

Because the district court denied the plaintiff's pre-trial motion to amend its complaint to include information concerning the copyright registration of the game's program code,[196] the court of appeals examined the relationship between a computer program and its audiovisual display. It questioned the district court's holding that "an infringement action based on a copyright of the audiovisual works does not involve the computer program. . . ."[197] The lower court's holding had prevented the plaintiff from introducing evidence of the similarities between its program code and the defendant's program code.

The Court of Appeals reaffirmed that video games are copyrightable, holding that "under the authorities copyrightability of video games as audiovisual works cannot be disputed."[198] However, the court went a

191. 783 F.2d 421 (4th Cir. 1986).

192. *Id.* at 425-27.

193. *Id.* at 427-28.

194. Notice again how the alleged infringement directly affects the original innovator's income stream.

195. *M. Kramer Mfg.*, 783 F.2d at 429.

196. *Id.* at 440.

197. *Id.* at 440. The plaintiff wished to introduce into evidence data concerning the program codes of the two programs. He believed that by doing this he could help demonstrate the similarities between the two programs.

198. *Id.* at 436. The court was referring to both the underlying program of a video game and its screen display. Both elements remain copyrightable, and until the Copyright

step further, holding "that a copyright in the audiovisual display of a computer program *protects not only the audiovisual display from copying, but also the underlying computer program to the extent the program embodies the game's expression.*[199] The program, as we have noted, is by definition a 'copy'. . .[of the screen display]."[200] Clearly, this was the correct holding. Because the Copyright Act gives the copyright owner the exclusive rights to reproduce copies of the copyrighted work, the copyright owner of a screen display has the right to control all *copies* of the screen display. Therefore, since program code is a *copy* of the screen display, fixed in the memory of the computer, the copyright owner of the display not only controls the rights to use the display, but also controls the rights to use the underlying code.[201] As the court concluded, a "computer program qualifies as a copy of plaintiff's audiovisual work and *as such is protected under the plaintiff's copyright* [of the screen display]."[202] The court's logic was as follows: if the display's copyright did not protect the underlying code, then the program code could not be a copy of the display. If the code was not a copy of the display, then the display could not be fixed in the code. If the display was not fixed in the code, then the display was itself not copyrightable, since it was not fixed in a tangible medium of expression.[203] If that were true, then *all* the video game cases were wrongly decided. More important, if displays were not copyrightable, there would be no way to protect creators from the dangers of "knock off" programs. Put more simply, the essence of the *M. Kramer Mfg.* holding is that "[i]t necessarily follows . . . that the audiovisual copyright may be infringed in one of two ways: The infringer may copy the audiovisuals themselves or the infringer may copy the underlying computer program."[204]

The *M. Kramer Mfg.* decision also demonstrates how a court can use the various facets of a computer program to protect the developer's interest in the program. By extending the audiovisual protection to the underlying program, the court may consider the *similarities between the underlying programs codes.* The court will not be limited to trying

---

Office's recent ruling to the contrary, often through two copyrights, one on the display and one on the code.

199. Note that under then current formulations, only the part of code that actually produces the display is a copy of the screen display. Other portions of the program code that do not produce the display are not copies of the display and are, therefore, not protected by the display's copyright.

200. *M. Kramer Mfg.* 783 F.2d at 442 (emphasis added). *Accord Softklone* 659 F.Supp. at 456. ("a computer program is a copy of a screen display."). *See supra* notes 177-181 and accompanying text.

201. *M. Kramer Mfg.*, 783 F.2d at 441; 17 U.S.C.§ 106(1).

202. *M. Kramer Mfg.* 783 F.2d at 441-42.

203. *Id.* at 441.

204. *Id.* at 445.

to determine whether the changes to the screen made by the defendants are enough to avoid infringement; the court can rely instead upon the "virtually identical" nature of the two program codes to prove copying.[205] Clearly, allowing a court to examine more facets of a program, rather than limiting the court to examining one aspect of the program (for instance the screen *or* code) provides better and more thorough protection for the software developer.

Both *Stern* and *M. Kramer Mfg.* concern the protectibility of video game screens. While there is criticism of the use of these cases to support the viability of protecting the screen display of an application program,[206] another court, in a case involving a communications program[207] and not a video game, and a textual display and not a graphic display, also protected the screen display of a computer program. Indeed, *Digital Communications v. Softklone Distributing Inc.*[208] may be the most important computer software case since *Apple*.[209] *Softklone* is important for the following reasons: first, the case does not concern a video game, but instead concerns a business application program; second, the court protected the screen of an application program that is *not* in pictures or in a game, but is comprised merely of words; third, the two program displays resulted from different underlying program codes; and fourth, it follows the *M. Kramer Mfg.* holding that programs are copies of screen displays.[210]

Softklone concerned the similarities in the status screens of two communications programs, the plaintiff's "Crosstalk XVI" and the defendant's "Mirror."[211] The plaintiff held copyrights on both the pro-

---

205. *Id.* at 446. The court's ability to look at the underlying program code would not enable the court to find infringement in this case if the defendant's program was a knock off of the plaintiff's program, since in that instance, the defendant's code would be different and the court would not be able to rely on the similarities in two program codes when determining infringement. Consequently, a better infringement standard is needed, one that will truly consider the totality of the program.

206. *See supra* note 162.

207. A communications program is an application program that allows the user to easily make use of a modem. Generally, communications programs enable the user to receive and transfer files through the modem via telephone connections with other computers.

208. 659 F.Supp. 449 (N.D. Ga. 1987).

209. Lotus Corporation's belief that the *Softklone* decision almost guarantees a victory in its case exemplifies the importance of the case in the current struggle to protect the "look and feel" of software. *Reuters*, April 27, 1987. *See A Thousand Clones*, *supra* note 6, at 208-20.

210. In addition, the court's decision protects the user interface in a very narrow fashion. This may be the most important aspect of the *Softklone* decision. *See A Thousand Clones*, *supra* note 6, at 210.

211. *Softklone*, 659 F. Supp. at 452-53. For a more detailed explanation of the screen and how it works, *see A Thousand Clones*, *supra* note 6, at 209.

gram *and* the status screen.[212] The defendant, presumably recognizing the plaintiff's successful program, "decided to develop a clone of Crosstalk XVI system."[213] The defendant clearly intended to exploit the innovative Crosstalk XVI screen, but also intended to avoid copyright infringement by writing Mirror in different program code than Crosstalk. Essentially, the court was faced with two programs that looked "virtually identical" but were written in different program codes.[214] Both programs used similar highlighting and arrangements on the screen to convey to the user the settings of current program parameters and the results of program functions. The court noted that the "Mirror status screen capture[d] the 'total concept and feel' of the Crosstalk XVI status screen."[215] *Softklone* epitomizes the danger to the computer industry of "knock offs"; if the court were to allow the defendant to continue to sell Mirror, it would be allowing a *non-innovator* to undercut the innovator's creative efforts. Clearly, this would be a disincentive for people to innovate, and an incentive for people to copy previous innovations.[216] Fortunately, the court understood this and prevented such a result.

Several aspects of the *Softklone* holding remain important to the inquiry of protecting screen displays. The court rejected the argument that the display, like a video game display, was an audiovisual work. The court did so, primarily out of its belief that if the status screen was on paper rather than a screen, it would have been a literary work and not an audiovisual display. More specifically, the court found that although the screen changed when the user changed the program's commands, it was not a "series of related images." A work must consist of a "series of related images" to constitute an audiovisual work under the Copyright Law.[217] Instead, the court held that the screen was copyrightable as a compilation.[218] The plaintiff argued that the screen could also be considered a derivative work of the code, but this argument was rejected because, in most cases, a program's screen display is designed *before* the program is encoded, whereas a derivative work implies that it

212. Note that under current Copyright Office procedures, a separate copyright on the screen display would be unavailable. Hence, displays like the one at issue in *Softklone* might currently be unprotectible. Indeed, the court held that had the plaintiff in *Softklone* not held a separate copyright on the screen display it would not have been able to protect it against the defendant's knock off program. *Id.* at 455-56. *See A Thousand Clones*, *supra* note 6, at 210; *Copyright Ruling*, *supra* note 16 at 153.

213. *Softklone*, 659 F.Supp. at 453.

214. *Id.* at 465.

215. *Id.*

216. *See infra* notes 312-329 and accompanying text.

217. *Softklone*, 659 F.Supp. at 462; 17 U.S.C. § 101 (1986).

218. The plaintiff, Digital Communication Associates, had registered the screen as a compilation. *Softklone*, 659 F.Supp. at 453.

has been *derived* from a preexisting work.[219] The court held that the screen was a compilation because it was an assembly of data (a listing of the program's command language) arranged in a certain original way. In this case, the display was designed to make the program easy for the user to use.[220] Most important, *the court found a textual display copyrightable.*

In addition, the court reaffirmed the idea that a program display is fixed in its code copy. In following the *M. Kramer Mfg.* holding, the court also explained why the program code copyright cannot, under then current copyright formulations, give reciprocal protection to its screen displays. The court correctly found that *M. Kramer Mfg.* based its holding on the fact that the program is a fixed copy of the screen display. Again, the code is a copy of the display because it will consistently produce the exact same screen display when run through the apparatus of a computer. However, "if one has a fixed screen display, one cannot, even with the aid of a machine, repeatedly create the same program [code] as many different programs can create the same screen display."[221] Hence, the display is not a copy of the code, and thus the copyright on a program code does not protect screen displays under the *Softklone* court's interpretation of then current copyright standards for computer software.[222] Consequently, possessing a copyright solely on the computer program code might not protect the developer from competing "knock off" programs that use different codes to create the same displays. The code copyright seems only to protect the developer from literal code copying, which, because of the ease in developing "knock off" programs, is in reality very little substantive copyright protection.

The screen display cases seem to demonstrate that copyright law will protect a program's screen display *only* if *it* is separately copy-

---

219. *Id.* at 462-63. *See supra* note 144.

220. *Softklone*, 659 F.Supp. at 462-63. The fact that the screen makes it easy for the user to use the program defeats any challenge that there is no expression in the screen; it clearly expresses information about what parameters are currently operative in the program and how to change them; 17 U.S.C. § 101 (defining a compilation as "a work formed by the collection and assembling of preexisting materials or of data that are selected, coordinated, or arranged in such a way that the resulting work as a whole constitutes an original work of authorship.").

221. *Softklone*, 659 F.Supp. at 456.

222. *But see Copyright Ruling, supra* note 16, at 152. The Copyright Office's ruling notes that the copyright on a program code does cover all aspects of the resulting program display. Because the Copyright Office ruling and *Softklone* are in conflict, it is unclear at this time whether a program's copyright protects *both* the code and the resulting screen display. In any case, since the standard proposed in this note, specifically enumerates the protectible elements of a work of software, it would not conflict with any other copyright doctrine. Without this statutory revision, despite the Copyright Office's decision, protecting the screen display with the same copyright that protects the code would seem to violate copyright standards.

righted.[223] However, the Copyright Office's recent ruling denying more than one copyright per computer program casts these decisions in doubt. It is currently not known, because courts have yet to examine the new Copyright Office regulations, whether the Copyright Office's new single copyright regulation will meet the proprietary needs of software developers.

In any event, the question of whether the user interface within the screen is protectible remains unanswered.[224] If the user interface is not protected, the programmer's actual protection of his investment will be minimal. The user interface is the most difficult and critical aspect of computer software protection; it must be examined in depth.

4. *Capturing the Feel: Protecting the User Interface and Other Potentially Utilitarian Aspects of Computer Software*

It should now be clear that any standard of protectibility for software hinges upon certain policy considerations. The key question concerns what degree of protection will encourage the most innovation. However, before discussing the merits and dangers of protecting a program's user interface, it is important to explicate exactly how this Note defines the somewhat nebulous term "user interface."[225] Put simply, the user interface is the part of the computer program output that controls and directs the *user's* communication with the program. It is the form the programmer devises to enable the user to use the program efficiently; generally, it is a system of regulating or guiding the user in the inputting of information into the computer program.[226] While nearly all commentators and courts generally agree that program code

223. One commentator has argued that this is a desired result, since it allows a court to analyze the copyrightability of the screen display in isolation. He believes that the program display and interface should be considered separately because each provides a unique function in facilitating communication between the computer program and the user. *A Thousand Clones*, *supra* note 6, at 212. While agreeing with these contentions, under this note's proposal, separate copyrights would not be required. Instead, the proposal requires a court to examine each element independently when determining the element's copyrightability, thus ensuring that a particular element's functions and ideas will not be protected. By also requiring the court to consider the similarities and differences of all the elements of the program when determining infringement, the proposal also engenders a more accurate infringement decision since the court is considering more elements of the program. This proposal, then, both protects creative work, allows fair competition, and encourages further innovation.

224. *But see A Thousand Clones*, *supra* note 6 at 214-216. The author does not differentiate between user interfaces and display screens.

225. A "user interface is [a] term used to describe any way in which a user accesses a computer system. . ." A. Chandler, Penguin Dictionary of Computers 472 (3rd ed. 1985).

226. *See Look and Feel*, *supra* note 17, at 123, n. 146. The author explains the concepts of user interfaces and provides several other examples. *See also PC Magazine*, *supra* note 11, at 155 (noting that the Macintosh graphic user interface is one of the biggest selling

and the non-functional aspects of display screens merit copyright protection, some legal commentators argue that there should be absolutely no protection of the program's user interface.[227] These skeptical commentators fear that protection of the user interface might "preclude others from using similar actions to perform functions, it might preclude other . . . interfaces from functioning as effectively."[228] If a particular type of interface becomes protectible, the argument continues, the use of menus, graphics, or command languages might be limited to one program. This would force other developers to continually develop new and perhaps inefficient interfaces to avoid infringing the original interface.[229]

However, the user interface, like the screen display, remains directly related to the financial success of the computer program.[230] Because a program is worthless to a user who cannot learn how to use it, or to one who believes it too cumbersome or simple for his tastes, the program's user interface can determine whether the program becomes a success or failure in the marketplace.[231] Consequently, if copyright law is to encourage new developments in software, it must protect the user interface,[232] while at the same time avoid giving the developer a monop-

---

points of the Macintosh computer); *A Thousand Clones*, *supra* note 6, at 197-98 (listing the primary interfaces designs as menus, command based, and direct video manipulation).

227. *Common Law, Uncommon Software*, *supra* note 133, at 1100-04. *But see A Thousand Clones*, *supra* note 6, at 219-20 (the author argues for copyright protection for aspects of the user interface).

228. *Common Law, Uncommon Software*, *supra* note 133, at 1101. *But See A Thousand Clones*, *supra* note 6, at 219-20 (the author argues that other interface styles, while incorporating basic interface designs, can be created).

229. *Common Law, Uncommon Software*, *supra* note 133, at 1101.

230. *The Difficult Path to the Easy-to-use Computer*, BUS. WK., Feb. 27, 1984, at 93.

231. Ranney, *supra* note 6, at 13 ("the ease of use. . . [the] user interface, is fast becoming a major selling point for computer software. Some believe that it will be the primary component that manufacturers will wish to protect. . ."). *See A Thousand Clones*, *supra* note 6, at 220 ("User interface design is critical to the market success of computer programs"). *See also Common Law, Uncommon Software*, *supra* note 133 at 1103 n.167 (noting that Multimate and Ansa's Paradox use the same user interface as Wang wordprocessors and Lotus 1-2-3 in an attempt to sell to those using the latters's products). The point is that a computer program might have an attractive screen, but if the logic and method of use of the program (the various command sets, symbols, and methods through which the user uses the program) is not pleasing, the program most likely will fail.

232. The innovation of a new style of user interface can mean the development of an innovative and superior product; this is something the copyright law should encourage. *See* Soucie, *Excel - Should You Switch?* PC WORLD, Mar. 1988, at 108. "Lotus's favorite has reigned supreme in the realm of spreadsheets. . . . But now a serious challenger, indeed a better product has made its debut: *Microsoft Excel*. . . *Excel* delivers a *stunning . . . interface*." (emphasis added). With little protection of the interface, developers will have little incentive to develop new concepts; it will be cheaper to copy existing interfaces. Indeed, unlike *Excel*, the designers of *Twin* and *VP Planner* (the targets of the Lotus Corp. suit) chose to emulate or copy the *1-2-3* interface and sell their programs more cheaply

oly upon an idea or a utilitarian work.[233] The standard proposed in this Note would achieve this goal. The proposed standard would direct the courts to use the plurality of expression analysis[234] to determine the user interface's underlying "type" and basic functions. Because the software would be covered by only one copyright, the court, when performing this examination, would also be able to consider the interface's role within the program's broad purpose. By using this analysis, the court would protect only those portions of the interface that are not part of the interface's basic design and that are not necessary to the operation of the program as whole.[235]

---

than Lotus. They chose quick profits rather than investing the time and money into creating their own interface design.

233. *Look and Feel*, *supra* note 17, at 105. The author vehemently argues that computer software competition will be eliminated if user interfaces are protected. *But see A Thousand Clones*, *supra* note 6, at 215-16 (noting that *not* protecting user interfaces will discourage innovation, because it is cheaper to copy an interface than to design a new interface style). It should be reemphasized that the purpose of copyright is to encourage innovation so society as a whole will intellectually and culturally progress.

234. It is important to remember the plurality of expressions test used by the courts in *Apple* and *Whelan*. Briefly, if there are a variety of ways to express the underlying idea, then the expression is separable and protectible. Conversely, if there is only one method of expressing the idea, then the idea and expression are said to have "merged", and the work cannot gain copyright protection.

235. *Softklone*, 659 F. Supp. at 458. For example, in the *Broderbund* facts, the court would determine that the purpose of the program was to make cards, posters and banners. *The purpose of the interface was to enable the user to use the underlying program through a menu driven interface system.* This purpose would not be copyrightable. However, everything not necessary to the operation of a menu driven computer printing program, yet was still part of the user interface (such as the particular order of loading in information, the particular commands, or the particular method of highlighting boxes to locate where a graphic would go) would be protectible. The point here is *not to protect the idea of a menu driven interface.* The protection here would only be for non-essential items such as menu structure, sequence, and word selection. Another programmer would be free to use a menu for a printing program, but would have to make the interface work somewhat differently to avoid copying. In addition, the programmer would have to avoid copying the other elements of the program as well. Such a requirement forces creativity among developers. *See A Thousand Clones*, *supra* note 6, at 213-14. The author also believes that the copyright law should protect those aspects of user interfaces that are not necessary to the interface's basic design and functions (i.e., there would be no protection on all of the elements of a menu system). This Note's proposed standard, while advocating protection only for innovation beyond the basic interface design, also requires that no part of an interface that is necessary to the functioning of a certain type of program (such as a printing program or word processing program) be protectible. This ensures that all works of software will possess interfaces that are viable to the particular purpose of the software. *But cf.* Synercom Tech., Inc. v. University Computing Co., 462 F.Supp. 1003, 1014 (N.D. Texas 1978) (the court denied copyright protection to the plaintiff's input formats because the only expression they contained was required in order to fulfill the program's underlying purpose, which was the creation of a *uniform system* of inputting data into a computer).

Still, several problems remain in protecting the user interface. Commentators have argued that there is no way to separate the idea underlying a user interface from its expression, thus preventing copyright protection.[236] Their argument is that an interface idea, such as a menu system, requires certain elements to function properly. Allowing copyright protection over these elements would give the copyright owner a monopoly over all variations of the interface, and this would be tantamount to granting a copyright on an idea. Clearly, protection of the *necessary* elements of the interface would violate copyright principles. However, this Note rejects the proposition that copyright principles forbid *any* protection of the interface. Instead, it suggests that only those elements which are *not necessary* to the basic interface, *as defined within the broad purpose and framework of the entire program*, should be protectible. The proposal goes no further. These non-essential elements must remain protectible in order to prevent copying of the interface and to encourage development of new and better interface styles and variations.[237]

---

236. *Common Law, Uncommon Software*, *supra* note 133, at 1101; *Protecting Look and Feel*, *supra* note 8, at 429. *But see A Thousand Clones*, *supra* note 6, at 213 (the author notes that the interface's idea should be defined as the basic interface design or interface type: menus, command based, or direct video manipulation. Anything beyond what is necessary for the basic interface design is protectible.

237. To reiterate, this Note does not take the extreme position that once a menu interface (or other basic interface design) is used, the developer possesses a copyright on the idea of a menu interface. That would overly inhibit the development of new software until a programmer could develop an entirely new interface. Instead, this Note rejects the argument that simply because two programs use the *same type of interface*, the interfaces will necessarily look and work *exactly* alike. *Look and Feel*, *supra* note 17, at 128-29 n. 175. This is simply not true. For instance, two programs that both use the same kind of menu interface would not have to use the same kind of command structure. An example of what might be copying are the command structures of Lotus Corp. *1-2-3* and Paperback Software's *VP-Planner* and Mosaic Software's *The Twin*. All use the same idea of a "two-line moving cursor menu" as part of their user interface, and this is not copyrightable since it is an *idea for an interface for a spreadsheet*. *In Their Own Words: Lotus, Mosaic and Paperback Software State Their Cases*, PC Mag., May 26, 1987, at 162 [hereinafter *In Their Own Words*]. But the two programs actually copy the command structure of *1-2-3*, such that "a keyboard command as complex and seemingly *arbitrary* as '/ P P R Backspace Home . End Home Enter O O U Q G Q ' will, in all three programs, define the same print range and send it to the printer without special formats." Taylor, *You be the Judge*, P.C. MAGAZINE, May 26, 1987, at 186 (emphasis added). Clearly, this kind of copying of command structures should be prohibited by copyright law. Neither *VP-Planner*, nor *The Twin* have made any innovations. Instead, they have simply copied the Lotus interface.

Interestingly enough, neither Paperback Software, nor Mosaic Software argue that they are not copying the exact expression of the Lotus program. Instead, they believe the similarities between the interfaces are necessary for compatibility. *In Their Own Words*, *supra* note 237 at 162-63. However, the *Apple* court noted that "compatibility. . . is a commercial and competitive objective which does not enter into the somewhat metaphysical

Because it is easier to simply copy the expressionable elements of an interface than to create a new one, providing no protection for the user interface would only encourage copying of existing interface designs and discourage innovation of new ones. The primary question remains one of policy. Those who favor strict and total non-protection of the user interface argue that protection of it "will eliminate clone programs, [and will] result in higher prices to software consumers."[238] However, the purpose of the copyright is *progress and innovation*, not the creation of inexpensive software.[239]

Although legal commentators and industry commentators have taken both sides of the issue,[240] the courts have been inconsistent in protecting the expressionable elements of the user interface. Indeed, although the court in *Broderbund Software, Inc. v. Unison World*[241] protected all elements of the plaintiff's program, extending the program code copyright to protect both the program display and user interface, it did so with erroneous reasoning. The court enjoined the defendant from distributing its printing program despite the fact that its program code was not similar to the plaintiff's program's code and that the plaintiff possessed no copyright on the program's screen display.[242] While the court came to generally the *correct policy result*[243]

---

issue of whether particular ideas and expressions have merged." *Apple*, 714 F.2d, at 1253. Indeed, *Microsoft Excel* demonstrates that new interfaces for spreadsheets continue to be possible; protection of expressionable aspects of user interfaces would force software developers to innovate and not copy. *See also A Thousand Clones*, *supra* note 6, at 219 (arguing that granting of command sets would not endanger the creation of other command sets or variations of user interfaces).

238. *Look and Feel*, *supra* note 17, at 134.

239. Some "clone" programs create arguably more efficient versions of existing programs. Most, however, are simply copies of existing works with no new originality or innovation. *See A Thousand Clones*, *supra* note 6, at 215-16.

240. *See* Warner, *Company Sues for Copyright Infringement*, *supra* note 8, at 1; LaPlante, *Suits Trigger Debate Over Ramifications*, INFOWORLD, Jan. 19,1987, at 1; Lach, *Court Backs 'Look & Feel' Copyright*, InfoWorld, Oct. 20, 1986, at 1; Warner, *Lotus Suit Stirs Debate Among Users*, *supra* note 8, at 1; *Ranney*, *supra* note 6, at 13; PC Mag., May 26, 1987, at 155-197; New York Times, Feb. 5, 1987; *Forbes*, June 15, 1987; *Reuters* April 27, 1987; Jun. 3, 1987; *Business Wire* Jun. 23, 1987.

241. 648 F.Supp 1127 (N.D. Cal 1986).

242. Under current formulations, however, because of the Copyright Office's recent ruling on computer software, Broderbund's copyright on its code *would* extend to the program display. *See Copyright Ruling*, *supra* note 16, at 152.

243. *But see Look and Feel*, *supra* note 17, at 132-34 (arguing that the result in *Broderbund* will result in the restriction of innovation and curtailing of competition). It should be noted that while Unison initially filed an appeal of the decision, *appeal docketed*, Docket No. 221, (9th Cir. December 30, 1986), the two companies have since settled. Unison agreed to pay Broderbund a certain sum of damages and also agreed to stop selling "Printmaster" until it had made certain changes requested by Broderbund. *Business Wire*, Jun. 23, 1987.

in this instance, its reasoning was unclear and *erroneous under then current legal formulations*. Because of this, the *Broderbund* decision demonstrates the inability of current copyright standards, if *applied logically and consistently*, to adequately protect computer software. The *Broderbund* court's poor reasoning, although leading to a correct policy result in that situation, may also lead to unpredictability; developers will not know whether other courts will also reason wrongly in order to arrive at a similarly correct policy result. Hence, the *Broderbund* court's mistakes demonstrate the need for a new standard of copyright protection for computer software.

Broderbund involved, as do many other software cases, a partnership that collapsed before the venture was complete. The plaintiff had developed and successfully marketed the software program "Print Shop" for the Apple II line of computers. The program enabled the user to design and print greeting cards, posters, signs and banners. The plaintiff, recognizing the market potential of the non-Apple II compatible MS-DOS market, began negotiating with the defendant, hoping to agree to terms whereby the defendant would create an MS-DOS compatible version of "Print Shop."[244] While negotiations between the parties continued, the defendant instructed its software engineer to "develop a program as identical to 'Print Shop' as possible— to 'imitate' it."[245] Meanwhile, in order to facilitate the conversion process, the plaintiff showed the defendant's engineer a copy of the program's source code. The plaintiff also gave the engineer several commercially available copies of the software from which he could create the MS-DOS version of it.[246] After several weeks of negotiations, during which time the defendant's engineer worked on the program, the negotiations collapsed. The defendant then instructed its engineer to "enhance" "Print Shop" rather than merely copy it. Actually, the engineer kept the same user interface of "Print Shop" since he had already completed work on it. He did, however, add a calendar function to his program, which did not exist in "Print Shop."[247] The defendant named its new program "Printmaster."[248] After releasing it, the plaintiff unsurprisingly, when considering how similar the programs actually were, sued the defendant for copyright infringement.

The *Broderbund* court focused upon whether the menu screens *and*

244. *Broderbund*, 648 F.Supp. at 1129-30.

245. *Id.* at 1131.

246. *Id.* at 1130-31.

247. It is conceivable that because of this addition "Printmaster" might be a derivative work of "Print Shop." In either case, however, an infringement action would lie since § 106(2) provides that the copyright owner controls the rights to all future derivative works.

248. *Broderbund*, 648 F.Supp at 1130-31.

the user interface portions of a computer program were copyrightable audiovisual works. The plaintiff asserted that although the underlying codes of the two programs were different, the ordering, sequence and structure of the user interface's menus screens were the same.[249] More specifically, not only did the program's display screens look the same, but the interface's system of menus directing the user how to input data and use the program was ordered the same, designed the same, and worked the same way in both programs. The plaintiff argued that both the screen display and more important, the user interface were protectible.[250]

The defendant, Unison World, Inc. asserted two arguments against this position, both of which were rejected by the court. First, Unison claimed that the idea underlying the user interface (the input formats, menus, and sequence of screens) was indistinguishable from its expression, thereby rendering the user interface unprotectible.[251] Unison argued that the underlying idea of the user interface was a "menu-driven computer program. . . . that allowed its user to print greeting cards, signs, banners and posters . . ."[252] Such an interface, Unison asserted, is limited in the kind of expression it can have.[253] The court, rejected this argument, holding that the interface's idea did not merge with its expression.[254] Second, the defendant argued that because the interface was a useful article,[255] and possessed no separable non-functional ele-

249. *Id.* at 1132.

250. *Id. See* Lach, *Court Backs 'Look & Feel' Copyright, supra* note 240, at 1 (. . .the court found that a program that closely copies a user interface violates copyright law").

251. *Broderbund*, 648 F.Supp at 1132.

252. Note that under this Note's proposed standard, this is how the user interface's idea would be defined, as it reflects the broad purpose of the interface with the entire program. Anything not necessary to this purpose would be protectible.

253. *Broderbund*, 648 F.Supp. at 1132. One commentator has noted that there are certain things and a certain order that such a menu-driven printing program must have. For instance, before asking the user for the information to be printed on the card, the computer must ask the user whether he wishes the computer to print a card or a banner. *Look and Feel, supra* note 17, at 127-28, n. 168. While this is true to some degree, a variety of menu interfaces might still be able to achieve operationality. In any case, under the proposed standard, the user interface is only one element of the court's infringement inquiry. Vast differences in the other elements would vitiate a finding of infringement even if the interfaces of the two programs are substantially similar. This provides added protection to both the copyright owner and subsequent developers.

254. *Broderbund*, 648 F.Supp. at 1132. This aspect of the holding has been subject to much criticism. *See Look and Feel, supra* note 17, at 122-26; *Protecting the Look and Feel, supra* note 8, at 414-15, 419-30, 436-39. These commentators argue that under current law the idea of a user interface should be considered separate from the program to avoid copyrighting the method of the interface. They generally conclude that user interfaces should not be protectible because they will limit competition.

255. A "useful article" is an article having an intrinsic utilitarian function that is not merely to portray the appearance of the article or to convey information. 17 U.S.C § 101.

ments, it was not copyrightable.[256]

In analyzing the problem, the *Broderbund* court misapplied then current copyright concepts. The court, in seeking to protect the display and the user interface,[257] stretched the boundaries of copyright law to the point of error. The mistake occurred primarily because Broderbund did not possess a copyright on the screen displays.[258] Consequently, if the court was to protect the program's display and interface, it had no choice to but to extend the program's code copyright to cover these *external* elements. In doing so, however, the court had to violate copyright principles. The *Broderbund* court misinterpreted *Whelan*, holding that *Whelan* held that the copyright on the code protected *all* elements, *including the external aspects*, of a computer program.[259] The court held this despite the fact that the video game cases clearly indicated that under then current law a screen display (and thus interface) must be copyrighted separately from its underlying program code.[260] In actuality, *Whelan* held only that code copyright extended to protect the structure or logic of the program design. In other words, the code copyright protected the internal program design. It mentioned only that display screens were admissible as partial evidence of the substantial similarity between the internal designs of the underlying programs.[261] The *Broderbund* court should not have, under then current law, protected the plaintiff's interface since Broderbund did not possess a valid copyright on the program's output. If Broderbund had registered its displays, "the court probably would have been able to support a holding

---

Furthermore, the Copyright Act notes that a useful article will only be copyrightable if its non-utilitarian aspects can be identified separately from its artistic elements. 17 U.S.C. § 101. For a survey of the problems with the useful article doctrine (judicial subjectivity and inconsistency) *see also* Note, *Works of Applied Art: An Expansion of Copyright Protection*, 56 SOUTHERN CAL. L. REV. 241, 247-253 (1982)[hereinafter *Works of Applied Art*].

256. *Broderbund*, 648 F.Supp. at 1133-34.

257. It should be reiterated that a user interface remains embodied in the output of the screen displays, just as the structure of a computer program remains embodied within the literal code of a computer program. The difference is that a screen display informs the user what the computer is doing or has done; the user interface informs the user what to do to the computer to make it work.

258. This problem appears to have been alleviated, since the Copyright Office now registers displays and program code under a *single* copyright. *Copyright Ruling, supra* note 16, at 153. Still, because the new rule remains untested, it is unclear exactly what effect it will have on the protection of computer displays and interfaces.

259. *Broderbund*, 648 F.Supp. at 1133. *Case Comment, supra* note 152, at 538-40.

260. *Case comment, supra* note 152, at 538-41. *See Look and Feel supra* note 17, at n. 181. *Contra Copyright Ruling, supra* note 16, at 153. Under the new Copyright Office regulations, such separate copyrights are not longer available. Whether *Broderbund* would still be decided in the same way under the new regulations, however, remains unclear.

261. *Whelan*, 797 F.2d at 1230; *Common Law, Uncommon Software, supra* note 133, at 1103.

that the screen outputs were copyrightable . . ."[262]

Still, commentators criticize any protection of the user interface because they believe its ideas and expressions merge.[263] These criticisms miss the point. Even if the interface is limited by the constraints of the underlying program, *it might still be able to be expressed in a variety of ways.* For example, the interface can vary the order of input, can allow the user to vary the level of instructions, can vary in the command language it requires, and can vary in the *types of commands* it uses. There are numerous ways that an interface can vary between programs, *even if the interfaces are of the same basic design and used for the same kind of program.*[264] While *Broderbund* is in error in protecting the interface, it is only in error because Broderbund did not possess a *copyright on any external program elements.*[265] If Broderbund had possessed a valid copyright on the displays, the court might have come to the same result without misinterpreting then current copyright law.[266]

The defendant, for reasons that are unclear,[267] also argued that the

---

262. *Case Comment supra* note 152, at 541. *Cf. Copyright Ruling, supra* note 16, at 153. The Copyright Office's ruling would presumably still allow for a finding that Unison infringed Broderbund's copyright by copying its screen displays. *But see Look and Feel, supra* note 17, at 128, (in which the author asserts that there was no other way for the programmers to develop a menu driven printing program.) However, this Note's author has used other printing programs in addition to Print Shop and Printmaster (Ashton-Tate's *Byline* for example) and has seen other menu systems that are quite different. The point is that because there is only a limited factual record published in the case, it is unclear whether other expressions of the interface exist when the interface idea is defined as "an adult oriented printing program."

263. *Look and Feel, supra* note 17, at 127-29.

264. This Note maintains that only those elements, beyond the basic interface design are protectible. Furthermore, this Note proposes that the interface should always be viewed within the scope of the underlying program's purpose. Those aspects of the interface, which are necessary to the basic interface design and which are necessary to the underlying program type (for example, a spreadsheet program), would not be protectible.

265. *But see supra* note 258.

266. Besides the unpredictability an erroneously reasoned case causes, there is still another danger with the *Broderbund* court's analysis. *The court did not conduct a separate idea/expression analysis upon each of the four elements of a program, instead holding that the program's idea was a designer printing program. Broderbund*, 648 F.Supp. at 134. The problem with this approach is that a court might determine that the basic interface design is not a necessary element to the program idea, and thus give the copyright owner a monopoly upon that basic interface design. This clearly would inhibit progress, as no other programmers would be able to use or vary a basic interface design without the copyright owner's consent. *See A Thousand Clones, supra* note 6, at 213 n.142. This note does not advocate such a policy.

267. *Look and Feel, supra* note 17, at 130-31. The author argues that the screens should be classified as useful articles, but he notes that "there is no direct authority for applying the useful article doctrine to the display screens generated by an application software program."

plaintiff's interface was unprotectible because it was a useful article.[268] As previously mentioned, a useful article is a work that possesses both utilitarian and aesthetic elements.[269] The defendant claimed that there was no way to physically separate the functional elements of the interface from its aesthetic elements; consequently, there could be no copyright protection of the interface.[270] In other words, it was impossible for the aesthetic or visual elements of the interface (the various menus, sequences and command codes) to exist separately from their functional qualities as part of a user interface.[271] In responding to this argument, the court, for reasons that are also unclear, erroneously applied a version of the plurality of expressions test, determining that the interface was chosen for purely aesthetic as opposed to utilitarian reasons.[272] However, this kind of analysis is irrelevant when a court is examining a potentially useful article. Instead, when deciding the work's copyrightability, "the judicial focus [should be] . . . whether the shape or design of a utilitarian article is capable of being 'identified separately from' and 'existing independently of' the utilitarian aspects.[273] Hence, if the court insisted upon treating the interface as a useful article, it should have focused upon whether the aesthetic (non-functional) elements of the in-

268. *Broderbund*, 648 F.Supp. at 1133.

269. *See Works of Applied Art*, *supra* note 255, for a good overview of the problems with the useful article doctrine. Several examples suffice to illustrate the point: In *Mazer v. Stein* 347 U.S. 201 (1954), the Supreme Court held that a lamp base in the shape of a ballerina was copyrightable as a sculpture since the base could exist separately from the functioning aspects of the lamp. The contrary was found in *Esquire v. Ringer*, 591 F.2d 796 (D.C. Cir. 1978), where the court held that modern designed lights did not contain physically separable aesthetic elements from the utilitarian purposes of the article thus it was not copyrightable. However, another court did not require physical separability, but only conceptual separability. In *Kielselstein-Cord v. Accessories by Pearl, Inc.*, 632 F.2d 989 (2d Cir. 1980), the court found that belt buckles, although clearly utilitarian, also had artistic value and thus were copyrightable.

Because determination of the copyrightablity of useful articles requires this separability analysis, screen displays and user interfaces do not fit into the rubric of the pictorial or graphic work of authorship category. *But see*, *Common Law, Uncommon Software supra* note 133, at 1100; *Look and Feel supra* note 17, at 130-31.

270. *Broderbund*, 648 F.Supp. at 1133. It should be noted that the interface graphics could have been deemed an audiovisual display by itself under current law. It is not clear why the interface graphics were considered "pictorial or graphic" works when the video game cases make clear that any graphic display should be considered an audiovisual work. If it is textually based, then it should be considered a compilation. *Softklone*, 659 F.Supp. at 463. On the other hand, displays have never been classified as pictorial or graphic works, although some commentators have argued that such a classification would be appropriate. *See Common Law, Uncommon Software, supra* note 133, at 1100; *Look and Feel, supra* note 17, at 130-31.

271. *Protecting the Look and Feel supra* note 8, at 434-35.

272. *Broderbund*, 648 F.Supp. at 1133-34. The court seemed to believe that if there were other ways to express the element, then it was not an useful article. *Id.*

273. *Works of Applied Art, supra* note 255, at 251.

terface were *separable* from the interface, not simply whether there were aspects of the interface that were aesthetic. If the court had focused upon the correct issue, it probably would not have protected the interface, since it is almost impossible to physically separate its aesthetic and utilitarian functions. Consequently, the user interface would remain unprotected.

Despite its erroneous legal reasoning, the *Broderbund* court, *in enjoining the distribution of "Printmaster,"* came to the correct result from a *policy* standpoint. If the court had correctly interpreted then current law, it most likely would not have extended the scope of the code copyright's protection to cover the program display and interface, and consequently, it would not have protected any external elements of the plaintiff's program. In doing so, the court's decision probably would have enabled the defendant to profit from its copying.[274] From a purely *policy perspective*, the *Broderbund* decision is desirable because it prevented unauthorized copying of a program's external elements. The important point to stress is that the two pieces of software remained so similar that it could hardly be said that Printmaster was a significant innovation upon Print Shop. Because they were essentially the same programs, a consumer would only be motivated to buy the cheaper of the two products, which would undoubtedly be "Printmaster," since Unison had avoided the research and development costs in originally creating the program design.[275] Thus, although the court reasoned erroneously, and created an unstable standard of protection, it still came to the correct policy result. If anything, the court's errors demonstrate the necessity of redefining the copyright law so a court can interpret the law correctly and consistently, while maintaining the proper policy result.

If, on the other hand, the two programs had been analyzed under this Note's proposed standard, the following might have resulted: first, the court would have examined the codes of the programs for literal, non-essential similarities. It would have found that the codes were completely different. Second, it would have applied the *Whelan* program design test, examining the program for structural and design similarities. Despite the fact that the defendant's engineer had added a calendar function, the court would probably have found that the programs possessed similar internal designs since both programs accomplished the same purpose in the same fashion. Third, the court would

274. Although the two programs had different program codes, the program code copyright might cover the internal design of the program, and the plaintiffs might have been able to establish copying by showing that both Print Shop and Printmaster possessed similar internal program designs.

275. *See generally Whelan*, 797 F.2d. at 1229-1231; *see infra* notes 321-323 and accompanying text.

have compared the programs's screen displays and found them to be virtually identical.[276] Again, the court would have had to filter out of its comparison those portions of the screen that were are required for a designer printing program. Finally, the court would have examined those portions of the user interface, for instance, the particular menu order and command terms, that were not necessary aspects of a menu driven printing program. After making this analysis, the court most likely would have found similarities between the protectible elements of the two program interfaces, since both programs used the same order of menus, and the same commands in their menus.[277] In any event, after consideration and cumulation of all the elements' similarities and differences, the court would then have determined, whether, if as a cumulative work, the defendant's work was substantially similar to the plaintiff's work of software. In other words, the court would have determined whether the defendant's software captured the "total concept and feel" of the plaintiff's work.[278] In this case, because three elements of Print Shop and Printmaster were so similar, a court applying this Note's proposed standard most likely would have found infringement and would have enjoined Unison from producing or distributing Printmaster.[279]

Digital Communications v. Softklone Distributing Corp.,[280] the most recent case concerning the "look and feel" issue, not only rejected the broad copyright protection accorded to the program copyright by *Broderbund*,[281] but also refused to grant copyright protection to the program's user interface elements. The court refused to protect the plaintiff's program's command structure and terms (the instructions

276. *Broderbund*, 648 F.Supp. at 1137. "The ordinary observer could hardly avoid being struck by the eerie resemblance between the screens of the two programs."

277. To reiterate, if the court found that there was no expressionable material within a certain element, that element (such as the user interface) would not be protectible. The court still would consider the similarities and differences between the remaining elements when determining infringement. It would merely exclude the non-protectible element from its inquiry.

278. *Softklone*, 659 F. Supp. at 465 (quoting *Roth Greeting Cards v. United Card Co.*, 429 F.2d 1106, 1110 (9th Cir. 1970)).

279. It is inherently a fact finder's decision whether works are substantially similar. This Note can only suggest a framework for such a determination; it cannot articulate, and still remain faithful to the substantial similarity doctrine, if it seeks to state a certain threshold of similarity that will always count as infringement. *See infra* note 336 and accompanying text.

280. 659 F.Supp. 449 (N.D. Ga. 1987).

281. *Softklone*, 659 F. Supp. at 455-56 (rejecting the *Broderbund* court's decision because it believed that *Broderbund* had erroneously determined that the external elements of the program were copies of the program code). *See supra* notes 199-223 and accompanying text.

typed in by the user which change program parameters)[282] concluding that such commands are ideas and therefore not copyrightable.[283] Instead, it stopped just short, protecting only the display element of the program.[284] It made this decision because the plaintiff had a separate copyright registration on the screen display as well as the underlying code.[285]

As previously noted,[286] *Softklone* concerned a defendant who intentionally copied the display and interface of the plaintiff's communications program "Crosstalk XVI."[287] Crosstalk had been a market success, and undoubtedly part of the reason was the program's effective and easy-to-use interface.[288] The defendant, believing that the program's external elements were not copyrightable, created the plaintiff's display and interface using different code. By doing so, the defendants believed that they had avoided infringing upon the plaintiff's program copyright.[289]

The program's interface consisted of a series of two symbol commands. The user, by typing a command (such as "SP" to change the program parameter that controlled the modem's baud rate), could easily adjust and change program parameters. The change was then instantly reflected on the display.[290] Although the defendants admitted to using the same interface as the plaintiff's, they asserted that the interface was not protectible.[291]

While the court protected the textual screen displays, the court did not protect the interface commands.[292] The court argued that the inter-

282. *Softklone*, 659 F.Supp. at 462; *A Thousand Clones*, *supra* note 6, at 218.

283. *Softklone*, 659 F.Supp. at 462; *A Thousand Clones*, *supra* note 6, at 219 n.173.

284. *Softklone*, 659 F.Supp. at 462; *see A Thousand Clones*, *supra* note 6, at 218.

285. *Softklone*, 659 F.Supp. at 455-56. Again, under current Copyright Office regulations, separate copyrights are no longer available. *Copyright Ruling*, *supra* note 16 at 152-53.

286. *See supra* notes 209-215 and accompanying text.

287. *Softklone*, 659 F.Supp. at 453.

288. The interface, the set of command terms that controlled the program, made the program unique. There are many different ways to design the interface of a communications program, as some use menu input, or as in *Softklone*, command structure interfaces. As will be shown, the protectible element under the proposed standard would be the facets of the interface that are not necessary to the functioning of the basic interface design for that particular type of program.

289. *Softklone*, 659 F.Supp. at 453.

290. The change reflected on the screen display is a part of the program display and not the interface. The court only protected this aspect of the program. It did not protect the program's command language, which forms the program's user interface. *Id.* at 459-63.

291. The defendants also argued that the display was not copyrightable. While they prevailed on the interface argument, they lost on the display issue.

292. *Softklone*, 659 F.Supp. at 459-463. The court noted that the particular arrangement of the commands is protectible, while the particular commands were not copyright-

face command language was not protectible because it was essential to the functioning of the computer program; the commands control the various program parameter settings and control how the program will work.[293] The court held that any element which controls or affects the operation of a computer program is functional and, therefore, not copyrightable.[294]

However, the court's reliance on the functionality of an element, and not on whether the same function can be carried out in a variety of methods, seems flawed. A command term or interface should not be uncopyrightable solely because it *eventually* affects the functioning of the underlying program. As the Supreme Court noted in *Mazer v. Stein:* "nothing in the copyright statute [supports] the argument that the intended use or use in industry of an article eligible for copyright protection bars or invalidates its registration. We do not read such a limitation into the copyright law."[295] Hence, when analyzing the user interface, the inquiry's focus should be on whether a particular command language can be designed differently yet still remain viable for that particular type of program. Generally, the command terms of a user interface need *not* remain similar to remain viable.[296]

If this Note's method of analyzing the user interface were applied to *Softklone*, the plaintiff's interface might be characterized as a "screen based, command structured, user interface for a communications system." Within that definition, whatever is not *necessary* to the functioning of the interface remains protectible. The particular command terms would thus be protectible, but the command ideas would

---

able. Clearly, this Note does not argue that the command functions themselves are copyrightable. However, it does argue that copyright law should protect the particular expression of a particular form of user interface, a standard that would go beyond the mere arrangement of the text on the screen, and would include the structure and command language of the interface. To the extent that *Softklone* is interpreted to mean that the commands of an interface, because they might be considered functional or unique are not copyrightable, this Note argues that current law should be changed to avoid this result. *See A Thousand Clones, supra* note 6, at 218-220.

293. *See A Thousand Clones, supra* note 6, at 218-219 n.173; *Softklone*, 659 F.Supp. at 460.

294. *Softklone*, 659 F.Supp. at 458-60. The court noted that the design of the screen display— the particular arrangement and highlighting— did not relate to the functioning of the program. However, the court held that the use of a command driven interface and in particular, two symbol commands did control the functioning of the program and thus were not copyrightable.

295. 347 U.S. at 218.

296. *See A Thousand Clones, supra* note 6, at 219, n.180. Again, the interface's idea would be the basic interface design (command or menu based or direct video manipulation) created for a particular type of program. Whatever is required for that type of program to make the basic interface design viable remains an idea; whatever is not necessary forms the protectible expression.

not be protectible. For example, typing "SP" to change the program's baud rate would be protectible, but the concept of a command that changed the program's baud rate would not be protectible, since any viable communications program must have this command to perform effectively. As long as a command idea can be expressed in different ways, the particular command term should be protectible, but the underlying function should never be protectible. Clearly, if a command can only be expressed in one viable way, then that expression would not be copyrightable. Finally, it is important to reemphasize that under the proposed standard, the interface inquiry would only be one fourth of the court's total analysis of the program.[297] Hence, the proposed standard's use of the plurality of expressions analysis forms a better method for determining the copyrightability of the user interface than the *Softklone* court's sole reliance on functionality as the test of copyrightability. The plurality of expressions analysis protects the creative portions of the interface, and gives the developer an incentive to develop. Furthermore, by not protecting the basic interface idea, the proposed standard allows for continued innovation existing interfaces.[298] In sum, the copyright law should protect those non-essential elements of the user interface as part of the total protection given to computer software.

The *Softklone* decision, although examining the user interface, did not protect it.[299] The *Softklone* court affirms that current law does not, if applied consistently, protect a program's user interface. Unfortu-

297. Because the proposed standard considers all four elements of a program, it might actually allow for more copying of the interface. For example, the second developer could copy the command structures and sequences and style of an interface, but apply them in a program that is designed differently, looks different, and encoded differently, and he probably would still avoid infringing the original program. However, the exact cumulative level of similarity that would rise to infringement is a matter for a court to decide.

In *Softklone*, however, it seems that the defendant would still lose. Clearly, the program code, while different, was most likely designed to work the same way. Second, as the court found, the screens were arranged exactly alike, and there were aspects of the screens that were not necessary for a communications program. Finally, because the user interfaces both operated using the *same* commands, they most likely would be found substantially similar. Consequently, it is likely that under the proposed standard the defendant would also be enjoined from distributing Mirror. However, to remove the injunction, the defendant would have to do more than just rearrange the elements on the screen. *It would have to reformulate the interface's command structure and potentially redesign the internal aspects of the program.*

298. While the creation of new interfaces would be the best innovation, such interfaces are difficult to create. A standard that requires the creation of an entirely new basic interface design for each program would be counterproductive; it would restrict innovative variations of current user interfaces. *See A Thousand Clones*, *supra* note 6, at 219-20.

299. However, because Mirror displayed its commands in the same way as Crosstalk XVI, the court still found infringement of the screen display and enjoined Softklone from distributing Mirror. *Softklone*, 659 F.Supp. at 464-65.

nately, such a standard does not protect developers, nor does it encourage innovation.[300] Hence, a new standard is in order.

5. *Conclusions of Current Law*

In the past decade, the explosion of computer technology has presented a peculiar and difficult problem for copyright law. The inherent dualistic nature of computer software, in that it contains both visual and non-visual elements and useful and non-useful elements, makes it difficult for courts to apply traditional copyright doctrine to it. The judicial decisions of the past decade concerning software demonstrate that courts are struggling to develop a workable, coherent standard of copyright protection for software. Their decisions show several important trends. First, it is clear that current copyright law protects, at a minimum, the source code and object code of a computer program against unauthorized literal copying of the code.[301] However, because of the Copyright Office's recent ruling on computer screen displays, its not clear as to what extent that current copyright law protects screen displays resulting from computer programs through a single copyright. The cases do make clear, however, that the displays do constitute an original work of authorship fixed in a tangible medium of expression and are copyrightable in some form. Unfortunately, it is not clear as to what class of authorship a screen display is; courts have classified them either as audiovisual works or compilations, depending upon the screen's graphic content.[302] Beyond these two points, the extent of current copyright protection of computer software under current law becomes even less and less certain.

Clearly, the *Whelan* case extends copyright protection beyond the code itself. Although less certain, the court's decision in *Whelan* appears to extend the scope of the program code's copyright to include the program's internal structure and design. Furthermore, the court's decision holds that a party may present similarities or differences in the program's screen displays, subroutines, and file modules as evidence to support or refute an infringement claim.[303] Beyond these holdings,

300. *Softklone* also demonstrates that until the Copyright Office's recent ruling on single copyright registration for computer software, developers were forced to obtain separate copyrights to fully protect their software creations. *Copyright Ruling, supra* note 16, at 153.

301. *See supra* text accompanying notes 107-134.

302. *Protecting the Look and Feel, supra* note 8, at 430-32 (noting that the scope of protection given to the screen will vary depending upon the kind of copyright given to it). *But see Copyright Ruling, supra* note 16 at 152-53 (holding that text based screens should not be copyrightable as compilations).

303. *See Defining the Scope, supra* note 28, at 529; *Common Law, Uncommon Software, supra* note 133, at 1103; *Protecting Look and Feel, supra* note 8, at 427. Note

Whelan is subject to much controversy.[304] However, an analysis of the holding demonstrates that *Whelan* should not be extended to protect anything beyond the internal structure of the program; in other words, a copyright on the program only protects the program code and program's internal design.[305]

Finally, the copyright law is even more confused regarding the protectibility of the user interface element of the computer program.[306] While the *Broderbund* court protected the user interface's menu sequences and menu structure,[307] the *Softklone* court protected only the particular arrangement of data on the screen, which is the screen display, and not the interface. The concept of user interface, while commented upon frequently in the literature surrounding the question of copyright protection for computer software, has not been *directly* protected by courts other than in *Broderbund*. Although it did not protect the program's interface, the court in *Softklone* did, however, note that a *system of inputting data* would be protectible if it demonstrated that it possessed "stylistic creativity" above and beyond the requirements needed for inputting data.[308] However, because the *Softklone* court only protected the program display and not the interface, the use of the court's opinion as support for protection of the use interface remains difficult.

While current standards are somewhat unclear and unresolved, especially when considering the user interface, the issues concerning the protectibility of non-literal aspects of computer programs remain critical to the software industry. Current law, because of its fuzziness in defining parameters of protection, risks discouraging innovation rather

---

that displays, subroutines and file modules are only evidence of copying of the internal design of the program.

304. One view is that *Whelan* extends to all elements of computer program; that is, the program's copyright covers all elements of the program. This is the *Broderbund* view. The more correct view is that *Whelan* only extends to the internal design of the program; the particular design that the programmer used to solve a certain problem. *Softklone*, 659 F.Supp. at 455-56; *Defining the Scope*, *supra* note 28, at 528-30; *Common Law, Uncommon Software*, *supra* note 133, at 1103.

305. *See supra* notes 138-170 and accompanying text. *Broderbund* was the only source to disagree with this analysis. *Softklone* expressly disagreed with the *Broderbund* holding. *But see Copyright Ruling*, *supra* note 16 at 152-53 (holding that one copyright registration covers all aspects of the computer program).

306. The user interface differs from the screen display as it helps control the inputting of data into the computer. The display reveals the results of the computer's functioning.

307. However, the *Broderbund* court reasoned incorrectly, and similar reasoning by other courts might lead to overprotection of the user interface. The sequence and structure of the menus should have been protected only to the extent they were not necessary to the idea of menu-driven printing program.

308. *Softklone*, 659 F.Supp. at 460. The court did not define stylistic creativity.

than encouraging it.[309] Requiring multiple copyrights, while being somewhat effective in protecting all currently protectible aspects of the software, only adds to the confusion and likelihood that a developer will not gain protection for his creative labor.[310] As the *Broderbund* court noted, "It would [be] . . . unreasonable to expect copyright holders to know at any given time the exact state of the law on the scope of copyright protection."[311]

Finally, and most important, current copyright standards may be failing to protect what the developers believe to be the most important aspects of software. Failure of the law to protect the totality or "look and feel" of a work of software could result in consequences that would undermine the intellectual progress that underlies the existence of copyright law. Without copyright protection of the totality of software, there will be a greater incentive to copy successful products rather than to develop new programs.[312] In order to ensure development of innovative software, the industry requires a new, clear and comprehensive standard of copyright protection that will encourage the development of innovative software. The new standard must be precise, yet also flexible enough to lead to consistent and *practical* judicial decisions; the new standard must also be comprehensive and reasonable enough to encourage innovation without stifling competition. This Note's proposal, by suggesting the creation of a new work of authorship, and by specifically extending the statutory protection of computer software to include all aspects of software, including code, internal design, screen displays, and user interfaces, will achieve these goals and further software development.

IV. PROPOSAL: TOWARD A CLEAR AND COMPREHENSIVE COPYRIGHT STANDARD FOR COMPUTER SOFTWARE

The CONTU Commission began its report by recognizing that the development of information technology is a living and ongoing process. Technology will only improve if there is continuous innovation; and, it is the purpose of copyright law to encourage that innovation. Unfortunately, legal structures designed for non-software entities and other

309. *See infra* note 316 and accompanying text.

310. *Defining the Scope*, *supra* note 28, at 530-31. It was precisely for this reason that the Copyright Office rejected registering multiple copyrights for computer software. *See Copyright Ruling*, *supra* note 16, at 152-53.

311. *Broderbund*, 648 F.Supp. at 1135.

312. "Even though copyrights stop developers from putting out products for a very cheap price, it encourages them to spend time and money to develop superior products." George Juarez, President of Nantucket Corp., quoted in PC MAGAZINE, May 27, 1987 at 8, at 175.

technologies may not keep pace with newly developed technologies.[313] Indeed, the Commission found that the "universe of works protectible by statutory copyright has expanded along with the imagination, communications media, and technical capabilities of society,"[314] from the original notions of copyright possessed by the framers of the Constitution. Copyright law, then, is fluid, flexible and changing. It must adapt to changing environments and changing technologies if it is to succeed in encouraging innovative and creative development.

Current copyright protection for computer software remains unclear, inconsistent, and potentially under-protective and damaging to an industry which thrives on innovation.[315] With standards that are questionable or confusing, and that fail to protect the financially important aspects of the work, *current* copyright law is not the most efficient method of protecting a software developer's intellectual creations. The current standard might actually discourage, rather than encourage, creative intellectual development.[316] Finally, because the current standard is unclear, judicial determinations risk being arbitrary. There is a danger that some decisions will overprotect software and other decisions will underprotect software, and that the decision will based upon such non-judicial factors as the character of the infringing parties.[317]

313. *CONTU Report*, *supra* note 1, at 3. The Commission noted that its entire existence was based upon the fact that "[t]he adequacy of the legal structure to cope with the pace and rate of technological change frequently has been called into question." *Id. See Protecting the Look and Feel*, *supra* note 8, at 445.

314. *CONTU Report*, *supra* note 1, at 11.

315. The CONTU Report noted that in order for computer programs to be disseminated, the programmers would have to be able to regain their investment in their work. The Commission concluded that the best way to do this was through copyright protection. *CONTU Report*, *supra* note 1, at 11. A *sui generis* method of protection, which might be cumbersome and complicated, would this goal. *See Defining the Scope*, *supra* note 28, at 530-34.

316. The CONTU Report left it to the courts to determine the scope of the protection the Copyright Act would give to software. *CONTU Report*, *supra* note 1, at 22-23. However, as Section III of this Note demonstrates, the courts have been unable to formulate a consistent and workable standard, and this could undermine future software innovation. As one member of the computer industry noted of the current standards, "I've talked to venture capitalists who don't want to fund companies because they don't think that what the guy is doing is protectible. It makes it hard for people to raise money." *Reuters*, Jun. 3, 1987. If current standards are causing reluctance to innovate on the part of developers, the standards must be modified to encourage rather than discourage innovation.

317. At this point, courts have erred both in overprotecting (*Broderbund*) and underprotecting (*Softklone*) software. As an example of the arbitrariness of the current standard, and effect that the parties have upon the decision of the case, one industry member noted: "To a great extent, we have to apply the 'smell' test to each situation. The *Whelan* case is one of those situations where the court was faced with a not very sympathetic defendant." Lee Bendekgey, quoted in InfoWorld, *Debate*, *supra* note 8, at 8. The effect and unpopularity of Lotus Corporation's involvement with the issue may have jaded the in-

The current standard of copyright protection for computer software is flawed. This Note, therefore, proposes a new standard of copyright protection for computer software, one that responds to marketplace realities and that will effectively work to encourage creative thought in the industry. The new standard should be statutorily based and specific enough to give guidance to courts applying it. The new standard should respond to current industry needs and should protect the totality or "look and feel" of the software work. Most important, the new standard must balance the conflicting interests of software innovators and software users; it must foster progress without constricting ideas; it must work to secure "the general benefits derived by the public from the labors of authors. . . ."[318] This Note's proposal, by focusing the infringement inquiry upon the program's code, design, display output, and user interface, and by creating the optimum level of protection for software, protects the totality of a computer program, responds to market realities, and encourages innovation in the industry.

A. THE IMPORTANCE OF LOOK AND FEEL

The *Whelan* court found that encoding a computer program was not the most difficult, nor the most expensive, aspect of computer programming.[319] Similarly, the program code is also not the most valuable aspect of the computer program. Instead, it is the "look and feel" or totality of all aspects of the software, when viewed as a whole, that forms the most valuable part of a computer program. Hence, the Copyright Act must address look and feel concerns because the issue concerns "proprietary rights worth hundreds of millions of dollars, [and] whether copyright laws will foster or discourage innovation."[320]Inadequate protection of software allows subsequent software manufacturers to capitalize upon existing works that are successful. The subsequent developer can exploit the existing work by recoding it and selling it at a less expensive price than the originator. Clearly, this practice interferes

---

dustry's view toward the issue in those cases. As Bill Howard, executive editor of PC Magazine has said, "Too many people confuse their feelings about Lotus with their feelings about the Lotus lawsuit. They don't particularly admire Lotus so they don't want Lotus to win. . . .The Supreme Court once noted its most-famous cases involve not very nice people. But that doesn't deny their day in court. Lotus has a legitimate beef. . . ." Rosch, *supra* note 8, at 161.

318. NIMMER, *supra* note 14, at 1.03[A], 1-31, 1-32.

319. *Whelan*, 797 F.2d at 1231.

320. N.Y. Times, Jan. 19, 1987, § D, at 2. In addition, one industry commentator has said "the rulings that result [from the determination of the look and feel controversy] may sharply abridge [the public's] ability to buy programs [it] want[s] or design programs that operate like other programs." PC MAGAZINE, *supra* note 8, at 155.

with the innovator's stream of revenues. [321] Such a situation, if left unchecked, will discourage innovation and *encourage copying* of sucessful existing programs.[322] Conversely, a standard that protects the totality (or "look and feel") of software will "force people to use their creativity and ingenuity to develop truly innovative software. They [will not] have to worry about being sued . . . and [will] make a real contribution to the industry."[323]

The debate, however, has not been one-sided, with many commentators, both legal and industrial, arguing that "look and feel" protection would be detrimental to the computer industry. They are against "look and feel" protection because such protection might conflict with current copyright formulations,[324] would inhibit competition,[325] would stifle innovation,[326] and might force software makers to constantly develop new user interfaces to avoid copyright infringement.[327]

---

321. *Forbes,* June 15, 1987. Another commentator's opinion exemplifies just how unoriginal and imitative the subsequent software packages are, writing:

> It looks like 1-2-3 [the name of Lotus's Spreadsheet program]. It works like 1-2-3. All the commands are the same, all your old Lotus worksheets load into it without a hitch, and all your hard-earned macros run just fine. It's a near perfect clone. At $50, it sounds too good to be true. *Or at least too good to be legal.* (emphasis added).

PC MAGAZINE, *supra* note 8, at 157.

322. *See In Their Own Words: Lotus Mosaic, and Paperback Software State Their Cases,* PC MAGAZINE, May 26, 1987, at 162-63, where Lotus Corporation outlines its arguments for look and feel protection, noting specifically that:

> [r]ecently, more and more software development talent, money, and time has been spent on imitation rather on true innovation. . . . Absent protection from copyright law, the true innovators in our industry (and those who might back them) might become increasingly concerned whether the rewards of innovation adequately compensate them for the time and money they risk.

323. Peter Marx, legal counsel for the Information Industry Association, quoted in LaPlante, *supra* note 240, at 8.

324. *See Protecting Look and Feel, supra* note 8, (the author argues that there is no way to separate idea and expression for the non-code elements of software).

325. *See Look and Feel, supra* note 17, at 134.

326. In terms of copyright analysis, this may be the most serious, indictment of "look and feel" protection. There is a fear that if software is overprotected, then "innovation, which often occurs by improving on the ideas used in other interfaces, will be curtailed." *Look and Feel, supra* note 17, at 133. Indeed, one commentator notes: "This industry has had a history of people taking existing software into garages and coming out later with vastly superior products." LaPlante, *supra* note 240, at 1, col. 5. Finally, another commentator adds, "The biggest concern. . . will be that creativity will be stifled if new developers can't improve upon previous software efforts and if existing software companies won't improve their products because they don't have to." Lohse, *Who Owns the Standards,* PC Magazine, May 26, 1987, at 176.

327. Michael Scott, Executive Director of the Center for Computer/Law (and the publisher of the *Computer/Law Journal*) has said: "Do you really want 20 different user friendly concepts? You would have to learn each one, which defeats the whole purpose." Ranney, *supra* note 6, at 13. Other commentators have offered similar arguments against

The proposed new standard addresses all of these concerns. First, while the new standard *changes* current copyright law, it does not violate basic copyright principles. The new standard, by requiring a court to conduct an independent idea/expression analysis upon each of the four enumerated elements of a computer program,[328] only mandates protection of the expressionable aspects of software. Furthermore, the standard's cumulative infringement test enables a designer to innovate on some aspects of existing works, to keep other aspects similar, and still to avoid infringing upon the copyrighted work.

Second, while clone programs, which *offer little, or nothing innovative to the consumer*, will be eliminated, thus reducing to some degree competition in the industry, copyright law, by design, is supposed to partially limit competition in order to encourage the development of new products. Instead of permitting lower priced copies of existing programs, the proposed standard will encourage the development of different and innovative products. The best copyright policy encourages *new and progressive products, not cheaper versions of existing products.*

Third, the proposed standard still enables developers to use aspects of prior programs when innovating new programs. The proposed standard will allow a developer to use a prior program's ideas; and, the proposed standard will allow an innovator to copy small parts of the existing work's expression, as long as the new program is not substantially similar, *as a whole*, to the old program.[329]

Finally, the proposed standard does not protect the basic interface

---

look and feel protection. *See* LaPlante, *supra* note 240, at 8. "every interface will now have to be different."

However, there has also been a response on this point from the industry. Ed Esber, President of Ashton-Tate (a software developer) has said:

> "Certain aspects of the user interface should be in the public domain, such as icons or moving cursor menus. . . . It would be detrimental if the precedent set was so tight that developers had to invent new interfaces just to get around the copyright, especially if that meant that users would have to learn 72 ways to interact with different software packages. On the other hand, I don't think protection should be limited to the code. . . . If the product looks exactly the same to the user as somebody else's product, I believe the line of illegality has been crossed.

Rosch, *supra* note 8, at 158. *See also A Thousand Clones*, *supra* note 6, at 214-16 (the author argues that compatibility arguments do not override the copyright law's purpose to encourage innovation. He concludes that compatibility concerns, therefore, should not trump innovation concerns).

328. Although some commentators argue that there is no way to extract the expressionable material from non-code elements of a computer program, this Note argues that the plurality of expressions test, when focused upon one software element at a time (and the not global approach taken by the *Broderbund* court) does adequately enable a court to separate protectible and non-protectible elements of the program.

329. For example, it might be possible to copy one aspect of the program and create completely new versions of the remaining three aspects of the original software.

design of a computer program. Instead, it protects only those aspects of the interface that are not necessary elements of the interface, or necessary to the underlying purpose of the program. Hence, developers will not have to constantly create an entirely new interface design each time they write a new computer program.

The scope of copyright protection for computer programs remains a difficult and major issue for both the computer and legal communities. The determination of the "look and feel" problem will have a tremendous impact upon the software industry. Although the courts have leaned toward protecting the totality of the computer program from copying, the standards remain inadequate. Consequently, a new standard is in order.

B. CREATING, CLARIFYING AND CODIFYING: A PROPOSAL FOR COMPREHENSIVE COPYRIGHT PROTECTION OF COMPUTER SOFTWARE

In order to respond to current industry realities and to fulfill the purposes of copyright, this Note proposes that Congress amend the Copyright Act to create a special new work of authorship called "computer software".[330] This work of authorship would provide courts with a framework for analyzing computer software and would protect the totality of the computer program. It would explicitly note that each work of "computer software" includes four specific aspects: program code, internal program design, display output, and user interface. All of these aspects will be considered to have its own separate (but related) ideas and expressions, though they need not be exclusive from each other. Clearly, only the expressionable part of each aspect would be copyright-

330. Under current law, a computer program is considered a literary work. This view comes the legislative history of the 1976 Act. H.R. Rep. No. 1476, 94th Cong. 2d Sess. 54. Because CONTU believed that Congress intended to include computer software within the category of literary works, CONTU rejected the idea of a separate work of authorship category for computer software as unnecessary. *CONTU Report*, *supra* note 1, at 16.

However, the current formulation is inadequate. Although Congress intended for the copyright on a program to extend beyond the literal code to include all facets of a program "to the extent that they incorporate authorship in the programmer's expression of original ideas, as distinguished from the ideas themselves," H.R. Rep. No. 1476, 94th Cong. 2d Sess. 51. the current categorization of literary works has led, until recently, to separate registration for the display aspects of the screen. *See Copyright Ruling*, *supra* note 16, at 152-53; *cf. Look and Feel supra*, note 17, at 123. It is unclear at this time exactly what effect the single registration requirment will have on the "look and feel" issue.

In any case, computer software simply does not fit into the current categories of authorship. "The limitations of the old concepts fail to reconcile the competing commercial and scientific interest of a modern industry." *Protecting the Look and Feel*, *supra* note 8, at 445.

able.[331] After conducting this analysis, the proposal instructs the court to cumulate the similarities and differences of all four elements of both programs when determining if there has been infringement.

The proposal, by setting out a framework of analysis for courts, will provide much needed consistency in the software copyright area. By separating the four elements of the program, and then cumulating them for the purpose of determining copyright infringement, the developer gains protection for the entirety of his work, but only for those parts of the work that are expressionable. Such a standard responds directly to criticisms of *Broderbund*, which argue that each element, particularly that of the user interface, has its own underlying idea and expression.[332] *The proposal's separate idea/expression analysis of each program element ensures that a program will not be overprotected.* Furthermore, the proposed cumulative infringement standard ensures that the work will not be underprotected. In addition, the work of authorship would be registered, in conformity with the recent decision of the Copyright Office, as a whole and not with separate copyrights.[333] This would ensure that when a developer registers his program with the Copyright Office, all aspects of the program will receive protection.

This Note's proposal requires that a court, when determining if there has been infringement, first examine each of the four ennumerated program elements for copyrightable material. After this initial inquiry, the court would then separately compare the similarities and differences of each element of the programs under dispute.[334] After considering the various degrees of copying within each of the four ennumerated elements of a software work (code, code design, display and interface), the court will have to cumulate the similarities and differences together and decide whether it believes that the two programs in dispute are, *as a whole*, substantially similar to each other.[335] The court will have to make "a qualitative, not quantitative judgment about the character of the work as a whole and the importance of the substan-

---

331. This does mean that the courts will have to determine, within each aspect, what is expression and what is non-protectible idea. There is no way, however, to legislate such an inquiry, as Judge Learned Hand noted, such an inquiry is inevitably "ad hoc". *Whelan*, 797 F.2d at 1235 (quoting Peter Pan Fabrics, Inc. v. Martin Weiner Corp. 274 F.2d 487, 489 (2d Cir. 1960)). However, the proposed legislation would recommend that the courts use the plurality of expression test within each aspect to determine the expressionable ideas.

332. *Look and Feel*, *supra* note 17, at 124-25 n.152.

333. *See Copyright Ruling*, *supra* note 16, at 152-53.

334. The court would only compare the copyrightable portions of each element.

335. *See Defining the Scope*, *supra* note 28, at 533, in which the author proposes a cumulative standard for copyright by which the screens and codes are accumulated for substantial similarity purposes. However, the author does not go as far as this proposal, as she continues to adhere to current categories of works of authorship.

tially similar portions of the work."[336]

The Note's proposal is innovative because it gives a court the ability to determine separately all the copyrightable elements of the work, and then allows it to integrate the elements together for its final determination of whether the works are similar.[337] In addition, dividing the aspects of the work will enable authors to showcase the differences between the works and elements, and perhaps demonstrate that while part of the plaintiff's work's expression was copied, the resulting work was substantially different. *The totality proposal could work in either direction, protecting whole works and also encouraging improvements upon existing works.*[338] Most important, by focusing the inquiry upon the whole of a program, the proposal forces courts to protect the *financial* interest of the developer. By protecting this financial interest, the proposal encourages others to invest time and money into developing new products, which enables society to progress intellectually.

Finally, while no standard can create a certainty of outcome for every potential case, the proposed standard provides a guide and framework to courts faced with cases in this area. It directs the court's inquiry and leaves only the actual application of the standard to the court. By providing such a framework, the proposal will create consistency in the law of this area, which is the most any proposal of this nature can hope to do.

V. CONCLUSION

The expansion of home computers and popular software develop-

336. *Whelan*, 797 F.2d at 1245. The court also said, "*There can be no bright line rules* as to when similarities are evidence of infringement and when they are legitimate— that is a determination to be made by the trier of fact." 797 F.2d at 1248 n.47.

Hence, this Note cannot not enumerate exactly what would rise to the level of infringement in any particular case; it cannot attempt to articulate that x amount of similarity in the codes, y amount in the structures of the codes, z amount in the screens, and d amount in the protectible portions of the user interfaces will rise to copyright infringement. That is clearly a matter for the courts. What this Note can do, however, is provide a *framework* in which the court can consistently make that determination. No standard can be so precise as to determine in advance how much copying will always rise to the level on infringement.

337. By analyzing each element separately, the court will be sure to protect only the expressionable aspects of the program.

338. For example, one can imagine a jury being shown the similarities between program interfaces which may or may not rise to the level of infringement. However, they would also be forced to consider the differences in program code, program design, and program displays which are not similar. In such a situation, it is likely that a jury would not find copying, thus allowing the defendant to build off a current work of software. Indeed, Sir Issac Newton's statement "if [he] had seen farther than other men, it was because [he] had stood on the shoulders of giants" exemplifies the need to allow subsequent developers to build from existing programs. *Whelan*, 797 F.2d at 1238 n.33.

ment has produced a vexing problem for copyright law. As the market expands and technology increases, the old categories and mechanisms of software protection rapidly become outmoded and fail to protect creators' interests. With the rapidly expanding marketplace, under current laws, the incentive may be not to innovate or develop a new software product, but instead to steal from other developers' ideas and infringe upon other's intellectual endeavors. The marketplace, with its emphasis on consumerism, has begun to view software differently, with an emphasis on its appearance and usability. When the copyright law was created for this area, it was not designed to meet these new issues.

Despite their struggling, courts have been unable to resolve the new problems concerning computer software. Most likely, their failures stem from the inadequacies of current law. In any case judicial decisions have tended to overprotect or underprotect software. In either case, the decisions have made it difficult for developers to gain adequate protection over their creative efforts.

This Note has examined the current standards for software protection and has specifically studied the copyright law and its relationship to the four distinguishable elements of a piece of software: code, design, display and interface. It has recognized the difficult task that courts face in extricating expressions from ideas. It has surveyed the courts' attempts to make the current standard, which defines a computer program as a "literary work," somehow respond to a marketplace that focuses not on the literary elements of software, but on the "look and feel" of software.

The current standard must be changed if it is to respond to the needs of the marketplace, to be administered coherently and consistently, and to answer to the needs of users and innovators. The Note has proposed a new, specially designed category of "computer software" as work of authorship in the Copyright Act. Such a category would specifically enumerate that a copyright on computer software includes the program code, the program's structure and design, the program's display output, and the program's user interface. The developer registering the work would register all aspects of the work under one copyright, and would be assured that his entire work was protected. This process would be more simple and reliable. Infringement determinations would be based upon the cumulative differences in each element of of the programs under dispute; in this way each element of the programs would still be subject to separate idea and expression analyses. This would prevent global protection of the entire work from inadvertently protecting program ideas, rather than solely protecting expression. The standard would punish those that infringed a substantial amount of the entire program. Conversely, those who simply built

upon an existing program, but made significant improvements, would still be free to reap the fruits of their innovation.

Finally, as with any innovative proposal, there are bound to be unforeseeable problems that arise in application. However, a new standard is clearly in order to eliminate the confusion and inconsistency of current copyright law. This Note's proposal seeks to fulfill the goals of copyright. They are not easy goals to meet. Lord Mansfield's statement, although made a long time ago, still holds true in the age of the personal computer:

> [W]e must take care to guard against two extremes equally prejudicial; the one, that men of ability, who have employed their time for the service of the community, may not be deprived of their just merits, and the reward for their ingenuity and labour; the other, that the world may not be deprived of improvements, nor the progress of the arts be retarded.[339]

The current standards are inadequate to the current tasks. The courts have shown that their struggled interpretations are not able to forge the consistency and depth that developers currently need. This Note's proposal provides a possible answer to the current pressing problems in the computer software industry.

Jack Sholkoff *

339. Sayre v. Moore, 102 Eng. Rep. 138, 140 n.6 (1785) (quoted in *Whelan* 797 F.2d at 1235 n.27).

© 1988 by Jack Steven Sholkoff.

* The author wishes to thank the following people for their generous assistance in helping to prepare this Note: Professor Erwin Chemerinsky, Laurie Balcom, Misara Shao, Richard Phillips, Daryl Teshima, Mignon Worman and Ami Rubinfeld. I would also like to thank Laura-Beth Berenson for her understanding and caring, and for many other reasons that are too numerous to name here. A special thank you goes to my Mother and Father, without whose support, motivation and patience, this Note would not have been possible. Of course, any errors are my own.