NOTE

COPYRIGHT, POTENTIAL MARKETS, AND THE USER INTERFACE: DEFINING THE SCOPE OF THE LIMITED MONOPOLY

I

INTRODUCTION

Since the enactment of the Copyright Act of 1976,[1] courts have faced the difficult task of applying venerable copyright principles to cases involving computer programs.[2] The 1976 Act protects both "application" and "system" computer programs as literary works.[3] Courts have protected both types of programs,[4] and have made it clear that copyright protects the literal aspects of computer programs,[5] including both source and object code.[6] This

---

In her article in this symposium (see page 311), Professor Pamela Samuelson analyzes the user interface of Lotus 1-2-3 and concludes that only the text of the opening screens, the explanatory text in the on-line facility, and the explanatory long prompts are expressive enough to satisfy traditional copyright standards. Professor Samuelson, herself, then points out that reasonable minds might disagree about what elements of the user interface are expressive enough to satisfy copyright standards. This author is one who disagrees, believing that the expressive elements of user interfaces are not limited to those elements listed by Professor Samuelson in her article, because user interfaces, like music, are expressive in ways that cannot be captured in a list of elements. This note focuses, however, on the next phase of inquiry, leaving the issue of expressiveness to be resolved by the courts.

[1] 17 USC §§ 101-810 (1982).

[2] The Copyright Act defines a computer program as "a set of statements or instructions to be used directly or indirectly in a computer program to bring about a certain result." 17 USC § 101 (1982).

[3] Copyright Law Revision, HR Rep No 1476, 94th Cong, 2d Sess 54 (1976), reprinted in 1976 USCCAN 5664. Computer programs can be classified as either "system" programs or "application" programs. System programs work together to form an operating system. An operating system creates the environment in which application programs function, and manages the various elements of a computer system's hardware, including the monitor and disk drives. Operating systems communicate with the computer rather than with the user. See Note, *A Thousand Clones: The Scope of Copyright Protection in the "Look and Feel" of Computer Programs*, 63 Wash L Rev 195, 196 (1988)(authored by Alan S. Middleton), citing *Apple Computer v Franklin Computer Corp.*, 714 F2d 1240, 1243 (3d Cir 1983) (operating system programs "generally manage the internal functions of the computer or facilitate use of application programs").

[4] See, for example, *Lotus Dev. Corp. v Paperback Software Intl*, 740 F Supp 37, 45 (D Mass 1990), citing *Stern Electronics, Inc. v Kaufman*, 669 F2d 852, 855 n3 (2d Cir 1982) ("written computer programs are copyrightable as literary works"); *SAS Inst. v S & H Computer Sys.*, 605 F Supp 816 (MD Tenn 1985) (protecting application program); *Franklin Computer*, 714 F2d 1240 (protecting systems program).

[5] Computer programs can be printed out and observed in text form. By comparing computer programs in their text form, mechanical duplications can be detected easily.

[6] See, for example, *Franklin Computer*, 714 F2d at 1243 (source and object code copyrightable). Object code is a program written in machine language that can be executed directly by the computers' central processing unit (CPU) without need for translation. *Paperback*, 740 F Supp at 43-44. Source code is a program written in a universal programming language, such as BASIC, that

note focuses on application programs, which allow the user to perform a specific task, such as word processing or cite checking.[7]

Application programs present unique challenges to courts that are trying to define the scope of copyright protection under the 1976 Act. Unlike traditional literary works, such as books or maps, an application program, when run on a computer, produces a nonliteral user interface. The user interface, however, or the "look and feel" of a computer program,[8] allows a user to communicate with a computer.[9] Some commentators have compared the relationship of a user interface and its underlying computer program to the face of a wristwatch and its inner workings.[10] The user interface, however, unlike the face of a watch, is actually a product of the underlying program. In this respect, the user interface is similar to music, and the computer program is like the compact disc in which the user interface is recorded. Both computer programs and musical compositions can be recorded in literal notation. However, a user interface, like music, is not experienced in its literal manifestation. Additionally, a user interface, like music, can be replicated without literal copying of its source. A musical work can be recreated by listening and imitating the sounds without mechanically copying the medium of fixation. A user interface can be recreated without mechanically copying the underlying program by a process known as reverse engineering. The term "reverse engineering" refers to a process of recreating a program by observing the responses and output of the program to various input and creating an independent program to achieve the same results. For these reasons, the user interface can be thought of as the "nonliteral" aspect of a computer program.

Moreover, computer programs that produce identical user interfaces may not be substantially similar when viewed only in their literal forms. Since

---

must be translated by an interpreter to be read by a computer's CPU. Id at 44. Object code is easily read by machines; source code is easily read by humans.

[7] Note, 63 Wash L Rev at 196-97 (cited in note 3), citing *Franklin Computer*, 714 F2d at 1243 ("Application programs usually perform a specific task for the computer user, such as word processing, checkbook balancing, or playing a game.").

[8] The term "look and feel" has been used to refer to different concepts. See, for example, Note, *Copyright Protection for Computer Languages: Creative Incentive or Technological Threat?*, 39 Emory L J 1293, 1294 n6 (1990) (authored by Elizabeth G. Lowry). For purposes of this note, the terms "user interface" and the "look and feel" of a computer program will be used interchangeably.

[9] See Anthony Chandor, *Penguin Dictionary of Computers* 472 (Penguin, 3d ed 1985) (defining "user interface" as a "term used to describe any way in which a user accesses a computer system, for example, through a visual display unit, a personal computer or a videotext terminal").

[10] See, for example, Pamela Samuelson, *Computer Programs, User Interfaces, and Section 102(b) of the Copyright Act of 1976: A Critique of* Lotus v. Paperback, 55 L & Contemp Probs 311 (Spring 1992). Most people do not think of the face of a wristwatch as a user interface for its inner workings. The typical wristwatch face consists of two hands, a numbered dial, and a device to adjust the time. The numbered dial and hands of the wristwatch's face convert unintelligible information from the inner workings of the watch into a readable and efficient format. Similarly, a computer program's user interface displays information from electronic signals in an intelligible format. The external device on a wristwatch that adjusts the time allows the user to communicate or interact with the inner workings. Likewise, a computer program's user interface allows a user to communicate with the underlying program. A computer program's user interface, like the face of a wristwatch, conveys information to the user and allows communication with the underlying machine.

most people who use computers experience only the user interface and never see the computer program in its literal form, copyright protection for only the literal aspects of computer programs, without protection for their nonliteral user interfaces, would be virtually meaningless.

Copyright protection for a traditional literary work gives the author a monopoly for a specified number of years.[11] However, if copyright protected only the literal aspects of computer programs, an author of a computer program would be protected only from mechanical duplication; and her monopoly would last only until another programmer could create a clone.[12] Through reverse engineering, clones often can be written in a relatively short amount of time, depending on the complexity of the original program. While some protection of the nonliteral aspects of computer programs is necessary,[13] the danger of chilling the development of new programs is great, because courts must decide the proper scope of the copyright monopoly without the convenience and familiarity of text to help them identify protectible expression. For these reasons, the debate now focuses on the scope of copyright protection for the nonliteral product of application programs.

Most of the cases that have dealt with the issue of copyright protection for the user interface arguably have been decided correctly, based on their facts.[14] The language in these decisions is often ambiguous,[15] however, and reveals a willingness to give the copyright monopoly a scope that is too broad.[16] None of the decisions has analyzed the goals of copyright in terms of the type of market for which the programs compete. Commentators have heavily criticized the latest decision involving user interfaces, *Lotus Development Corp. v. Paperback Software International*,[17] because it can be interpreted to provide too much protection for computer programs.[18] The uncertainty in

---

[11] For example, an author of a book written on or after January 1, 1978 has a complete monopoly for the work from the moment of its creation until 50 years after the author's death. 17 USC § 302(a) (1982).

[12] Clones are computer programs that have virtually identical user interfaces and functions but whose literal aspects are entirely dissimilar to the original program. Note, 63 Wash L Rev at 215 (cited in note 3). Although the user interfaces are usually identical and thus interchangeable to the user, the computer code that creates the "clone" is an independently written program. Id. This is possible for two main reasons: computer programs can be written in many different languages; and a myriad of different commands can achieve the same result on a computer. Therefore, a clone would not infringe the copyright of the source or object code in the original program if copyright protection did not extend to the nonliteral aspects of user interfaces.

[13] Copyrights of more traditional forms of literary works have been infringed by works with nonliteral similarities. See, for example, *Twentieth Century-Fox Film Corp. v MCA, Inc.*, 715 F2d 1327, 1329 (9th Cir 1983) (13 alleged distinctive plot similarities between *Battlestar Galactica* and *Star Wars* may be basis for finding copyright violation).

[14] See text accompanying notes 45-96.

[15] See Samuelson, 55 L & Contemp Probs at 314 n14 (cited in note 10) (arguing that copyright cases have tended to be either lengthy and elaborately flawed, or so cryptic as to provide little guidance on the court's reasoning).

[16] Paul Goldstein, *Infringement of Copyright in Computer Programs*, 47 U Pitt L Rev 1119, 1125-26 (1986).

[17] 740 F Supp 37 (D Mass 1990).

[18] See generally, for example, Samuelson, 55 L & Contemp Probs 311 (cited in note 10); Ronald Abramson, *Why* Lotus-Paperback *Uses the Wrong Test and What the New Software Protection*

the level of protection for computer programs has stifled innovation and chilled the "speech" that copyright seeks to encourage.[19] Unfortunately, the *Paperback* decision will not be reviewed on appeal due to an out-of-court settlement.[20]

This note will argue that courts can reduce the amount of uncertainty about the level of copyright protection for user interfaces by considering the type of market for which programs compete. Computer programs with substantially similar user interfaces compete in one of three main types of markets: the actual market; a potential market for different hardware; and a potential market for programs with different underlying purposes. By considering for which market the programs compete, courts can analyze defenses to copyright infringement more accurately and further the goals of copyright law. This note will further argue that the monopoly afforded by copyright should extend only to the actual market, not to any potential markets. Even if courts do not agree that copyright protection for user interfaces should be limited to programs that compete for the same actual market, an analysis that uses the language of actual or potential markets would illuminate the courts' reasoning and inject a greater degree of predictability into their decisions.

Part II of this note provides a brief summary of the goals of copyright law and outlines some major copyright principles that apply to user interface decisions. Part III presents the facts of the main copyright decisions involving the protection of user interfaces to illustrate the current state of the law and demonstrate the uncertainty in the level of applicable copyright protection. Part IV defines the types of markets for user interfaces and examines their differences with respect to market incentives and the goals of copyright. Part IV also discusses how the validity of certain infringement defenses changes according to the type of market at issue, and concludes that copyright protection should be limited to programs that compete in the same actual market. Part V proposes an infringement test that would retain the ad hoc nature of copyright decisions, yet allow software developers to predict the outcome of future user interface cases more accurately. The conclusion suggests that courts should recognize the different types of markets for

---

Legislation Should Look Like, 7 Computer L 6 (Aug 1990); D. Lee Antton & Gary M. Hoffman, *The Impact of* Lotus Development v Paperback Software, 7 Computer L 1 (Aug 1990); Richard H. Stern, *Legal Protection of Screen Displays and Other User Interfaces for Computers: A Problem in Balancing Incentives for Creation Against Need for Free Access to the Utilitarian*, 14 Colum-VLA J L & Arts 283 (1990); Note, Lotus Development Corp. v Paperback Software International: *Broad Copyright Protection For User Interfaces Ignores the Software Industry's Trend Toward Standardization*, 52 U Pitt L Rev 689 (1991) (authored by Gerard J. Lewis, Jr.); Note, 39 Emory L J at 1330-35 (cited in note 8).

[19] See Alfred C. Yen, *A First Amendment Perspective on the Idea/Expression Dichotomy and Copyright in a Work's "Total Concept and Feel"*, 38 Emory L J 393, 431 n192 (1989) (" 'Look-and-feel lawsuits are already impacting product development. And because hardware and software development takes so long, *the uncertainty brought on . . . will affect the products we see for up to 3 years*. Some of this effect will be irreversible.' " (quoting PC Magazine at 169 (May 26, 1987) (emphasis added), in turn quoting Dan Bricklin, developer of the VisiCalc spreadsheet program).

[20] See Samuelson, 55 L & Contemp Probs at 313 n10 (cited in note 10) (discussing details of settlement).

computer programs and examine the market incentives, the goals of copyright, and the role of standardization in light of the distinct characteristics of each market.

II

SOME FUNDAMENTALS OF COPYRIGHT LAW

The U.S. Constitution authorizes Congress "[t]o 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."[21] Congress grants copyright monopolies to "serve the public welfare by encouraging authors to generate new ideas and disclose them to the public, being free to do so in any uniquely expressed way they may choose."[22] Copyright law attempts to balance the author's interest in profit against society's interest in use.[23] Thus, courts must achieve a delicate balance between granting too strong a monopoly and failing to offer sufficient incentives to encourage authors to contribute to the public welfare. Either result would chill the development of technology and undermine the goals of copyright.

Copyright only protects the expression of ideas, not the ideas themselves.[24] In *Baker v. Selden*,[25] the leading case on the distinction between idea and expression, the Supreme Court ruled that a copyright on a book, which described a new system of bookkeeping and contained forms employing the new system, was not infringed by another book that contained similar forms. Although the Court recognized that the book contained copyrightable expression, it refused to extend copyright protection to the ideas expressed by the book.[26] Thus, subsequent authors have been free to use the ideas contained in a work if they do not infringe on the expression of those ideas.[27]

The doctrines of merger and *scènes à faire* limit the copyright monopoly to ensure that copyright does not protect ideas only or chill the development of

---

[21] US Const Art I, § 8, cl 8.

[22] *Paperback*, 740 F Supp at 52, citing *Harper & Row Publishers v Nation Enterprises*, 471 US 539, 546 (1985).

[23] "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." *Twentieth Century Music Corp. v Aiken*, 422 US 151, 156 (1975).

[24] 17 USC § 102(b) (1982); *Mazer v Stein*, 347 US 201, 217-18 (1954); *Baker v Selden*, 101 US 99, 102-03 (1880).

[25] 101 US 99; see also Samuelson, 55 L & Contemp Probs at 326 n70 (cited in note 10) (discussing *Baker v. Selden*).

[26] *Baker*, 101 US at 104 ("[A]ny person may practice and use the art itself which he has described and illustrated therein"). The Supreme Court later interpreted *Baker* to hold "that a copyrighted book on a peculiar system of bookkeeping was not infringed by a *similar* book using a *similar* plan which achieved *similar* results where the alleged infringer made a *different arrangement of the columns and used different headings*." *Mazer*, 347 US at 217 (emphasis added). Thus it appears that the Court will protect authors against verbatim, or close to verbatim copying, but not against loose paraphrasing. Goldstein, 47 U Pitt L Rev at 1124 (cited in note 16).

[27] *Nichols v Universal Pictures Corp.*, 45 F2d 119, 121 (2d Cir 1930).

technology. Under the doctrine of merger, if two works are similar because the underlying idea can be expressed only in a limited number of ways, the expression is not protected because the idea is said to have "merged" with the expression.[28] Courts frequently give factual works less protection than fictional works due to the merger doctrine.[29] Similarly, the doctrine of *scènes à faire*, or "scenes which 'must' be done,"[30] limits the protection of certain expressions of ideas that, as a practical matter, are indispensable to the treatment of a topic.[31]

To prove infringement of a copyright, a plaintiff must demonstrate ownership of the copyright and actual copying by the defendant.[32] Because direct evidence of actual copying is often difficult to obtain, courts usually infer actual copying by proving access to the copyrighted work[33] and substantial similarity[34] between the works. To prove substantial similarity, courts typically use the bifurcated test of *Arnstein v. Porter.*[35] Under *Arnstein*, the fact-finder must first decide whether there is sufficient similarity between the two works to conclude that the alleged infringer used the copyrighted work.[36] Expert witnesses testify on this issue, which has sometimes been called the "extrinsic" test of substantial similarity.[37] If the answer to the first question is affirmative, the fact-finder then moves to the "intrinsic" test, which is decided without the aid of expert testimony. Under this test, the fact-finder decides, from the perspective of the "lay observer," whether the copying was "illicit" or "an unlawful appropriation" of the copyrighted work.[38]

Traditionally, copyright has protected authors from infringements by works in the same market as the copyrighted work, as well as from infringements in "potential" markets.[39] In fact, Congress requires courts to

---

[28] See, for example, *Herbert Rosenthal Jewelry Corp. v Kalpakian*, 446 F2d 738, 742 (9th Cir 1971) (applying the merger doctrine to a jeweled bee pin).

[29] Note, 63 Wash L Rev at 205 (cited in note 3), citing *Landsberg v Scrabble Crossword Game Players*, 736 F2d 485, 488 (9th Cir 1984).

[30] Melville B. Nimmer & David Nimmer, 3 *Nimmer on Copyright* § 13.03[B][4] (1991) ("*Nimmer on Copyright*"), citing *Walker v Time Life Films, Inc.*, 784 F2d 44, 50 (2d Cir 1986).

[31] *Atari, Inc. v North American Philips Consumer Electronics Corp.*, 672 F2d 607, 616 (7th Cir 1982).

[32] Note, Manufacturers Technologies, Inc. v. Cams, Inc.—*The Legal Fiction Created by a Single Copyright Registration of a Computer Program and Its Display Screens*, 65 Notre Dame L Rev 536, 545-46 (1990) (authored by Ginamarie A. Guadio).

[33] Although access is generally defined as the opportunity to copy, some courts have required proof that the duplicating author knew of and viewed the original work. 3 *Nimmer on Copyright* at § 13.02 (cited in note 30).

[34] Courts with difficulty must draw lines to determine substantial similarity:

> Somewhere between the one extreme of no similarity and the other complete and literal similarity lies the line marking off the boundaries of "substantial similarity." Judge Learned Hand has said that this line "wherever it is drawn will seem arbitrary" and that "the test for infringement of a copyright is of necessity vague."

3 *Nimmer on Copyright* at § 13.03[A] (cited in note 30), quoting *Nichols*, 45 F2d at 122, and *Peter Pan Fabrics v Martin Weiner Corp.*, 274 F2d 487, 489 (2d Cir 1960).

[35] 154 F2d 464, 468-69 (2d Cir 1946).

[36] *Whelan Associates, Inc. v Jaslow Dental Lab., Inc.*, 797 F2d 1222, 1232 (3d Cir 1986).

[37] Id.

[38] Id.

[39] See *Harper*, 471 US at 568 (protecting a book against quotation before publication in a news article); but see *Sony Corp. v Universal City Studios*, 464 US 417 (1984) (holding that to allow "time

consider the effect of an infringing use upon the potential market of a work when evaluating a fair use defense.[40] Fair use is a statutory defense that excuses infringements of copyright for certain purposes including criticism, comment, news reporting, teaching, scholarship, and research.[41] Courts have interpreted the protection of the potential market as the single most important element of fair use,[42] but have not required any showing of actual harm, only a showing by a "preponderance of the evidence that *some* meaningful likelihood of future harm [to the potential market] exists."[43] The fair use defense, though somewhat beyond the scope of this note, is mentioned here to illustrate that the concept of potential markets is firmly rooted in copyright law. It is also important to note that Congress does not require courts to protect potential markets but only to "consider[] . . . the effect [of a copyright infringement] upon the potential market [of a copyrighted work]."[44]

III

SIGNIFICANT USER INTERFACE CASES

The facts of four of the most significant decisions involving user interfaces illustrate the current uncertainty about the relevant level of copyright protection. The following discussion of these cases is limited to the aspects of the cases that illustrate the ambiguity in the courts' decisions and the role that an analysis of potential markets could play.

A. *Whelan Associates, Inc. v. Jaslow Dental Laboratory, Inc.*

Whelan was the first decision to address directly the copyright issues relating to protection for computer programs' user interfaces.[45] Whelan

---

shifting" through home videotaping would not cause any likelihood of nonminimal harm to the potential market of copyrighted broadcasts).

[40] "In determining whether the use made of a work in any particular case is a fair use the factors to be considered shall include . . . (4) the effect of the use upon the potential market for or value of the copyrighted work." 17 USC § 107 (1988).

[41] 17 USC § 107. The statute names four factors to be considered when evaluating a defense of fair use:

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

Id (emphasis added).

[42] *Harper*, 471 US at 566.

[43] *Sony*, 464 US at 451. Included in the protection of potential markets is protection for derivative works. *Harper* at 568. "A derivative work is a work based upon one or more preexisting works, such as a translation . . . ." 17 USC § 101.

[44] 17 USC § 101.

[45] *Whelan*, 797 F2d 1222. Courts had faced similar issues in earlier decisions involving videogame copyrights. See *M. Kramer Mfg. Co. v Andrews*, 783 F2d 421 (4th Cir 1986); *Williams Elec., Inc. v Artic Intl, Inc.*, 685 F2d 870 (3d Cir 1982); *Midway Mfg. Co. v Strohon*, 564 F Supp 741 (ND Ill 1983).

Associates developed a computer program named "Dentalab" to perform basic bookkeeping and administrative tasks for Jaslow Dental Laboratory.[46] Whelan and Jaslow then entered into a contract, in which Whelan agreed to continue to improve the program, and Jaslow agreed to market it.[47] Jaslow realized that there might be a market for a program that served essentially the same function as Dentalab but was written in a different computer language to operate on different hardware.[48] Jaslow then independently created a new program named "Dentcom" and marketed it as "a new version of the Dentalab computer system."[49] Jaslow developed the Dentcom program to capture the potential market of Dentalab for computers with different hardware. Although Dentcom was not a literal translation of Dentalab, the programs had virtually identical screen outputs, file structures, and subroutines.[50] Whelan filed suit in the federal district court, claiming that the Dentcom program infringed its copyright for Dentalab. The district court ruled for Whelan, finding that Dentcom was substantially similar to Dentalab due to its structure and general organization, although it was not a direct transliteration and was written in a different computer language.[51] The Third Circuit affirmed.[52]

The Third Circuit held that copyright protection could extend beyond the literal source and object code to the program's "structure, sequence, and organization" ("SSO").[53] The *Whelan* court then departed from established copyright principles by rejecting the *Arnstein*[54] bifurcated substantial-similarity test[55] and formulating a new method of distinguishing between unprotectable *ideas* and protectable *expression* in computer software. The court based this new test on its analysis of *Baker v. Selden*,[56] and suggested that "the purpose or

---

[46] *Whelan*, 797 F2d at 1226.

[47] Id.

[48] Id. Computer programs are written in specific languages for specific types of hardware. External constraints limit the types of software that will work on a computer. For example, a program written to operate on IBM computers will not operate on the Macintosh system.

[49] Id at 1227.

[50] Id at 1228. The five subroutines that controlled order entry, invoicing, accounts receivable, end-of-day procedure, and end-of-month procedure were performed almost identically in both programs. Id.

[51] Id at 1229.

[52] Id at 1248.

[53] Id. The *Whelan* court used the terms "structure," "sequence," and "organization" interchangeably and intended them to be synonymous. Id at 1224. Although "structure, sequence, and organization" (SSO) is sometimes used to refer to a user interface, the phrase can refer to many aspects of a computer program that are unrelated to a user interface. For example, the *Whelan* court rests its holding mainly on the similarities between the SSO of two computer program's file structures, id at 1242-48, that the user does not experience in day-to-day use of the program. While the file structure may have an indirect effect on the way that a program "looks and feels," it is not a part of the user interface. Thus, although the SSO includes the user interface, it is not limited to aspects of the user interface.

[54] See text accompanying notes 35-38.

[55] *Whelan*, 797 F2d at 1232. The court rejected the test due to the complexity of computer programs, the unfamiliarity of this technology to most members of the public, and the doubtful value of the test when the same trier of fact is used for both extrinsic and intrinsic tests (since the fact-finder cannot forget the expert testimony for the intrinsic part of the test). Id at 1232-33.

[56] 101 US 99 (1880).

function of a utilitarian work would be the work's idea, and everything that is not necessary to that purpose or function would be part of the expression of the idea."[57] The court rejected the argument that copyright protection for computer programs would retard progress by interfering with the "stepping-stones" method of progress used by programmers.[58] According to the court, "progress in computer technology or technique [was not] qualitatively different from progress in other areas of science or the arts."[59]

The *Whelan* court rested its decision primarily on the similarity of the SSO in the file structures and subroutines of the two programs,[60] both of which are not a part of the user interface. In fact, Whelan did not even claim infringement with respect to the screen outputs.[61] While the decision did mention their similarity, the court did not consider the screen output to be covered by the same copyright as the computer program, and referred to the similarity as merely "indirect, inferential evidence of the nature of material covered by another copyright."[62] Thus, the *Whelan* decision has limited value in user interface cases beyond its holding that copyright protection can extend beyond the literal source and object code of computer programs.

B. *Broderbund Software, Inc. v. Unison World, Inc.*

Broderbund,[63] like *Whelan*, involved a program that exploited the potential market for different hardware. In *Broderbund*, however, the claim was based entirely on the similarity of the user interfaces. Broderbund Software owned the rights to a program called "Print Shop" that created greeting cards.[64] Print Shop was developed for Apple computers and would not function on IBM computers because the two systems' hardware are incompatible. Broderbund began negotiations with Unison World to convert Print Shop into a program that could be used on IBM computers.[65] As a result of the negotiations, Unison began writing an independent program for IBM computers that simulated Print Shop as closely as possible.[66] During the

---

[57] *Whelan*, 797 F2d at 1236. The *Whelan* purpose test has been heavily criticized by commentators. See, for example, Note, *Idea, Process, or Protected Expression?: Determining the Scope of Copyright Protection of the Structure of Computer Programs*, 88 Mich L Rev 866, 879-85 (1990) (authored by Steven R. Englund); Note, Whelan Associates v. Jaslow Dental Laboratories: *Copyright Protection for Computer Software Structure—What's the Purpose?*, 1987 Wis L Rev 859 (1987) (authored by Thomas M. Gage). One of the main faults of the "purpose" test is that it falsely assumes that a computer program is composed of only one idea. Thus, courts using this test may wrongly protect many ideas as expression.

[58] *Whelan*, 797 F2d at 1238. The "stepping-stones" method of progress refers to the process of making a series of improvements, each one building on the last. See Note, *Copyright Infringement of Computer Programs: A Modification of the Substantial Similarity Test*, 68 Minn L Rev 1264, 1292 (1984) (authored by Howard Root).

[59] *Whelan*, 797 F2d at 1238.

[60] Id at 1245-48.

[61] Id at 1244.

[62] Id at 1243-46.

[63] *Broderbund Software, Inc. v Unison World, Inc.*, 648 F Supp 1127 (ND Cal 1986).

[64] Id at 1129-30.

[65] Id at 1130.

[66] Id.

development process, negotiations between Broderbund and Unison broke down.[67] Consequently, Unison stopped trying to replicate Print Shop, but it continued to develop a program named "Printmaster," which had started as the IBM duplicate of Print Shop.[68] Although Unison was no longer trying to simulate Print Shop, it did not discard the previously written user interfaces that emulated Print Shop.[69] Printmaster was actually an enhanced version of Print Shop; the programmers added improved features after the negotiations broke down, and they were no longer constrained to create an exact duplicate of Print Shop.[70]

Even though *Whelan* only considered the screen displays as inferential evidence of other infringement. The court in *Broderbund* read *Whelan* as holding that copyright protection "is not limited to the literal aspects of a computer program, but rather . . . extends to the overall structure of a program, including its audiovisual displays;"[71] using the *Arnstein* substantial similarity test,[72] the court found that Printmaster infringed the Print Shop copyright because the "look and feel" of the user interfaces was substantially similar.[73] The *Broderbund* case thus protected the potential market of Print Shop against an enhanced program with added features solely due to the similarity between their nonliteral user interfaces.

C. *Digital Communications Associates, Inc. v. Softklone Distributing Corp.*

The dispute in *Softklone* concerned the rights to a computer program owned by Digital called "Crosstalk," which allowed computers to communicate with one another.[74] Softklone obtained a commercially available copy of the program and independently wrote a program named "Mirror" that was identical to Crosstalk in both look and operation.[75] Although Softklone did not copy any of the literal aspects of Crosstalk, the two programs' user interfaces were virtually identical.[76] Crosstalk and Mirror competed in the same actual market; both were designed to serve the same primary purpose, facilitating communication between computers, and to operate on the same hardware.

---

[67] Id at 1131.

[68] Id.

[69] Id.

[70] Id.

[71] Id at 1133.

[72] However, the court opined that an integrated test involving expert testimony and analytic dissection would be the wave of the future. Id at 1136.

[73] Id at 1137. The court observed that "a mere list[] of similarities [could] not adequately convey the impression of overall similarity . . . ." Id. See Pamela Samuelson & Robert Glushko, *Comparing the Views of Lawyers and User Interface Designers on the Software Copyright "Look and Feel" Lawsuits*, 30 Jurimetrics J 121 (1989) (criticizing "look and feel" lawsuits).

[74] *Digital Communications Associates, Inc. v Softklone Distrib. Corp.*, 659 F Supp 449, 452 (ND Ga 1987).

[75] Id at 453. The Mirror program is a classic example of a clone.

[76] Id.

The *Softklone* court held that the user interface was copyrightable as a literary work,[77] and found that Mirror was an infringement of Digital's copyright of Crosstalk. In so doing, it rejected Softklone's arguments that the idea and expression had merged,[78] the screen was an uncopyrightable blank form,[79] and the user interface should not be protected due to the importance of standardization in the computer industry.[80] The *Softklone* court followed the *Whelan* "purpose" test to distinguish between idea and expression.[81] Since the user interfaces of Crosstalk and Mirror were identical, the court did not have to analyze the programs to establish substantial similarity.[82] It is important to note that the *Softklone* court relied on *Whelan* to reject the defendant's standardization argument without pointing out the differences between the types of programs and markets at issue in *Whelan*.[83] This failure laid the groundwork for the broad generalizations found in later decisions that have caused much uncertainty about the level of copyright protection for user interfaces.

D. *Lotus Development Corp. v. Paperback Software International*

Paperback is the most recent case to struggle with the difficulties of defining copyright protection for the user interface.[84] The defendant, Paperback Software, developed a spreadsheet program named VP-Planner for IBM personal computers.[85] While improving VP-Planner, Paperback observed Lotus 1-2-3 for the first time and noted its commercial success.[86] Based on this observation, Paperback concluded that it would have to make VP-Planner compatible with Lotus 1-2-3 to compete in the market.[87] Paperback then proceeded to convert VP-Planner into a "workalike for 1-2-3" by revising the hierarchical menu structure, ensuring that all keystroke sequences produced the same result, adding certain functions of 1-2-3, and deleting functions that were not part of 1-2-3.[88] The VP-Planner manual described itself as a "feature-for-feature workalike for 1-2-3," and claimed that it was "designed to

---

[77] Id at 465. The court concluded that a computer program's copyright protection did not extend to a program's screen displays and thus required the plaintiff to establish actual copying by a basis other than merely looking at the screen and creating a clone, such as proving that defendant copied the source or object code. Id at 455-56. However, the plaintiff in this case had obtained copyright registration on the Crosstalk status screen; the court used this copyright to protect the program. Id at 456. The Copyright Office later ruled that a program and its screen displays are one work, protected by a single copyright. Notice of Registration Decision: Registration and Deposit of Computer Screen Displays, 53 Fed Reg 21,817 (1988).

[78] *Softklone*, 659 F Supp at 465.

[79] Id at 460.

[80] Id at 462.

[81] Id at 458.

[82] They did, however, cite the substantial similarity test used in *Arnstein*. Id at 464.

[83] Id at 462. See text accompanying notes 54-55 (discussing the "stepping-stone" argument rejected in *Whelan*).

[84] *Paperback*, 740 F Supp 37.

[85] Id at 68-70.

[86] Id at 69.

[87] Id.

[88] Id.

work like Lotus 1-2-3, keystroke for keystroke."[89] Lotus 1-2-3 and VP-Planner competed in the same actual market because both programs functioned on the same hardware and had the same basic purpose of spreadsheet creation.

The court in *Paperback* departed from both the bifurcated substantial-similarity test of *Arnstein* and the purpose test of *Whelan*. In place of these tests, the court adopted a three-part test to determine copyrightability:

> FIRST, . . . the decisionmaker must focus upon alternatives that counsel may suggest, or the court may conceive, *along the scale from the most generalized conception to the most particularized* and choose some formulation—some conception or definition of the idea—for the purpose of distinguishing between the idea and its expression. . . . SECOND, the decisionmaker must focus upon whether an alleged expression of the idea is limited to elements essential to the expression of that idea (or is one of only a few ways of expressing the idea) or instead includes identifiable elements of expression not essential to every expression of that idea. THIRD, having identified elements of expression not essential to every expression of the idea, the decisionmaker must focus on whether those elements are a substantial part of the allegedly copyrightable "work."[90]

The court stated that "[if] the decisionmaker, weighing the relevant factors, determines that the legal test applying the idea-expression distinction is satisfied, copyrightability is established."[91] Applying this standard, the court found that VP-Planner infringed the copyright of Lotus 1-2-3.[92] Although the court used "nonliteral" terminology, the opinion relies on "verbatim copyright" to support its finding of infringement.[93] The court rejected the defendant's defenses based on the need for compatibility and standardization.[94] While the court's rationale for rejecting the defenses may have been sound on the facts, its language was ambiguous and potentially overbroad for two reasons. First, the court neglected to point out the differences between the types of programs and markets at issue in *Paperback* with those of *Broderbund* and *Whelan*. Second, the court used the general term "standardization" to refer to a defense that differed significantly from the defense presented in *Broderbund*, which referred to the similar defense used in *Whelan*. The *Paperback* decision has been heavily criticized because its ambiguity can be interpreted as protecting an overly broad copyright monopoly for user interfaces.[95]

While most would agree that the cases discussed above reached the desired result based on their facts,[96] each decision has added ambiguity to the

---

[89] Id. Although VP-Planner was virtually identical to 1-2-3, there were differences in the start-up screens, placement on the screen of the menu lines, exact wording of the long prompts, organization of the help screens, and in the increased width of the VP-Planner screen, which allowed the user to hide certain columns. Id at 70.

[90] Id at 60-61 (emphasis in original).

[91] Id at 61.

[92] Id at 84.

[93] Id at 70.

[94] Id at 77-79.

[95] See note 18 (listing articles criticizing *Paperback*).

[96] Although most commentators agree that these cases reached the desired result, some argue that copyright law was not the proper medium for reaching those results. See, for example, Richard A. Beutel, *Trade Dress Protection For the "Look and Feel" of Software: A New Source of Proprietary Rights*

law because the courts neglected to recognize the differences in the facts and used general phrases to describe significantly different concepts. The next section will argue that this uncertainty can be lessened by analyzing user interfaces in terms of their potential markets and recognizing the differences in defenses, such as standardization, when raised in each market.

IV

ANALYZING THE PROTECTION OF USER INTERFACES IN TERMS OF POTENTIAL MARKETS

For traditional literary works such as books, copyright protects all potential markets and makes no attempt to classify the potential markets into conceptual categories.[97] Similarly, none of the major decisions involving user interfaces has attempted to distinguish the differences among potential markets.[98] However, the potential markets for user interfaces, unlike those for other literary works, can be organized into distinct categories. By using these categories, courts can properly identify relevant market incentives and evaluate defenses based on standardization. Although courts and commentators typically refer to standardization as if it were a single concept that has equal force regardless of the type of program[99] or market at issue, standardization arguments and market incentives differ greatly according to

---

Protection For the Software Industry?, 5 Computer L 1 (Oct 1988) (suggesting that a trade dress theory rather than copyright should be used to protect software).

[97] Although the courts distinguish these derivative works, parody and criticism are not considered exploitations of potential markets for the purposes of this note. See note 43 (explaining derivative work).

[98] See text accompanying notes 45-96.

[99] Most commentators base their standardization arguments on cases that involved spreadsheet programs. See, for example, Note, 39 Emory L J at 1293 (cited in note 8) (basing standardization and language arguments on *Ashton-Tate v Fox Software and Santa Cruz Operation*, No 88-6837-TJH (CD Cal filed Nov 18, 1988), and *Paperback*, 740 F Supp 37); Note, 52 U Pitt L Rev at 689 (cited in note 18) (basing standardization argument on *Lotus*, 740 F Supp 37). However, spreadsheet programs are distinguishable from most other computer programs. A spreadsheet program, such as Lotus 1-2-3, helps the user create worksheets, graphs, and documents based on the information calculated or stored in the worksheet. *Lotus 1-2-3 Release 3.1: Tutorial* vii (Lotus Dev. Corp., 1990). A worksheet is a "grid that provides a structure for entering and *calculating* data, and storing and organizing information." Id at 1-1 (emphasis added). Thus, a worksheet is actually a type of program that a user creates by using Lotus 1-2-3 or any spreadsheet program. When the worksheet is conceptualized as a program, the user interface can be analogized to a language that is used to create the "worksheet" program. When the user interface is discussed in this analytical framework and analogized to a language, the arguments for not protecting the user interface sound compelling. The user interface for spreadsheet programs presents only the exception to the rule, however, not the rule itself.

Most programs, such as word processors like WordPerfect 5.1, graphics programs like CorelDraw 3.0, and graphic interfaces like Windows 3.0, allow the operator to use the program only to create output or perform the functions originally included with the program, not to create new programs such as worksheets. Although even non-spreadsheet programs allow the user to create mini-programs called macros, which are "a series of keystrokes and []commands that perform [the main program] tasks," id at 5-1, the user cannot create independent programs, beyond these shortcuts for existing keyboard commands..

While programs will not always fall cleanly into either the category of spreadsheet-type programs, which create independent programs, or non-spreadsheet programs, which can create only macros (for example, WordPerfect 5.1 has limited spreadsheet-like capabilities that allow the user to store the potential market at issue. A proper evaluation of these potential markets will aid courts in providing the proper balance of copyright protections.

Computer programs that have substantially similar user interfaces compete in one of three main markets: the actual market, potential markets for different hardware, and potential markets for programs with different underlying purposes. The following sections will discuss each type of market and identify the associated market incentives and standardization arguments.

A. The Actual Market

User interfaces with underlying programs that operate on compatible hardware and share the same basic purpose[100] compete in the same actual market. Programmers create clones[101] of popular programs to compete in the same actual market for two main reasons. First, they create clones to attract purchasers who are familiar with the existing program but cannot afford its price. A programmer can duplicate an existing program through reverse engineering without investing as much money in research and development as the developer of the original program.[102] Since the existing program is popular, the second programmer does not need to spend as much money for marketing. Consequently, the second programmer can offer the clone for a lower price, attracting both users who could not afford the existing program and new users who could afford it but choose to save money by purchasing the clone. The facts in *Softklone* represent this situation.[103] Mirror was a practically identical clone of Crosstalk that sold for a significantly lower price.[104]

Second, programmers create clones of popular programs to attract users who are hesitant to learn a different user interface or users who have files created by the existing program that cannot operate properly with a new user

---

commands in a document to calculate data), recognition of the main purpose and operation of a program would help a trier of fact evaluate standardization arguments.

Some commentators base standardization arguments relating to retraining costs on the training costs for spreadsheet programs. While complex programs such as Lotus 1-2-3 do have relatively high training costs, training costs for programs vary with the complexity of the program. See Note, *Litigation as a Mechanism for Inefficiency in Software Copyright Law*, 39 UCLA L Rev 397, 400 n10 (1991) (authored by William H. Wright) (rudimentary training cost for each user of Lotus 1-2-3 is approximately one thousand dollars). Additionally, retraining costs are greatly reduced if the competing program directs training at users who are familiar with other, similar programs. See, for example, *Product Profiles*, 3 PC Sources 504 (Jan 1992) (describing Word for Windows 2.0). Thus, the barriers created by training costs will vary greatly, depending on both the type of programs involved and the knowledge of the user.

[100] For example, the purpose of an underlying program may be word processing, spreadsheets creation, or graphics generation.

[101] See note 12 and accompanying text.

[102] Some commentators contend that developers, after reverse engineering another program, incur substantially equal costs in manufacturing a competing product. See, for example, Dennis S. Karjala, *Copyright, Computer Software, and the New Protectionism*, 28 Jurimetrics J 33, 56 (1987). However, clone programs often sell for an amount that is substantially less than that of the original program. See, for example, note 104.

[103] See text accompanying notes 74-83.

[104] The list price for Crosstalk was $195 while Mirror sold for $69.95. Note, 63 Wash L Rev at 215 n154 (cited in note 3), citing PC Clones, Premier Edition 1987, at 8.

interface. This type of clone usually runs faster or contains additional valuable features not included in the original program.[105] The facts of *Paperback* illustrate this situation.[106] Clones created for this purpose arguably advance the science and useful arts by improving the existing program. However, if courts do not protect the original program from competition by clones, the original author may not recover his or her original development and marketing costs.[107]

Some argue that standardization of user interfaces encourages the development of the underlying programs.[108] If a user does not have to consider the time and expense involved with learning a new user interface, the user will choose a program based solely on the functionality of the underlying program, not on the user interface.[109] Additionally, clone manufacturers argue that standardization serves the public interest by encouraging the development of improved and less expensive competing programs with common interfaces. While standardization could encourage competition in the underlying program, it also could have a chilling effect on the development of new underlying programs because it endangers the software developer's ability to recoup development expenses.[110] For the same reasons, standardization removes the incentives to invest time and money[111] to develop improved user interfaces.

Other commentators argue that standardization is necessary where the original program has a monopoly created by user preference or resistance to change.[112] If a product, such as a typewriter keyboard, gains widespread public acceptance, and copyright protects the layout of the keys for the keyboard, users become locked-in to the original product since competitors cannot emulate the popular layout of keys.[113] The commentators argue that once the lock-in phenomenon begins, competitors no longer have incentives

---

[105] This type of clone is a derivative work. See note 43.

[106] See text accompanying notes 84-95.

[107] Traditional copyright theories for literary works reserve the right to make derivative works to the original author. See note 43.

[108] The defendant in *Softklone* argued that the need for standardization in the computer industry justified its creation of Mirror, a clone of plaintiff's Crosstalk program. Note, 63 Wash L Rev at 214 n151 (cited in note 3). The defendant in *Paperback* also presented a standardization defense. See *Paperback*, 740 F Supp at 77-79 (rejecting standardization argument). However, the exact reasoning of the defense was not recorded in the decision. Id.

[109] See Note, 63 Wash L Rev at 214-16 (cited in note 3) (discussing standardization arguments).

[110] The user interface is often the determinative factor in the marketability of a program. Id at 216.

[111] Software developers invest substantial amounts of time and money in developing user interfaces. "For example, Xerox employees 'devoted about thirty work-years to the design of the Star user interface . . . [and] worked for two years before [writing] a single line of actual product software.' " Id at 216 n158, quoting Dr. David C. Smith, et al, *Designing the Star User Interface*, BYTE 242, 246 (April 1982) (emphasis in original).

[112] See, for example, Karjala, 28 Jurimetrics J at 44-48 (cited in note 102) (discussing monopoly through user preference or resistance to change); Note, *The Incompatibility of Copyright and Computer Software: An Economic Evaluation and a Proposal for a Marketplace Solution*, 66 NC L Rev 977, 997-98 (1988) (authored by Vance Franklin Brown) (arguing that protection of the user interface would create an insurmountable barrier to entry due to retraining costs).

[113] Karjala, 28 Jurimetrics J at 45-46 (cited in note 102).

to create products that offer small sequential improvements to the existing product, because users will not learn the new keyboard unless the improvements are significant enough to justify the time and expense.[114] In this situation, competing developers may not invest money into research and development because it is difficult to predict the significance of improvements until after the development process is complete.

For user interfaces that compete in the same actual market, the user preference monopoly argument does not justify infringement of expression that meets the traditional tests for copyright. First, the user-preference monopoly argument assumes that improvement can be achieved only through separate, self-sufficient products. However, computer programmers can create add-in programs[115] to achieve the desired sequential improvements. By working with the existing program, users do not have to leave the widely accepted user interface of the existing program.[116] Thus, if a programmer develops an improved spell-checking section for an existing word processing program, he or she could market the new spell-checker as an "improvement" program that alters the popular existing word processor. Users who own the original program could then purchase the improved spell-checking program and load it into their computer.[117] Since the users must purchase both the original and the improvement program, add-in programs preserve the incentives and monopolies for both authors. In this manner, competing programmers can create small, sequential improvements in existing computer programs without encountering the obstacle of user-preference monopolies and without destroying the incentives of the original programmer.[118]

Second, the user-preference monopoly argument, when stating that retraining costs create an "insurmountable barrier" to competition,[119] fails to recognize that these costs differ with various types of programs and users.[120]

---

[114] Id at 46.

[115] Add-in programs are independent programs that either alter or work with an existing program. Add-in programs, such as Norton Desktop for Windows, are fairly common in today's computer software market.

[116] This possibility was not practical with past technology due to the difficulty in physically modifying mechanical products, such as the typewriter example used to justify the standardization argument.

[117] This is true only for programs that are designed to compete in the same actual market. If the programs are designed for use on different hardware, the improvement program cannot work with the existing program. Similarly, if the programs are designed for different purposes, the improvement will often not work with an existing program because the improvements are often program specific. However, add-in programs are quite effective for programs that compete for the same actual market. Unlike clumsy mechanical adapters, add-in programs merge so effectively with the original program that the user cannot tell that it was not an original part of the program.

[118] The court in *Paperback* recognized the faults of the user preference monopoly argument and implicitly rejected it by discussing the role of add-in programs. *Paperback*, 740 F Supp at 78-79.

[119] See Note, 66 NC L Rev at 997-98 (cited in note 112).

[120] Retraining costs are related to the complexity of the program. For example, the retraining costs for a relatively simple program such as Crosstalk will be substantially less than for a more complex program such as Lotus 1-2-3. See note 99. Further, retraining costs will be less for individuals and small businesses than for major corporations with numerous employees to retrain. For example, if a program's retraining cost is equal to *X*, a business owning one program that needs to train one employee will experience a total retraining cost equal to *X*. However, a business owning one program that needs to train five employees will experience a total retraining cost equal to 5*X*.

Additionally, retraining costs are zero for new users in the market who are unfamiliar with any programs. Thus, it is improbable that retraining costs will create an insurmountable barrier to competition because competitors can target either new users in the market or users who have lower retraining costs.[121] Competitors can capture part of the market by offering a product that sells for a lower price or one that contains improved features.[122] For these same reasons, it is unlikely that owners of existing copyrights for user interfaces would take unfair advantage of the monopoly created by user preference by raising the price too high.[123]

Finally, commentators argue that standardization is necessary to allow transfer of computer files among users.[124] However, secondary programs could use conversion programs to overcome incompatibility created by different user interfaces.[125] While conversion programs do require an extra step to gain compatibility, they are a feasible solution to the compatibility problems created by transfering files between programs with different user interfaces.

Courts should protect user interfaces that compete for the same actual market because standardization is not necessary and could erode the market incentives that copyright seeks to preserve. When the benefits of standardization can be achieved through alternate means, standardization should not be used as a justification for infringements that endanger the incentives that copyright seeks to preserve.

While this example oversimplifies the problem by ignoring the possibilities for group training discounts, the point remains true that training costs vary with the number of people to be trained.

[121] Evidence of competition can be seen by looking at the market for computer spreadsheets. Although Lotus 1-2-3 holds the largest share of the market, a computer user may still choose another competitor such as Microsoft Excel, Quattro Pro, Wingz, Allways, or Supercalc V.

[122] The competitor's success will depend upon a number of factors, including the complexity of the program, the average retraining costs, the number of new users entering the market, the amount of improvement over the existing program, and the price difference between the new and existing program.

[123] Some commentators have argued that granting copyright protection to the user interface would result in unreasonable prices for computer software. See note 99. However, a recent advertisement listed the price of Lotus 1-2-3, even after its successful copyright suit, as only $389.00 (version 3.1) as compared with other popular spreadsheets such as Microsoft Excel ($293.00), Quattro Pro ($308.00), and Wingz for Windows ($329.00). *PC Sources* 96 (Jan 1992). The higher the price of the program, the more likely competitors can lure away new users to the market and users that have lower retraining costs. Further, if a developer raises the price of software too much, he or she will encourage the creation and distribution of illegal, private copies among consumers to avoid paying the prohibitive price. Additionally, it is unlikely that a software producer would ever raise the price of the program to a level that is above the price of competing programs plus average retraining costs. Even if the software producer priced a program at a price equal to the cost of the competing program plus the cost of retraining, a program that is more efficient or easier to use could sway purchasers to lose money in the initial retraining period in favor of long range considerations.

[124] Note, 52 U Pitt L Rev at 693 (cited in note 18).

[125] The court in *Lotus* rejected this argument recognizing that translation programs, such as the one used by *Microsoft Excel*, are a practical solution to incompatibility problems. *Lotus*, 740 F Supp at 78.

B. Potential Markets for Different Hardware

An existing program can be translated or rewritten for use on different types of hardware.[126] Each type of incompatible hardware represents a potential market for existing programs. The *Whelan* and *Broderbund* cases involve disputes about programs attempting to exploit this type of potential market.[127]

If the courts protect this potential market, hardware that is too new to have a large market,[128] or that has a distinct but limited market, might never benefit from the technology developed for other hardware.[129] Although it may be fairly easy to translate some computer programs for use on other types of hardware, the programmer of the original program does not have an incentive to translate the original program unless he or she perceives a large enough market to justify the expense of translation. Even if courts do not protect this potential market, the original author has a great advantage over those seeking to translate the program because he or she is intimately familiar with it.[130] Thus, by not protecting this potential market, courts encourage the original programmer to take advantage of his or her knowledge of the program by quickly translating it for use on other hardware before competitors develop similar programs. If courts protect the potential market for different hardware, they could chill the development of technology, because some types of hardware might never benefit from advances made in software programs developed for other hardware.

The need for standardization in the computer industry is compelling in this situation because programs written for different types of hardware will not erode the economic incentives that copyright seeks to secure for an author. The programmer of a clone written for different hardware than the original program is not competing for the same customers as the original. The clone will harm the original program's market only if the hardware for which the clone is designed is superior to the hardware of the original program. Software should not be protected by copyright against market losses caused by consumers choosing to invest in improved hardware.

The user-preference monopoly argument is more persuasive when the program competes in the potential market for two reasons. First, protection for the potential market in this situation could chill the development of technology by causing consumers to choose hardware based on the available software rather than on the merits of the hardware itself. Consumers might be reluctant to choose improved hardware, due to fears about the availability

---

[126] This situation is analogous to the traditional translation of a literary work from one language to another, which is protected as a derivative work.

[127] See text accompanying notes 45-73.

[128] New types of hardware may never gain widespread acceptance unless programs that have a widespread following are quickly available for use on the new hardware due to the consumer "lock-in" problems discussed in notes 112-14 and accompanying text.

[129] See Karjala, 28 Jurimetrics J at 78 (cited in note 102) (discussing the dangers of offering too much protection to programs against programs that operate on different hardware).

[130] See id at 79 (discussing advantage of knowledge to original programmer).

of software, and thus the new hardware might not gain a sufficient market to create the incentives necessary to convince programmers to transliterate existing programs for the new hardware. By not protecting potential markets for different hardware, courts would create the incentive for software developers to translate their programs for as many types of hardware as possible, and consumers would not be discouraged from investing in improved hardware due to fears about software unavailability.

Second, add-in programs cannot be used to achieve sequential improvements in this type of potential market.[131] Add-in programs work with existing programs. Thus, a secondary programmer is not able to provide sequential improvements in software for this market unless he or she has the right to create a clone of the original program that was written for use on different hardware.

For these reasons, copyright should not protect programs from clones that operate on hardware that is incompatible with the type for which the original program was designed.[132] Although the *Broderbund* and *Whelan* decisions arguably reached the correct result based on their facts, the court should not have based protection in copyright law.[133]

C. Potential Markets for Programs with Different Underlying Purposes

The same user interface can be used for programs with underlying programs that have different purposes. For example, the user interface for Windows, a popular graphical menu program, could be imitated and used for a word-processing program. The purpose of Windows is to organize multiple programs and run them under a common user interface. The purpose of a word-processing program is the production of documents, a goal unrelated to the management of multiple programs. Although Windows can run a word-processing program under its user interface, the purposes of the two programs are entirely different. Thus, the two programs are competing for different markets. Each type of program serving a purpose other than that of the original represents a potential market for the user interface of the original program.

Although no decisions have discussed copyright disputes based on these facts, protection of the potential market that exists for adapting user interfaces to programs with different purposes would significantly chill the development of technology. First, the potential market in this situation is too speculative to warrant protection. A programmer could probably convert the user interface for only a few programs, if any, before the program was

---

[131] See note 117 and accompanying text.

[132] One commentator has compared the activity of converting a program for use on different hardware to "using someone else's idea for an internal combustion engine inside a four-wheel vehicle to design a motorcycle." Karjala, 28 Jurimetrics J at 57 (cited in note 102).

[133] Perhaps courts could remedy these types of fact situations by protecting the programs under other doctrines, such as unfair competition under the Lanham Act, 15 USC § 1125(a) (1982), or trade dress theory. See, for example, Beutel, 5 Computer L 1 (cited in note 96).

outdated.[134] Second, the original author might not have the ability nor the inclination to write other programs with different purposes.

The success of the Apple computer system testifies to the desirability of common user interfaces for programs with different underlying purposes.[135] Standardization of user interfaces in this situation promotes efficiency and reduces the need for undesirable retraining.[136] Standardization also encourages users to employ more products.[137] The user preference monopoly argument becomes further persuasive in this situation because add-in programs cannot be used to achieve sequential improvements.[138] Unless courts allow a programmer to create clones of popular user interfaces, users will be forced to learn a new interface for each program.[139] Since programs with different underlying purposes do not compete for the same consumers, no useful purpose is served by requiring differences in user interfaces.[140] The author's copyright monopoly is not damaged by programs that compete for different consumers. In fact, the original author's program may increase in value if its user interface gains popularity and is used by several programs.

V

A PROPOSED TEST FOR INFRINGEMENT

Courts should adopt an infringement test with clearer language and a stronger conceptual basis that would allow programmers to predict the outcome of future software cases more accurately. Uncertainty in the level of protection for computer programs stifles innovation and discourages programmers from using the ideas and expressions of other authors in beneficial ways that cause no harm to the original authors' monopolies.[141]

---

[134] Although some programs have retained the same name for a number of years, they are usually not the same original program because most are updated every few years to reflect changes in hardware or software technology. Even if a program remains useful for five years, its lifespan is too short to allow the average programmer to both update the original program and adapt the user interface for programs with different purposes.

[135] Karjala, 28 Jurimetrics J at 70 (cited in note 102); see Note, 52 U Pitt L Rev at 717-18 n126 (cited in note 18) (explaining advantages of the Macintosh system).

[136] Karjala, 28 Jurimetrics J at 70 (cited in note 102). Standardization reduces training costs and lost time since users do not have to learn how to use different user interfaces. See Note, 52 U Pitt L Rev at 715 n116 (cited in note 18), citing John Markoff, *For PC's, the Mouse Will Reign*, NY Times D1, col 3 (Oct 12, 1988) (corporate managers prefer standardized user interfaces like the Macintosh because they reduce training time and expense).

[137] See Note, 52 U Pitt L Rev at 715-16 n117 (cited in note 18), citing Peter H. Lewis, *New Software Shows Off Mac's Best Features*, NY Times C9 col 1 (Aug 23, 1988) (discussing surveys showing that users of the Macintosh, which sports a standardized user interface across programs, use twice as many programs as do their counterparts who use programs with non-standard user interfaces).

[138] See note 117 and accompanying text.

[139] Currently, to avoid the inconvenience of learning a different user interface for each program, a user either must buy a computer system that operates under a standard user interface, like the Macintosh or software that creates a standard user interface, like Windows.

[140] Karjala, 28 Jurimetrics J at 70 (cited in note 102), citing Comment, *Protecting the "Look and Feel" of Computer Software*, 1 High Tech L J 411, 425 (1986) (authored by John Pinheiro & Gerard La Croix).

[141] See note 19 and accompanying text.

This note proposes a three-part test for copyright infringement of the user-interface.

First, the trier of fact, aided by testimony from expert witnesses,[142] should examine the source and object code of the computer program for evidence of literal similarity. Since the first part of the test focuses only on the literal aspects of the source and object codes, the trier of fact should rely on the traditional tests for copyright infringement developed for literary works such as books. If the trier of fact finds literal or fragmented literal similarity in the source or object code of the programs, the plaintiff should win without further examination of the two programs. If literal or fragmented literal similarity is not found, the trier of fact should move to the second part of the infringement test.

Under the second part of the test, the trier of fact, again aided by testimony from expert witnesses, should define the scope of copyright protection for the nonliteral aspects of the user interface. To do this, the factfinder should determine the specific audiences that the programs target. If the programs do not compete in the same *actual* market, the factfinder should conclude that the defendant's program does not infringe the plaintiff's copyright. In this manner, only the actual market for the program will be protected, not the potential markets for different hardware or for programs with different underlying purposes.[143] If the programs compete in the same actual market, the trier of fact should then examine the user interface by operating the program to determine which aspects of the user interface are eligible for copyright protection.[144]

---

[142] Expert witnesses should testify about similarities that are caused by external constraints and technological considerations. Additionally, the expert witnesses can aid the trier of fact to identify portions of programs that should not be protected under the doctrines of merger or *scènes à faire* due to conventions of the software programming industry. See text accompanying notes 28-31.

[143] The "commercial competitiveness" approach to separating idea from expression when literal copying is absent reaches a similarly-limiting result by confining protection to programs that are directly competitive with the original in a market that the original developer is likely to enter, and to programs that represent the essential core of the two products, thus creating "direct and pervasive" competition based on the original developer's work. Raymond Nimmer & Patricia Krauthaus, *Copyright and Software Technology Infringement: Defining Third Party Development Rights*, 62 Ind L J 13, 48 (1986); see also Karjala, 28 Jurimetrics J at 80-81 (cited in note 102) (criticizing "commercial competition" approach).

[144] A full discussion of this point is beyond the scope of this note. However, careful application of traditional copyright theories should be adequate to properly limit the scope of copyright protection. The CONTU report contained extensive discussions of tradition doctrines that are applied on a case-by-case basis and rejected proposed alternative limits on protection in favor of such doctrines. Note, 88 Mich L Rev at 890 (cited in note 57). The "successive filtering approach" is an example of a test that utilizes traditional copyright theories to eliminate all non-protectable elements from analysis. See 3 *Nimmer on Copyright* at § 13.03[F] (cited in note 30); see also Note, 65 Notre Dame L Rev at 548-50 (cited in note 32).

Some commentators argue that traditional copyright theories preclude the protection of any aspects of the user interface. See, for example, Note, 39 Emory L J at 1324-25 (cited in note 8) (arguing that the terms of the command language, such as copy or save, are not protected by copyright due to the doctrine of merger). However, commentators who fear that copyright protection of the user interface will result in the appropriation of specific terms to the exclusion of others misconstrue the nature of copyright protection. For example, if the phrase "Euclid alone has looked on Beauty bare" were protected by copyright, others would still be free to use words such as "Euclid" or "Beauty." Similarly, when copyright protects a certain passage of a song, others are free

Finally, the trier of fact should assume the role of a computer-user in the target audience and decide whether the competing program is an unlawful appropriation of the plaintiff's program.[145] At this stage, expert testimony should be limited to explaining the use and functions of the program to enable the trier of fact to assume the role of a user in the intended audience.[146] Thus, as in the intrinsic part of the *Arnstein* audience test, the third part of this test is "merely an alternative way of formulating the issue of substantial similarity."[147]

If the trier of fact correctly applies the first two steps of the test, and thus limits the inquiry of the third step to the aspects of the user interface that are properly subject to copyright protection, the dangers of offering too much or too little copyright protection will be greatly reduced. Although similar audience tests have been rejected by most courts in computer cases, the proposed test limits the scope of the appropriation test through the testimony of experts. The test also limits the examination by the audience to those aspects of the program specifically designed for use by the audience.[148] In this manner, none of the typical problems raised by technical issues are present since expert testimony will only help the trier of fact assume the role of the intended audience.

VI

CONCLUSION

Although most of the cases involving copyright protection of the user interface appear to have been decided correctly on their facts, the language of the decisions has left computer programmers uncertain as to the scope of protection given to user interfaces. The courts should adopt an infringement test that allows software programmers to predict the scope of copyright

---

to use the individual notes. Thus, copyright protection of the arrangement of words, such as that proposed for the user interface, would not result in the appropriation of words themselves. Copyright would protect against only identical, or nearly identical, arrangements within a user interface. If identical or nearly identical arrangements are necessary for another reason, a court can recognize this and not find an appropriation.

[145] The issue of appropriation amounts to a question of "whether the defendant took from plaintiff's works so much of what is pleasing to the . . . lay [users], who comprise the audience for whom such [program is written], that defendant wrongfully appropriated something which belongs to the plaintiff." *Arnstein*, 154 F2d at 473.

[146] Since the user interfaces are designed for the user and the user makes the market choices whether to purchase a program, a user in the program's intended audience is the perfect trier of fact to decide if the competing program infringes the plaintiff's copyright.

[147] 3 *Nimmer on Copyright* at § 13.03[E] (cited in note 30), citing *Ideal Toy Corp. v Fab-Lu, Ltd*, 360 F2d 1021 (2d Cir 1966).

[148] The "ordinary observer" test that is normally used in cases involving books, music, and other traditional copyright works has been criticized when used in cases involving computers due to the technical nature of software. See *Whelan Associates, Inc. v Jaslow Dental Lab., Inc.*, 797 F2d 1222, 1232-33 (3d Cir 1986) (discussing the "ordinary observer" test); Note, *Infringement of the Exclusive Right to Prepare Derivative Works: Reducing Uncertainty*, 73 Minn L Rev 1521, 1529 (1989) (authored by Michael Wurzer), citing Note, *Copyright Infringement of Computer Programs: A Modification of the Substantial Similarity Test*, 68 Minn L Rev 1264, 1285-88 (1984) (authored by Howard Root) (arguing that expert testimony should control determination of infringement in computer software cases).

protection for computer programs more accurately. An unambiguous test for copyright infringement will remove the chilling effect of the current test and further the goals of copyright.

In formulating a new infringement test for user interfaces, courts should recognize the differences in market incentives and the role of standardization for each type of market. Computer programs with substantially similar user interfaces compete in one of three types of markets: the actual market, a potential market for different hardware, or a potential market for programs with different underlying purposes. This note argued that copyright protection of the nonliteral aspects of user interfaces should be limited to those uses involving the same actual market. However, even if protection is not limited to the same actual market, the unique characteristics of each type of market for user interfaces should inform courts' decisions. By adopting either the test proposed in this note, or another test that considers the potential markets for the user interface, courts can communicate the breadth of copyright protection for user interfaces and thus encourage the advancement of science and technology.

Todd D. Daubert