Fordham Intellectual Property, Media and Entertainment Law Journal

Volume 3 *Volume III*
Number 1 *Volume III Book 1*

Article 9

1992

# Copyright Protection and Computer Programs: Identifying Creative Expression in a Computer Program's Nonliteral Elements

Lisa C. Green

# NOTE

# Copyright Protection and Computer Programs: Identifying Creative Expression in a Computer Program's Nonliteral Elements

## INTRODUCTION

Congress granted copyright protection to computer programs when it passed the new Copyright Act of 1976.[1] Furthermore, Congress chose to protect computer programs as a type of literary work.[2] Though at least one commentator has taken the position that the "code in which the programs are written is analogous to the text of other literary works,"[3] courts and other commentators have disagreed on how much copyright protection should be granted to a programmer's work based on this analogy.[4]

[1] Copyright Act of 1976, 17 U.S.C. §§ 101-914, 102(a) (1988 & West Supp. 1992). The 1976 Copyright Act provides protection for "original works of authorship fixed in any tangible medium of expression, now known or later developed, from which they can be perceived, reproduced, or otherwise communicated, either directly or with the aid of a machine or device." 17 U.S.C. § 102(a). Works of authorship include literary works, and Congress categorized computer programs as literary works. *See* 17 U.S.C. § 102(a); *see also infra* notes 18, 25.

[2] H.R. REP. NO. 1476, 94th Cong., 2d Sess. 54, *reprinted in* 1976 U.S.C.C.A.N. 5659, 5667 ("The term 'literary works' . . . includes . . . computer programs . . . .").

[3] Mary L. Kevlin, *Computer Cases—Bound for the Supreme Court?*, N.Y. L.J., July 31, 1992, at 3.

[4] Whelan Assocs., Inc. v. Jaslow Dental Labs., Inc., 797 F.2d 1222 (3d Cir. 1986) (granting expansive protection), *cert. denied*, 479 U.S. 1031 (1987); *see also* Anthony L. Clapes, et al., *Silicon Epics and Binary Bards: Determining the Proper Scope of Copyright Protection for Computer Programs*, 34 UCLA L. REV. 1493 (1987) [hereinafter Clapes, *Silicon Epics*] (favoring expansive protection); *but see* Computer Assocs. Int'l, Inc. v. Altai, Inc., No. 91-7893, 1992 U.S. App. LEXIS 14305 (2d Cir. June 22, 1992) (granting limited protection); Peter G. Spivack, Comment, *Does Form Follow Functions? The Idea/Expression Dichotomy in Copyright Protection of Computer Software*, 35 UCLA L. REV. 723 (1988) (favoring limited protection).

When courts have analogized the verbatim copying of a program's code to the verbatim copying of a play or novel's text, they have had little difficulty in applying the new law.[5] Literal copying often means infringement, whether the infringed work is a novel or a computer program.

However in cases involving the copying of the "nonliteral"[6] elements of programs, courts have had more difficulty in applying the analogy.[7] Traditional copyright law protects against copying of the plot or structure of a novel or play. However, when called upon to grant the same protection to the structure or organization of a computer program, courts and commentators have faltered. Some argue that these elements should not be protected at all under copyright law.[8] Even those who believe that nonliteral program elements warrant protection have struggled to find a workable test for determining when infringement of these elements has occurred.[9] Most of the difficulty has been in finding a means of distinguishing when a program's nonliteral elements are the author's expression and when they are mere ideas. Making this distinction is of

[5] *See* Apple Computer, Inc. v. Formula Int'l, 725 F.2d 521 (9th Cir. 1984); Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240 (3d Cir. 1983), *cert. denied*, 464 U.S. 1033 (1984).

[6] The cases involving infringement of the nonliteral elements of computer programs can be divided into two main categories: (1) the "touch and feel" cases, which involve copying the way a program appears to the user; and (2) the "structure, sequence and organization" (SSO) cases which involve copying the way a program is designed and organized. The SSO cases involve copying that is opaque to the user.

[7] *Compare* Johnson Controls, Inc. v. Phoenix Controls Sys., Inc., 886 F.2d 1173, 1175 (9th Cir. 1989) (structure of program was held to be copyrightable expression) *and* Lotus Dev. Corp. v. Paperback Software Int'l, 740 F. Supp. 37 (D. Mass. 1990) (flowchart, if sufficiently detailed, could be afforded copyright protection) *with* Plains Cotton Coop. Ass'n v. Goodpasture Computer Serv., Inc., 807 F.2d 1256 (5th Cir.) (holding that because program's structure was dictated by external market forces it was an unprotectable idea under copyright law), *cert. denied*, 484 U.S. 821 (1987).

[8] *See* Randall M. Whitmeyer, Comment, *A Plea for Due Processes: Defining the Proper Scope of Patent Protection for Computer Software*, 85 NW. U. L. REV. 1103, 1123-25 (1991) *cited with approval in* Computer Assocs. Int'l, Inc. v. Altai, Inc., No. 91-7893, 1992 U.S. App. LEXIS 14305, at *57 (2d Cir. June 22, 1992).

[9] SAS Inst. Inc. v. S & H Computer Sys., Inc., 605 F. Supp. 816 (M.D. Tenn. 1985); Johnson Controls, Inc. v. Phoenix Control Systems, Inc., 886 F.2d 1173 (9th Cir. 1989).

paramount importance since the Copyright Act only protects an author's expression and not the idea that is being expressed.[10]

In 1986, the United States Court of Appeals for the Third Circuit addressed this issue in *Whelan Associates, Inc. v. Jaslow Dental Laboratory, Inc.*[11] In *Whelan*, the court enunciated a test for separating idea from expression that extended protection to a program's nonliteral elements. Since that time the *Whelan* test has been widely applied and rigorously debated.[12] However, in 1992, the United States Court of Appeals for the Second Circuit, in *Computer Associates International, Inc. v. Altai, Inc.*[13] denounced the *Whelan* test as overly broad and, relying heavily on expert opinion, formulated a new, more restrictive test for distinguishing idea from expression.

This note contends that both the *Whelan* test and the *Altai* test fail to accord copyright protection for software to the extent contemplated by the Copyright Act of 1976 and traditional copyright doctrine. This is true despite the fact that *Whelan* and *Altai* reached opposite conclusions as to the proper scope of copyright protection for computer programs. Sometimes a programmer employs pure creativity to effect a design solution. Other times, the programmer merely complies with rigid system dictates or business demands. *Whelan* viewed the programmer's work solely as the former; *Altai* severely overemphasized the latter. Both failed by adhering to a one-sided view of the process of computer programming.

Computer programs are protected as literary works under the Copyright Act of 1976 and Part I of this note reviews some of the traditional tests used by courts to distinguish between idea and expression when dealing with literary works. Part II examines the *Whelan* decision and discusses how it granted overly broad protection to a programmer's works. Part III discusses the recent

[10] *See infra* notes 51-52 and accompanying text.

[11] 797 F.2d 1222 (3d Cir. 1986).

[12] *See* discussion *infra* Part II. Debate over the *Whelan* test ranged from questioning whether computer programs should be protected by copyright at all to the overbreadth of the test itself.

[13] No. 91-7893, 1992 U.S. App. LEXIS 14305 (2d Cir. June 22, 1992).

*Altai* decision and analyzes how it unduly restricted copyright protection for a programmer's work. Part IV argues that neither the *Whelan* nor the *Altai* test is correct, and that absent Congressional clarification, any court that attempts its own solution is unlikely to have success fashioning a workable test. This note concludes that if computer programs are to be ensured the proper amount of copyright protection, Congress needs to clarify the copyright law as it pertains to computer programs. Only then will courts be able effectively to apply the vast body of copyright law and doctrine to the newest form of literary work.

## I. THE HISTORY AND ROLE OF THE IDEA/EXPRESSION DICHOTOMY

In order to understand more fully the present debate[14] over how to determine infringement of copyright for the nonliteral elements of computer programs, it is useful to begin with two brief inquiries: First, an examination of the history and purpose of the statutory protection of computer programs as literary works; and second, a delineation of the common law tests for copyright infringement of literary works that have been applied in computer program cases.[15]

### A. *Congress' Decision to Protect Computer Programs as Literary Works*

Copyright originates in the Constitution which provides that Congress may "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."[16] Traditionally,

[14] *See supra* notes 11-13 and accompanying text.

[15] These judicially created tests were being used at the time Congress decided to include computer programs as literary works in the copyright statute. *See infra* notes 31-55 and accompanying text.

[16] U.S. CONST. art. I, § 8, cl. 8. From the time of the Founding there has been a duality of purpose in copyright law: "to promote the public disclosure of and dissemination of works of 'authorship'; [yet] on the other hand, . . . to confer on the creators the power to restrict or deny distribution of their works." Peter Jaszi, *Towards a Theory of Copyright: Metamorphoses of Authorship*, 1991 DUKE L.J. 455, 463 (1992). This commentator has also suggested that this tension is the basis for most copyright

Congress has accorded authors and inventors protection for their works depending on the nature of the work: Copyright law protects literary works, while patent law protects novel utilitarian works.[17] Since the Constitution was written, Congress has passed several copyright statutes, each statute enlarging the scope of works protected by copyright.[18] However, when Congress began extensively revising the federal copyright law in the 1970s, it faced a difficult decision: whether computer programs should be protected by copyright, or whether some other means of intellectual property protection was more appropriate.[19] To aid in deciding these questions, Congress established the National Commission on

doctrines such as merger and scenes à faire. *Id.* at 464.

[17] Peter S. Menell, *An Analysis of the Scope of Copyright Protection for Application Programs*, 41 STAN. L. REV. 1045, 1046 (1989). Also, the patent and copyright laws vary as to the type and scope of protection they give.

> When Congress exercised its broad Constitutional Authority to enact patent and copyright laws in order to promote the progress of science and the useful arts, Congress took substantially different approaches in balancing competing interests. Under copyright law, authors are afforded much weaker protection than inventors are afforded under patent law: copyright gives no protection for ideas, no matter how original, and no protection against independent creation. In light of this weak protection, the subject matter capable of being protected is much broader in scope than under patent law (original authorship rather than novel and non-obvious inventions) and the terms of protection is much longer. These trade-offs reflect a reasonable and carefully crafted balancing by Congress of the various diverse interests it was seeking to further.

Allen R. Grogan, *Bonito Boats and Whelan: A Simple Contrast Between Patent and Copyright Law*, COMPUTER LAW., July 1989, at 33, 34.

[18] Clapes, *Silicon Epics*, *supra* note 4, at 1493 n.4:

> The first congressional copyright statute, passed in 1790, governed only maps, charts, and books. In 1802, the Act was amended in order to grant protection to any person "who shall invent and design, engrave, etch or work . . . any historical or other print or prints . . . ." Protection was extended to musical compositions when the copyright laws were revised in 1831. In 1865, at the time when Matthew Brady's pictures of the Civil War were attaining fame, photographs and photographic negatives were expressly added to the list of protected works. Again in 1870, the list was augmented to cover paintings, drawings, chromos, statuettes, statuary, and models or designs of fine art.

(citing Goldstein v. California, 412 U.S. 546, 555-56 (1973) (citations omitted)). In 1909, the Congress greatly revised the statute and extended copyright protection to motion pictures. The 1909 Act was amended in 1971 to include sound recordings. *Id.*

[19] *See* Menell, *supra* note 17, at 1046-47.

New Technological Uses of Copyrighted Works (CONTU).[20]

"The purpose of [CONTU] was . . . to study the reproduction and use of copyrighted works in conjunction with computers, and to make recommendations as to any necessary changes to be made in the copyright law."[21] While CONTU was studying the issue, Congress passed the Copyright Act of 1976,[22] which explicitly mandated that computer programs were to be treated as literary works for copyright purposes.[23] All works covered under the new statute—including computer programs—were protected[24] insofar as they embodied an author's particular expression; the ideas behind that expression, however, were denied protection.[25] In copyright

[20] Act of Dec. 31, 1974, Pub. L. No. 93-573, § 201, 88 Stat. 1873. The CONTU Report is usually considered the legislative history of the Copyright Act of 1976 with respect to computer programs. *See, e.g.*, Vault Corp. v. Quaid Software Ltd., 847 F.2d 255, 261 (5th Cir. 1988) (courts "rely on the CONTU Report as an expression of legislative intent"); Micro-Sparc, Inc. v. Amtype Corp., 592 F. Supp, 33, 35 n.7 (D. Mass. 1984) (the CONTU Report is considered an expression of legislative intent); Midway Mfg. Co. v. Strohon, 564 F. Supp. 741, 750 n.6 (N.D. Ill. 1983) (same); Steven Englund, Note, *Idea, Process, or Protected Expression?: Determining the Scope of Copyright Protection of the Structure of Computer Programs*, 88 MICH. L. REV. 866, 886 (1990) ("[S]everal courts have considered the CONTU Report a reflection of legislative intent . . . with respect to the 1976 [Copyright] Act."); Cary S. Kappel, Note, *Copyright Protection of SSO: Replete With Internal Deficiencies and Practical Dangers*, 59 FORDHAM L. REV. 699, 701 (1990) ("The report is generally considered the legislative history of the 1976 Copyright Act."). *But see* Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1242 (3d Cir. 1986) (holding that the CONTU Report is only the legislative history for those provisions which Congress amended in response to CONTU's suggestions).

[21] Melville Nimmer, Declaration Regarding the National Commission of New Technological Uses of Copyrighted Works (CONTU) Final Report ¶ 3 (Nov. 15, 1984), *reprinted in* Clapes, *Silicon Epics*, *supra* note 4, at 1585 [hereinafter Melville B. Nimmer Declaration].

[22] Copyright Act of 1976, 17 U.S.C. §§ 101-914 (1988 & West Supp. 1992).

[23] *See supra* note 2 and accompanying text.

[24] The owner of a valid copyright has the exclusive right to copy, prepare derivative works, distribute, perform or display the copyrighted work, subject to certain limitations. 17 U.S.C. § 106.

[25] *See supra* note 1. Further, "[i]n no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work." 17 U.S.C. § 102(b).

cases involving literary works, this distinction had often played a critical role, and in the new law Congress explicitly stated that it should also apply to computer programs: "[C]omputer programs [should be protected under the new statute] to the extent that they incorporate authorship in the programmer's expression of original ideas, as distinguished from the ideas themselves."[26] In its final report, CONTU concurred with Congress' decision, stating that "computer programs, to the extent that they embody an author's original creation, are [the] proper subject matter of copyright."[27] CONTU also did not criticize Congress' decision to classify computer programs as literary works.[28] Congress' decisions, both to include computer programs under the existing copyright category of literary works and to apply the idea/expression distinction,[29] indicated its broader intention to integrate computer programs into existing copyright law and theory.[30]

### *B. The Role of the Idea/Expression Distinction in Literary Copyright Infringement Cases*

When applying the copyright law in computer software copyright infringement cases, courts have looked to traditional literary copyright cases for guidance as to how to apply the idea/expression

[26] H.R. REP. NO. 1476, 94th Cong., 2d Sess. 54, *reprinted in* 1976 U.S.C.C.A.N. 5659, 5667.

[27] Final Report of the National Commission on New Technological Uses of Copyrighted Works (1979), *reprinted in* 3 COMPUTER L.J. 53 (1981) [hereinafter CONTU REPORT]. CONTU recommended three changes to the new 1976 Copyright Statute. *Id.* One recommendation was that "[t]he new copyright law should be amended: (1) to make it explicit that computer programs, to the extent that they embody an author's original intention, are proper subject matter for copyright." *Id.* The other two recommendations are beyond the scope of this discussion.

[28] *Id.*

[29] Lotus Dev. Corp. v. Paperback Software Int'l, 740 F. Supp. 37, 54 (D. Mass. 1990). Some commentators have suggested that this is not what Congress intended and that the mention of the idea/expression distinction was nothing more than a restatement of 17 U.S.C. § 102(b). *See supra* note 25 for text of § 102(b).

[30] *Lotus Dev. Corp.*, 740 F. Supp. at 54 (Congress' "conclusion is also consistent with the object and policies of copyright—to encourage the creation and dissemination of new ideas by protecting, for limited times, the specific way that an author has *expressed* those ideas.") (emphasis in original).

distinction. "In order to prove infringement a plaintiff must show ownership of a valid copyright and copying by the defendant."[31] The infringement may consist of literal duplication or the appropriation of nonliteral elements.[32] Further, evidence of copying may be direct or indirect.[33] Though courts have developed several different approaches for using indirect evidence in proving nonliteral copying, they usually follow one of the two approaches outlined below.[34] Though each of these approaches is often referred to as the "substantial similarity test,"[35] the two are in reality very different tests. Each test asks the same question: Is there substantial similarity? The tests differ, however, as to when in the sequence of inquiry the court asks this question. This difference in timing the question usually determines how a court will approach the larger issue of illicit copying.

One clearly articulated test for determining if two works are substantially similar is the two-pronged test offered in *Arnstein v. Porter*.[36] Under this test, there is substantial similarity on a finding: (a) that defendant copied from plaintiff's copyrighted work and (b) that the copying, if proved, went so far as to constitute improper

[31] Novelty Textile Mills v. Joan Fabrics Corp., 558 F.2d 1090, 1092 (2d Cir. 1977).

[32] Nichols v. Universal Pictures Corp., 45 F.2d 119, 121-22 (1930) (Judge Learned Hand stated that "[i]t is . . . essential to any protection of literary property . . . that the right cannot be limited literally to the text, else a plagiarist would escape by immaterial variations.").

[33] Copying is rarely proved by direct evidence of the "physical act of copying." 3 MELVILLE B. NIMMER & DAVID NIMMER, NIMMER ON COPYRIGHT, § 13.01[B], 13.10 (1991) [hereinafter NIMMER]. Copying is more often proved by indirect evidence, such as by proving access and substantial similarity. *Id.* § 13.10.01 (citing cases).

[34] Howard Root, Note, *Copyright Infringement of Computer Programs: A Modification of the Substantial Similarity Test*, 68 MINN. L. REV. 1264, 1280 (1984) (tracing the use of the substantial similarity test in cases involving literary works).

[35] *Id.* at 1276.

[36] 154 F.2d 464 (2d Cir. 1946), *cert. denied*, 330 U.S. 851 (1947). The late Professor Latman stated that "the clearly articulated copyright approach of [the *Arnstein* case], when properly understood, remains the most instructive guide to proving infringement." Alan Latman, *Probative Similarity As Proof of Copying: Toward Dispelling Some Myths In Copyright Infringement*, 90 COLUM. L. REV. 1185, 1191 (1990).

appropriation.[37]

> If there is evidence of access[38] and similarities exist, then the trier of fact [may reasonably infer copying] . . . . If the evidence of access is absent, the similarities must be so striking as to preclude the possibility that plaintiff and defendant independently arrived at the same result . . . . If copying is established, then only does there arise the second issue, that of illicit copying (unlawful appropriation).[39]

Professor Alan Latman further explicated the parameters of illicit copying as set forth in *Arnstein*:

> [A]ssuming that adequate proof is made of copying, that is not enough; for there can be "permissible copying," copying which is not illicit. Whether . . . [the] defendant unlawfully appropriated [is the second] issue of fact . . . . The question, therefore, is whether defendant . . . wrongfully appropriated something which belongs to the plaintiff.[40]

Since only the author's expression "belongs" to him, it is at this second stage of the inquiry that the court must differentiate between protected expression and unprotected idea.[41] Because copying has been proved, the court now asks if what the defendant copied was protectable expression. Asking the questions in this order properly delays the complicated task of distinguishing idea from expression. This is important because proof that there was "copying of uncopyrighted components of a work clearly has some probative value in establishing the copying of other, copyrighted, components of the same work."[42] Therefore, when a court

[37] *Arnstein*, 154 F.2d at 467.

[38] Access has been defined as "the opportunity to view the copyrighted work." *See* NIMMER, *supra* note 33, § 13.02[A]. Others have held access to be the actual viewing and knowledge of the copyrighted work. *See* Bradbury v. Columbia Broadcasting Sys., Inc. 287 F.2d 478, 479 (9th Cir. 1961), *cited with approval in* Root, *supra* note 34, at 1278 n.88.

[39] *Arnstein*, 154 F.2d at 466. Expert testimony may be considered by the trier of fact during the first prong but not during the second. *Id.*

[40] Latman, *supra* note 36, at 1192-93.

[41] *Id.* at 1195.

[42] *Id.*

distinguishes unprotectable idea from protectable expression it does so knowing that the defendant has copied. This keeps the focus of the court's inquiry on the defendant's copying, not the protectability of the plaintiff's works.

Since *Arnstein*, other courts have approached the issue in a different way. These courts have collapsed the two prongs of the *Arnstein* inquiry into a test of access and substantial similarity.[43] This test will be referred to as the compressed version of the *Arnstein* test.[44] If access is not shown, but the works are strikingly similar, an inference of copying may be drawn without proof of access.[45] If, however, access is found, then the court asks if the works are substantially similar and if "the [substantial] similarities relate to copyrightable material."[46] By this means, the court ends up evaluating whether works are similar (because they share unprotected ideas or protected expression) without establishing whether there was copying in general. Neither the *Arnstein* two-pronged test or the compressed version, however, offers a court guidance on how to make the distinction between idea and expression.

In *Nichols v. Universal Pictures Corp.*,[47] Judge Learned Hand articulated his abstractions test as a means of separating and distinguishing idea from expression when faced with circumstantial evidence of nonliteral copying.[48] The *Nichols* court began by assuming, *arguendo*, that the defendant had—to some degree—used the plaintiff's copyrighted work in preparing his own.[49] The court

[43] Novelty Textile Mills v. Joan Fabrics Corp., 558 F.2d 1090, 1092 (2d Cir. 1977); *Altai*, 1992 U.S. App. LEXIS 14305, at *19; *Altai*, 775 F. Supp. 544, 557 (E.D.N.Y. 1991) (both citing Walker v. Time Life Films, Inc., 784 F.2d 44, 48 (2d Cir.), *cert. denied*, 476 U.S. 1159 (1986).

[44] Latman, *supra* note 36, at 1188.

[45] 3 NIMMER, *supra* note 33, §13.03[B].

[46] *Walker*, 784 F.2d at 48.

[47] 45 F.2d 119 (2d Cir. 1930).

[48] *Id.* The case involved the motion picture play "The Cohens and The Kellys" which was held not to infringe the copyright in the play characters in "Abie's Irish Rose." Both works were comedies involving secret inter-faith marriages and the complications that arise as the disapproving families gradually discover the truth. The infringement claim focused on similarity of the plot and characters.

[49] *Id.* at 120.

then went on to discuss at length the similarity of the two works and the extent to which that similarity was comprised of protected expression.

> Upon any work, . . . a great number of patterns of increasing generality will fit equally well, as more and more of the incident is left out. The last may perhaps be no more than the most general statement of what the [work] is about, . . . but there is a point in this series of abstractions where they are no longer protected, since otherwise the [author] could prevent the use of his "ideas," to which, apart from their expression, his property never extended. Nobody has ever been able to fix that boundary, and nobody ever can.[50]

Thus in *Nichols*, as in *Arnstein*, the final determination of whether there was illicit nonliteral appropriation depended on how the court characterized what was taken—whether it was idea or expression, whether it was material that belonged to the public or to the plaintiff.

This final distinction between idea and expression defines the boundary of copyright protection,[51] since material deemed an idea will not be accorded copyright protection. To draw this boundary, a court must resolve the conflicting interests present in copyright law—viz. providing enough protection for authors' work to maintain creative incentive while encouraging the dissemination of knowledge and science to the public.[52] Thus, every time a court must make the idea/expression distinction, it must also resolve these

[50] *Id.* at 121 (citations omitted).

[51] *See* Clapes, *Silicon Epics*, *supra* note 4, at 1550.

[52] *See* Jaszi, *supra* note 16, at 464-65. *See also* Lotus Dev. Corp. v. Paperback Software Int'l, 740 F. Supp. 37, 54 (D. Mass. 1990) ("[T]he objects and policies of copyright [are] to encourage the creation and dissemination of new *ideas* by protecting, for limited times, the specific way that an author has *expressed* those ideas.") (emphasis in original); Herbert Rosenthal Jewelry Corp. v. Kalpakian, 446 F.2d 738, 742 (9th Cir. 1971) (The principal consideration when drawing the line between idea and expression "is the preservation of the balance between competition and protection reflected in the patent and copyright laws."); Sony Corp. of Am. v. Universal Studios, Inc., 464 U.S. 417, 432 (1984) (granting a limited monopoly will provide sufficient incentive for authors to create new works).

tensions anew.[53]

It is the complexity inherent in the idea/expression distinction—and not the unsuitability of protecting computer programs as literary works under the copyright law—that has perplexed courts adjudicating computer program copyright cases involving non-verbatim copying.[54] The difficulty courts have experienced in resolving this distinction has been further compounded because "computer programs as a form of expression are not well understood by makers of law and policy."[55] Courts thus find themselves balancing elements they do not fully understand. For a test of copyright infringement to be meaningful in computer program cases, both these areas must be addressed.

## II. A BROAD SCOPE OF COPYRIGHT PROTECTION UNDER *WHELAN ASSOCIATES, INC. V. JASLOW DENTAL LABORATORY, INC.*

The copyright infringement action brought in *Whelan Associates, Inc. v. Jaslow Dental Laboratory, Inc.*[56] involved two computer programs for managing the business operations of a dental laboratory.[57] The defendant Jaslow owned and operated a dental laboratory and hired the plaintiff Whelan, a computer programmer, to write a program for managing the business operations of his dental lab.[58] The two agreed that Whelan would

[53] *See* Jaszi, *supra* note 16, at 464-65; *see also Nichols*, 45 F.2d at 121.

[54] Despite the consensus between Congress and CONTU to include computer programs under the umbrella of copyright law, their decision has been criticized for being at odds with the fundamental principle that copyright cannot protect "any idea, procedure, process, system, method of operation, concept, principle or discovery." *See* Menell, *supra* note 17, at 1046-47; Computer Assocs. Int'l, Inc. v. Altai, Inc., 775 F. Supp. 544, 560 (E.D.N.Y. 1991) ("Since the behavioral aspect of a computer program falls within the statutory terms 'process,' 'system,' and 'method of operation,' it may be excluded by statute from copyright protection. Indeed, it has been suggested that computer software is better protected by patent law than by copyright law.") (citing Whitmeyer, *supra* note 8, at 1123-25).

[55] *See* Clapes, *Silicon Epics*, *supra* note 4, at 1499.

[56] 797 F.2d 1222 (3d Cir. 1986).

[57] *Id.* at 1225-26.

[58] *Id.* at 1225. *See* Thomas M. Gage, Note, *Whelan Associates v. Jaslow Dental Laboratories: Copyright Protection for Computer Software Structure— What's the*

own the copyright in the program while Jaslow would use the program.[59] Further, Jaslow was to act as Whelan's sales representative in marketing the program, called DENTALAB, to other dental laboratories.[60] Two years after Whelan completed the DENTALAB program for Jaslow, Jaslow began selling a similar program, called DENTACOM, which was written in a different language.[61] Whelan brought suit for copyright infringement.[62]

Following a trial without a jury, the United States District Court for the Eastern District of Pennsylvania held that sales of the DENTCOM system violated Whelan's copyright.[63] Jaslow appealed to the United States Court of Appeals for the Third Circuit. The sole issue before the Third Circuit was whether the district court was correct in holding that a computer program's structure was protected by copyright law.[64] The Third Circuit affirmed the district court's decision: A computer program's structure was

*Purpose?*, 1987 WIS. L. REV. 859, 875-78 (more elaborate discussion of the facts in *Whelan*). Whelan owned half the company that Jaslow had hired to write the program. *Id.*

[59] *Whelan*, 797 F.2d at 1225. Initially the company that Whelan worked for—and was half owner of—held the copyright to DENTALAB. After completing the DENTALAB program for Jaslow, Whelan formed her own company, Whelan Associates, which became the new holder of the DENTALAB copyright. *Id.*

[60] *Id.*

[61] *Id.* at 1226-27. The business relationship worked successfully for two years. However, Whelan had written DENTALAB in the computer language EDL, and many of the smaller dental labs did not own computers that could process EDL. Jaslow realized that there was a market for a similar program written in a more popular computer language. Jaslow produced such a program using BASIC instead of EDL. That new program, called DENTCOM, became the alleged copyright infringer. *Id.* at 1226.

[62] *Id.* at 1227.

[63] *Whelan*, 609 F. Supp. 1307, 1321-22 (E.D. Pa. 1985). The district court found copyright infringement because DENTCOM was not independently created. Jaslow had access to DENTALAB, and there was substantial similarity between DENTCOM and DENTALAB. *Id.* The district court *did* find that the two programs were substantially similar in their structure and overall organization. *Id.* at 1321-22.

[64] On appeal Jaslow contended that the district court had erred in finding copyright infringement because of the substantial similarity of the two programs' structure and overall organization. Jaslow asserted that only the literal elements of the program—in this case the code—were protectable expression. *Whelan*, 797 F.2d at 1233.

protected by copyright law. Further, there was sufficient evidence to uphold the district court's finding of substantial similarity between DENT-COM and DENTALAB.[65]

The Third Circuit began by noting that the Copyright Act of 1976 "extends copyright protection to literary works, and that computer programs are classified as literary works for the purposes of copyright."[66] Further, "the copyrights of other literary works can be infringed even when there is no substantial similarity between the works' literal elements."[67] The court reasoned that "[b]y analogy to other literary works, it would . . . appear that the copyrights of computer programs can be infringed even absent copying of the literal elements of the program."[68]

The court then considered the defendant's contention that "what is true of other literary works is not true of computer programs."[69] The court noted that "[i]t is axiomatic that copyright does not protect ideas, but only expressions of ideas,"[70] and that "[i]t is frequently difficult to distinguish the idea from the expression thereof."[71] The court disagreed, however, with Judge Hand's assertion that such a distinction will "inevitably be *ad hoc*."[72] Instead, the court undertook to "formulate a rule applicable in this case."[73]

In formulating its own test, the court stated that

> because the line between idea and expression is elusive, we must pay particular attention to the pragmatic considerations that underlie the distinction and copyright law generally . . . . We must remember that the purpose of copy-

[65] *Id.* at 1248.

[66] *Id.* at 1234 (citing H.R. REP. No. 1476, 94th Cong., 2d Sess. 54, *reprinted in* 1976 U.S.C.C.A.N. 5659, 5667).

[67] *Whelan*, 797 F.2d at 1234 (citations omitted).

[68] *Id.* (citing 3 NIMMER, *supra* note 33, § 13.03[A]).

[69] *Id.* at 1234.

[70] *Id.* (citing Baker v. Selden, 101 U.S. 99 (1879); 17 U.S.C. § 102(b)).

[71] *Id.* at 1235.

[72] *Id.* (citing Peter Pan Fabrics, Inc. v. Martin Weiner Corp., 274 F.2d 487, 489 (2d Cir. 1960)).

[73] *Id.*

> right law is to create the most efficient and productive balance between protection (incentive) and dissemination of information, to promote learning, culture and development.[74]

The court used *Baker v. Selden*[75] as the starting point for distinguishing idea from expression.[76] It reasoned that "[j]ust as *Baker v. Selden* focused on the end sought to be achieved by Selden's book, the line between idea and expression may be drawn with reference to the end sought to be achieved by the work in question."[77] The court concluded that

> the purpose or 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 . . . . Where there are various means of achieving the desired purpose, then the particular means chosen is not necessary to the purpose; hence, there is expression, not idea.[78]

Thus, under the *Whelan* test, how the court distinguishes idea from expression depends exactly on how the court defines the program's end purpose or objective.[79]

Applying its new test to the facts of the case, the court found that "the purpose of the utilitarian DENTALAB program was to aid

[74] *Id.* (citing U.S. CONST. art. I, § 8, cl. 8).

[75] 101 U.S. 99 (1879). The case involved a book that described a method of accounting and forms that were to be used with that method. The Court held that where the method of accounting described in the book

> cannot be used without employing the methods and diagrams used to illustrate the book, or such as are similar to them, such methods and diagrams are to be considered as necessary incidents to the art, and given [therewith] to the public.
>
> Applying this test, the Court held that the blank forms were necessary incidents to Selden's method of accounting, and hence were not entitled to any copyright protection.

*Whelan*, 797 F.2d at 1236 (quoting Baker v. Selden, 101 U.S. at 103) [citations omitted].

[76] *Whelan*, 797 F.2d at 1236.

[77] *Id.*

[78] *Id.* (citation omitted).

[79] Gage, *supra* note 58, at 881.

in the business operations of a dental laboratory."[80] Having defined the purpose, the court stated that a programmer could write other programs, using a different structure and organization, that would aid in the business operations of a dental lab. The court reasoned that, because any number of structures could have been used, no one structure was a necessary part of the program's purpose and idea. Therefore, the structure Whelan had chosen to employ was part of her expression and was thus protectable under copyright law.[81]

The *Whelan* court was aware that the copyright protection it was granting to computer programs was broad and that such broad protection had been criticized in the past.[82] Criticism of broad protection for computer software such as that allowed by the *Whelan* test is based in large part on the belief that "progress in the [computer industry] is achieved by means of 'stepping stones,' a process that 'requires plagiarizing in some manner the underlying copyrighted work.'"[83] As a result, "giving computer programs too much copyright protection will retard progress in the field."[84]

The *Whelan* court responded that it did not believe that computer technology was significantly different from

> progress in other areas of sciences or the arts. In balancing protection and dissemination, the copyright law has always recognized and tried to accommodate the fact that all intellectual pioneers build on the work of their predecessors. Thus, copyright principles derived from other areas are

[80] *Whelan*, 797 F.2d at 1238.

[81] The court noted when this test should not be applied:

> Different computer systems may functionally serve similar purposes without being copies of each other. There is evidence in the record that there are other software programs for the business management of dental laboratories in competition with plaintiff's program. There is no contention that any of them infringe although they may incorporate many of the same ideas and functions.

*Whelan*, 797 F.2d at 1238-39.

[82] *Id.* at 1237-38.

[83] *Id.* at 1238 (citing Root, *supra* note 34, at 1292).

[84] *Id.*

> applicable in the field of computer programs.[85]

Therefore, if application of those principles results in broad protection, then that is consistent with copyright law as applied to literary works.

The *Whelan* court's approach has been applied to some degree in the majority of computer software cases since the decision was announced in 1986.[86] Cases rejecting the *Whelan* approach have been in the minority.[87] By contrast, *Whelan* has been widely criticized in the academic community. Criticism has focused on two aspects of the decision: (1) The overly broad definition of expression, and (2) the negative impact that broad copyright protection could have on the computer industry.[88]

[85] *Id.* (citation omitted).

[86] *See generally* Lotus Dev. Corp. v. Paperback Software Int'l, 740 F. Supp. 37, 55 (D. Mass. 1990) (citing *Whelan* for the proposition that "[c]opyright protection of computer programs may extend beyond the program's literal code to their structure, sequence and organization"); Bull HN Info. Sys., Inc. v. American Express Bank, Ltd., No. 88 Civ. 2103, 1990 WL 48098, at *3 (S.D.N.Y. Apr. 6, 1990); Healthcare Affiliated Servs., Inc. v. Lippany, 701 F. Supp. 1142, 1151-52 (W.D. Pa. 1988) (analyzing whether there was substantial similarity in the structure, sequence and organization of a program within the meaning of *Whelan*); Broderbund Software Inc. v. Unison World, Inc., 648 F. Supp. 1127, 1133 (N.D. Cal. 1986) (following the reasoning of *Whelan* to hold that the overall structure, sequence and arrangement of computer software is within the scope of copyright protection); Digital Communications Assocs. Inc. v. Softklone Distrib. Corp., 659 F. Supp. 449, 455 (N.D. Ga. 1987) (citing *Whelan* for the proposition that a program's structure, sequence and organization are copyrightable); Dynamic Solutions, Inc. v. Planning & Control, Inc., 1987 Copyright L. Dec.(CCH) § 26,062, 20,912 (S.D.N.Y. 1987) ("I agree with the Third Circuit's careful analysis in *Whelan Associates* that to the extent structural similarities in a computer program are 'not necessary to [the] purpose or function' of the program, they constitute a protectable expression of an idea."); Richard Raysman & Peter Brown, *Computer Protection for Software Redefined*, N.Y. L.J., July 14, 1992, at 3, 4.

[87] Plains Cotton Co-op v. Goodpasture Computer Serv., Inc., 807 F.2d 1256, 1262 (5th Cir.), *cert. denied*, 484 U.S. 821 (1987); *cf.* Synercom Technology, Inc. v. University Computing Co., 462 F. Supp. 1003, 1014 (N.D. Tex. 1987). *See generally* Raysman & Brown, *supra*, at 4.

[88] Whitmeyer, *supra* note 8, at 1128; Menell, *supra* note 17, at 1074, 1082; Englund, *supra* note 20, at 881; Gage, *supra* note 58, at 882; Marc T. Kretschmer, Note, *Copyright Protection For Software Architecture: Just Say No!*, 3 COLUM. BUS. L. REV. 823, 837-39 (1988); Spivack, *supra* note 4, at 747-55; 3 NIMMER, *supra* note 33, §

Critics[89] noted a crucial flaw in the *Whelan* test: It identified the idea of a program solely in terms of the end function of a program. This presented two problems. First, most programs performed more than one function; thus—under the *Whelan* definition of idea—they appeared to contain more than one idea.[90] However, under *Whelan,* only one function—the end function—was *unpro-tected* idea. This implied that the ideas of all of a program's sub-functions became protected expression.[91] Such an outcome would be in direct conflict with the copyright law, which held that ideas were unprotectable.

Second, the breadth of a program's idea was dictated by how broadly or narrowly the court defined the end function of the program.[92] If the end purpose of a program was broadly defined, then the idea of the program was broadly defined;[93] this left little of the program as protectable expression. Conversely, if the end purpose of a program was narrowly defined, the idea of the program was narrow. This in turn left most of the program as protectable expression. A court could thus provide any level of protection it chose based on the breadth of the definition it selected for a program's end purpose.[94] In the end, the inaccuracy of the *Whelan* test left it both in direct conflict with copyright law and open to manipulation by the courts.

Commentators also feared the long-range effects of the *Whelan* test on the computer software industry. They believed that a broad definition of expression would "enable first comers to 'lock up' basic programming techniques as implemented in programs to perform particular tasks."[95] They argued that since the computer industry grows by building on the work of others, overly broad

13.03[F] at 13-78.33.

[89] Kretschmer, *supra* note 88, at 837-38; Spivack, *supra* note 4, at 747-48; 3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.33.

[90] *See* Kretschmer, *supra* note 88, at 836; Spivack, *supra* note 4, at 747-48.

[91] *See* Kretschmer, *supra* note 88, at 838-39.

[92] *See* Englund, *supra* note 20, at 881; *see also* Gage, *supra* note 58, at 881.

[93] *See* Englund, *supra* note 20, at 881; Spivack, *supra* note 4, at 750; Gage, *supra* note 58, at 881.

[94] *See* Englund, *supra* note 20, at 881; Gage, *supra* note 58, at 882.

[95] Menell, *supra* note 17, at 1082.

protection of the basic building blocks of the field could greatly hamper future development.[96] The outcome of this would be that the "development of new programs may decrease and new program costs might increase relative to what would occur in the absence of copyright protection for program structure."[97]

Commentators posed alternatives to the *Whelan* test. Some simply argued that patent law, not copyright law, was the correct form of protection for computer programs.[98] Others maintained that copyright was the correct form of protection[99] but that a new test was needed to identify protectable expression.[100] One proposed alternative was the "successive filtering method" set forth by David Nimmer.[101]

Nimmer's test called for a successive application of various tests to filter out the non-protectable material in a computer program. One of the various tests to be applied was the abstractions test.[102] The other tests were based on copyright doctrines such as merger[103] and scenes à faire.[104] Nimmer's "successive filtering

[96] *See* David Nimmer, et al., *A Structured Approach to Analyzing the Substantial Similarity of Computer Software in Copyright Infringement Cases*, 20 ARIZ. ST. L.J. 625, 630 (1988) [hereinafter Nimmer, *A Structured Approach*].

[97] Gage, *supra* note 58, at 881. At the time of its *Whelan* decision, the Third Circuit was aware that some commentators believed that "giving computer programs too much copyright protection will retard progress in the field." However, the court had stated that it did not believe that progress in computer technology was significantly different from "progress in the others areas of sciences or the arts. In balancing protection and dissemination, the copyright law has always recognized and tried to accommodate the fact that all intellectual pioneers build on the work of their predecessors. Thus, copyright principles derived from other areas are applicable in the field of computer programs." *Whelan*, 797 F.2d at 1238 (citation omitted).

[98] *See* Menell, *supra* note 17, at 1083.

[99] *See* Englund, *supra* note 20, at 867; Clapes, *Silicon Epics*, *supra* note 18, at 1546.

[100] *See* Englund, *supra* note 20, at 881 (suggesting that identifying the protectable expression in a computer program involves distinguishing process from expression as well as idea from expression); *see also* 3 NIMMER, *supra* note 33, § 13.03[F], 13-78.26 to .30.

[101] 3 NIMMER, *supra* note 33, § 13.03[F].

[102] *Id.* at 13-78.30-.35; *see supra* notes 47-50 and accompanying text for explanation of the abstractions test.

[103] *Id.* at 13-78.35 to .36; *see infra* note 139 for explanation of the merger

method" took into account the realities of computer programming. It also took into account the need for a test that could be applied by non-specialists and which would yield just results.[105] This "successive filtering method" formed the basis for the test that the Second Circuit applied in *Computer Associates International, Inc. v. Altai, Inc.*[106]

## III. A NARROWER SCOPE OF COPYRIGHT PROTECTION UNDER *COMPUTER ASSOCIATES INTERNATIONAL, INC. V. ALTAI, INC.*

The copyright infringement action brought in *Computer Associates International, Inc. v. Altai, Inc.*[107] involved a job scheduling program for IBM mainframe computers.[108] Since "mainframe computers use different operating systems[109] . . . , normally a program would have to come in different versions for each operating system."[110] However, Computer Associates' job scheduling program, called CA-SCHEDULER, was designed so that a single version could be run on different operating systems. This was done through the use of an internal translator, or adapter,

doctrine.

[104] *Id.* at 13-78.36 to .42; *see infra* note 140 for an explanation of the scenes à faire doctrine.

[105] *See* Nimmer, *A Structured Approach*, *supra* note 96, at 626.

[106] Computer Assocs. Int'l v. Altai, Inc., No. 91-7893, 1992 U.S. App. LEXIS 14305 (2d Cir. June 22, 1992).

[107] *Id.*

[108] *Id.* at *10 ("[The] primary functions [of a job scheduling program] are straightforward: to create a schedule specifying when the computer should run various tasks, and then to control the computer as it executes the schedule.").

[109] Operating systems were correctly described for the *Altai* court as:

> the programs that manage the resources of the computer and allocate those resources to other programs that need them. For example, functions that operating system software might perform include providing blocks of memory to an application program that requires them, or allocating processing time among several application programs running on the computer at the same time.

Brief for Appellant at 5, Computer Assocs. Int'l, Inc. v. Altai, Inc., No. 91-7893, 1992 U.S. App. LEXIS 14305 (2d Cir. June 22, 1992) [hereinafter Brief for Appellant].

[110] Kevlin, *supra* note 3, at 3.

module called ADAPTER.[111]

The defendant Altai also marketed a job scheduling program called ZEKE. Though initially Altai's ZEKE program could run on only one operating system,[112] Altai eventually created a new job scheduling program that could run on multiple operating systems.[113] Altai's new product contained an adapter module which Altai named OSCAR 3.4.[114] The OSCAR 3.4 adapter module was similar to Computer Associates' ADAPTER[115] and, in fact, was created by a former employee of Computer Associates.[116]

Altai marketed its new product for three years before Computer

[111] *Altai*, 1992 U.S. App. LEXIS 14305, at *10-11. More specifically, the job scheduling program was broken down into two distinct components. The first component performed the actual job scheduling. The second component served as an interface or adapter between the first component and the operating system. According to the Appellant:

> In a program constructed in this way, whenever the first, task-specific, component needs to ask the operating system for some resource, it calls the second component instead of calling the operating system directly. Since the first, task-specific component calls the adapter component rather than the operating system, the first component need not be customized to any operating system. The adapter component insures that all the system calls are performed properly for the particular operating system in use . . . . This approach has two very desirable effects. First, to adapt a program to a new version of an existing operating system or an entirely new operating system, the developer generally need only modify the adapter component. Second, revisions to the task-specific component of the program to correct problems or add feature will not affect the program's ability to run under all operating systems.

Brief for Appellant, *supra* note 109, at 6-7.

Though the infringement action involved the ADAPTER module, "ADAPTER [was] not an independently marketed product of [Computer Associates]; it [was] a wholly integrated component of CA-SCHEDULER and [had] no capacity for independent use." *Altai*, 1992 U.S. App. LEXIS 14305, at *10-11.

[112] *Altai*, 1992 U.S. App. LEXIS 14305, at *13. Initially ZEKE could only run on VSE. *Id.*

[113] *Id.* at *14-15.

[114] *Id.* at *15.

[115] *Id.*; *see also* Kevlin, *supra* note 3, at 3.

[116] The former employee had worked on an ADAPTER enhancement project at Computer Associates and was able to complete the Altai project in only three months. *Altai*, 1992 U.S. App. LEXIS 14305, at *14-15.

Associates learned that Altai might have appropriated part of ADAPTER.[117] Computer Associates filed suit, and after litigation began, the former Computer Associates employee admitted that he had copied substantial amounts of Computer Associates' ADAPTER source code when he "created" Altai's adapter module, OSCAR 3.4.[118] Upon the advice of counsel, the president of Altai ordered a rewrite of the offending version of OSCAR 3.4.[119] The rewritten program was entitled OSCAR 3.5.[120] Though the new version did not contain copied code, Computer Associates claimed that Altai's OSCAR 3.5 "copied the structure, sequence and organization of its Adapter program,"[121] and thus infringed on Computer Associates' copyright for ADAPTER.

Following a trial without a jury, the United States District Court for the Eastern District of New York found that Altai was liable for copying ADAPTER into OSCAR 3.4 but that it was not liable for copyright infringement in developing OSCAR 3.5.[122] On appeal,

[117] *Id.* at *15.

[118] *Id.* The former employee had illegally taken a copy of Computer Associates' ADAPTER source code with him when he left Computer Associates. *Id.* at *14. Thirty percent of OSCAR 3.4 was copied from ADAPTER. *Id.* at *15. This version of OSCAR was ultimately found to be infringing by both the district court and the appellate court. Computer Assocs. Int'l, Inc. v. Altai, Inc., 775 F. Supp. 544 (E.D.N.Y. 1991), *aff'd*, No. 91-7893, 1992 U.S. App. LEXIS 14305 (2d Cir. June 22, 1992).

[119] *Altai*, 1992 U.S. App. LEXIS 14305, at *16. Altai's goal was to use as much of OSCAR 3.4 as could be saved without infringement and to remove those portions that had been copied from ADAPTER. The former employee was entirely excluded from the process, and his copy of the ADAPTER code was locked away. Eight programmers were put on the project, none of whom had been involved with the development of OSCAR 3.4. The president of Altai, who had worked closely with the former employee in developing OSCAR 3.4, provided the eight programmers with a description of the portions that had to be reworked. While it had taken the former employee three months to write all of OSCAR 3.4, it took the eight programmers six months to complete the rewrite. *Id.*

[120] *Id.* at *16-17. After the rewrite, Altai only shipped the new version, OSCAR 3.5, to its new customers. Further, Altai shipped OSCAR 3.5 as a "free upgrade" to its customers who had already purchased OSCAR 3.4. However, the damage was already done, and the lawsuit over the issue of OSCAR 3.4 remained. *Id.* at *16-17.

[121] Kevlin, *supra* note 3, at 3.

[122] Computer Assocs. Int'l, Inc. v. Altai, Inc., 775 F. Supp. 544, 570-71 (E.D.N.Y. 1991). Because Altai conceded that it had copied Oscar 3.4., the district court focused solely on OSCAR 3.5. The court began by stating that one of the two

the United States Court of Appeals for the Second Circuit affirmed the judgment of the district court in its entirety.[123] In its decision, the Second Circuit voiced dissatisfaction both with Congress' decision to use copyright law as the vehicle to protect computer programs and with the doctrines courts have developed in their efforts to extend copyright protection to computer programs.

First, the Second Circuit stated its belief that copyright law is an ill-suited means of protection for computer programs, and that computer programs would be better protected by patent law.[124] Of course the Second Circuit recognized that its opinion differed from Congress', and that it was bound to follow the copyright law Congress had passed.[125] However, the Second Circuit stated that it did not intend to follow the case law that had developed since

elements necessary for a copyright infringement action had been met: The plaintiff, Computer Associates, owned a valid copyright in ADAPTER. The court then turned its attention to the remaining element: Was there copying of the copyrighted work by the defendant? *Id.* at 555. The court stated that in order to prove its work had been copied, Computer Associates had to prove "access and substantial similarity between the works." *Id.* at 557. In addition, Computer Associates had to show that its "expression was 'improperly appropriated,' by proving that the similarities relate to copyrightable material." *Id.* (citations omitted). The court assumed that Altai did have access. *Id.* at 558. The court then devoted the rest of its inquiry to the second element of copying—substantial similarity.

[123] Computer Assocs. Int'l, Inc. v. Altai, Inc., No. 91-7893, 1992 U.S. App. LEXIS 14305, at *4 (2d Cir. June 22, 1992). The trial court awarded $364,444 in damages for OSCAR 3.4, which Altai did not challenge on appeal. *Id.* at *4.

[124] *Id.* at *57-58. The Second Circuit registered its belief that:

> [C]opyright registration—with its undiscriminating availability—is not ideally suited to deal with the highly dynamic technology of computer science. Thus far, many of the decisions is this area reflect the courts' attempt to fit the proverbial square peg in a round hole. . . . [P]atent registration, with its exacting up-front novelty and non-obviousness requirements, might be the more appropriate rubric of protection.

*Id.* at *57.

The court further stated that "the resolution of this specific issue could benefit from further legislative investigation—perhaps a CONTU II." *Id.* at *58.

[125] *Id.* at *58. "Congress has made clear that computer programs are literary works entitled to copyright protection . . . . Of course, we shall abide by these instructions, but in doing so we must not impair the overall integrity of copyright law." *Id.*

*Whelan.*[126] The Second Circuit held that the *Whelan* test distorted the balance between protection and dissemination intended by the copyright law[127] because it was over-expansive in its protection of a programmer's work.[128] This balance, the Second Circuit maintained, could only be properly achieved by narrowing the scope of copyright protection for computer programs.[129]

### A. *Narrowing the Scope with a New Test for Substantial Similarity*

In order to narrow the scope, the Second Circuit affirmed the district court's decision to use a new test for substantial

[126] *Altai*, 1992 U.S. App. LEXIS 14305, at *34-37 (citing Whelan Assocs. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1224 n.1 (3d Cir. 1986), *cert. denied*, 479 U.S. 1031 (1987)).

[127] *Altai*, 1992 U.S. App. LEXIS 14305, at *3-4. The court stated: "[T]he copyright law seeks to establish a delicate equilibrium. On the one hand, it affords protection to authors as an incentive to create, and on the other, it must appropriately limit the extent of that protection so as to avoid the effects of monopolistic stagnation." *Id.* at *3 (citation omitted).

[128] *Id.* at *56. The court stated that "the crucial flaw in [*Whelan's*] reasoning is that it assumes that only one 'idea,' in copyright law terms, underlies any computer program, and that once a separable idea can be identified, everything else must be expression." *Id.* at *34 (citing 3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.33) (the author has updated the *Altai* court's Nimmer citations to reflect the pagination of the most recent edition of the treatise). Further, the Second Circuit averred that "the sweeping scope of copyright protection engendered by the *Whelan* rule . . . 'enables first comers to 'lock up' basic programming techniques as implemented in programs to perform particular tasks.'" *Altai*, 1992 U.S. App. LEXIS 14305, at *56 (citations omitted). The court concluded that *Whelan* "results in an inhibition of creation by virtue of the copyright owner's quasi-monopoly power." *Id.* at *56-57 (quoting Spivack, *supra* note 4, at 765).

[129] . The court stated:

> While incentive based arguments in favor of broad copyright protection are perhaps attractive from a pure policy perspective [citation omitted], ultimately, they have a corrosive effect on certain fundamental tenets of copyright doctrine. If the test we have outlined results in narrowing the scope of protection, as we expect it will, that result flows from applying, in accordance with Congressional intent, long-standing principles of copyright law to computer programs.

*Altai*, 1992 U.S. App. LEXIS 14305, at *58-59.

similarity.[130] The new test was "a three-step procedure, based on the abstractions test utilized by the district court, in order to determine whether the non-literal elements of two or more computer programs are substantially similar."[131] The abstractions test looks at a work in increasingly general or abstract terms. Each view contains less and less detail. At a certain point the view is sufficiently general to be considered pure idea. It is at this point that copyright protection ends. Any view of the work past this point—such as a work's general outline—would be unprotectable as mere idea and not the author's expression. The abstractions test was the first of three steps proposed by the Second Circuit: abstraction, filtration, and comparison.[132]

In the first step of the new test, a court would apply the abstractions test[133] in order to "dissect the allegedly copied

[130] *Id.* at *35-36. In discussing why the use of other tests for substantial similarity had failed, the Second Circuit agreed with the district court that "[i]n the context of computer programs, many of the familiar tests of similarity prove to be inadequate, for they were developed historically in the context of artistic and literary, rather than utilitarian, works." *Id.* at *62 (citation omitted). However, the basis of the new test was the abstractions test, which was also developed in the context of literary works. It is difficult to determine whether the court was aware of this inconsistency, for all the court says on the point was that "[w]hile the abstractions test was originally applied in relation to literary works such as novels and plays, it is adaptable to computer programs." *Id.* at *38. Further, while the court believed that the other substantial similarity tests had failed because they had been developed for literary works, the court was fully aware that Congress intended computer programs to be considered literary works. *Id.* at *23-24.

[131] *Altai*, 1992 U.S. App. LEXIS 14305, at *36. The new test delineated by the Second Circuit and the new test used by the district court were essentially the same: "We note that Judge Pratt's method of analysis effectively served as a road map for our own." *Id.* at *64. The Second Circuit, however, formalized the test into a three step process. *Id.* at *36.

[132] *Id.* at *36-37.

[133] *Id.* at *38. The court quoted Judge Learned Hand's articulation of the abstractions test:

> Upon any work . . . a great number of patterns of increasing generality will fit equally well, as more and more of the incident is left out. The last may perhaps be no more than the most general statement of what the [work] is about . . . but there is a point in this series of abstractions where they are no longer protected; since otherwise the [author] could prevent the use of his "ideas", to which, apart from their expression, his property never extended.

program's structure and isolate each level of abstraction contained within it. This process begins with the code and ends with an articulation of the program's ultimate function."[134] The Second Circuit observed that since the abstractions test mirrors the programming process;[135] the test is well suited to computer program infringement cases.[136]

Having determined the levels of abstraction that contain protectable expression, a court would then apply the second step—filtration.[137] At this phase a court would "sift out all non-protectable material"[138] by applying various copyright doctrines such as merger,[139] scenes à faire,[140] and public domain.[141] Thus,

---

*Id.* (citing Nichols v. Universal Pictures Corp., 45 F.2d 119, 121 (2d Cir. 1930), *cert. denied*, 282 U.S. 902 (1931)). *See* discussion, *supra* Part I.B.

[134] *Altai*, 1992 U.S. App. LEXIS 14305, at *39.

[135] *Id.* at *39-40 (citing 3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.32).

[136] *Id.* For the proposition that the programming process mirrors the abstractions test, the Second Circuit relied on the following passage from Nimmer's copyright treatise:

> [A] programmer usually will start with a general description of the function that the program is to perform. Then, a specific outline of the approach to this problem is developed, usually by studying the needs of the end user. Next, the programmer begins to develop the outlines of the program itself, and the data structures and algorithms to be used. At this stage, flowcharts, pseudo-code, and other symbolic representations often are used to help the programmer organize the program's structure. The programmer will then break down the problem into modules or subroutines, each of which addresses a particular element of the overall programming problem, and which itself may be broken down into further modules and subroutines.
>
> Finally, the programmer writes specific source code to perform the function of each module or subroutine, as well as to coordinate the interaction between modules or subroutines. In many ways, the process capsulized above mirrors the . . . abstractions test, [which is] readily adaptable to analyzing computer software. At the start of the process, the programmer only has a general notion of what the program is supposed to do . . . [which] falls into the realm of unprotectable ideas. When the program is completed, the programmer will have produced code which will likely constitute protectable expression. At some point between these extremes, the level of specificity is sufficient to cross the line between idea and expression.

3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.31 to .32 (citations omitted).

[137] *Altai*, 1992 U.S. App. LEXIS 14305, at *40.

[138] *Id.* at *37.

[139] *Id.* at *41-42. The merger doctrine's underlying principle is that "when there

application of the merger doctrine would filter out "those elements of a computer program that are necessarily incidental to its function [and] are [therefore] unprotectable."[142] Similarly, an application of

is essentially only one way to express an idea, the idea and its expression are inseparable and copyright is no bar to copying that expression." *Id.* (citing Concrete Mach. Co. v. Classic Lawn Ornaments, Inc., 843 F.2d 600, 606 (1st Cir. 1988)). The merger doctrine was first developed in *Baker v. Selden.* The case involved the issue of whether ledger sheets, which were to be used with an accounting system explained in plaintiff's books, enjoyed copyright protection. The Court held no, since they were "necessarily incident to" the system of accounting described. *Baker*, 101 U.S. 99, 103 (1879).

[140] *Altai*, 1992 U.S. App. LEXIS 14305, at *48. The scenes à faire doctrine rests on the principle that "where 'it is virtually impossible to write about a particular historical event or fictional theme without employing certain "stock" or standard literary devices,' such expression is not copyrightable." Hoehling v. Universal City Studios, Inc. 618, F.2d 972, 979 (2d Cir.), *cert. denied*, 449 U.S. 841 (1980). *Hoehling* was an infringement suit involving several works on the Hindenberg disaster. The court concluded that "similarities in representations of German Beer Halls, scenes depicting German greetings such as 'Heil Hitler,' or the singing of certain German songs did not lead to a finding of infringement because they were 'indispensable, or at least standard, in the treatment of' life in Nazi Germany." *Altai*, 1992 U.S. App. LEXIS 14305, at *48. As applied to computer programs, the doctrine would exclude those design choices that are circumscribed by external factors such as hardware specifications of the computer on which software is to be run, business demands of the industry to be served, and programming practices that are so widely accepted as to be "stock." *Id.* at *49-50.

[141] *Altai*, 1992 U.S. App. LEXIS 14305, at *51. Material that is in the public domain is not protected by copyright even when it is incorporated into a copyrighted work. 3 NIMMER, *supra* note 33, § 13.03[F] at 13-79. In the context of computer programs, this means public domain software. Such software is often shared and distributed through computer "bulletin boards." Also, some "programming texts may contain examples of actual code that programmers are encouraged to copy." *Id.* § 13.03[F] at 13-80.

[142] *Altai*, 1992 U.S. App. LEXIS 14305, at *31-32. However, as will be discussed *infra* in Part IV, the court extended this doctrine so that "when one considers the fact that the programmers generally strive to create programs 'that meet the user's needs in the most efficient manner', the applicability of the merger doctrine becomes compelling . . . . The more efficient a set of modules are, the more closely they approximate the idea or process embodied in that particular aspect of the program's structure." *Id.* at *43 (citations omitted). This expansion of the merger doctrine is a powerful tool by which the Second Circuit narrowed copyright protection for computer programs. *But see* 3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.36 ("The merger doctrine should be applied to deny protection to those elements dictated purely by efficiency concerns.").

the scenes à faire doctrine would filter out those structural elements that "flow naturally from considerations external to the author's creativity."[143] Finally, an application of the public domain doctrine would filter out any remaining expression that is unprotectable because it is not the expression of the author.

After applying the filtration step, a court would have isolated "a core of protectable expression. In terms of a work's copyright value, this is the golden nugget."[144] The court would then apply the third step—comparison.[145] At this point "the court's substantial similarity inquiry [would focus] on whether the defendant copied any aspect of this protected expression, as well as an assessment of [any] copied portion's relative importance with respect to the plaintiff's overall program."[146]

The Second Circuit held that an application of this "three step approach . . . not only comports with, but advances the constitutional policies underlying the Copyright Act."[147] A closer look at how the court expanded and applied the test will demonstrate that the Second Circuit was incorrect in this assertion.

### *B. Applying the New Test*

The first step in the new test formulated by the Second Circuit was to apply the abstractions test.[148] The court observed that since the abstractions test mirrors the programming process, the test is well suited to computer program infringement cases.[149] The court

[143] 3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.37. Again, the Second Circuit's application of this doctrine is instrumental in narrowing the scope of copyright protection for computer programs. *See* discussion *infra* in Part IV; *but see id.* § 13.03[F] at 13-79 ("[A] court should apply this factor cautiously. Computer programming is a highly creative and individualistic endeavor.").

[144] *Altai*, 1992 U.S. App. LEXIS 14305, at *52.

[145] *Id.*

[146] *Id.*

[147] *Id.* at *53.

[148] *See supra* notes 47-50 and accompanying text for a discussion of the abstractions test. The abstractions test was also used by the district court. *See supra* notes 133-136 and accompanying text.

[149] *Altai*, 1992 U.S. App. LEXIS 14305, at *38-39.

then reviewed the programming process: The programmer starts by identifying the program's ultimate purpose and then goes on to identify the necessary sub-tasks to fulfill that purpose. From there, the programmer arranges these sub-tasks or modules, into a flowchart.[150] "Once each necessary module has been identified, designed and its relationship to the other modules has been laid out conceptually,"[151] the programmer codes the program into source code. Once the source code is complete, it is translated into object code.[152]

However, the five levels of abstraction that the district court identified and the Second Circuit accepted bore little resemblance to the programming process as described in the opinion. The five levels identified were: object code, source code, parameter lists, services required, and general outline.[153] These five levels are not five increasingly general views of a computer program, nor are they a schema that mirrors the programming process; they are merely five different aspects of a computer program.[154] Further, when

[150] *Id.* at *6.

[151] *Id.* at *9.

[152] *Id.*

[153] *Id.* at *65.

[154] "Object code is a program expressed as binary numbers comprehensible to the computer, a pattern of ones and zeros that causes the computer to execute a coherent set of operations leading to a useful result. Most programs are written in some other language more comprehensible to humans, and are then translated into object code . . . ." ANTHONY CLAPES, SOFTWARE, COPYRIGHT & COMPETITION, 31 (1989) [hereinafter CLAPES, SOFTWARE]. Source code is what a programmer produces when he writes or "codes." Usually a programmer works in a "high [] level programming language, . . . such as FORTRAN, which uses arithmetic statements and English-like word statements." Menell, *supra* note 17, at 1051 n.27 (citations omitted). Going from source code to object code is not a process of abstraction, but of translation. The vice-chairperson of CONTU stated that "the translation from source code to object code simply converts the program into a different physical form . . . . Object code is simply a machine translation, or conversion, of the source code." Melville B. Nimmer Declaration, *supra* note 21, ¶ 16.

Parameter lists are the values that are passed between sub-programs or macros of the program. Passing parameters can be analogized to filling in the blanks in a form letter. CLAPES, SOFTWARE, *supra* at 72. Services required would be the function or functions that the program was called upon to deliver. So for the ADAPTER or OSCAR portion of the job-scheduling programs, this would be the system calls the program was taken in sequence, these five views do not reflect a logical progression through "patterns of increasing generality [that] fit equally well, as more and more of the incident is left out."[155] Thus, there is no identifiable point in the sequence where the "level of specificity if sufficient to cross the line between idea and expression."[156] The court's use of the five-level abstraction model is in fact at odds with the whole point of the abstraction test—viz. finding the point where idea and expression diverge.

In addition, the court's application of the abstractions test failed to include a level that referred to the program's detailed structure. So, given the nature of the court's three-pronged test, this restrictive analytical framework tended to deflect the court's inquiry from the question of possible infringement involving the structure, sequence, or organization of a program. Since the court's three-pronged test limited the court's attention to the elements identified during the first step—the abstraction test step—any element that was not included in an abstraction level was dropped from the inquiry. Therefore, since none of the abstraction levels contained ADAPTER's sequential structure, the *Altai* court never asked whether that structure had been infringed.

This omission flowed from the court's reliance on an expert witness and especially the expert's opinions on program structure. Though the Second Circuit discussed the district court's reliance on the expert witness,[157] it never reviewed in detail the expert's view of a program's structure. This is unfortunate because the Second

called upon to make. Computer Assocs. Int'l, Inc. v. Altai, Inc., 775 F. Supp 544, 559 (E.D.N.Y. 1991). The court did not discuss what the general outline was beyond stating that it was the "high level structure." *Altai*, 775 F. Supp. at 561-62. A high level structure has been analogized to the chapters in a book, with each chapter devoted to a different major capability of the program. CLAPES, SOFTWARE, *supra* at 31.

[155] Nichols v. Universal Pictures Corp., 45 F.2d 119, 121 (2d Cir. 1930), *cert. denied*, 282 U.S. 902 (1931).

[156] *See* Nimmer, *A Structured Approach*, *supra* note 96, at 638.

[157] The court stated that the expert was "instrumental in dismantling the intricacies of computer science so that the court could formulate and apply appropriate rules of law." Computer Assocs. Int'l, Inc., v. Altai, Inc., No. 91-7893, 1992 U.S. App. LEXIS 14305, at *63 (2d Cir. June 22, 1992). Unfortunately, much of the expert's opinion refocused the issues so that the court was unable to apply the appropriate rule of law.

Circuit's application of the abstraction test was clearly influenced by the expert's analysis. The expert witness maintained that as far as program structure went, programs could be viewed in two ways: the program as process, and the program as literal text.[158] The expert witness maintained that the sequential structure of a program only emerged when a program processed.[159] Thus, what a layperson would think of as the logical flow of the program—the structure, sequence and organization of the program—only existed when the program processed. Since sequential structure was solely a component of the program's process, any discussion of the text's structure was limited to the literal structure of the words.[160]

Categorizing the sequential structure of a program as part of the program's process had an enormous impact on the way in which the

[158] *Altai*, 775 F. Supp. at 559-60. The court expert began by stating that a program may be examined in its text or in its behavior. Further, each program view—both as text and as behavior—has a structure. The program as text has the structure of its literal text, i.e., the relationship of one word to another. The program as behavior has the structure of its sequence of operations, i.e., the overall design and organization of the program. Thus if the sequences of operations in a program are part of the behavior, then they are not part of the text. *Id.* at 557.

The sequence of operations is what a layperson would think of as the "structure, sequence and organization" of the program. The conclusion to be drawn from this reasoning is that the "structure, sequence and organization" of a program are part of the program's process; that is, there is only sequential structure to a program when the program is processing. The expert based his view of the literal text on the on the fact that the commands in a program do not necessarily appear in the exact order in which they are performed. Therefore, it is not always possible to glean the organization of the program's commands from observing how a program processes. From this the expert concluded that the only meaningful structure of the text is the literal structure of the words, without any thought of "what's next." *Id.* at 557. While this might strictly be true, one commentator noted that:

> the simplest fashion in which control is passed [in a program] . . . is seriatim. Unless the [programmer] decides differently, the instructions in [his] program will be presented to the processor in the order in which they appear in the text of the program. For good reason or convenience, the program author can decide that control of the computer should be passed from one part of the program to a remote program, rather than to a contiguous part. This can be done by calling or macro, or by branching to a subroutine.

CLAPES, SOFTWARE, *supra* note 154, at 84.

[159] *See supra* note 158.

[160] *Id.*

court applied the abstractions test, since the court was aware that the copyright law did not protect any process or method.[161] The court, thus, logically focused its inquiry on the view of program as text, implicitly adopting the expert's opinion that the only structure to be found in the program as text was the structure of the literal words. As a result, none of the levels of abstraction that the court identified from the program as text contained the program's detailed design or flow.

Applying the first step of the abstraction test, the court excluded the program's structure, sequence, and organization from its levels of abstraction. This effectively prevented any further inquiry by the court as to whether the program had been infringed by copying the structure, sequence, and organization. By not considering this question, the court greatly constricted the scope of copyright protection that Congress intended for computer programs.[162] Further, it did so without ever explicitly stating that it had excluded the structure, sequence, and organization of a computer program from copyright protection. Finally, the expert's opinion and the court's own conclusions conflicted with the way in which other commentators viewed programs' flow. One such commentator had stated that "the flow of a program is roughly akin to the flow of a plot in literature. It answers the question 'what happens next ?'"[163] This latter view is consistent with the legislative history.[164] CONTU had made no recommendations that would limit copyright protection for the detailed design, structure, and flow of a computer program, particularly not in circumstances where copyright protection would be available for the structure and flow of a novel or a play.[165]

The court also used the second step of its test (filtration) to narrow greatly the scope of copyright protection for computer programs.[166] The Second Circuit based its filtration step on the

[161] *Altai*, 1992 U.S. App. LEXIS, at *27.

[162] *See supra* notes 26-30.

[163] Clapes, *Silicon Epics*, *supra* note 4, at 1529.

[164] For a discussion of CONTU as legislative history, *see supra* note 20.

[165] Melville B. Nimmer Declaration, *supra* note 21, ¶¶ 30-31.

[166] Although the Second Circuit formalized the application of various copyright

"successive filtering method" suggested by David Nimmer.[167] However, in applying its filtration step, the court ignored the caveats and explanatory notes that had accompanied Nimmer's explication of his successive filtering method.[168]

The first copyright doctrine that the court applied was the merger doctrine.[169] The court began by stating what the merger doctrine meant when applied to computer programs: "[W]hen specific instructions, even though previously copyrighted, are the only and essential means of accomplishing a given task, their later use by another will not amount to infringement."[170] The court then stated that programmers are often concerned with efficiency[171] and that often times there is one way to program a function which is known to be the most efficient. From this, the court correctly concluded what Nimmer had concluded:[172] If a programmer uses

doctrines into its second filtration step, the district court had also employed the scenes à faire doctrine in its analysis. Computer Assocs. Int'l, Inc. v. Altai, Inc., 775 F. Supp. 544, 559-60 (E.D.N.Y. 1991).

[167] Computer Assocs. Int'l, Inc. v. Altai, Inc., No. 91-7893, 1992 U.S. App. LEXIS 14305, at *40 ("Professor Nimmer suggests, and we. endorse, a 'successive filtering method' for separating protectable expression from non-protectable material." (citing 3 NIMMER, *supra* note 33, § 13.03[F])).

[168] *See infra* note 172.

[169] *Altai*, 1992 U.S. App. LEXIS 14305, at *41; *see supra* note 139 for explanation of the merger doctrine.

[170] *Id.* at *42-43 (citations omitted).

[171] *Id.* at *45 ("Efficiency is an industry-wide goal.").

[172] *See also* 3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.35 to .36, which states in pertinent part:

> Although theoretically many ways may exist to implement a particular idea, efficiency concerns can make one or two choices so compelling as to virtually eliminate any other form of expression. In such a case, applying the merger doctrine may dictate the conclusion that a particular way to implement an idea is unprotectable . . . .
>
> Computer searching and sorting algorithms provide good examples of this phenomenon . . . . A great deal of computer science research has been devoted to developing methods of sorting or searching through data, and to analyzing the relative efficiency of various methods.
>
> When efficiency trade-offs between methods are substantial, . . . common sense would dictate that a programmer choose the most efficient method. . . .
>
> In such cases, the merger doctrine should be applied to deny protection to those elements of a program *dictated purely by efficiency concerns.*

the most efficient method, he is not infringing because such a compelling choice is said to have merged with the idea.[173]

However, the court then expanded its application of the merger doctrine to efficiency concerns in programming. It concluded that "the more efficient a set of modules are, the more closely they approximate the idea or process embodied in that particular aspect of the program[]."[174] Thus an experienced programmer who codes efficient programs would lose his copyright protection.[175] This is hardly what Nimmer intended when he stated that "the merger doctrine should be applied to deny protection to those elements of a program dictated *purely by efficiency concerns*."[176] Further, the Second Circuit's application of the merger doctrine far exceeded recent applications of the doctrine in copyright law.[177]

The Second Circuit also applied the doctrine of scenes à faire[178] to computer programs far beyond the scope foreseen by Nimmer.[179] The Second Circuit began by stating the doctrine of scenes à faire as it applied to computer programs: "[I]n many instances it is virtually impossible to write a program to perform particular functions in a specific computing environment without employing standard techniques."[180] "This is a result of the fact that a programmer is

*Id.* (emphasis added).

[173] *Altai*, 1992 U.S. App. LEXIS 14305, at *43.

[174] *Id.* at *43.

[175] One commentator noted that it is "perhaps [the Second Circuit's] unwillingness to protect by copyright factors dictated by efficiency that most sharply distinguishes *Altai* from its predecessors." David Bender, *Computer Associates v. Altai: Rationality Prevails*, COMPUTER LAW., Aug. 1992, 1, 10.

[176] 3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.36 (emphasis added).

[177] Concrete Mach. Co. v. Classic Lawn Ornaments, Inc. 843 F. 2d 600, 606 (1st Cir. 1988) ("When there is essentially only one way to express an idea, the idea and its expression are inseparable and copyright is no bar to copying that expression."); Digital Communications Assocs., Inc. v. Softklone Distrib. Corp., 659 F. Supp. 449, 458 (N.D. Ga. 1987) ("'accepted test' in the computer area is: if 'there is only one way to express the idea, the "idea" and "expression" merge and there is no copyrightable material'") (citation omitted).

[178] *See supra* note 140 for an explanation of the scenes à faire doctrine.

[179] 3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.35 to .42.

[180] *Altai*, 1992 U.S. App. LEXIS 14305, at *49 (citing 3 NIMMER *supra* note 33, § 13.03[F] at 13-78.37).

often circumscribed by extrinsic considerations."[181] Such extrinsic factors include: the mechanical specifications of the hardware on which a program is to run; compatibility requirements of other programs with which the program must interact; computer manufacturer design standards; business needs; and widely accepted programming processes.[182]

According to the Second Circuit's statements, a court should ask if a program element is original, or if it "flows naturally from the considerations external to the author's creativity."[183] Instead, the Second Circuit extended the doctrine to mean that "functional [program] elements . . . do not qualify for copyright protection."[184] Since all elements of a computer program are ultimately functional, this is a far more potent application of the doctrine than that proposed by Nimmer.[185] Nimmer suggested comparing the influence of the programmer's creativity with the influence of the functional consideration.[186] Further, Nimmer warned that "a court should apply this factor cautiously. Computer programming is a highly creative and individualistic endeavor. A court should not be led by defense counsel to believe that complex [computer] programs consist only of commonly known techniques and materials strung together without significant originality or skill."[187]

The court's application of the public domain doctrine to computer programs did not differ from that suggested by Nimmer. Both held that those "elements of a computer program that have entered the public domain by virtue of freely accessible program exchanges and the like"[188] are not protected by copyright.[189]

Upon a completion of the second step, the court stated that it

[181] *Id.* (citing 3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.36 to .42).

[182] *Id.* (citing 3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.37 to .42).

[183] 3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.37.

[184] *Altai*, 1992 U.S. App. LEXIS 14305, at *66.

[185] Mere functionality was not on the list of external factors Nimmer proposed to be used as examples of scenes à faire. 3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.36 to .42.

[186] *Id.* § 13.03[F] at 13-78.37 to .38.

[187] *Id.* § 13.03[F] at 13-79.

[188] *Altai*, 1992 U.S. App. LEXIS 14305, at *51.

[189] *Id.*; 3 NIMMER, *supra* note 33, § 13.03[F] at 13-79.79 to .80.

had "sifted out all elements of the allegedly infringed program which are 'ideas' or are dictated by efficiency or external factors, or taken from the public domain, [but that] there may remain a core of protectable expression."[190] In the third step, the court focused its attention on this core of protectable expression. "At this point, the court's . . . inquiry focuses on whether the defendant copied any aspect of this protected expression, as well as an assessment of the copied portions relative importance with respect to the plaintiff's overall program."[191] The core of protectable material identified through steps one and two must be compared with the defendant's program. This comparison is to "ascertain if there is a sufficient degree of similarity to justify a finding of infringement."[192]

The Second Circuit then reviewed the district court's comparison of the two programs.[193] The Second Circuit agreed with the district court's analysis of the first levels of abstraction: the source and object codes. Both courts agreed that since the literal texts of ADAPTER and OSCAR 3.5 were not identical, there was no substantial similarity.[194] The Second Circuit also concurred with the district court's analysis of the third level of abstraction: the parameter lists. The district court had stated that "only a few of the [parameter] lists . . . were similar to protected elements in ADAPTER; the others were either in the public domain or dictated by the functional demand of the program."[195] The district court had concluded that the similarity of the remaining protected parameter lists did not "warrant a finding of infringement given their relative contribution to the overall program."[196] The Second Circuit did not review the district court's holding on the fourth and fifth levels of abstraction—the list of services and the general outline. At trial, the district court had accepted the expert's opinion as to the

[190] *Altai*, 1992 U.S. App. LEXIS 14305, at *52.

[191] *Id.*

[192] 3 NIMMER, *supra* note 33, § 13.03[F] at 13-80.

[193] *Altai*, 1992 U.S. App. LEXIS 14305, at *58.

[194] *Id.* at *66. *See also Altai*, 775 F. Supp. at 562.

[195] *Altai*, 1992 U.S. App. LEXIS 14305, at *66 (quoting *Altai*, 775 F. Supp. at 560).

[196] *Id.* at *66-67.

copyrightability of both these levels.[197] The court accepted the expert's opinion that the fourth level of abstraction (the list of services) is of "minuscule" importance in the overall picture of similarity.[198] The district court also adopted the expert's evaluation on the fifth level of abstraction—the general outline. The general outline "was not important, because it was so simple and obvious to anyone exposed to the operation of the program."[199] Thus the district court found, and the Second Circuit agreed, that ADAPTER and OSCAR 3.5 were not substantially similar, there was no copying, and Altai had not infringed Computer Associates' copyright.[200]

However, in its review of the district court's decision, the Second Circuit did not discuss the district court's use of the expert's chart.[201] This chart assigned a weighted value to each of the abstraction levels: "code" was ranked at 1000, and all the other levels were ranked at 100, 1, or nil.[202] The purpose of this chart was to "quantify the relative importance" of the abstraction levels that the court had identified.[203] Thus, when the court compared the programs' similarity at each level, it would know how important that similarity was. The court concluded that "the factor which is

[197] *Altai*, 775 F. Supp. at 560.

[198] *Id.* at 562.

[199] *Id.*

[200] *Id.* at 560; *Altai*, 1992 U.S. App. LEXIS 14305, at *69.

[201] The chart appeared as:

| | |
|---|---|
| Code | 1000 |
| Parameter lists | 100 |
| Macros | 100 |
| List of services | 1 |
| Organization chart | Nil. |

*Altai*, 775 F. Supp. at 560.

[202] *Id.*

[203] This aspect of the comparison step was not discussed by the Second Circuit, although the court did declare that it "affirmed the judgment of the district court in its entirety." *Altai*, 1992 U.S. App. LEXIS 14305, at *4. Further, the court had earlier declared its intention to join the company of courts that had extended copyright protection to the nonliteral elements of computer programs. However, the weighting scheme, which gave 1000 points to the computer code, and nil to the structure, would make it difficult—if not impossible—for a plaintiff ever to prevail in a case for infringement of nonliteral elements.

by far the most important—code, rated at 1000—presents no similarity at all because the code was rewritten."[204] Despite the enormous impact of such methodology, the Second Circuit never reviewed the use of the chart or discussed the use of a similar chart in future infringement actions. Therefore, the district court, by its system of comparison, made it more difficult to find infringement in the absence of literal copying.[205]

In total, the court's analysis denied copyright protection to the nonliteral elements of computer programs. Because the court omitted any view of the program's structure, sequence, or organization from its abstractions test, it never even considered whether there was substantial similarity of these elements. The court's application of the filtration step excluded material that was arguably protectable. Further, the system of comparison used by the district court disfavored a finding of copyright infringement absent literal copying. The Second Circuit was correct in its belief that its new test would narrow the scope of copyright protection for computer programs.[206] The court was incorrect, however, to think that its decision was "in accordance with Congressional intent,"[207] which had held that copyright protection should extend to the nonliteral elements of computer programs.[208]

[204] *Altai*, 775 F. Supp. at 560.

[205] However, the Second Circuit never addressed the use of the chart, its effect on the court's findings, or the use of similar charts in future actions.

[206] *Altai*, 1992 U.S. App. LEXIS 14305, at *58.

[207] *Id.* at *59.

[208] The vice-chairperson of CONTU, Melville Nimmer stated that:

> not only the sequence of instructions but the selection, arrangement and coordination of the elements of a program are also protectable under the traditional principles of copyright protections of compilations. (17 U.S.C. § 103) . . . .
>
> Another of the traditional copyright principles CONTU recognized as applicable to computer programs is that, while copyright does not protect against unauthorized use of very generalized abstractions, it does protect against unauthorized use of plots, designs, arrangements and the like which are sufficiently concrete to constitute an expression of the sequence of events, episodes, incidents and segments and the structure of their development, coordination and interplay.

Melville B. Nimmer Declaration, *supra* note 21, ¶ 16. He further stated that:

### *C. The New Test Viewed in Its Larger Context*

It is also important to discuss the court's application of its new three-step test taken in the context of the court's larger test for copyright infringement. The court applied its new test as the second part of its two-part question: Was there access, and was there substantial similarity?[209] For the purposes of analysis, the district court assumed that Altai had access to the ADAPTER code when creating OSCAR 3.5. Thus, in determining whether Altai had unlawfully copied protected aspects of Computer Associates' ADAPTER, the district court narrowed the focus of its inquiry to ascertaining "whether Altai's OSCAR 3.5 was substantially similar to ADAPTER."[210] The Second Circuit stated that it approved of the district court's analysis.[211] This test for copying—access and substantial similarity—is what was discussed above as the compressed *Arnstein* test.

Unlike the full, two-pronged *Arnstein* test, the compressed test requires the court to evaluate the copyrightability of the plaintiff's work without knowing whether there was copying in general. When applied to computer cases, the lack of such probative knowledge can greatly color the court's decision. When a court is asked to determine the degree to which two works are similar because of the same external constraints or efficiency concerns, it would be useful for the court to know if there was copying in

> The general copyright principles applicable to [computer] programs have been, and remain, those which are applicable to novels, plays, . . . textbooks, musical works, maps, motion pictures, sound recordings and other categories of work . . . CONTU had no views, and made no recommendations which would negate the availability of copyright protection for the detailed design, structure and flow of a program under the copyright principles that make copyright protection available, in appropriate circumstances, for the structure and flow of a novel, [or] a play.

*Id.* ¶¶ 12, 28 (cited in Brief for Appellant, *supra* note 109, at 16).

[209] *Altai*, 1992 U.S. App. LEXIS 14305, at *19; *Altai*, 775 F. Supp. at 558 (E.D.N.Y. 1991) (both citing Walker v. Time Life Films, Inc. 784 F.2d 44, 48 (2d Cir.), *cert. denied*, 476 U.S. 1159 (1986)).

[210] *Altai*, 1992 U.S. App. LEXIS 14305, at *19.

[211] *Id.*

general before it made such a decision. The court could then better determine if an element was in both programs because legitimately there was no programming alternative, or rather because the defendant simply copied.

The Second Circuit lacked such probative information, and thus its inquiry focused away from the possibility of copying and instead on the issue of the protectability of the plaintiff's program. Nimmer had warned that "the mere fact that an element of a program might fit within one of the [unprotected] categories . . . does not necessarily mean that such an element is unprotectable, or that its appearance in defendant's program cannot be the result of illegal copying."[212] In sum, the Second Circuit did not examine the elements of the program to see if they were illegally copied; they examined the elements to see if they could be excluded from protection.

## IV. NEITHER *ALTAI* NOR *WHELAN* PROPOSED A VIABLE TEST FOR IDENTIFYING THE PROTECTABLE NONLITERAL ELEMENTS IN A COMPUTER PROGRAM

As we have seen, drawing the line between idea and expression is a difficult task. Judge Learned Hand declared that "nobody has ever been able to fix that boundary, and nobody ever can."[213] Instead that distinction must "inevitably be [made] *ad hoc*."[214] Both the Second and the Third Circuits tried to fix that boundary, and both failed. Neither test sufficiently ensured that computer programs would receive the scope of copyright protection intended by Congress.

The Third Circuit tried to formulate a test that would ensure that computer programs received copyright protection comparable to that accorded other literary works.[215] The court believed that this was necessary because of Congress' decision to include

[212] Nimmer, *A Structured Approach*, *supra* note 96, at 654.

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

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

[215] *See supra* notes 74-79 and accompanying text.

computer programs as a literary work under the new copyright law.[216] Further, Congress gave the idea/expression distinction as the test that marked the limit of copyright protection for a computer program.[217] Since the idea/expression dichotomy is the benchmark used in other literary cases involving nonliteral copyright infringement,[218] there was no evidence that Congress intended to grant computer programs less copyright protection than other literary works.[219] In deciding to grant computer programs the same broad range of protection enjoyed by other literary works, the Third Circuit rejected policy arguments similar to the ones later asserted by the Second Circuit,[220] which advocated granting software less protection than other literary works.[221] The Third Circuit concluded that the policy reasons advanced did not sufficiently prove that Congress had intended computer programs to enjoy less copyright protection than other literary works.[222]

Unfortunately, Third Circuit's *Whelan* test failed to accomplish the goal of granting computer programs copyright protection on a par with that granted other literary works.[223] The test was so overly broad that it protected ideas as well as expression, and thus it violated copyright law.[224] Obviously, by protecting ideas, this was more copyright protection than traditional literary works had ever received. Thus, the Third Circuit failed to formulate successfully a test that ensured computer programs the same range of copyright protection extended to other literary works.

By contrast, the Second Circuit felt compelled to narrow the scope of copyright protection for computer programs for two reasons. The first was the failure of the *Whelan* test. The court correctly asserted that *Whelan*'s overly broad protection "enables first comers to lock up basic programming techniques as

[216] *See supra* notes 2, 64-65 and accompanying text.

[217] *See supra* notes 26-30 and accompanying text.

[218] *See supra* notes 34-45 and accompanying text; *Whelan*, 797 F.2d at 1234.

[219] *Whelan*, 797 F.2d at 1234.

[220] *See supra* notes 17-22 and accompanying text.

[221] *See supra* notes 8-9, 66-73 and accompanying text.

[222] *Whelan*, 797 F.2d at 1234-35.

[223] *See supra* notes 89-94 and accompanying text.

[224] *Id.*

implemented in programs to perform particular tasks."[225] The second reason that the court advanced for narrowing the scope of copyright protection for computer programs was its belief that the balance of interests inherent in the copyright law was better served by a greater dissemination of computer programs to the public. Of course, increasing dissemination also meant limiting protection.[226] As noted above, the balance between dissemination and protection is involved whenever a court makes the idea/expression distinction.[227] Thus a court wishing to change the balance in favor of dissemination need only alter the idea/expression distinction so that there is less protected expression. The Second Circuit wished to change the balance, and it did so by proposing a new test that altered the distinction.

The Second Circuit based its test on the successive filtering method developed by Nimmer. Nimmer had developed the successive filtering method in the wake of *Whelan*'s failures.[228] The method conscientiously and rigorously applied copyright doctrines and judicial tests in order to determine what was expression and what was idea, what was protectable and what was not.[229] Further, the successive filtering method took into account the need for a test that could be applied by non-specialists, yet would yield just results.[230] Unfortunately, the test formulated by the Second Circuit needed to be applied by an expert, and it yielded unjust results.

The three-pronged test formulated by the Second Circuit simply discouraged a finding of copyright infringement absent a finding of literal copying of the program's text. This was far less protection than other literary works received.[231] Thus, the Second Circuit

[225] *See supra* notes 124-126 and accompanying text

[226] *See supra* notes 127-128 and accompanying text.

[227] *See supra* notes 50-54 and accompanying text.

[228] *See supra* note 106 and accompanying text; 3 NIMMER, *supra* note 33, § 13.03[F] at 13-78.32.

[229] *See supra* notes 98-105 and accompanying text.

[230] *See* Nimmer, *A Structured Approach*, *supra* note 96, at 626.

[231] Whelan Assocs., Inc. v. Jaslow Dental Lab., Inc., 797 F.2d 1222, 1234 (3d Cir. 1986) ("The copyrights of other literary works can be infringed even when there is no substantial similarity between the works' literal elements. One can violate the copyright of a play or book by copying its plot or plot devices.") (citations omitted); *see*

failed to formulate successfully a test that ensured computer programs the same range of copyright protection as other literary works, as intended by Congress.[232] In doing this, the Second Circuit established a sort of second-class patent law specifically for computer programs:

> [In exercising] its broad Constitutional Authority to enact patent and copyright laws in order to promote the progress of science and the useful arts, Congress took substantially different approaches in balancing competing interests. Under copyright law, authors are afforded much weaker protection than inventors are afforded under patent law: copyright gives no protection for ideas, no matter how original, and no protection against independent creation. In light of this weak protection, the subject matter capable of being protected is much broader in scope than under patent law (original authorship rather than novel and non-obvious inventions) and the term of protection is much longer. These trade-offs reflect a reasonable and carefully crafted balancing by Congress of the various diverse interests it was seeking to further.[233]

Thus by drastically reducing the portions of a computer program that can be protected against copyright infringement, the Second Circuit upset the balance of protections created by Congress. Congress must reestablish this balance by removing any ambiguities from the copyright law that would prevent a broad scope of protection for computer programs.

Over three years ago, Congress began to investigate the ambiguities in the copyright law as it applied to computer programs. In 1989, the Congressional Subcommittee on Intellectual Property and Judicial Administration asked Congress' Office of Technological Assessment (OTA) to prepare a report that addressed computer technology and the American intellectual property system.[234] In

---

*also supra* note 208.

[232] *See supra* notes 206-208.

[233] Allen R. Grogan, *Bonito Boats and Whelan: A Simple Contrast Between Patent and Copyright Law*, COMPUTER LAW., July, 1989, 33, 34.

[234] Richard Raysman & Peter Brown, *OTA Report: Finding a Balance for*

May of 1992, OTA presented its report to Congress.[235] After careful analysis of the existing law and policies, OTA "identified three policy areas—the appropriate scope of copyright protection, patent protection for software-related inventions and algorithms and complications facing libraries and users of digital information—that Congress might want to address, and outlined possible options for legislative action for each."[236]

OTA noted that the first policy area—determining the proper scope of copyright protection for software—was complicated by the functional aspect of computer software.[237] OTA listed several approaches to the problem.[238]

First, Congress could continue to treat computer programs as literary works, but "could clarify or modify the scope of patent and/or copyright [through the use of] definitional changes."[239] Thus, Congress could specifically exclude or include certain program elements from copyright protection.[240] However, OTA noted that if Congress chose this route, Congress would have to keep the list of definitions in the Copyright Act current.[241] OTA's second suggestion was that Congress "could establish a separate category in the Copyright Act for 'computer programs' instead of treating them as literary works."[242] OTA noted that one "advantage of this approach would be that the courts would not have to apply the same principles to software and other literary works, whose economics, patterns of innovation/dissemination and useful life spans are quite different."[243] However, OTA also stated that "there

*Software Protection*, N.Y. L.J., Nov. 10, 1992, at 3.

[235] OFFICE OF TECHNOLOGY ASSESSMENT, U.S CONG., PUB NO. OTA-TCT-527, FINDING A BALANCE: COMPUTER SOFTWARE, INTELLECTUAL PROPERTY, AND THE CHALLENGE OF TECHNOLOGICAL CHANGE (1992).

[236] Raysman & Brown, *supra* note 234, at 3. Only the first of these three policy areas, the appropriate scope of copyright protection, is within the scope of this note.

[237] *See* OTA REPORT, *supra* note 235, at 28-29.

[238] *Id.* at 29.

[239] *Id.*

[240] *Id.* at 30.

[241] *Id.*

[242] *Id.*

[243] *Id.*

would be a period of uncertainty as a new body of case law developed."[244] As a third proposal, OTA suggested that "[i]nstead of establishing a separate category for software within Section 102(a), Congress might limit that scope of literary copyright to the code, with the possibility of adopting a *complementary* regime for elements of software design and functionality."[245] OTA's fourth suggestion was that computer programs could remain within the category of literary works, but Congress could "establish legislative bounds holding the extent of copyright as a literary work to the code (as text), not to the behavior of the program when it is executing or to 'interfaces.' [Congress could then] determine whether the latter are to be covered by a complementary, *sui generis* regime."[246] Finally, OTA suggested replacing copyright protection for computer programs with a *sui generis* framework of protection that would include the code as well as other program elements.[247] OTA pointed out that law that is specifically tailored to software could include features not permitted under copyright law, such as a shorter term length, or specific infringement criteria.[248] However, OTA also pointed out that it would be "difficult to define what is and what is not covered under copyright law and under the new, *sui generis* law. A new body of case law would have to develop, as would international agreements, particularly regarding the *sui generis* mode of protection."[249]

OTA's suggestion that Congress simply clarify the existing copyright law is the soundest solution for courts now facing infringement cases of the nonliteral elements of software. A "new system would create unacceptable uncertainty."[250] Further, a new body of law will inevitable create new ambiguities that will eventually have to be clarified. A better approach would be for

[244] *Id.*

[245] *Id.* (emphasis in original).

[246] *Id.*

[247] *Id.* at 30-31.

[248] *Id.* at 30.

[249] *Id.*

[250] *Id.* at 28 (OTA notes that this is the contention of those commentators who argue that the existing copyright law is working well, both domestically and internationally) (citations omitted).

Congress to clarify the ambiguities that have already surfaced. That would provide courts with a large body of existing case law, and some direction as to how to apply it.

Congress should begin by stating in clear and unmistakable terms that computer programs are granted the full panoply of rights afforded to more traditional literary works.[251] Congress should state that the copyright law applies equally to the nonliteral and literal elements of a computer program, and it should reiterate that the idea/expression distinction is the test for copyright protection.[252] Further, Congress should add that the issue of nonliteral versus literal elements in an alleged infringement case should have no bearing on the idea/expression distinction. The test should be applied equally in instances of nonliteral and literal appropriation. If Congress clearly indicates that nonliteral and literal elements are equally protected under the copyright law, a future court could not institute an approach explicitly designed to provide less copyright protection for nonliteral infringement than for literal infringement of computer programs, as the Second Circuit did in *Altai*.[253]

In order to ensure that computer programs are granted the same copyright protection as other literary works, Congress should briefly explain that computer programs are to be treated the same as other literary works because both types of work contain potentially protectable expression outside the literal text. Congress should also make it clear to courts that they can grant computer programs copyright protection based on the same understanding of similarity that is applied in cases involving other literary works.[254] Courts should feel encouraged to draw analogies between a novel and a program's structure.

[251] *See supra* notes 26-30 and accompanying text.

[252] *See supra* note 26 and accompanying text.

[253] *See supra* notes 149-209 and accompanying text.

[254] Melville Nimmer stated that "[i]n my treatise, I have analyzed in some depth the standards for determining when a copyrighted work has been infringed through the duplication of its fundamental plot, structure and arrangement (the pattern test). In my opinion, CONTU fully expected that these traditional principles be applied to computer programs." Melville B. Nimmer Declaration, *supra* note 21, ¶ 25; *see also supra* notes 121-126 and accompanying text.

Thus, a court should consider any organizational construct that a programmer utilizes to structure a program as, at least, potentially protectable as expression. This would eliminate a court's ability to construe sequential structure as process, and thus to deny copyrightability.[255] Congress should also consider providing a non-inclusive list of the nonliteral elements of a computer program that are to enjoy copyright protection to the extent that they embody an author's expression. However, if Congress does provide such a list, it must emphasize that the list is only illustrative, and not exclusive. Such a list should include the organization, sequence, and structure of the overall program and its subroutines and modules, to the extent that they do not embody mere ideas. Stating that the idea/expression distinction could be made at more than one level would prevent future courts from providing the same overly broad protection found in the *Whelan* test, which identified a program with a single idea.[256]

If Congress emphasizes its intention to grant computer programs the same broad copyright protection as other literary works, Congress will encourage courts to apply the vast body of case law and doctrines.[257] This is extremely appropriate, given that the creative process of computer programming is similar to that of writing a more traditional literary work.[258] Further, by clearly indicating that computer programs contain protectable expression beyond the literal text, Congress will ensure the proper balance between dissemination and incentive.[259] Much of a programmer's skill finds expression in a program's overall design.[260] To deny

[255] *See supra* notes 154-159 and accompanying text.

[256] *See supra* notes 87-90 and accompanying text.

[257] *See* Clapes, *Silicon Epics*, *supra* note 4, at 1501 ("Traditional rules of copyright law adapt very comfortably to [computer programs].").

[258] *Id.* at 1507 ("That such an authorship takes place in the context of a programming language does not change its essential nature.").

[259] *See supra* notes 47-52 and accompanying text.

[260] Clapes, *Silicon Epics*, *supra* note 4, at 1534-35.

> As with structure, flow and logic, design is an attribute that may be considered at a high level of abstraction or at a low level of abstraction. The low level of abstraction, the program's detailed design, is a complex web of structure, sequence, pattern and organization. The resulting combination is a tapestry of decisions and actions that is the essence of the author's expression.

copyright protection for that expression is to eliminate incentive.[261] Congress must make clear that the creative expression found in a computer program's nonliteral elements is protected under the copyright law.

## CONCLUSION

Other U.S. courts should not follow either the *Altai* decision or the *Whelan* decision. Though opposite in their approaches, both decisions failed. The *Altai* decision should not be followed because it grants far less copyright protection to computer programs than was intended by Congress. Further, the *Altai* decision upsets the balance of protections envisioned by Congress.[262] The *Whelan* decision errs in the opposite direction because it grants protection to a program's incorporated ideas, and thus it fails to strike the proper balance between public dissemination and individual incentive mandated by the Constitution. The *Altai* decision was an answer to the failings of the *Whelan* decision. Yet the *Altai* decision failed as strikingly as the *Whelan* decision. A third test is not the answer. The *Altai* test demonstrated that even the most thoughtful test does not guarantee just results.

United States courts are now faced with alternatives in deciding software copyright infringement cases. A court wishing to grant protection to the nonliteral elements of a program follows *Whelan*; a court wishing to deny such protection follows *Altai*. A Supreme Court resolution may be required in the short term to avert the forum shopping that will become inevitable. However, a Congressional clarification of the copyright as it applies to computer programs would be a superior solution.

> Because computer programs as a form of expression are not well-understood by the makers of law and policy, they are presently at risk of being relegated to the backwaters of copyright, to an inferior status at law in which the authors

*Id.* (citing P. BRUCE & S. PEDERSON, THE SOFTWARE DEVELOPMENT PROJECT: PLANNING AND MANAGEMENT 85-86 (1982)).

[261] *Id.* at 1509-10.

[262] Grogan, *supra* note 233, at 34.

> of this class of literary work would be accorded less than full protection against the taking of their original works of authorship.[263]

Armed with a clarification of the law, courts could successfully apply the broad body of copyright law and doctrines to computer programs cases on an *ad hoc* basis, which after all, seems to be inevitable.

Lisa C. Green

[263] Clapes, *Silicon Epics*, *supra* note 4, at 1499.