NORTH CAROLINA LAW REVIEW

Volume 66 | Number 5 Article 5

6-1-1988

The Incompatibility of Copyright and Computer Software: An Economic Evaluation and a Proposal for a Marketplace Solution

Vance Franklin Brown

Recommended Citation

Vance F. Brown, *The Incompatibility of Copyright and Computer Software: An Economic Evaluation and a Proposal for a Marketplace Solution*, 66 N.C. L. Rev. 977 (1988).
Available at: http://scholarship.law.unc.edu/nclr/vol66/iss5/5

COMMENT

The Incompatibility of Copyright and Computer Software: An Economic Evaluation and a Proposal for a Marketplace Solution*

Since the advent of the computer age the legal community has been seeking, pondering, and debating ways in which to group, define, and protect a form of intellectual property commonly called computer software.[1] Revisions to the Copyright Act[2] and the decision in *Apple Computer, Inc. v. Franklin Computer Corp.*[3] and its progeny firmly place the bulk of software protection in the copyright domain.[4] Nevertheless, the reliance on copyright law as applied to this particular form of intellectual property has become destructive to the competitive process in the software industry.

Because copyright law was not originally designed to protect computer-related property, the courts have had to struggle with semantics, battle with well-settled copyright precedent, and grapple with the wording of the copyright statute and its legislative history to mold copyright law into society's protective device for software property. The *Apple* decision and subsequent cases identified the scope of such protection for literal, verbatim copies of computer programs. The United States Court of Appeals for the Third Circuit delivered a landmark opinion in *Whelan Associates v. Jaslow Dental Laboratories,*[5] expanding the scope of copyright protection from the literal, identical copying of a program's code to the copying of its structure, sequence, and organization. This expansion distorts copyright law by failing to adhere to a crucial principle: copyright protection extends only to the *expression* and *not* to the *idea* expressed.[6] Unfortunately, the *Whelan* court could achieve equitable results only by an incorrect application of copyright law.

Recent court decisions have extended protection beyond the underlying software code to the literal elements of a program's executable image, commonly

* Portions of this Comment are derived from the author's prize-winning entry in the 1987 Nathan Burton Memorial Competition.

[1] In a market economy, to facilitate effective and efficient allocation of scarce resources, it is necessary to grant individuals exclusive property rights. *See* Demsetz, *Toward a Theory of Property Rights*, 57 AM. ECON. REV. 347 (1967).

[2] Pub. L. No. 94-553, 90 Stat. 2541 (1976) (codified at 17 U.S.C. §§ 101-810); *see infra* notes 34-50 and accompanying text.

[3] 714 F.2d 1240 (3d Cir. 1983), *cert. dismissed,* 464 U.S. 1033 (1984); *see infra* notes 183-86 and accompanying text.

[4] The majority of the legal community believes that copyright protection of software is effective. *See* Baumgarten & Meyer, *Program Copyright And the Office of Technology Assessment* (part 1), COMPUTER LAW., Oct. 1987, at 8; Mantle, *Trade Secret and Copyright Protection of Computer Software,* 4 COMPUTER L.J. 669 (1984).

[5] 797 F.2d 1222 (3d Cir. 1986), *cert. denied,* 107 S. Ct. 877 (1987); *see infra* notes 62-78 and accompanying text.

[6] Commonly referred to as the "idea-expression" dichotomy, this is a basic principle of copyright law. 17 U.S.C. § 102(b) (1982).

called the program's "look and feel" or user interface.[7] In this situation a correct application of copyright law leads to anticompetitive results. The first developer of successful software who is able to set an industry standard may now use the copyright law as a shield against competitive market forces. Because copyright law and software are incompatible, this Comment suggests that copyright protection for computer software be abandoned.

Part I reviews the history of copyright protection as applied to computer software and then examines the effect of recent copyright decisions on the competitive process. Further, it examines the incompatibility between copyright protection and a socially desirable economic and competitive result in the computer software industry. Part II recommends that Congress, courts, and software developers place the onus of protection on the marketplace by encouraging use of contract law, complemented by trademark and trade secret law. Moreover, for those circumstances that fall through the cracks of these established bodies of law, Congress should adopt a federal law of misappropriation, a catch-all category that would preserve a first developer's legitimate lead time and thereby provide the finishing touch to a complete scheme of software protection. This solution would protect both the idea and the expression of software, but only for a time that would be determined by market forces. Courts would then be free to render equitable and procompetitive decisions without distorting a body of law that is ill-suited for the task of software protection.

Before examining the basics of copyright law as applied to software, an examination of the economic and constitutional objectives of an effective scheme of legal protection is useful.[8] First, the plan should provide the economic incentive needed to stimulate innovation and creativity in the computer software industry. This can be accomplished by allowing individuals to achieve either monetary or other personal gain, commensurate with the creativity involved, as measured by the software's usefulness to society.[9] The plan must therefore protect intellectual endeavors by preventing others from benefiting from the creator's innovations. This reward for innovation, however, must be tempered by society's desire to promote further innovation and competition. New ideas must be disseminated throughout society so that further progress can be made without having to reinvent the wheel.[10] The preservation of competition in the mar-

[7] *See* Waldman, *Software-Copyright Laws Are in State of Confusion*, Wall Street Journal, March 21, 1988, at 23 col.4.

[8] The economic goals expressed in this Comment are not universally accepted by lawyers and economists, but evince this author's views on objectives that should be followed while searching for an optimal scheme of legal protection. A debate or comparison of the various schools of thought is beyond the scope of this Comment. For a review of the conflicting economic views on granting monopoly rights to inventors, see Machlup, *An Economic Review of the Patent System*, Study No. 15, Subcommittee on Patents, Trademarks and Copyrights, Senate Committee on the Judiciary, 85th Cong., 2d Sess. (1958), *reprinted in* P. GOLDSTEIN, COPYRIGHT, PATENT, TRADEMARK AND RELATED STATE DOCTRINES 16-19 (2d ed. 1981).

[9] *See, e.g.*, Mazer v. Stein, 347 U.S. 201, 219 (1954) (discusses the economic philosophy behind intellectual property protection).

[10] Even Sir Isaac Newton realized the importance of building on the work of his predecessors. He has been quoted as proclaiming, "[I]f I had seen farther than other men, it was because I had stood on the shoulders of giants." Whelan Assoc. v. Jaslow Dental Laboratories, 797 F.2d 1222, 1238 n.33 (3d Cir. 1986), *cert. denied*, 107 S. Ct. 877 (1987).

ketplace is desirable to keep consumer prices down, quality high, and to trigger innovation and technological advancements; therefore, it would be inefficient to allow software developers to secure greater monopoly profits than is necessary to stimulate creativity and innovation.[11] Second, the plan must comply with the guideline set forth by the United States Constitution that empowers Congress "[t]o Promote the Progress of Science and useful Arts, by securing for limited time to authors and inventors the exclusive right to their respective writings and discoveries."[12]

I. The Evolution of Copyright Protection for Software

Computers were first developed in the 1940s, but these early computers were programmed by the setting of electrical switches by hand.[13] In the 1950s and 1960s programmers began writing programs in source code to automate the manual switching task, but software misappropriation was not yet a problem because most programs were custom-developed to meet the needs of individual users, and the software was packaged and sold with the computer hardware.[14] Additionally, these big computers were affordable only to businesses and institutions with sufficient financial resources to purchase and maintain the machines.

With the advent of semiconductor technology in the 1970s came the problem of software misappropriation. Microcomputers became affordable to small

[11] *See* Hirshleifer, *The Private and Social Value of Information and the Reward to Inventive Activity*, 61 Am. Econ. Rev. 561 (1971).

> "The advantages arising from a system of copyright are obvious. It is desirable that we should have a supply of good books: we cannot have such a supply unless men of letters are liberally remunerated. . . .
>
> . . . .
>
> . . . [T]he least exceptionable way of remunerating them is by a monopoly. Yet monopoly is an evil. For the sake of the good we must submit to the evil; but the evil ought not to last a day longer than is necessary for the purpose of securing the good."

Baird, *Common Law Intellectual Property and the Legacy of* International News Service v. Associated Press, 50 U. Chi. L. Rev. 411, 416 n.15 (1983) (quoting Macaulay, *Copyright* (pt. 1), in Prose and Poetry 731, 733-35 (G. Young ed. 1952)).

[12] U.S. Const., art. I, § 8, cl. 8.

[13] *See* Wharton, *Use and Expression: The Scope of Copyright Protection for Computer Programs*, 5 Computer L.J. 433 (1985).

[14] A programmer initially writes the software in a source code, which contains words within programming statements often equivalent to English (*e.g.*, "begin" or "while"). An entire statement or instruction may contain, for example, "While X < 100 Do," or "IF X > Y GOTO 50." The source code may be in one of several languages, such as COBOL, FORTRAN, BASIC, Pascal, C, or EDL. Programmers use different languages often depending on the type of application program they are writing. For example, the acronym "COBOL" stands for "COmmon Business Oriented Language" and is used in many business environments; "FORTRAN" stands for "FORmula TRANSlation" and is commonly used for scientific applications; and "BASIC" refers to "Beginners All-purpose Symbolic Instruction Code" and is commonly the language used in schools to teach the fundamentals of computer programming. The source code can then be edited much like one would edit existing text in a word processor. Once the program is written in source code, it is then automatically translated by the computer into object code, or machine language, which is a binary code of "0"s and "1"s that represent low and high voltages and that tells the hardware in the computer whether to turn switches or gates "on" or "off." Instead of hand-setting electrical switches as was done in the 1940s, this is now done by the software, a term that may include either the source code or the object code, but only object code actually directs the computer to perform functions.

For more information outlining the computing process, see L. Dickey, Introduction to Computer Concepts Hardware and Software (1974); *see also* Appendix to this Comment.

businesses and to home users. Various vendors produced microcomputers that would run the same software. These hardware advances led to entry into the market by individual software producers who wanted to take advantage of the high demand for programs that would do everything from word processing to spread sheet calculations. By the beginning of the 1970s, an estimated 10,000 computer programs were being written each day.[15] Vendors began mass marketing programs stored on a machine readable medium, such as floppy disks, to users worldwide. The problem of misappropriation surfaced when consumers soon found it very profitable, convenient, and easy to copy software written by others rather than purchase an authorized copy.[16] Some competitors also began taking copied software and selling it, or passing it off as their own.[17]

As the value to society of computer software rapidly increased, so came the plea for protection from its piracy.[18] Before determining the most effective method of protecting computer programs, the legislature and courts were confronted with the problem of how this new form of property should be categorized and defined. The legal community had to decide whether software should be copyrighted like a book or patented like a machine.[19]

To better understand this dilemma and the complexity of the problem, consider the following: A program is designed to solve a particular problem. An applications program, which tells the computer how to execute specific functions such as word processing and spread sheet calculations, begins as an algorithm, recipe, or flow chart. Usually first written in the vernacular, these procedures are then coded into a programming language such as COBOL, Pascal, or FORTRAN, commonly referred to as the source code.[20] The source program is automatically translated by the computer into object code, which is a binary

[15] Milgrim, *Proprietary Rights in Software,* COMPUTER LITIGATION 1985 TRIAL TACTICS AND TECHNIQUES 15, 17 (1985) (citing Koller, *Computing Software Protection: Report of an Industry Clinic,* 13 IDEA 351, 361 (1970)).

[16] Copying a program stored on a floppy disk to another floppy can usually be accomplished in a matter of seconds at a cost of under $2.00 (the cost of the blank floppy). Today it is estimated that, excluding backups, for every legitimate copy of a computer program there exists an unauthorized copy. *See* Berkin, *A Look at Software Piracy—1987,* COMPUTER LAW., Sept. 1987, at 11.

Computer crime's harmless image has probably been enhanced because until recently our society did not portray computer pirates in a bad light. Popular movies applaud computer criminals instead of chastising them. In one recent film, "War Games," the hero improves his mediocre high school grades by accessing and manipulating the school's computer system. The news media also has lauded computer criminals. The term "computer piracy," first embraced by the media, gives these criminals a certain Robin Hood attractiveness. Society in general seems delighted with the fact that computers are not yet a substitute for swashbuckling human ingenuity. *See* Pender, *Perspectives on Computer Ethics and Crime,* 36 BUSINESS 30 (1986).

[17] Sometimes piracy takes the form of copying a program stored on cassette tape, floppy disk, hard disk, or some other external memory device. However, some programs are stored on a silicon chip, known as a ROM (read only memory) chip. Everyday users do not copy programs from such a device, but experts can copy programs from this chip. *See, e.g.,* Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240, 1249 (3d Cir. 1983) (holding that programs stored in ROM are copyrightable), *cert. dismissed,* 464 U.S. 1033 (1984).

[18] *See, e.g.,* Parker v. Flook, 437 U.S. 584, 587 n.7 (1978) (Court observed that by 1976 the value of software in our society exceeded $43 billion).

[19] Although other forms of protection, such as trade secret and trademark law, are being used to protect software (*see* sources listed *infra* note 161), the legal community chose to group this form of intellectual property under one of the traditional headings of "patent" or "copyright."

[20] *See supra* note 14 and accompanying text.

representation ("1"s and "0"s).[21] This machine language instructs the hardware when to turn internal switches on ("1") and off ("0") within the computer.

This program, the creative solution to a problem, is first expressed in the English language and eventually expressed as electrical signals. A program may be grouped and defined in several ways because it changes form several times during its existence. At its two highest levels, the flow chart and source code, the program is a readable expression and one might therefore assume that it should be copyrighted like a book. At its lowest level, the machine language, one might suggest that software be patented like a machine.

Patent law protects "any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof."[22] The owner of a patent is granted a monopoly over the use, manufacture, and sale of the invention for the statutory period of seventeen years and is protected against independent discovery.[23] Obtaining a patent is usually a long, laborious, and expensive process, often taking three to five years. Moreover, the patent applicant must meet stringent tests of usefulness,[24] novelty,[25] and non-obviousness.[26] As the *quid pro quo* for the grant of the patent monopoly, the inventor is required to make a full disclosure of the work to the public.[27]

The patentability of computer programs has been rigorously debated by legal scholars and the courts.[28] A well-settled principle in patent law is that laws of nature, mathematical formulas, abstract ideas, or other fundamental truths are not patentable.[29] For example, Einstein could not have patented the formula $E = mc^2$, nor Pythagoras his theorem, nor Newton the laws of gravity.

[21] "Compilation" is a term used by programmers to describe the process in which the high-level source code is automatically transformed into low-level object code. The "compiler" that accomplishes this task is simply another program.

[22] 35 U.S.C. § 101 (1982); *see* S. MANDELL, COMPUTERS, DATA PROCESSING, AND THE LAW 124-30 (1984).

[23] *See* 35 U.S.C. § 154 (1982).

[24] *See* 35 U.S.C. § 101 (1982).

[25] *See* 35 U.S.C. §§ 102(a), (e) (1982).

[26] *See* 35 U.S.C. § 103 (1982) ("A patent may not be obtained . . . if the differences between the subject matter . . . and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art . . . .").

[27] *See* 35 U.S.C. §§ 111, 112 (1982).

[28] *See* H. HANNEMAN, THE PATENTABILITY OF COMPUTER SOFTWARE (1985); Comment, *The Protection of Property Rights in Computer Software*, 14 AKRON L. REV. 85 (1980); Comment, *Intellectual Property Protection for Computer Programs: Are Patents Now Obtainable*? 26 CATH. U.L. REV. 835 (1977); Note, *Patentability of Computer Programs*, 56 DET. J. URB. L. 289 (1978); Note, *The Policy Implications of Granting Patent Protection to Computer Software: An Economic Analysis*, 37 VAND. L. REV. 147 (1984); Annotation, *Patentability of Computer Programs*, 6 A.L.R. FED. 156 (1971).

In 1966 a Report by the President's Commission on the Patent System concluded that patent laws are unsuitable for the protection of computer software. S. DOC. NO. 5, 90th Cong., 1st Sess., 12-13 (1966).

[29] In O'Reilly v. Morse, 56 U.S. 62 (1853), the Court held that Samuel B. Morse could not obtain exclusive rights for any device using electromagnetism to communicate over distances. To do so, the Court reasoned, would grant an exclusive right to electromagnetism. *See also* Rubber-Tip Pencil Co. v. Howard, 87 U.S. (20 Wall.) 498, 507 (1874) ("[a]n idea of itself is not patentable"); Le Roy v. Tatham, 55 U.S. (14 How.) 156, 175 (1852) ("A principle, in the abstract, is a fundamental truth; an original cause; a motive; these cannot be patented, as no one can claim in either of them an exclusive right.").

Because computer programs often utilize mathematical principles, past courts have usually denied their patentability.[30] However, in the 1981 case of *Diamond v. Diehr*,[31] the claimed invention involved a process for molding raw, uncured synthetic rubber into cured precision products. The United States Supreme Court concluded that a program may be included as part of an otherwise patentable machine or process: "[W]hen a claim containing a mathematical formula implements or applies that formula in a structure or process which, when considered as a whole, is performing a function which the patent laws were designed to protect,"[32] then the claim satisfies the requirements of patent law.

Diehr implicitly questions the applicability of patent protection per se for computer software. One can infer from *Diehr* that computer programs by themselves are not patentable. However, more recently in *Paine, Webber, Jackson & Curtis v. Merrill Lynch*,[33] the United States District Court for the District of Delaware upheld the patentability of a sophisticated set of business-oriented computer programs. Even if the decision in *Paine, Webber* stands and application programs are patentable per se, using the patent laws as a means of protection for software may be impracticable: by the time a patent is issued the rapid technological change in the industry would render many programs obsolete.

There never has been an obvious method of protecting computer software. The legislative and judicial selection of copyright law as the solution indicates only that this body of law seemed the most appropriate way to deal with the problems of the time. Initial appropriateness, however, does not necessarily mean that copyright law is the most effective way to deal with the problems courts are addressing today.

[30] *See, e.g.*, Parker v. Flook, 437 U.S. 584 (1978) (patent denied for mathematical method of updating "alarm limits," the numbers which indicated safe temperature and pressure levels measured during catalytic conversion of hydrocarbons); Dann v. Johnston, 425 U.S. 219 (1976) (patent denied for computer and computer program on ground of obviousness under § 103); Gottschalk v. Benson, 409 U.S. 63 (1972) (patent denied for computerized process for converting binary-coded decimal (BCD) numbers into ordinary decimal numbers, for a "mere idea" is an unpatentable mental process).

Two cases, *In re* Freeman, 573 F.2d 1237 (CCPA 1978) and *In re* Walter, 618 F.2d 758 (CCPA 1980), limited the mathematical algorithm objection to patentability. One commentator explaining the significance of these two decisions stated:

> The *Freeman-Walter* test sets forth a two-step analysis for determining whether a mathematical algorithm was statutory subject matter. First, the claim was analyzed to determine whether a mathematical algorithm was directly or indirectly recited in the claim. Second, if the mathematical algorithm was found, the claim as a whole was further analyzed to determine whether the algorithm was applied in any manner to physical elements to process steps. If the algorithm applied to physical elements or the process steps, then it passed muster under [35 U.S.C. § 101] and the overall invention was patentable.

R. RAYSMAN & P. BROWN, COMPUTER LAW: DRAFTING AND NEGOTIATING FORMS AND AGREEMENTS § 5.07[3] (1987).

[31] 450 U.S. 175 (1981).

[32] *Id.* at 192.

[33] 218 U.S.P.Q. 212 (D. Del. 1983) (upholding the validity of a patent for a "Securities Brokerage-Cash Management System").

A. *The Legislative Initiative*

Congress enacted the Copyright Act of 1790[34] to provide legal protection for publishers and authors against unauthorized copying of printed material. Congress had no way of predicting the evolution of nontraditional printed subject matter.[35] In 1964 the Copyright Office reluctantly began accepting computer programs for registration as books.[36] The 1976 Copyright Act (1976 Act), effective January 1, 1978,[37] provided for copyright protection in "original works of authorship fixed in any tangible medium of expression," including "literary works."[38] Although the 1976 Act did not expressly include computer software as copyrightable subject matter, the legislative history and the definitions contained within the Act clearly indicate that Congress intended the term "literary works" to encompass computer programs.[39]

However, section 102(b) of the 1976 Act precludes copyright protection for "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."[40] Section 102(b) is a codification[41] of the unanimous Supreme Court decision in *Baker v. Selden*[42] where the Court found that although the plaintiff owned a valid copyright in a book describing an accounting system, which included blank forms consisting of ruled lines and headings (based on the universally used T-accounts), the copyright did not preclude others from using that accounting system.[43] The Court was concerned that copyright law not be used to *monopolize* ideas or utilitarian processes, but that it

[34] Act of May 31, 1790, ch. 15, 1 Stat. 124.

[35] The most notable development in American copyright law since 1790 has been its recurring expansion. It now embraces "original works of authorship fixed in any tangible medium of expression . . . [including] literary works; musical works; dramatic works; pantomimes and choreographic works; pictorial, graphic, and sculptural works; motion pictures and other audiovisual works; and sound recordings." 17 U.S.C. § 102(a) (1982).

[36] M. SCOTT, COMPUTER LAW § 3.4 (1987).

[37] Pub. L. No. 94-553, 90 Stat. 2541 (1976) (codified at 17 U.S.C. §§ 101-810).

[38] 17 U.S.C. § 102(a) (1982).

[39] M. SCOTT, *supra* note 36, § 3.5, at 3-9. Section 101 defines "literary works" as "works, other than audiovisual works, expressed in words, numbers, or other verbal or numerical symbols or indicia, regardless of the nature of the material objects, such as books, periodicals, manuscripts, phonorecords, film, tapes, *disks*, or cards, in which they are embodied." 17 U.S.C. § 101 (1982) (emphasis added). The legislative history of § 101 reveals that "literary works . . . include . . . computer programs to the extent that they incorporate authorship in the programmer's expression of original ideas, as distinguished from the ideas themselves." H.R. REP. No. 1476, 94th Cong., 2d Sess. at 54, *reprinted in* 1976 U.S. CODE CONG. & ADMIN. NEWS 5659, 5667.

[40] 17 U.S.C. § 102(b) (1982).

[41] *See* H.R. REP. No. 1476, 94th Cong., 2d Sess., *reprinted in* 1976 U.S. CODE CONG. & ADMIN. NEWS 5659, 5670 (Section 102(b) is intended to "restate . . . that the basic dichotomy between expression and idea remains unchanged.").

[42] 101 U.S. 99 (1879).

[43] Professor Nimmer observed:

> The doctrine of *Baker v. Selden* rests upon the premise that the copyright laws may not be used to obtain a monopoly on a system or method for performing commercial or scientific functions. . . . From this acceptable premise, the courts have reasoned that if the only manner of performing the system or method is by copying a written expression, then the copyright liability must be denied for such writing when used to perform such system or method, else the system or method itself will be monopolized.

1 M. NIMMER, NIMMER ON COPYRIGHT § 2.18(C) (1978).

only be used to protect the *expression* of the idea or process.[44]

In 1974 Congress created the National Commission on New Technological Uses of Copyrighted Works (CONTU)[45] to evaluate the problems associated with protecting proprietary rights in software and to make specific recommendations for legislation that would adequately protect the interests of software developers.[46] After three years of data collection, hearings, analysis and deliberation, the majority of CONTU recommended that the 1976 Copyright Act "be amended to make it explicit that computer programs, to the extent that they embody an author's original creation, are proper subject matter of copyright."[47] Based on the recommendations of CONTU, Congress responded in the 1980 Computer Software Copyright Act[48] by adding a provision defining computer programs and by replacing the old section 117 with a new section 117, authorizing the owner of a copy of a computer program to make essential adaptations for use with the computer or for archival purposes.[49]

The CONTU Report remains significant because it represents the only legislative history reflecting the meaning and scope of the revisions enacted by Congress. Moreover, several court decisions applying copyright law to computer programs have looked to the Report for guidance when rendering their decisions.[50]

[44] *Baker*, 101 U.S. at 103-04; *see also* Mazer v. Stein, 347 U.S. 201, 217 (1954) ("Unlike a patent, a copyright gives no exclusive right to the art disclosed; protection is given only to the expression of the idea—not the idea itself.") (citation omitted).

[45] NATIONAL COMM'N ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS, FINAL REPORT 9 (1978) [hereinafter CONTU REPORT].

[46] *See, e.g.*, 120 CONG. REC. 41415 (1974) (the evaluation by CONTU "is inherently valuable in our forthcoming review of the copyright laws. With the advent of computer technology, recall electronics, data storage and the like, it is impossible for our courts to handle cases of copyright infringement under the [former copyright] law.") (statement by Rep. Danielson).

[47] CONTU REPORT, *supra* note 45, at 2. Commissioner Hersey dissented from this recommendation, arguing that "[t]he Act of 1976 should be amended to make it explicit that copyright protection does not extend to a computer program in the form in which it is capable of being used to control computer operations." *Id.* at 2. Hersey based his dissent on the belief that "[w]orks of authorship have always been intended to be circulated to human beings and to be used by them—to be read, heard, seen, for either pleasurable or practical ends. Computer programs, in their mature phase, are addressed to machines." *Id.* at 70.

[48] 17 U.S.C. § 101 (1982) ("[A] computer program is a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result.").

[49] Section 117 currently provides in part:

> It is not an infringement for the owner of a copy of a computer program to make or authorize the making of another copy or adaptation of that program provided:
>
> (1) that such new copy or adaptation is created as an essential step in the utilization of the computer program in conjunction with a machine and that it is used in no other manner, or
>
> (2) that such new copy or adaptation is for archival purposes only and that all archival copies are destroyed in the event that continued possession of the computer program should cease to be rightful.

17 U.S.C. § 117 (1982).

[50] *See, e.g.*, Apple Computer, Inc. v. Formula Int'l, Inc., 725 F.2d 521 (9th Cir. 1984) (making reference to CONTU REPORT as authority for its decision); Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240, 1251 (3d Cir. 1983) (relied on CONTU REPORT), *cert. dismissed*, 464 U.S. 1033 (1984). "Although the Congressional action in 1980 does not appear to be supported by a legislative history, it is fair to conclude, since Congress adopted its recommendations without alteration, that the CONTU Report reflects the Congressional intent." Midway Mfg. Co. v. Strohon, 564 F. Supp. 741, 750 n.6 (N.D. Ill. 1983).

B. *The Judicial Attempt at Applying Copyright Law*

Armed with a congressional mandate, the courts were ready to resolve software misappropriation disputes. Courts, however, were faced with the dilemma of applying traditional copyright principles to a nontraditional copyright subject matter. First, under traditional principles the protected literary expression was meant to communicate with humans. The programmer's literary expression, in contrast, is intended in its mature state to communicate with a machine.[51] Second, a computer program, unlike traditional forms of subject matter, is a utilitarian work rather than an artistic or fictional one in that it causes a computer to perform various tasks.[52] Third, the judiciary system as a whole lacks the highly intricate understanding of computer functioning and thus is not equipped to differentiate one programmer's expression from another coded expression, thereby frustrating application of the "substantial similarity" test.[53]

[51] As observed by one commentator:

> The extension of copyright protection to computer software requires the courts to apply copyright principles to a type of literary work which is different in certain ways from any other protected by the copyright laws. The most fundamental of these differences is that computer programs lack the communicative function of traditional literary works.

Gesmer, *Developments in the Law of Computer Software Copyright Infringement*, 26 JURIMETRICS J. 224, 225 (Spring 1986). Another commentator stated that "[n]ew tests and modified legal principles will have to be developed to deal with computers—a form of 'literary machines.' " Davis, *Computer Software—The Final Frontier: Clones, Compatibility and Copyright*, COMPUTER LAW., June 1985, at 1, 3.

[52] The functional component of a work does not receive protection under the copyright laws. *See, e.g.*, Esquire, Inc. v. Ringer, 591 F.2d 796 (1978) (denying validity of copyright for overall shape of outdoor lighting fixture because granting protection for mere utilitarian articles would lead to widespread protection for industrial designs). The definition of "pictorial, graphic, and sculptural works" in section 101 of the Copyright Act includes "works of artistic craftsmanship insofar as their form but not their mechanical or utilitarian aspects." 17 U.S.C. § 101 (1982).

Often it is difficult, if not impossible, to distinguish between a work's artistic expression and its functional component. In Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240 (3d Cir. 1983), *cert. dismissed*, 464 U.S. 1033 (1984), Franklin argued that Apple's operating system (a collection of systems software programs that allows the computer to perform programming tasks, execute programs, and direct input and output tasks) was a purely "utilitarian work" which could not be copyrighted and that Apple was seeking to block the use of the art embodied in its operating systems. *Id.* at 1251. The court rejected Franklin's argument because nothing in the copyright statute supports the argument that an intention to use a copyrightable article in industry invalidates its registration, so long as it contains copyrightable expression. *Id.* at 1251-52.

[53] Because direct evidence of copying is rarely available, copying may be established by proof of "access" and "substantial similarity." *See* Midway Mfg. Co. v. Strohon, 564 F. Supp. 741 (N.D. Ill. 1983) (applying "substantial similarity" test). The leading case in this area is Arnstein v. Porter, 154 F.2d 464 (2d Cir. 1946), *cert. denied*, 330 U.S. 851 (1947), in which the court established a two-step test for determining substantial similarity. The two-step test requires the application of (1) an "extrinsic" test directed at determining whether there exists a substantial similarity in the underlying ideas; and (2) an "intrinsic" test to determine whether there is substantial similarity in the expression of the underlying idea. *See* Sid & Marty Krofft Television Prods., Inc. v. McDonald's Corp., 562 F.2d 1157, 1164 (9th Cir. 1977). Traditionally, expert testimony is admissible to prove similarity under the extrinsic test, but the intrinsic test is administered only by the "ordinary lay observer." *Id.* This caused obvious difficulties when an ordinary lay observer was asked to compare the similarity of two computer programs. To alleviate this problem, the courts generally have abandoned the bifurcated test of substantial similarity in computer copyright disputes in favor of an integrated substantial similarity test in which both lay and expert testimony is admissible. *See* Whelan Assocs. v. Jaslow Dental Laboratories, 797 F.2d 1222, 1233 (3d Cir. 1986) ("[W]e believe that the ordinary observer test is not useful and is potentially misleading when the subjects of the copyright are particularly complex, such as computer programs."), *cert. denied*, 107 S. Ct. 887 (1987); E.F. Johnson Co. v. Uniden Corp. of Am., 623 F. Supp. 1485, 1492 (D. Minn. 1985) (substantial similarity test un-

Despite these obstacles, the courts solidified the applicability of copyright law to computer software in several first generation cases. Although Congress had helped define the term "computer programs," courts were left with the task of determining whether the definition extended to programs written in object code, whether operating system programs were to be protected,[54] and whether tangible mediums of expression included programs embedded in silicon chips, such as ROMs,[55] PROMs,[56] and EPROMs.[57] Although one pre-1980 amendment court opinion stated that a ROM silicon chip containing a computer program was not a copy of the program subject to copyright protection,[58] since the landmark case of *Apple Computer, Inc. v. Franklin Computer Corp.*[59] the overwhelming majority of courts have found that copyright protection extends to all computer programs, whether in object or source code, or whether stored on any fixed media, tape, disk, ROM chips, or any other such device.[60]

Although courts had to overcome some fundamental difficulties with this nontraditional and complex subject matter, the old copyright concepts adapted well because these first generation cases involved extensive and verbatim copying of the plaintiff's underlying code. These cases could easily be equated to a classic hypothetical case in which an infringer duplicated an entire book, made relatively few, if any, changes, and then sold the book as her own.

workable in computer cases); Midway Mfg. Co. v. Strohon, 564 F. Supp. at 752-53 (relied entirely on expert testimony to find substantial similarity).

One recommendation is that courts shift the emphasis from the "substantial similarity" test and focus on the defendant's conduct and on the advantage the defendant gained from the infringement. Conley & Bryan, *A Unifying Theory for the Litigation of Computer Software Copyright Cases*, 63 N.C.L. REV. 563 (1985); *see also* Note, *Copyright Infringement of Computer Programs: A Modification of the Substantial Similarity Test*, 68 MINN. L. REV. 1264, 1285-88 (1984) (substantial similarity test has little value when applied to computer programs); Note, *Copyright Infringement Actions: The Proper Role for Audience Reactions in Determining Substantial Similarity*, 54 S. CAL. L. REV. 385, 399-404 (1981) (criticizing lay observer standard in certain instances).

[54] *See supra* note 52.

[55] Read Only Memory.

[56] Programmable Read Only Memory.

[57] Erasable Programmable Read Only Memory.

[58] *See* Data Cash Sys. v. JS&A Group, Inc., 480 F. Supp. 1063, 1066-69 (N.D. Ill. 1979), *aff'd*, 628 F.2d 1038 (7th Cir. 1980).

[59] 714 F.2d 1240 (3d Cir. 1983), *cert. dismissed*, 464 U.S. 1033 (1984).

[60] *See* Apple Computer, Inc. v. Formula Int'l, Inc., 725 F.2d 521 (9th Cir. 1984) (all computer programs as embodied in ROMs and diskettes are copyrightable subject matter); *Franklin*, 714 F.2d 1240 (source and object code copyrightable, notwithstanding ROM embedded on silicon chip); Williams Electronics, Inc. v. Artic Int'l, Inc., 685 F.2d 870 (3d Cir. 1982) (statutory requirement of "fixation" satisfied through embodiment of expression in ROM); E.F. Johnson Co. v. Uniden Corp. of Am., 623 F. Supp. 1485 (D. Minn. 1985) (object program stored in EPROM protected by copyright); Midway Mfg. Co. v. Strohon, 564 F. Supp. 741 (N.D. Ill. 1983) ("instruction ROMs" copyrightable subject matter); GCA Corp. v. Chance, 217 U.S.P.Q. 718 (N.D. Cal. 1982) (object code part of operating system loaded onto EPROM copyrightable subject matter); Tandy Corp. v. Personal Micro Computers, Inc., 524 F. Supp. 171 (N.D. Cal. 1981) (object code part of an operating system on a ROM was copyrightable subject matter); *see also* Horning, *Copyright Protection of Computer Software*, in COMPUTER LITIGATION 1985 TRIAL TACTICS AND TECHNIQUES 115, 123-24 (Practicing Law Inst. ed. 1985) (listing computer-related cases construing "copyrightable subject matter").

1. The Merger of Idea and Expression

The current focus has shifted from issues regarding the copyrightability of certain forms of programs and various acceptable mediums of expression to questions about the degree of similarity required before a second work constitutes an infringing copy.[61] In general these cases do not involve the line by line, verbatim duplication encountered in earlier cases, but the copying of a program's overall structure, sequence, organization, and even its manner of operation.

In one of the most important and controversial computer copyright cases to date, *Whelan Associates, Inc. v. Jaslow Dental Laboratories, Inc.*,[62] the United States Court of Appeals for the Third Circuit ruled that "copyright protection of computer programs may extend beyond the program's literal code to their structure, sequence, and organization."[63] Plaintiff did not allege, nor did the court find, any copying of the literal elements of a program—the source or object codes. Rather, the court found that the overall structure of defendant's program was substantially similar to the overall structure of plaintiff's program. An overview of the facts in this case is essential to an understanding of the court's conclusion.

Defendant-appellant Jaslow Dental Laboratory and plaintiff Whelan Associates, a custom-software developer, entered into a contractual arrangement whereby plaintiff agreed to design and code a computer program that would automate the business functions of defendant's dental laboratory.[64] The program, "Dentalab", was written for an IBM Series One machine in a computer language known as EDL.[65] The contract stipulated that the program would remain the property of plaintiff and that defendant would receive a ten percent royalty for every other copy sold.

While this business relationship continued, defendant Rand Jaslow, realizing the economic potential of a comparable program that would run on an IBM PC or compatible, began to develop such a program written in the BASIC programming language. Because Rand did not possess the computer expertise to properly finish this project, he employed a professional computer programmer to complete the program. Rand then terminated the existing contract with Whelan and began marketing his "Dentcom" program, which he advertised as "a new version of the Dentlab [sic] computer system."[66] Rand also sent a letter to Whelan Associates warning them against the further marketing of the Dentalab program, which Rand maintained contained valuable trade secrets of Jaslow Dental

[61] *See* Radcliffe, *Recent Developments in Copyright Law Related to Computer Software,* 4 COMPUTER L. REP. 189 (1985); Note, *Defining the Scope of Copyright Protection for Computer Software,* 38 STAN. L. REV. 497 (1986).

[62] 797 F.2d 1222 (3d Cir. 1986), *cert. denied,* 107 S. Ct. 877 (1987).

[63] *Id.* at 1248.

[64] *Id.* at 1225. The "business functions" included registering and processing inventory, updating customer lists, invoicing, billing, and other general accounting tasks.

[65] Event Driven Language.

[66] *Whelan,* 797 F.2d at 1227.

Laboratory.[67]

Because the programs were written in two different languages, the code could not be an exact duplication. Moreover, plaintiff did not allege that defendant's Dentcom program was a "translation" of plaintiff's Dentalab program (*i.e.*, the logical equivalent of a translation from English to French).[68] Rather, plaintiff argued that its copyright had been infringed because defendant's program was substantially similar in three respects: the file structures, the screen outputs, and five subroutines.[69] Rand Jaslow also admitted to having access to the Dentalab source code.

After discounting the need for the traditional ordinary observer test when applied to complex computer software cases,[70] the court focused its attention on the scope of computer software protection. The court resolved that because copyrights of other literary works can be infringed even absent substantial similarity between the works' literal elements, there is no reason to establish a rule denying structural protection when applied to software.[71] Defendants contended that the nonliteral components of a program necessarily entailed the program's idea and thus could not receive protection.[72] Recognizing that copyright law only protects the expression of an idea, the court agreed with the district court's all-encompassing view that the expression in a software program is "the *manner in which the program operates, controls and regulates* the computer in receiving, assembling, calculating, retaining, correlating, and producing useful information either on a screen, printout or by audio communication."[73]

The language of the court seems to expressly ignore the "procedures," "processes," and "methods of operation" excluded from copyright by section 102(b),[74] thereby crossing the threshold from the protection of expression into the protection of the underlying idea.[75] Yet, instead of quibbling with the

[67] *Id.* at 1226.

[68] As one court observed:

> [I]t is as clear an infringement to translate a computer program from, for example, FORTRAN to ALGOL, as it is to translate a novel or play from English to French. In each case the substance of the expression . . . is the same between original and copy, with only the external manifestation of the expression changing.

Synercom Technology, Inc. v. University Computing Co., 462 F. Supp. 1003, 1013 n.5 (N.D. Tex. 1978).

[69] *Whelan*, 797 F.2d at 1228. A "subroutine" is a group of programming statements that are logically grouped together to perform a specific task.

[70] *See supra* note 53.

[71] *Whelan*, 797 F.2d at 1234. The *Whelan* court relied on cases that prohibited copying of plots or plot devices in plays or books. *See* Twentieth Century-Fox Film Corp. v. MCA, Inc., 715 F.2d 1327, 1329 (9th Cir. 1983) (plot similarities between "Battlestar Galactica" and "Star Wars" may be basis for infringement); Sid & Marty Krofft Television Prods., Inc. v. McDonald's Corp., 562 F.2d 1157, 1167 (9th Cir. 1977) (similarities between McDonaldland characters and H.R. Pufnstuf characters can be established by showing the similarity in the "total concept and feel" of the two); Nichols v. Universal Pictures Corp., 45 F.2d 119, 121 (2d Cir. 1930) (copyright "cannot be limited literally to the text, else a plagiarist would escape by immaterial variations").

[72] *Whelan*, 797 F.2d at 1235.

[73] *Id.* at 1239 (emphasis added). The *Whelan* court was encouraged by the holding in SAS Inst., Inc. v. S & H Computer Sys., 605 F. Supp. 816 (M.D. Tenn. 1985), despite the *SAS* court's finding evidence of similarity in both literal and organizational aspects. *Whelan*, 797 F.2d at 1239.

[74] *See supra* text accompanying note 40.

[75] This holding is in direct opposition to the position taken by the Copyright Office, which is

court's misuse of statutory language or engaging in a semantic debate, it may be more prudent to examine the inherent practical problems that surface when protection is extended to the organization and structure of computer software.

The court's formulation of protected expression represents a broad view of the idea of the program. The court stated that the idea in this program was the "efficient organization of a dental laboratory."[76] The court formulated a line between the idea and expression by making reference to the end sought to be achieved: "In other words, 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."[77] Thus, the court reasoned, "[w]here 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]

This "means-end" analysis is dangerous in the context of computer programming. Various "means" chosen by programmers are often dictated by reasons of efficiency, including programmer's time, speed of computer execution, and size of compiled code.[79] These efficient means and methods of programming are taught to computer science students in school and business. For purposes of illustration in a nonprogramming context, students are taught to learn multiplication tables (5 × 4 = 20) so that they refrain from multiple addition (5 + 5 + 5 + 5 = 20). Students of computer programming are taught to choose certain statements over others for the same efficiency reasons. For example, a programmer writing in Pascal could achieve the same desired purpose of printing the word "dangerous" on the screen ten times in various ways, one of which is as follows:

```
Writeln (“dangerous”);
Writeln (“dangerous”);
Writeln (“dangerous”);
Writeln (“dangerous”);
```

the agency responsible for administration of the Copyright Act. The Copyright Office publishes "circulars" that explain copyright law to lay persons. The Copyright Office's Circular R61 (May 1983) supports a more narrow scope of copyright protection:

> EXTENT OF COPYRIGHT PROTECTION
>
> Copyright protection extends to the literary or textual expression contained in the computer program. *Copyright protection is not available for ideas, program logic, algorithms, systems, methods, concepts, or layouts.*

Circular R61 (May 1983) (emphasis added). The *Whelan* court stated that the circular should be given marginal weight and that "to the extent that the Copyright Office's circular differs, it should not be followed." *Whelan,* 797 F.2d at 1242 n.38.

[76] *Id.* at 1240.

[77] *Id.* at 1236.

[78] *Id.* The court further stated: "This test is necessarily difficult to state, and it may be difficult to understand in the abstract." *Id.* at 1236 n.28.

[79] *See* Petition For A Writ of Certiorari To The United States Court Of Appeals For The Third Circuit at 16, Whelan Assocs., Inc. v. Jaslow Dental Laboratories, Inc., 797 F.2d 1222 (3d Cir. 1986), *cert. denied,* 107 S. Ct. 877 (1987) [hereinafter Petition For Certiorari]. Moreover, this test may be problematic in the realm of microprocessors, in which the sequence and ordering of the microinstructions are often dictated by the microprocessor hardware. *See* Steinberg, NEC v. Intel: *The Battle Over Copyright Protection For Microcode,* 27 JURIMETRICS J. 173 (1987).

```
Writeln ("dangerous");
Writeln ("dangerous");
Writeln ("dangerous");
Writeln ("dangerous");
Writeln ("dangerous");
Writeln ("dangerous");
```

Or, the programmer could implement the following statements:

```
For  I := 1 to 10 do begin
        Writeln ("dangerous");
end;
```

Obviously, the latter method would be the more desirable coding technique. With a computer program there are often optimal methods of programming and various programming techniques that are dictated by reasons of efficiency. This is not true with traditional literary works. There is no most efficient way to write poetic lines, musical notes, or romantic novels.[80] Yet efficient "structures" and "organization" are commonplace in the programming arena. In fact, most sophisticated programming students are required to take courses in "data structures" and "computer organization."[81] Judicial protection of a program's structure, sequence, organization, and methods of operation may, in certain instances, be analogized to protection of multiplication tables, an event that would be precluded even by patent law.[82]

As this example illustrates, application of traditional copyright analysis to computer programs fails because software is not a traditional form of literary work.[83] Granted, several efficient ways exist to produce a given result in a computer program, but courts are not equipped to determine the extent to which a program's structure has been copied.[84] The holding may ultimately depend on which party's expert is the most persuasive and which expert the court deems

[80] Petition For Certiorari, *supra* note 79, at 16.

[81] *See, e.g.*, R. KRUSE, DATA STRUCTURES & PROGRAM DESIGN (1984); A. TANENBAUM, STRUCTURED COMPUTER ORGANIZATION (1984).

[82] The court attempted to rid itself of this inconsistency by stating in a footnote: "Of course structural similarities between two programs can also arise in completely legitimate ways—*e.g.*, where the authors of the two programs have included subroutines from common, unprotected subroutine libraries, or where the authors of both programs have consulted common, public domain, reference books." *Whelan*, 797 F.2d at 1248 n.47. But then even the court recognized the difficulty: "There can be no bright line rules as to when similarities are evidence of infringement and when they are legitimate." *Id.*

The United States Court of Appeals for the Fifth Circuit recently "decline[d] to embrace" the *Whelan* Court's strict holding that the mere showing of substantial similarity of organization and structure together with access is sufficient to establish copyright infringement. Plains Cotton Coop. Ass'n v. Goodpasture Computer Serv., 807 F.2d 1256, 1262 (5th Cir.), *cert. denied*, 108 S. Ct. 80 (1987). The court recognized that similarity in organization and structure do not necessarily denote a copyright infringement. In *Plains* the court affirmed the lower court's refusal to grant a preliminary injunction because plaintiff failed to show substantial likelihood of success on its copyright claim. The court reasoned that the similarities between the two programs in this case were dictated by the externalities of the cotton market, for "market factors play a significant role in determining the sequence and organization of cotton marketing software, and we decline to hold that those patterns cannot constitute 'ideas' in a computer context." *Id.*

[83] Davis, *Computer Software—The Final Frontier: Clones, Compatibility And Copyright*, COMPUTER LAW., June 1985, at 1, 3.

[84] *See, e.g.*, SAS Inst., Inc. v. S & H Computer Sys., 605 F. Supp. 816, 832 (M.D. Tenn. 1985)

most credible.[85] Moreover, a good programmer can take source code and rearrange subroutines, change variable names, and disguise copying in ways that make tenuous any determination of structural similarity, even by another programming expert.

The cases that expanded the traditional copyright protection from the copying of the literal elements of a program's code to the copying of its overall structure and organization are tainted with facts in which the losing party was guilty of egregious conduct. Faced with the choice of denying equitable relief or distorting the copyright law by crossing the idea-expression barrier, courts chose the latter.

The proposition that the overall structure and organization of a computer program is an important consideration in a software copyright infringement suit was first stated by a court in *SAS Institute, Inc. v. S & H Computer Systems.*[86] The case involved the alleged copying of plaintiff's "SAS" statistical analysis program. The court took note of at least 44 instances of defendant's direct copying from plaintiff's source code and could have rested its opinion on literal, line-by-line copyright infringement without looking to the elements of organization and structure. However, the determination that defendant's conduct was especially egregious seemed to be integral to the court's conclusions.[87] The court noted that defendant made obvious attempts to disguise evidence of copying, made no preliminary design plans for its DEC version of SAS, and that it entered into a licensing arrangement with the SAS Institute to obtain the program's source code knowing from the outset that it would breach the express terms of the contract. The court concluded that defendant's conduct relative to the licensing agreement "cannot be said to comply with its legal duty of good faith and fair dealing."[88]

Most courts evaluating a copyright infringement claim have looked only to two classic elements: 1) proof that the injured party owned the copyrighted material; and 2) the existence of copying usually established by the determination that the two competing works are substantially similar and that the alleged infringer had access to the copyrighted material.[89] However, use of copyright

("This Court is not a computer programmer, and is simply not able to determine, out of 186,000 lines of integrated product code, which of them reflect misappropriation.").

[85] The trial court in *Whelan* gave great weight to plaintiff's more credible expert: "I conclude . . . that Dr. Moore, plaintiff's expert had greater knowledge as to the particular programs at issue . . . . To the extent that Dr. Moore's testimony supports plaintiff's contentions of copying, I find his testimony more credible and helpful . . . ." *Whelan*, 609 F. Supp. at 1321-22.

[86] 605 F. Supp. 816, 830 (M.D. Tenn. 1985) ("the copying proven at trial does not affect only the specific lines of code cited by [plaintiff's expert]. . . . Rather, to the extent that it represents copying of the organization and structural details of SAS, such copying pervades the entire S & H product.") (relying on the reasoning of Meredith Corp. v. Harper & Row Publishers, Inc., 378 F. Supp. 686 (S.D.N.Y), *aff'd*, 500 F.2d 1221 (2d Cir. 1974), *opinion after trial*, 413 F. Supp. 385 (S.D.N.Y 1975)).

[87] The court concluded that the SAS source code was "extensively and systematically used by S & H in the preparation of S & H's product." *SAS*, 605 F. Supp. at 822.

[88] *Id.* at 828. The court also found instances of "slavish copying." *Id.* at 826.

[89] *See* Plains Cotton Coop. Ass'n v. Goodpasture Computer Serv., 807 F.2d 1256, 1260 (5th Cir.), *cert. denied*, 108 S. Ct. 80 (1987); Miller v. Universal City Studios, 650 F.2d 1365, 1375 (5th Cir. 1981); 3 M. NIMMER, *supra* note 43, § 12.11[D].

law to provide equitable relief has led at least one commentator to suggest that courts should explicitly recognize that conduct of a party is a legitimate element to be considered in a software copyright dispute: "It often will be inappropriate in software cases simply to compare the competing works to determine whether they are substantially similar. Rather, the technology dictates a focus on the defendant's conduct and any advantage he has gained from it."[90]

In 1986 a Congressional report examined the impact recent and anticipated advances in technology will have on the intellectual property system and particularly copyright law.[91] Congress's Office of Technology Assessment ("OTA") report concluded that an inherent incompatibility exists between computer software protection and copyright law:

> Regardless of how legal scholars resolve the issue of idea and expression, the Federal courts, in interpreting copyright law, will eventually face a dilemma; either: 1) the copyrightable expression in a computer program will be limited to the strict line-by-line program code, in which case the unscrupulous might easily escape liability for infringement by simply varying the code in a trivial way, or 2) the copyrightable expression will be extended to the logic, design, structure, performance or even the output of the program, in which case one has copyrighted a "procedure, process, system, or method of operation." The cases that have been decided thus far indicate that the courts are adopting the latter alternative, and have extended the meaning of expression in computer programs to include the processes that the programs implement. . . .
>
> In theory, none of these rulings is permitted under traditional copyright principles. This is not because the courts have misinterpreted copyright law, but because copyright law cannot be successfully applied to computer programs.[92]

·2. The "Look and Feel" Cases

The previously discussed cases involved the scope of protection extended to

[90] Conley & Bryan, *supra* note 53, at 612.

[91] U.S. CONGRESS, OFFICE OF TECHNOLOGY ASSESSMENT, INTELLECTUAL PROPERTY RIGHTS IN AN AGE OF ELECTRONICS AND INFORMATION, OTA-CIT-302 (Washington, DC: U.S. Government Printing Office, April 1986) [hereinafter OTA REPORT]. This report was requested by Senator Charles Mathias, Jr., Chairman of the Senate Judiciary Committee, Subcommittee on Patents, Copyrights and Trademarks; and by Congressman Peter W. Rodino, Jr., Chairman of the House Judiciary Committee, Congressman Robert W. Kastenmeier, Chairman of the Subcommittee on Courts, Civil Liberties, and the Administration of Justice, Congressman Hamilton Fish, and Congressman Carlos Moorhead. *Id.* at iii.

The OTA's conclusion that copyright law is fundamentally incompatible with software technology protection has already been sharply criticized for reflecting a "conceit that permeates human history: the belief that the present is so special—so utterly distinct from the past—that solutions found by previous generations for their problems offer little if any guidance today." Baumgarten & Meyer, *supra* note 4, at 10. These commentators, however, wholeheartedly applaud how well traditional copyright law concepts have adapted to recent cases without examining the effect such application has on the competitive process. Such an evaluation is analogous to concluding that a chocolate cake has an exquisite flavor because all the right ingredients were used, without ever tasting it.

[92] OTA REPORT, *supra* note 91, at 81.

the underlying code of computer software—the source and object code. The most recent controversy in the computer law arena involves the scope of protection copyright affords to the visual output of computer software. The visual output of software is analogous to a clock face and the object code is like the clockworks. It appears from the current judicial trend that a second programmer will be precluded from independently developing software that emulates the functions and operations of an already existing program.

Technological growth has always depended on innovators improving and building on the work of their predecessors.[93] This is often described as a desire not to force inventors to re-invent the wheel. Similarly, the competitive process can thrive only when producers may freely enter markets and compete with similar products.

Suppose, for example, that A was the sole producer of Calvin Keen jeans in the town of Brookeville. A great demand for these jeans arose among teenagers because of the stylish, slender fit and the designer look. Producer A had a limited supply of these jeans and was able to sell them at double the cost of production. Competitor B, in a neighboring town, wanted to take advantage of these tremendous profits so she designed a similar jean, entered A's market, and sold her jean at two-thirds the price of Producer A's jeans. All the teenagers in Brookeville began buying this less-expensive jean because it looked just as good and the lower price made their parents much happier. Once Producer A recognized that he was losing business to Competitor B, he quickly lowered his prices below that of B's jeans. Soon this competitive struggle between Producer A and Competitor B drove the price of the jeans down to their cost of production. Meanwhile, instead of becoming fat and lazy and resting on the success of his Calvin Keen jean, Producer A realized that the competitive market would not allow him to reap profits forever. Accordingly, Producer A was ready to market his sleek new designer shirt that would bear the well-known and cherished Calvin Keen label . . . and the process continues.

In much the same way, it is important that software competitors be allowed to produce similar, yet non-copied programs to compete with those programs that are already reaping profits. Copyright protection was not originally intended to prevent competitors from offering similar products. The United States District Court for The Middle District of Tennessee in *SAS Institute, Inc. v. S & H Computer Systems*[94] recognized the importance of allowing programmers to write independent, yet similar code, " 'one is always free to make the machine do the same thing as it would if it had the copyrighted work placed in it, but only by one's own creative effort rather than by piracy.' "[95]

In practice this statement is no longer accurate. The protection now pro-

[93] The court in Whelan Assocs. v. Jaslow Dental Laboratories, 797 F.2d 1222 (3d Cir. 1986), *cert. denied*, 107 S. Ct. 877 (1987), acknowledged that "copyright law has always recognized and tried to accommodate the fact that all intellectual pioneers build on the work of their predecessors." *Id.* at 1238.

[94] 605 F. Supp. 816 (M.D. Tenn. 1985).

[95] *Id.* at 829 (quoting National Commission on New Technological Uses of Copyrighted Works, Final Report, 21 (1978)).

vided by copyright law for software's audiovisual display, derivative works, and compilations creates an insurmountable barrier to competitor entry in the software market.

a. Audiovisual and Derivative Works

The subject matter of a valid copyright may contain an original work of authorship in literary works, motion pictures, or other audiovisual works.[96] Audiovisual works are defined as any work consisting of "a series of related images which are intrinsically intended to be shown by the use of machines . . . together with [any] accompanying sounds."[97] The audiovisual copyright can be infringed without infringing the literary copyright in the computer program's object and source codes.[98]

The test for the copyrightability of an audiovisual display is the same as that for a literary work: whether the work conveys expression that is distinguishable from the underlying idea.[99] The early software cases that relied on audiovisual copyright were the long line of entertainment-oriented videogame cases.[100] Each case involved computer programs that generated artistic and fanciful audiovisual displays. *Atari, Inc. v. North American Philips Consumer Electronics Corp.*[101] involved the alleged infringement of the popular Pac-Man game, which displays a "gobbler" being guided through a maze appearing to gobble up dots in its path while being chased through the maze by several "ghost monsters." The United States Court of Appeals for The Seventh Circuit held that the play of the game, strategy, and the underlying maze-chase concept was not protected, but that the expression, as articulated in the "gobbler" and "ghost monsters" was protected in that these characters were "wholly fanciful creations, without reference to the real world."[102]

At some point, however, the screen display becomes too plain and ordinary to be distinguishable from the idea. The United States Court of Appeals for The Second Circuit in *Stern Electronics v. Kaufman*[103] realized the limitation of the audiovisual copyright protection:

> We need not decide at what point the repeating sequence of images would form too insubstantial a portion of an entire display to warrant a copyright, nor the somewhat related issue of whether a sequence of images (e.g., a spaceship shooting down an attacking plane) might con-

[96] 17 U.S.C. § 102(a) (1982).

[97] 17 U.S.C. § 101 (1982).

[98] *See* Atari, Inc. v. North American Philips Consumer Electronics Corp., 672 F.2d 607 (7th Cir.), *cert. denied,* 459 U.S. 880 (1982).

[99] 17 U.S.C. § 102(b) (1982); *see* Russo and Derwin, *Copyright in the "Look and Feel" of Computer Software,* COMPUTER LAW., Feb. 1985, at 5.

[100] *See* Williams Electronics, Inc. v. Artic Int'l, Inc., 685 F.2d 870 (3d Cir. 1982); Midway Mfg. Co. v. Strohon, 564 F. Supp. 741 (N.D. Ill. 1983); *see also* Stern Electronics, Inc. v. Kaufman, 523 F. Supp. 635 (E.D.N.Y. 1981), *aff'd,* 669 F.2d 852 (2d Cir. 1982) (establishing the basic validity of audiovisual copyright for videogame displays).

[101] 672 F.2d 607 (7th Cir.), *cert. denied,* 459 U.S. 880 (1982).

[102] *Id.* at 618.

[103] 669 F.2d 852 (2d Cir. 1982).

> tain so little in the way of particularized form of expression as to be only an abstract idea portrayed in a non-copyrightable form.[104]

The current judicial trend has shifted from allowing software manufacturers to use the audiovisual copyright to protect entertainment software involving fanciful creatures and characters to allowing this protection to extend to the user interface of productivity or commercial software containing little or no artistic or creative originality.

Audiovisual copyright may now protect the appearances, executable images, and input formats produced by the software on the monitor screens and the sequence of keystrokes used to manipulate information or desired functions by the user. This interaction between a user and a software program is often referred to as the "user interface" of a program. Some commentators equate the protection of a program's user interface with the protection of a program's "look and feel."[105]

Consider a hypothetical word processor. When a user wants to print a document, he might achieve this desired function by pressing the escape key and then selecting the appropriate "Print document" option from the "menu," which contains other options including "cOpy document," "Create document," "Revise document," and "Erase document." The items could be selected by pressing the highlighted letter on the keyboard. These options are all part of the program's user interface. Before examining the landmark case that first applied the audiovisual copyright to the protection of the user interface for productivity software, it would be helpful to look first at a pre-*Whelan* decision that addressed a similar problem.[106]

In *Synercom Technology, Inc. v. University Computing Co.*[107] the United States District Court for the Northern District of Texas considered whether the "sequence and ordering" of plaintiff's input formats used in a structural analysis program was protected expression or an unprotected idea. Synercom furnished its customers printed instructions describing the order in which data should be entered. Because Synercom's program was a commercial success and was the market leader, defendants tailored their program to accept input data in a sequence identical to that of plaintiff's program and provided its users with similar printed input instructions. The court held that the sequence and ordering of data for the input formats is inseparable from the idea underlying the formats

[104] *Id.* at 857; *see* Amici Curiae Brief Relating to Copyright Issues and Protection of Look and Feel of Computer Programs at 21, Broderbund Software, Inc. v. Unison World, Inc., 648 F. Supp. 1127 (N.D. Cal. 1986) (No. C85 3457).

[105] *See, e.g.*, Davidson, *The Whelan Decision: Missing the Middle Ground*, 5 COMPUTER L. REP. 335, 337 (1986); Letter by Henry Davis, *"Look and Feel" Isn't Everything in This Case*, BYTE, July 1986, at 22; Russo and Derwin, *Copyright in the "Look and Feel" of Computer Software*, COMPUTER LAW., Feb. 1985, at 1.

[106] Crucial for this discussion is the understanding that although the user interface is generated by the underlying code, two entirely different codes can yield an identical user interface. *See, e.g.*, *Whelan*, 797 F.2d at 1244 n.45 ("Different program codes in different computer languages are capable of producing identical screen outputs."). It is equally important to understand that two programs may look identical on the screen to the user, but the "guts" or code in one program may be superior to the other (*e.g.*, faster, more efficient, less compiled code, etc.).

[107] 462 F. Supp. 1003 (N.D. Tex. 1978).

and therefore could not receive protection.[108] As Judge Higginbotham questioned: "If sequencing and ordering is expression, what separable idea is expressed?"[109]

Although Synercom Technology based its complaint on a registered literary copyright of its printed instruction forms, its facts are analogous to the audiovisual copyright extended to plaintiff in *Broderbund Software, Inc. v. Unison World, Inc.*[110] The only difference in *Broderbund* was that the data entry instructions were displayed in menu screens generated by software. The input formats in *Synercom* and the menus in *Broderbund* served the same function: "[both] told the user what type of data to enter, where to place them and how to use it."[111]

Plaintiff Broderbund marketed a program for Apple computers called "Print Shop," which assisted the user in the creation of customized greeting cards, banners, posters, and signs that contain variable combinations of text, graphics, and borders. Defendant Unison World, a company that specialized in converting existing software to make it compatible with other computers, met with Broderbund about the possibility of writing an IBM-compatible version of Print Shop. Broderbund insisted that if Unison were to receive the right to convert Print Shop, it had to "produce an exact reproduction of the original."[112] Plaintiff "very briefly" showed defendant the source code of the Print Shop program, just so defendant could get a feel for the complexity of the project.[113] Six months later, Unison World began developing this specified imitation, but it did so without access to plaintiff's source code. After substantial progress on the written conversion of Print Shop, negotiations for the IBM rights broke down and Unison World instructed its programmers to complete the project, but no longer to confine themselves to producing an exact replica. Defendant's programmers enhanced their version of Print Shop by adding a calendar function, a streamlining of certain design selections, and the ability to save various designs. Unison World then marketed its program under the product trademark "Printmaster."

[108] The court in *Whelan* questioned the holding in *Synercom*: "To the extent that *Synercom* rested on the premise that there was a difference between the copyrightability of sequence and form in computer context and in any other context, we think that it is incorrect." *Whelan*, 797 F.2d. at 1240.

[109] *Synercom*, 462 F. Supp. at 1013.

[110] 648 F. Supp. 1127 (N.D. Cal. 1986). The court extended the copyright in the underlying code to its audiovisual screen displays: "copyright protection is not limited to the literal aspects of a computer program, but rather . . . extends to the overall structure of a program, including its audiovisual displays." *Id.* at 1133. Another federal court has rejected this approach by holding that the audiovisual screen displays are separate works and may be registered separately. *See* Digital Communications Assocs. v. Softklone Distributing Corp., 659 F. Supp. 449, 455 (N.D. Ga. 1987). To settle the issue, the Copyright Office is currently considering adopting one of various registration schemes. *See Notice of Public Hearing Registration and Deposit of Computer Screen Displays*, 52 Fed. Reg. 28,311 (1987) (Public Hearing September 9-10, 1987); Katchman, *Copyright Registration of Computer Screen Displays from the Perspective of the Copyright Office*, COMPUTER LAW., Oct. 1987, at 16.

[111] *Broderbund*, 648 F. Supp. at 1132.

[112] *Id.* at 1130.

[113] *Id.*

Because defendant did not have access to the Print Shop source code during the writing of its version of the program, plaintiff could not allege literary copyright infringement and instead charged that Unison World infringed Print Shop's audiovisual copyright. Specifically, plaintiff maintained that Unison World's version copied Print Shop's menu screens, input formats, and its sequencing of screens. Defendant argued that the *Synercom* rationale controlled and that these input formats and menu options contained expression that was indistinguishable from the idea expressed. However, the court was "persuaded by the reasoning of *Whelan*" and concluded that "the separable idea of 'Print Shop' is the creation of greeting cards, banners, posters and signs that contain infinitely variable combinations of text, graphics, and borders."[114] The Court concluded: "A rival software publisher is completely free to market a program with the same underlying idea, but it must express the idea through a substantially different structure."[115] Because Broderbund offered evidence of another competitor's program called "Stickybear Printer" which could produce similar banners and greeting cards but had different menu screens that accomplished essentially the same function as those of Print Shop, the court reasoned that "the existence of 'Stickybear Printer' proves that there do exist other, quite different ways of expressing the ideas embodied in 'Print Shop.' "[116]

One is no longer free "to make the machine do the same thing as it would if it had the copyrighted work placed in it" even if "one's own creative effort" is used.[117] It is unlikely that even the *Whelan* court would take its own words to the *Broderbund* extreme: "The rule proposed here, which allows copyright protection beyond the literal computer code, would provide the proper incentive for programmers by protecting their most valuable efforts, *while not giving them a stranglehold over the development of new computer devices that accomplish the same end*."[118]

Although some might argue that the existence of Stickybear Printer proves that there was not a stranglehold over the development of new computer devices that accomplish the same "general" end, the opinion seems to prohibit new programs that accomplish the exact or substantially similar end. Protecting a program's exact end result, the "look and feel" or executable image of the software, is potentially destructive to the competitive process and consumer welfare. American society has long benefited from having the first successful market entrant set the industry standard, with other competitors soon following suit.

To provide an example in a non-computer context, when people first learn to drive, they do not have to worry about whether the car is a Ford, Chrysler, or Chevrolet, because the gas pedal is always going to be on the right, and the brake will be on the left. The court in *Synercom* posed a similar analogy:

The familiar "figure-H" pattern of an automobile stick is chosen arbi-

[114] *Id.* at 1133.

[115] *Id.*

[116] *Id.* at 1132.

[117] CONTU REPORT, *supra* note 45, at 21.

[118] *Whelan*, 797 F.2d at 1237 (emphasis added).

> trarily by an auto manufacturer. Several different patterns may be imagined, some more convenient for the driver or easier to manufacture than others, but all representing possible configurations. The pattern chosen is arbitrary, but once chosen, it is the only pattern which will work in a particular model. The pattern (analogous to the computer "format") may be expressed in several different ways: by a prose description in a driver's manual, through a diagram, photograph, or driver training film, or otherwise. Each of these expressions may presumably be protected through copyright. But the copyright protects copying of the particular expressions of the pattern, and does not prohibit another manufacturer from marketing a car using the same pattern. Use of the same pattern might be socially desirable, as it would reduce the retraining of drivers.[119]

In much the same way, copyright should not protect a program's user interface. Businesses spend hundreds of millions of dollars each year training employees how to use word processors and spread sheets developed by the industry leaders. If a competitor develops another word processor or spread sheet which is offered at a lower price but which has a different appearance, and if the competitor attempts to enter the industry leader's market, a business in need of additional copies would be foolish to purchase this new software because it would have to retrain employees. The first developer can therefore keep prices high and continually reap economic profits without being subjected to competitive market forces. This constitutes an insurmountable barrier to entry and, in effect, creates a long-term, patent-like monopoly in the software.

Imitation of industry leaders is a means of survival for competitors in all industries. It now appears that a software company successful in setting an industry standard may be able to shield itself from any competition. Allowing programmers to copyright the user interface in this manner is akin to allowing the first developer of a car to exclude all others from placing a brake pedal to the left of the gas pedal in a car. As one commentator suggested, "The software developer needs the freedom to use the interface of another program, just as a watchmaker might borrow the concept of a watch's face while revolutionizing its clockworks."[120] Moreover, consumers will be forced to continue paying exorbitant prices for business software developed by the industry leaders.

Copying the clock's face was recently precluded in *Digital Communications Associates v. Softklone Distributing Corp.*[121] The facts in this case were undisputed. During the early 1980s the original plaintiff in this action, Microstuff, Inc., developed and marketed the "Crosstalk" asynchroneous data communica-

[119] *Synercom*, 462 F. Supp. at 1013. *Cf.* Plains Cotton Coop. Ass'n v. Goodpasture Computer Serv., 807 F.2d 1256, 1262 (5th Cir. 1987) (substantial similarity in organization and structure may be dictated by various market factors).

[120] Statement made by Daniel Bricklin, reported by David Sanger, *A Divisive Lotus 'Clone War'*, New York Times, Feb. 5, 1987, at 30.

[121] 659 F. Supp. 449 (N.D. Ga. 1987). Although the *Broderbund* court concluded that the user interface in question constituted an audiovisual work, the *Digital* court was of the opinion that the productivity user interface was a literary work. *Id.* at 462. The court further held that the elements of the menu screen were protected subject matter as a compilation. *Id.* at 463; *see infra* text accompanying note 147.

tion system.[122] The Crosstalk program, utilizing a microcomputer's modem and the telephone lines, enabled its users to access information and data stored in other microcomputers or in a remote mainframe computer. Although competitors market similar communication packages that accomplish the same general end, Crosstalk and its updates (or enhancements) have "been extremely successful in the marketplace."[123]

Defendant Softklone is a corporation created for the purpose of marketing and distributing "clone" computer programs.[124] In 1985 after observing Crosstalk's success, Softklone completed and began to market a program called "Mirror" that essentially emulated the user interface of Crosstalk.[125] Softklone did not use or copy plaintiff's source code, but programmed its version independently. The evidence at trial showed that Crosstalk XVI sold at retail for one hundred ninety-five dollars, while Mirror was sold initially for just under fifty dollars.[126]

As usual the court had to struggle with what elements in the menu screen constituted idea and what elements constituted expression. The court concluded that the cloned elements of the program that were nonessential to program operation constituted the expression of Crosstalk and, therefore, were infringements. For example, "first entering a 'SPeed' command and then entering a 'DAta' command [on the main menu] has the same effect as first entering a 'DAta' command and then entering a 'SPeed' command." The court determined that the particular way the menu items were arranged was nonessential and therefore protected.[127]

This trend will likely continue. Lotus Development Corporation has recently filed a lawsuit against Paperback Software International for allegedly "cloning" or "identically recreating," with only "cosmetic or superficial variations, the essential user interface of Lotus 1-2-3."[128] There is no allegation that Paperback copied the underlying code of 1-2-3, but only the program's familiar grid-like screen appearance and the sequence of keystrokes used to manipulate functions and information. Lotus 1-2-3 was first introduced in 1983 and has

[122] In 1986 Microstuff was purchased by Digital Communications Associates, Inc. *Id.* at 453.

[123] *Id.* at 452.

[124] *Id.* at 453.

[125] Softklone's parent, ForeTec Development Corporation, actually first coded and began marketing the Crosstalk clone. In August, 1985 ForeTec created Softklone, a wholly owned subsidiary, to market and distribute "clone" computer programs such as the one in dispute. *Id.* It is essential to note that Softklone did not have access to plaintiff's source code for Crosstalk. Softklone only had access to the executable image, or machine language, and therefore had to independently code its "Mirror."

[126] *See* Smith and Elgison, *DCA v. SoftKlone: The Continuing Saga of Copyright, Computers, and Clones*, COMPUTER LAW., April 1987, at 14.

[127] *Digital*, 659 F. Supp. at 459. For a more thorough discussion of "compilations" and their history, see *infra* text accompanying notes 134-46.

[128] Lotus Dev. Corp. v. Paperback Software Int'l, No. 87-0076-K. (D. Mass., filed January 12, 1987) [hereinafter Lotus Complaint]. Lotus Development Corp. is also seeking to preclude another competitor from marketing its Lotus clone called "The Twin." *See* Lotus Dev. Corp. v. Mosaic Software, Inc., No. 87-0074-K (D. Mass., amended complaint filed March 6, 1987) ("The Twin" is advertised as being "amazingly similar" to Lotus 1-2-3); Apple Computer, Inc. v. Microsoft Corp., (N.D. Cal., complaint filed March 17, 1988); *Apple's Copyright Lawsuit is Seen as Effort to Lock in Technical Lead,* Wall Street Journal, March 21, 1988, at 23 col.4.

since been near or at the top of best-seller lists for software. Because Paperback's version, called "VP Planner," carries a list price of only one hundred dollars, while 1-2-3 costs approximately four hundred ninety-five dollars, one can easily understand Lotus's concern. The user manual for "VP Planner" does not hide the fact that it has copied the industry standard set by 1-2-3, but instead emphasizes this fact in its marketing: "VP Planner works just like 1-2-3. The commands and the way you enter and manipulate your data are the same."[129] Given the current state of the law, Lotus Development Corporation will likely win its suit against Paperback Software, further eroding the competitive process in the computer software industry.

A potentially far-reaching change in the 1976 Copyright Act was the grant to an author of the exclusive right to make and to authorize any "derivative works based upon the copyrighted work."[130] A derivative work is defined as any "work based upon one or more preexisting works, such as . . . abridgment, condensation, or any other form in which a work may be recast, transformed, or adapted."[131]

This definition may imply that more than the literal aspects of a program's code is protected.[132] Although a work evidently is not derivative unless it has been substantially copied from a prior work,[133] it is difficult to imagine how any business or productivity software that uses similar menu screens and keystrokes could overcome such scrutiny. In *Lotus*, for example, it will be difficult for Paperback to argue that its "clone" is not "based upon" 1-2-3 and has not been "recast," "adapted," or "transformed" into its own version.

Lotus Development Corporation, the undisputed spread sheet marketing leader, has reaped millions in profits from its development of 1-2-3. Several years transpired before a competitor was able to market a program that accom-

[129] Lotus Complaint, *supra* note 128, at count I, para. 22.

[130] 17 U.S.C. § 106 (1982). Other "exclusive rights" in a copyrighted work include the right to copy, distribute, perform, and display the work. *Id.*

[131] *Id.* § 101.

[132] As was concluded in *Whelan*:

> Although the Code does not use the terms "sequence," "order" or "structure," it is clear from the definition of compilations and derivative works, and the protection afforded them, that Congress was aware of the fact that the sequencing and ordering of materials could be copyrighted, *i.e.*, that the sequence and order could be parts of the expression, not the idea, of a work.

Whelan, 797 F.2d at 1239. Compare the definition of "derivative work" with what Professor Nimmer calls "Comprehensive Nonliteral Similarity," 3 M. NIMMER, *supra* note 43, at § 13.03[A][1]. Generally, if the overall structure of the defendant's work has been copied, not just the literal aspects, the "substantial similarity" test will be satisfied under this approach. *See* Nichols v. Universal Pictures Corp., 45 F.2d 119 (2d Cir. 1930), *cert. denied*, 282 U.S. 902 (1931); *see also* Midway Mfg. Co. v. Artic Int'l, Inc., 704 F.2d 1009 (7th Cir.) (video game "speed-up" kit, which directs the characters to move in different ways, constitutes a derivative work), *cert. denied*, 464 U.S. 823 (1983). The extent of legal protection afforded derivative works, however, remains unclear in the computer software context. *See* Digital Communications Assocs. v. Softklone Distributing Corp., 659 F. Supp. 449, 463 (N.D. Ga. 1987) (court describes application of derivative work in computer screen context a "thorny issue"); Hazen, *Contract Principles as a Guide for Protecting Intellectual Property Rights in Computer Software: The Limits of Copyright Protection, The Evolving Concept of Derivative Works, and Proper Limits of Licensing Arrangements*, 20 U.C. DAVIS L. REV. 105, 108 (1986).

[133] *See* 3 M. NIMMER, *supra* note 43, § 3.01.

plishes virtually the same end result. In hindsight, the economic incentive that Lotus would need to create such a product has been realized for years. Although any further protection would result in economic waste, the current status of copyright law forces a court to side with Paperback or ignore the express wording of the copyright law.

b. Compilations and the Idea-Expression Inconsistency

Since the decision in *Baker v. Selden*,[134] legal scholars have been lulled into a false sense of security that the idea-expression analysis always leads to correct results. "Idea-expression"—what does it mean? The words themselves epitomize a subjective amorphousness that escapes precise definition. Until its application to the visual screens produced by computer software, idea-expression was usually a term used by courts to support their socially beneficial opinions.

If the goal is to achieve a socially optimal and competitive outcome, the idea-expression dichotomy is incompatible with computer software protection. As this section will demonstrate, the fundamental gap in copyright law is that the *Baker v. Selden* idea-expression dichotomy often fails in the essential purpose of intellectual property protection: to reward authors and inventors for a limited time in an amount commensurate with the value of their works to society. The Supreme Court in *Mazer v. Stein*[135] suggested the same economic goal:

> The economic philosophy behind the clause empowering Congress to grant [intellectual property protection] is the conviction that encouragement of individual effort by personal gain is the best way to advance public welfare through the talents of authors and inventors in "Science and useful Arts." Sacrificial days devoted to such creative activities deserve rewards commensurate with the services rendered.[136]

To understand the "incompatibility" conclusion reached by this Comment, one must trace the development of copyright law back to the time the idea-expression threshold test was first introduced to fact-based works.

At one end of the spectrum of protected works under the copyright law is a novel or play. The author has the incentive to write a novel because the value of her efforts are embodied in what copyright law clearly considers "expression." Consumers purchase Hemingway's works because they like to read the unique and descriptive stories or expressions that are embodied in the works. Therefore, protecting the expression directly compensates the author for the fruits of his labor and copyright idea-expression analysis works well.

At the other end of the spectrum are fact-based works such as maps, directories, and news and historical accounts. The true value embodied in a map, for example, comes from surveying the land and determining proper elevations and boundaries. Society benefits from the mapmaker's efforts and should therefore reward him accordingly. However, after this information is gathered, the result-

[134] 101 U.S. 99 (1879); *see supra* text accompanying notes 42-44.

[135] 347 U.S. 201 (1954).

[136] *Id.* at 219.

ing geographic dimensions become facts or ideas that, standing alone, do not merit protection under copyright law. In theory, others are free to take these facts and use them in their own maps, thus reaping the benefit of the first mapmaker's labor.[137]

Faced with the inevitable consequence that the exploration of new geographical areas would cease, United States Court of Appeals for the Third Circuit insinuated that copyright law requires "sweat of the brow": "[T]he presentation of information available to everybody, such as is found on maps, is protected only when the publisher of the map in question obtains originally some of that information by the sweat of his own brow" because "[a]most anybody could combine the information from several maps onto one map, but not everybody can go out and get that information originally and then transcribe it into a map."[138]

Although most courts have steadfastly adhered to the traditional principle that the amount of labor expended in producing a work in a copyright infringement action is irrelevant,[139] "sweat of the brow" reasoning is evident in cases that involve the protection of factual compilations.[140] Although courts first have had to struggle with how to extricate themselves from the copyright limitation that ideas are in the public domain, they have invariably denied copyrightability to works of free-riders. In *Adventures in Good Eating, Inc. v. Best Places to Eat, Inc.*,[141] for example, the United States Court of Appeals for The Seventh Circuit affirmed the validity of a copyright in a directory listing of fine restaurants.[142] Although the individual restaurant names were facts/ideas, the court explained, "[a]ny person may produce a directory of restaurants, but he must 'start from scratch,' do his own collecting, his own appraisal, and his own description and editing."[143]

[137] When the idea and expression are indistinguishable and merge, no copyright protection exists. *See, e.g.*, Herbert Rosenthal Jewelry Corp. v. Kalpakian, 446 F.2d 738, 742 (9th Cir. 1971) (denying copyright protection for the expression of jewel-encrusted "bee" pin because it could not distinguish the idea from the expression).

[138] Amsterdam v. Triangle Publications, Inc., 189 F.2d 104, 106 (3d Cir. 1951). In Donald v. Zack Meyer's T.V. Sales and Serv., 426 F.2d 1027 (5th Cir. 1970), the court held that a common legal form was not copyrightable because its creators expended only "negligible efforts." *Id.* at 1031. The court justified its holding under the copyright law by concluding that the works were not "original" works under section 102(a) of the Copyright Act. *Id.*; *see* 17 U.S.C. § 102(a). For in-depth discussions of copyright protection of facts, see Denicola, *Copyright in Collections of Facts: A Theory for the Protection of Nonfiction Literary Works*, 81 COLUM. L. REV. 516 (1981); Gorman, *Copyright Protection for the Collection and Representation of Facts*, 76 HARV. L. REV. 1569 (1963); *see also* United States v Hamilton, 583 F.2d 448, 452 (9th Cir. 1978) (synthesis from public sources and independent observation in creating a map supports the validity of a copyright on the map).

[139] *See* Rockford Map Publishers v. Directory Serv. Co., 768 F.2d 145, 148 (7th Cir. 1985) ("The copyright laws protect the work, not the amount of effort expended."); Eckes v. Card Prices Update, 736 F.2d 859, 862 (2d Cir. 1984) ("the fruits of another's labor in lieu of independent research obtained through the sweat of a researcher's brow, does not [usually] merit copyright protection . . . ."); Miller v. Universal City Studios, Inc., 650 F.2d 1365, 1369 (5th Cir. 1981).

[140] The telephone directory cases are illustrative of this labor theory approach. *See* Rand McNally & Co. v. Fleet Management Sys., 591 F. Supp. 726, 732 (N.D. Ill. 1983) (listing cases adhering to this reasoning).

[141] 131 F.2d 809 (7th Cir. 1942).

[142] *Id.* at 815.

[143] *Id.* at 812-13.

The economic reality is that consumers who buy maps, for example, are not paying solely for the unique colors and other methods of expression that the mapmaker employs in representing various elevations and boundaries, but are also rewarding the mapmaker for the ability, time, and expense exhausted to gather the facts. The other features, such as the expressions implemented, generally have only minimal significance when compared to the value of the whole work. People do not pay just for the expression—they are paying primarily for the facts, the ideas.

If society only offered protection for the expression embodied in the map, the mapmaker would be undercompensated for the fruits of his labor and society would be discouraging the optimal level of geographical explorations. If geographical explorations and other fact-gathering is curtailed, then society loses. This analysis leads to only two conclusions, both of which point to the failure of the idea-expression analysis in light of our economic objectives: either 1) it fails from an economic perspective because we are not rewarding fact-gatherers in an amount commensurate to the entire value of their efforts to society and thus are discouraging their efforts (society is only rewarding expression); or 2) it fails because we are rewarding fact-gatherers for the entire value of the work (which is embodied in both the idea and expression).

Courts generally circumvent the idea-expression dichotomy and thereby reward fact-gatherers by protecting, as expression, the unique arrangement and order that has been implemented by the gatherer to display the facts on a printed page.[144] Here lies the fundamental downfall in software cases: merely protecting the arrangements only indirectly compensates the gatherer of the information. Refusing to expressly reward the time and effort expended in gathering the facts leads to an improper outcome when applied to the output of software.

The 1976 Copyright Act embodies this fact-based/idea-expression analysis by including "compilations" within its purview.[145] A compilation is "a work formed by the collection and assembling of preexisting materials or of data that are selected, coordinated, or arranged in such a way that the resulting work as a whole constitutes an original work of authorship."[146]

The *Digital* court correctly concluded that the items on the Crosstalk "Main Menu" status screen were arranged so as to constitute an "original work of authorship."[147] Most business/productivity related computer displays or menus, including that of Lotus 1-2-3, satisfy the definition of compilation under the copyright law. Accordingly, although giving competitors the right to inde-

[144] *See* West Publishing Co. v. Mead Data Central, Inc., 799 F.2d 1219, 1227 (8th Cir. 1986) (LEXIS's appropriation of page references from West's legal case reports would infringe West's copyright in arrangement).

[145] 17 U.S.C. § 103 (1982).

[146] 17 U.S.C. § 101 (1982).

[147] *See Digital,* 659 F. Supp. at 463 ("[T]he status screen is a 'compilation' of parameter/command terms."). "Originality" has been interpreted to be a very minimal threshold requirement to copyrightability: "The standard for 'originality' is minimal. It is not necessary that the work be novel or unique, but only that the work have its origin with the author—that it be independently created." *West Publishing,* 799 F.2d at 1223.

pendently copy the look and feel of productivity software is socially desirable, doing so would contravene a correct application of copyright law.

The inherent difference between compilations in computer screen displays and traditional fact-based infringements against maps or directories is that for a programmer to implement the menu or visual display of a market leader's product without copying the underlying code, he must accomplish this task by coding independently, by the "sweat of the brow." In other words, the defendant in *Digital* went out and surveyed the land but was denied the right to make the map. Or, using a previous illustration, the defendant in *Digital* independently developed the internal mechanics of a clock by "the sweat of the brow" but was denied the right to copy the clock's face. Such a result would not have occurred with traditional fact-based work cases. As the court in *Rand McNally & Co. v. Fleet Management Systems*[148] explained after discussing past compilation cases in a non-software context: "None of the above cases disputes one's right to compile a list of facts through one's own labor."[149]

In search of a copyright solution, one might argue that similar productivity software is fair use and falls under the express limitation on the exclusive rights under section 107.[150] Applying the *Lotus* facts, however, it is disingenuous to argue that Paperback copied the user interface of 1-2-3 for "purposes such as criticism, comment, news reporting, teaching, . . . scholarship, or research."[151] To the contrary, Lotus 1-2-3 was copied because it was the spreadsheet leader, the state of the art in business software, and was reaping handsome economic profits. Paperback's purposes included economic commercial gain and all the other motivations commonly associated with the dynamic market structure. Furthermore, the relative low price of "VP Planner" would have the effect of driving down the price of 1-2-3, thus diminishing its commercial value. Although Paperback's motivations may not seem as altruistic as teaching, schol-

[148] 591 F. Supp. 726 (N.D. Ill. 1983).

[149] *Id.* at 733.

[150] 17 U.S.C. § 107 (1982). The factors to be considered for determining whether the infringement constitutes fair use are as follows:

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

Id. Probably the most famous fair use defense was used in Sony Corp. of Am. v. Universal City Studios, Inc., 464 U.S. 417 (1984). Universal, owners of copyrights on some of the television programs that are broadcast on the public airwaves, claimed that Sony's video cassette recorders (VCRs), which enabled owners to record the programs, assisted users in copyright infringement. *Id.* at 420. The Court held that this use constituted fair use because the average user of the public uses VCRs for "time shifting" purposes and such usage would not impair the commercial value of plaintiff's copyrights. *Id.* at 456; *see also* Harper & Row Publishers v. Nation Enterprises, 471 U.S. 539 (1985) (when defendant's copy undermines or diminishes the value of the copyrighted work, fair use is not a defense).

A parody may constitute fair use, but one can only take from the copyrighted work enough to conjure up the original. *See* Elsmere Music, Inc. v. National Broadcasting Co., 482 F. Supp. 741, 747 (S.D.N.Y), *aff'd*, 623 F.2d 252 (2d Cir. 1980); Columbia Pictures Corp. v. National Broadcasting Co., 137 F. Supp. 348, 354 (S.D. Cal. 1955).

[151] 17 U.S.C. § 107 (1982).

arship, or research, its function as a competitor is just as important to our society. Nevertheless, section 107 and its fair use defense are not the solution.

C. *Summary*

The *Whelan* and *SAS* courts battled with copyright semantics to reach equitable results in software cases involving alleged infringements of the underlying program code. In doing so the courts distorted copyright law by crossing the idea-expression threshold.

Recent cases involve infringement actions against competitors who have copied the user interface. Although *Broderbund*'s application of the audiovisual copyright to nonfanciful, productivity visual displays is questionable, like protection is certainly available as a compilation (or possibly as a derivative work) under the copyright law. As has been shown, however, a correct application of the copyright law in computer software compilation cases leads to anticompetitive results. All these decisions point to the incompatibility of copyright protection for computer software.

The current trend in computer software disputes is to provide protection that far exceeds that which is needed to stimulate innovation and creativity.[152] Furthermore, the increased protection sidesteps constitutional guidelines in that such a plan of protection discourages rather than promotes the "Progress of Science and useful Arts."[153]

II. A New Scheme of Protection

> Although the application of our intellectual property rules has been adjusted over time in response to changing commercial practice and evolving technologies, the continuing stream of the new scientific advances calls for rethinking the very concepts derived from earlier centuries on which those rules are based. New concepts of what intellectual property is and how it should be protected—beyond patents, trademarks, trade secrets, and copyrights—may well be needed, as well as sweeping changes in intellectual property laws and the way they are administered and enforced.[154]

This statement, made in a report by The President's Commission of Industrial Competitiveness, reflects a growing awareness that copyright law is incompatible with computer software protection. Technology has outgrown the law that was first developed for pure literary works. The direction software development is taking indicates that copyright application will become even more troublesome in the future as "artificial intelligence" becomes commonplace in our

[152] As stated by CONTU, "[C]opyright should not grant anyone more economic power than is necessary to achieve the incentive to create." CONTU REPORT, *supra* note 45, at 12.

[153] U.S. CONST., art. I, § 8, cl. 8.

[154] OTA REPORT, *supra* note 91, at 61 (quoting GLOBAL COMPETITION: THE NEW REALITY, The President's Commission on Industrial Competitiveness, January 1985 Appendix D, "A Special Report on the Protection of Intellectual Property Rights," at 305).

society.[155] This new generation of software is designed to make the computer mimic the human brain. Computers will be able to interpret the visual environment, drive cars, and will even be able to learn. Computers are actually becoming intelligent. Programs are being written that, in turn, will independently generate new programs. Within the next decade programs will be able to write poetry. Can machines be "authors" under the copyright law? Are machine produced expressions protected?

This Comment has demonstrated that even a correct application of copyright law leads to anticompetitive results. Although it may be true that rejection of the idea-expression dichotomy could lead to protection for ideas, the result would not necessarily be objectionable. Essentially, the approach offered in this Comment is the same as that which drives a free-market economy: if the underlying idea of the software is useful to society, then that idea will be insulated by the market—but only for a limited time. This exclusive monopoly duration will be equal to the time it takes the idea to become successful in the marketplace, together with the time it takes another person to implement the same idea (commonly referred to as the "lead time").[156]

Consider the situation in *Lotus* and *Digital.* If Lotus 1-2-3 and Crosstalk had not been based on ideas that were useful to society, then the products would not have been successful in the marketplace and the ideas, although still protected, would be in a state of perpetual uselessness. However, because the products were based on ideas that were particularly useful to society, as evidenced by handsome profits, they have been protected by an exclusive monopoly for a duration equal to the time it takes a competitor to implement the same end-result. Once competitors drive the price of the successful software down to the cost of production, monopoly power ends.

There is no need to grant a fixed monopoly of fifty years plus the life of the author to provide the incentive needed to stimulate creative and inventive software in the marketplace.[157] The copyright system as applied to computer software is inherently foolish because it protects all works for the same duration without focusing on the usefulness to society. The free market provides a perfectly variable system that grants a monopoly according to the product's usefulness. If a product is extremely useful, the monopoly may be short-lived but the economic reward during this period will be great.

Congress should dismantle copyright protection of computer software in favor of a marketplace approach. Students of basic microeconomics will recognize that the system described herein is essentially the same one that drives all unregulated industries: Adam Smith's "invisible hand." If property rights are adequately defined, protected, and transferable, the invisible hand will ensure

[155] *See generally* Nycum and Fong, *Artificial Intelligence And Certain Resulting Legal Issues,* COMPUTER LAW., May 1985, at 1.

[156] *See generally* Breyer, *The Uneasy Case For Copyright: A Study of Copyright in Books, Photocopies, and Computer Programs,* 84 HARV. L. REV. 281, 300 (1970) (lead time provides protections for the original authors of books which partially obviates the need for copyright protection).

[157] For any work created on or after Jan. 1, 1978, the copyright monopoly is for 50 years after the life of the author or 75 years for an "anonymous" work. 17 U.S.C. § 302 (1982).

that resources flow to their most valuable uses.[158]

Although copyright's long term monopoly would be taken away, it is essential that software be adequately protected. Judge Posner illustrates the problems that a society would face in the absence of property rights:

> A farmer plants corn, fertilizes it, and erects scarecrows, but when the corn is ripe his neighbor reaps and sells it. The farmer has no legal remedy against his neighbor's conduct since he owns neither the land that he sowed nor the crop. After a few such incidents the cultivation of land will be abandoned and the society will shift to methods of subsistence (such as hunting) that involve less preparatory investment.[159]

Although software is less tangible than the real property described in the above hypothetical, providing exclusive rights is equally essential for all types of property. Without protection, software developers would be unwilling to invest their time and money and start-up companies would find it impossible to acquire venture capital. The protection needs to be both certain and predictable.

The remainder of this Comment proposes a marketplace system of protection for software and discusses how it would be implemented.[160] Generally, the scheme relies on contract law, the same law that is implemented to drive our market economy. Trade secret and trademark law provide additional protection. Further, it employs a federal misappropriation law that serves to protect a first developer's legitimate lead time.

A. *Contract, Trademark, and Trade Secret*[161]

The software industry is already taking the marketplace initiative by pro-

[158] R. POSNER, ECONOMIC ANALYSIS OF LAW 29 (2d ed. 1977).

[159] *Id.* at 27.

[160] Although copyright law has undergone little dynamic alteration, Congress has made subtle changes to respond to problems in copyright law when applied to technology subject matter. See Semiconductor Chip Protection Act of 1984, Pub. L. No. 98-620, § 302, 98 Stat. 3335, 3347 (codified at 17 U.S.C. §§ 901-914 (Supp. IV 1986)); Note, *A Chip Off the Old Block: Copyright Law and the Semiconductor Chip Protection Act,* 7 COMPUTER L.J. 245, 255-58 (1986).

[161] This Comment does not examine closely the various contractual, trademark, and trade secret methods that can be utilized for the protection of computer software. Other writers have thoroughly commented on the effectiveness, validity, and methods of implementation of these forms of protection.

For a discussion of the protection of software under contract law, *see* M. SCOTT, *supra* note 36, at §§ 5.32-6.124; R. RAYSMAN & P. BROWN, *supra* note 30, at §§ 7.01-.13A; S. MANDELL, *supra* note 22, at 8-16; Hazen, *supra* note 132.

For an overview of the protection of software under trademark law, see Trademark Act of 1946 (Lanham Act), *as amended,* 15 U.S.C.A. §§ 1051-1127 (1982) (§ 1120 provides that any person damaged by the use of any false description or representation has a cause of action); J. LATSCH, AMERICAN STANDARD HANDBOOK OF SOFTWARE LAW, at 197-216 (1985); J. MCCARTHY, TRADEMARKS AND UNFAIR COMPETITION (1973).

For a general discussion of trade secret protection of software, *see* 12A R. MILGRIM, BUSINESS ORGANIZATIONS, R. MILGRIM ON TRADE SECRETS §§ 2.06A[5], 2.09[5] (1981); OTA REPORT, *supra* note 91, at 87-88; R. RAYSMAN & P. BROWN, *supra* note 30, at §§ 6.01-.05A; M. SCOTT, *supra* note 36, at §§ 5.1-.28; Bender, *Protection of Computer Programs: The Copyright/Trade Secret Interface,* 47 PITT. L. REV. 907 (1986); Bender, *Trade Secret Protection of Software,* 38 GEO. WASH. L. REV. 909 (1970); Conley & Bryan, *supra* note 53, at 573-77.

tecting computer programs through contractual agreements with consumers.[162] Discouraged by the uncertainty of copyright protection, software producers ensure protection for their creations through contracts that explicitly identify the rights of a purchaser or licensee.

Most software today is distributed under contract lease or license arrangements in which the user does not buy all ownership rights of the program, but purchases only the right to use it.[163] In smaller, more specialized markets in which software is generally more expensive, it is cost effective for the software producer and users to negotiate contractual terms on an individual basis. Problems arise, however, in the mass marketing of lower priced software. Negotiating individual contracts would be cost prohibitive given the large number of purchasers and the low price. Vendors of mass-market programs therefore have relied on "shrink-wrap" licenses: the vendor displays the terms of the licensing contract on the cover of the package containing the disk and the user is theoretically bound by its terms if the package is opened.[164] Because these agreements are not signed, their enforceability is unclear.[165] Realizing that software needs to be both protected and freely transferable, some states have passed statutes that specifically validate the use of these agreements.[166]

Trademarks are devices used by a producer to identify her goods, thereby distinguishing them from goods sold by others. Trademark protection becomes extremely crucial if competitors are allowed to independently code programs that are clones of the market leaders. Programs that may duplicate the user interface of Lotus 1-2-3, for example, may be inferior in terms of speed of execution or size of compiled code. Trademarks enable consumers who want to pay for these features to distinguish between the products. Trademarks, servicemarks, and trade names also eliminate some information costs because consumers often know what kind of quality and service to expect from particular vendors.[167]

A developer also may protect software as a trade secret if the software is

[162] *See* OTA REPORT, *supra* note 91, at 89. Fifty-four percent of the public favors the marketplace approach to resolving problems associated with intellectual property protection. *Id.*

[163] Copyright law protects against making copies of software. It does not, however, prohibit use. Contract arrangements have therefore been essential to elucidate the rights that a user purchases and the rights that a developer retains.

[164] Also commonly referred to as "tear-open" or "blister pack" licensing agreements.

[165] *See* Note, *Tear-Me-Open Software License Agreements: A Uniform Commercial Code Perspective on an Innovative Contract of Adhesion,* 7 COMPUTER L.J. 261, 270-71 (1986); Note, *The Protection of Computer Software Through Shrink-Wrap License Agreements,* 42 WASH. & LEE L. REV. 1347 (1985); *see also* Guilbert, *Louisiana Software License Enforcement Act Under the Scrutiny of a Louisiana District Court: What Impact on Shrink Wrap License Agreements?* 5 COMPUTER L. REP. 840, 843 (1987) (at common law or under the Uniform Commercial Code, beginning performance may be deemed acceptance of the terms of the contract).

[166] *See* Illinois Software License Enforcement Act, Ill. Rev. Stat. CL 29 Par. 801-808 (1986); Louisiana Software License Enforcement Act, LA. REV. STAT. ANN. §§ 51:1961-1966 (West 1984). *But see* Vault Corp. v. Quaid Software Ltd., 655 F. Supp. 750, 762 (E.D. La. 1987) (court held that Louisiana's statute was preempted by federal copyright law, 17 U.S.C. § 301).

[167] Critics also recognize the disadvantages of trademark protection. Primarily, it encourages excessive advertising expenditures to develop brand loyalty. Strong brand loyalty can be a formidable barrier to entry. For an analysis of the role and effectiveness of trademarks in our economy, see Backman, *The Role of Trademarks in Our Competitive Economy,* 58 TRADEMARK REP. 219 (1968).

used in a business and it gives that business an opportunity to obtain an advantage over competitors who do not know or use the software.[168] To succeed on the merits of a trade secret claim, plaintiff must show that (1) it possessed a trade secret[169] and (2) defendant used that trade secret in breach of an agreement, confidence, or duty, or as a result of discovery by improper means.[170]

Trade secret law is a creature of state statutory and common law.[171] Accordingly, state law defines what a trade secret is and what rights it affords.[172] Although others can develop the same trade secret, they must not do so with the aid of an unauthorized disclosure nor by obtaining the secret through improper means.

B. *Federal Misappropriation*

Federal misappropriation should be used only as a last resort. Circumstances may arise, however, that will escape the grasp of contract, trademark, and trade secret law even though the equities will demand relief. Evaluations of which party is "good" and which party is "bad" are embedded in the dictum of many opinions. Courts have also been troubled when a competitor seeks to reap the benefit of another's labor without expending similar effort. A federal law of misappropriation would explicitly recognize some of these concerns. The purpose of this law is twofold. First, it provides a federal forum; second, and more important, it protects the legitimate lead time into the marketplace of a first developer.

Federal misappropriation would be rooted in a well-established common law extension of unfair competition law that protects against unauthorized taking of valuable intangible assets that do not otherwise enjoy protection under copyright, patent, or trademark law.[173] This doctrine, known under the relatively vague label of "common law misappropriation," allows a court to grant equitable relief without creating a broad monopoly.[174]

Common law misappropriation was first pronounced by the United States

[168] RESTATEMENT OF TORTS § 757 comment b (1939).

[169] The factors to be considered in determining whether claimant has a trade secret include the following:

> (1) the extent to which the information is known outside of his business; (2) the extent to which it is known by employees and others involved in his business; (3) the extent of measures taken by him to guard the secrecy of the information; (4) the value of the information to him and to his competitors; (5) the amount of effort or money expended by him in developing the information; and (6) the ease or difficulty with which the information could be properly acquired or duplicated by others.

Id.

[170] RESTATEMENT OF TORTS § 757 (1937).

[171] *See, e.g.*, N.C. GEN. STAT. § 66-152 (1985).

[172] The United States Supreme Court held that patent law does not preempt states' trade secret law. *See* Kewanee Oil Co. v. Bicron Corp., 416 U.S. 470 (1974).

[173] *See* United States Golf Ass'n v. St. Andrews Sys., Data-Max, Inc., 749 F.2d 1028, 1034-35 (3d Cir. 1984).

[174] Reback & Hayes, *Copyright Gone Astray: The Misappropriation Alternative*, COMPUTER LAW., April 1986, at 3 (suggesting common law misappropriation doctrine is a desirable way to deal with intellectual property not falling under the copyright act). *See generally* Franklin, *Misappropriation Law, Copyright Law, and Product Compatibility*, COMPUTER LAW., Feb. 1987, at 8.

Supreme Court in *International News Service v. Associated Press.*[175] The Associated Press (AP) employed reporters who gathered and prepared news to be printed in member newspapers and posted on bulletin boards for the public to read. Defendant International News Service (INS) copied these stories from the bulletin boards and from early editions of AP's newspapers and sold them after a careful revision of the expression. Although AP expended all of the effort and absorbed all the cost of gathering and investigating the "hot" news items, INS was often able to take these AP stories directly from bulletin boards on the east coast and publish them on the west coast ahead of AP's publication there. The Court recognized that this practice did not fall under copyright law which was not "intended to confer upon one who might happen to be the first to report a historic event the exclusive right for any period to spread the knowledge of it."[176] Nevertheless, the Court refused to let such an unfair practice continue:

> [The] defendant, by its very act, admits that it is taking material that has been acquired by complainant as the result of organization and the expenditure of labor, skill, and money, and which is salable by complainant for money, and that defendant in appropriating it and selling it as its own is endeavoring to *reap where it has not sown,* and by disposing of it to newspapers that are competitors of complainant's members is appropriating to itself the harvest of those who have sown. . . . [A] court of equity ought not to hesitate long in characterizing it as unfair competition in business.[177]

Because the misappropriation doctrine was first promulgated by the Supreme Court in a diversity case and its decision was based on federal common law, with the subsequent decision in *Erie Railroad Co. v. Tompkins*[178] the doctrine has lost its binding authority. Nevertheless, common law misappropriation still has significance in state courts and is considered a branch of unfair competition law.[179] The misappropriation doctrine has been substantially limited by cases recognizing that when the states provide a broader scope of protection

[175] 248 U.S. 215 (1918). *See* 2 L. ALTMAN & R. CALLMANN, UNFAIR COMPETITION, TRADEMARKS, AND MONOPOLIES Ch.15 (1981); J. MCCARTHY, *supra* note 161, at §§ 10.23-.35; Callmann, *He Who Reaps Where He Has Not Sown: Unjust Enrichment in the Law of Unfair Competition,* 55 HARV. L. REV. 595 (1942); Sell, *The Doctrine of Misappropriation in Unfair Competition,* 11 VAND. L. REV. 483 (1958); Note, *Unfair Competition: Application to News Service,* 4 CORNELL L.Q. 223 (1919); Note, *Property in News,* 32 HARV. L. REV. 566 (1919); Note, *Unfair Competition,* 17 MICH. L. REV. 490 (1919); Note, *Misappropriation as Unfair Competition,* 6 SYRACUSE L. REV. 317 (1955); Note, 28 YALE L.J. 387 (1919).

[176] *International News Service,* 248 U.S. at 234.

[177] *Id.* at 239-40 (emphasis added).

[178] 304 U.S. 64 (1938).

[179] Misappropriation has been an actionable ground for relief in the state courts in various noncomputer contexts. *See, e.g.,* Standard & Poor's Corp. v. Commodity Exchange, Inc., 683 F.2d 704 (2d Cir. 1982) (applying New York law and holding that the use of Standard & Poor's 500 Index for futures contracts is prohibited); Columbia Broadcasting Sys. v. Melody Recordings, Inc., 134 N.J. Super. 368, 341 A.2d 348 (1975) (concerning record piracy which is now covered by copyright statute); Pottstown Daily News Publishing Co. v. Pottstown Broadcasting Co., 411 Pa. 383, 192 A.2d 657 (1963) (news context); Southwestern Broadcasting Co. v. Oil Center Broadcasting Co., 210 S.W.2d 230 (Tex. Civ. App. 1947) (broadcast context).

In computer contexts a state law misappropriation claim has been held preempted by federal copyright law. *See* Videotronics, Inc. v. Bend Electronics, 564 F. Supp. 1471 (D. Nev. 1983); Synercom Technology, Inc. v. University Computing Co., 474 F. Supp. 37 (N.D. Tex. 1979).

than an applicable federal statute, the federal law must preempt the state law under the supremacy clause of the Constitution.[180]

A 'federal misappropriation statute would alleviate this preemption problem. Such a federal statute should be enacted, specifically taking computer software out of the copyright domain and placing it in the realm of federal misappropriation. A claim for federal misappropriation would consist of three elements:[181] 1) the coding of a valuable software program through extensive effort and skill and with the expectation of profit; 2) the appropriation of that software by the defendant at little or no cost, thereby gaining an unfair competitive advantage in a way in which the court can characterize defendant's action as "reaping where it has not sown"; and 3) economic harm to the owner of the original software directly caused by the wrongful appropriation.

In addition, the statute would implement the following mandatory guidelines:

> 1) A showing of enough similarity between two works to constitute federal misappropriation is a difficult burden for plaintiff. A claim that defendant copied or translated the plaintiff's program line-by-line (of the program's entire code, of a substantial amount of the entire code, or of an important procedure or function) either in the object code level, the assembler level, or at a higher level, is a clear showing of misappropriation, absent some contract arrangement.
>
> 2) Implicit in every finding of misappropriation is a recognition that defendant's actions would likely abate the incentive for the injured party and others in a similar position to engage in similar software development.
>
> 3) One is free to copy the "look and feel" (the user interface) of productivity software. The executable image of entertainment software, like the "PAC-MAN" game, also can be copied. However, fanciful and artistic characteristics such as PAC-MAN's "gobbler" may find protection under this section;
>
> 4) Every software developer is under a stringent duty to protect software through contract. Any intentional distribution of software unaccompanied by a contract identifying the rights and obligations of the parties will be deemed a waiver of rights under the act.
>
> 5) Federal misappropriation legislation does not preempt state trade secret claims nor claims falling under state trademark law.

Legislated federal law of misappropriation would preempt state misappropriation and would thereby provide a unified approach currently provided by the copyright act but would do so without granting unnecessary monopolies that stifle competition.[182] Paragraph four ensures that federal misappropriation is

[180] U.S. CONST., art. VI, § 2; *see* Sears, Roebuck & Co. v. Stiffel Co., 376 U.S. 225 (1964); Compco Corp. v. Day-Brite Lighting, Inc., 376 U.S. 234 (1964); *cf.* Goldstein v. California, 412 U.S. 546 (1973) (states free to provide protection for subject matter that does not come within the specific federal statutory provisions) (holding later codified in section 301 of the Copyright Act).

[181] *See* J. MCCARTHY, *supra* note 161, § 10:25, at 322; Hazen, *supra* note 132, at 137 (listing basic elements considered in a state misappropriation claim); Reback & Hayes, *supra* note 174, at 3.

[182] When the dispute involves a situation that is inherently federal, the courts can assert federal

not merely a substitute for copyright law because it will encourage the parties to individually define property rights by way of contract. The emphasis of protection would shift to the marketplace, but there would be a remedy against third parties who shorten a first developer's lead time by unfairly competing. Further, this is not common law misappropriation. Federal misappropriation has a similar rationale and goal, but this law is narrowly tailored for a specific subject matter.

Courts would no longer be bound by precedent established for a completely different type of intellectual property and would no longer need to participate in semantic maneuvering to render equitable decisions. The implementation of this federal scheme would also enable computer illiterate courts to rely less on determination of what a program's organization and structure entails, and would shift their focus to areas in which they are versed; for example, the determination of whether or not a contract term has been breached or whether the parties exhibited good faith and fair dealing.

C. *An Application*

It may be helpful to an understanding of this new scheme of protection to apply it to several of the core cases discussed above. *Apple Computer, Inc. v. Franklin Computer Corp.*[183] represents a clear case for application of federal misappropriation. Franklin sought to compete with Apple by copying the underlying code of Apple's operating system programs. The programs sold by Franklin implemented code that was identical to that employed in the Apple software.[184] One expert testified that the Franklin code was "unquestionably copied from Apple and could not have been independently created."[185] Franklin did not even dispute that it copied the Apple programs. Instead, it attempted to cover its misappropriation with the copyright law. The bulk of the opinion dealt with whether a functional operating system program can receive copyright protection and whether a program stored in ROM could be protected. The end result, however, was that Franklin clearly reaped where it had not sown and as a result thwarted Apple's lead time in the marketplace.

Although the amount of labor employed is not an element of copyright infringement, the *Apple* court was rightly concerned that Franklin was attempting to benefit from the fruits of Apple's labor: "Apple estimated the 'works in suit' took 46 man-months to produce at a cost of over $740,000, not including

common law authority. Illinois v. City of Milwaukee, Wisconsin, 406 U.S. 91, 103 (1972) (" 'It is not uncommon for federal courts to fashion federal law where federal rights are concerned.' " (quoting Textile Workers v. Lincoln Mills, 353 U.S. 448, 457 (1956)).

Compare this approach to the federal scheme that governs the clean up of hazardous waste. This scheme includes two federal statutes and the application of traditional and evolving federal common law principles. *See* United States v. Chem-Dyne Corp., 572 F. Supp. 802, 810 (S.D. Ohio 1983); The Resource Conservation Recovery Act of 1976, 42 U.S.C §§ 6901-6987 (1982); The Comprehensive Environmental Response, Compensation, and Liability Act of 1980 (CERCLA), 42 U.S.C. §§ 9601-9657 (1982).

[183] 714 F.2d 1240 (3rd Cir. 1983), *cert. dismissed*, 464 U.S. 1033 (1984).

[184] *Id.* at 1245.

[185] *Id.*

the time or cost of creating or acquiring earlier versions of the programs or the expense of marketing the programs."[186] Franklin would not have been precluded from legitimately competing with Apple by independently coding its own version of Apple's operating system that accomplished the same end result.

The "look and feel" cases are on the other end of the legitimate competition spectrum. Defendants in *Broderbund, Digital,* and *Lotus* did not copy plaintiffs' underlying code nor did they have access to the source;[187] instead, they "sweated at the brow" by independently coding programs that accomplished the same end. This represents legitimate competition. In these cases defendants never would have sought to compete unless plaintiffs' products were successful and profitable. The market provided each plaintiff with a monopoly in both the idea and the expression of the programs for a duration equal to the time it took each product to become successful and the time it took defendants to independently recreate plaintiffs' user interfaces. With all these software products the monopoly duration without copyright amounted to several years. More importantly, however, plaintiffs were handsomely rewarded for their creative endeavors.

Before applying this new scheme to the cases of *SAS* and *Whelan,* an understanding of how programs are marketed would be helpful. Most software is distributed in the object code form (or executable image). If a user purchases Lotus 1-2-3, for example, all that is received is the object code, not the source code. Source code is only needed to modify, change, or enhance a program. There is no excuse for allowing a nonowner of software to obtain, have access to, or even see the program's source code without appropriate individual licensing arrangements, nondisclosure agreements, or other contractual restrictions. When only the object code is distributed competitors will not be able to curtail substantially a first developer's lead time without making extensive verbatim copies or translations.[188]

[186] *Id.*

[187] The defendant in *Broderbund* was "briefly" shown the source code of the "Print Shop" program so defendant could get a feel for the size and complexity of the project. Defendant, however, did not have access to the source code while programming its version. *See Broderbund,* 648 F. Supp. at 1130.

[188] Coincidence does not explain why the plaintiffs in software cases that alleged copyright infringement of the organization and structure (as opposed to only verbatim copying) of the code involved a defendant that had access to plaintiffs' source codes. *Compare Apple,* 714 F.2d 1240 (3d Cir. 1983) (extensive verbatim copying but defendant only had access to object code), *cert. dismissed,* 464 U.S. 1033 (1984) *with Whelan,* 797 F.2d 1222 (3d Cir. 1986) (only copying of organization and structure alleged but defendant had access to source), *cert. denied,* 107 S. Ct. 877 (1987). If a competitor only has access to the object code of a program, technical problems generally make it much easier for her to write a new version to accomplish the same end result than to rely on the organization and structure of the first developer's object code—unless verbatim copying occurs. *See* Appendix.

An exaggerated fear exists in the legal community that recipients of the object code will reverse engineer (or disassemble) the code, modify the resulting assembler code, reconstruct it into a high level source, compile the source and sell the new object code, all at little or no cost relative to the labor and expense employed by the original programmer. This belief is a fallacy. In reality, it is an extremely difficult, laborious, and time-consuming task to take assembler code that has been disassembled from object code and to reconstruct it into its original or similar high level source code. The disassembled version of object code loses the original comments, the variable names, and the program's logical groupings (or structure indentation). All are important to the understanding of the

Whenever source code is distributed an alarm should sound. In the absence of contractual provisions defining the licensee's use of the code, a first developer will be without a remedy. In both *SAS* and *Whelan* defendants had access to plaintiffs' source code. Plaintiff in *SAS* properly negotiated contractual restrictions in the licensing agreement before distributing the source. The license agreement permitted defendant to modify plaintiff's program, but only for its own use.[189] It further provided that "no redistribution of such an updated work was allowed."[190] The *SAS* case could have easily been decided under contract law. As the court noted, "This case primarily involves a license agreement between the [plaintiff] and [defendant]."[191] Defendant S & H breached an express term of the licensing agreement.

In *Whelan* defendant also obtained access to plaintiff's source code, which he used extensively in developing a microcomputer version of plaintiff's program,[192] but plaintiff had no contractual protection. Under the parties' only contractual arrangement, defendant Jaslow agreed to use its "best efforts to act diligently in the marketing of the Dentalab package," and plaintiff Whelan agreed to use its "best efforts and to act diligently to improve and augment the previously successfully designed Dentalab package."[193] Marketing software does not require having access to the source code, but only the object code. Under a marketplace analysis the burden is on the software owner to have ap-

function of a particular section of code. Disassembly of object code is often accomplished for purposes such as 1) overriding copy protection or encryption schemes; 2) gaining a glimpse at a particular file structure so that the new program could interface with an existing program; or 3) getting a look at the "idea" behind a particular procedure or function and then subsequently writing one's own code to accomplish the same end.

A competitor does not significantly curtail a first developer's lead time by disassembling code unless large chunks of the code are literally copied or translated and inserted into the competitor's program. If a competitor makes verbatim or literal copies or translations of sections of the disassembled code, then an action for federal misappropriation would be available.

Copyright law treats disassembly in much the same way. It prevents a competitor from making literal copies of the disassembled code. Copyright law in certain instances permits reverse engineering or disassembly under its fair use defense: "For copyrighted works, including programs, fair use analysis applies. If reverse engineering, in the software context, means observing how a program works, and then writing an original program to accomplish the same ends, then it is probably permissible." Baumgarten & Meyer, *Program Copyright and the Office of Technology Assessment* (part 2), COMPUTER LAW., Nov. 1987, at 6-7. As one court recognized:

> The mere fact that defendant's engineers dumped, flow charted, and analyzed plaintiff's work does not, in and of itself, establish pirating.... Had [defendant] contented itself with surveying the general outline of the [plaintiff's] program, thereafter converting the scheme into detailed code through its own imagination, creativity, and independent thought, a claim of infringement would not have arisen.... While defendant may have permissibly dumped, flow charted, and analyzed plaintiff's code, it could not permissibly copy it.

E.F. Johnson Co. v. Uniden Corp. of Am., 623 F. Supp. 1485, 1501 n.17 (D. Minn. 1985). The *E.F. Johnson* court found iterative or verbatim reproduction of data tables and 38 of 44 subroutines. Defendants had reverse engineered plaintiff's object code program. *Id.* at 1497 n.10. *See* Hazen, *supra* note 132, at 124-25; *see* Brooks, *Reverse-Engineering Computer Software: Is It Fair Use or Plagiarism?* COMPUTER LAW INSTITUTE 799 (1986); *see also* Appendix.

[189] *SAS*, 605 F. Supp. at 821.

[190] *Id.*

[191] *Id.* at 819.

[192] *See* Whelan Assocs. v. Jaslow Dental Laboratories, 797 F.2d 1222, 1232 n.22 (3d Cir. 1986) (defendant Jaslow had "surreptitiously and without consent [of plaintiff] obtained a copy of the [Dentalab] source code"), *cert. denied*, 107 S. Ct. 877 (1987).

[193] *Id.* at 1226.

propriate contractual restrictions before source code is unveiled to any nonowner. Under the facts in *Whelan*, however, defendant Jaslow wrongfully took the source code from plaintiff without permission; therefore, plaintiff would have an action against defendant for federal misappropriation or conversion. An action for federal misappropriation would be available because plaintiff's distribution of its source code was unintentional.

A prudent software owner having a close business relationship with an employee or with a person responsible for marketing should take precautions by negotiating contractual restrictions including a nondisclosure provision or a reasonable covenant not to compete.[194] One advantage of the marketplace approach is that the parties can negotiate a variety of protective provisions. If the parties had defined the scope of protection in a contract the *Whelan* case would have been easily resolved.

III. Conclusion

Protection of property is fundamental for the survival of a market economy. Absent protection, members of society would lack the incentive to create or to preserve valuable property. Since the 1970s, when computer software became increasingly valuable, the legal system has struggled with which form of protection best serves society. Protection for this evolving form of property initially was problematic because no one knew how computer software should be categorized and defined. After years of debate, the bulk of protection for computer software resides in the copyright domain. Copyright law was originally established to protect words that were placed on paper and subsequently perceived by the human senses. Other forms of intellectual property, such as paintings and audiovisual displays, were added later to the copyright scheme, but the primary function of these works was to stimulate and communicate with human senses.

Although computer software is perceived by machines rather than humans, the introduction of software to copyright law seemed to work well at the outset. Early cases involved the literal, line-by-line copying of the program code, which was then sold for profit at relatively little cost to the infringer. Disallowing protection in these cases would have been devastating to the competitive process because the economic incentive needed to stimulate innovation and creativity in the computer software industry would have vanished. Later courts granted protection to the organization and structure of software to reach equitable results, and in doing so crossed the idea-expression threshold.

Recent cases extended copyright protection to the literal elements of the user interface of software. Audiovisual copyright has been expanded from a means of protection for fanciful creations in entertainment software to the protection of the user interface of productivity software. Although the concept of derivative work has not yet been used by a court to prohibit copying of the user

[194] *See* Plain's Cotton Coop. Ass'n v. Goodpasture Computer Serv., 807 F.2d 1256, 1258 (5th Cir.) (defendants, former employees of plaintiff, left plaintiff company with a copy of the source code that they developed while in scope of employment; none of the employees had been required to sign confidentiality agreements as a condition of employment), *cert. denied*, 108 S. Ct. 80 (1987).

interface, its far-reaching potential renders it a time bomb. Further, the concept of compilation has been extended from prohibiting free-riders in a noncomputer context to precluding legitimate software competitors. As a result of this ad hoc application of the copyright law, one is no longer free to create software that will duplicate the end result of another copyrighted program. This protection far exceeds that which is needed to provide an incentive to create.

This Comment recommends a marketplace approach to software protection using contract law as the foundation. The burden of protection would shift to software developers and the scope of protection would be negotiable. The scheme would protect both the idea and the expression, but the monopoly duration would be limited to a first developer's lead time. Further, Congress should adopt a federal law of misappropriation. Based on the classic decision in *International News Service* that persons should not be able to "reap where they have not sown," this specific scheme protects a first developer's lead time and encourages the marketplace to rely on contract law as its protective device.

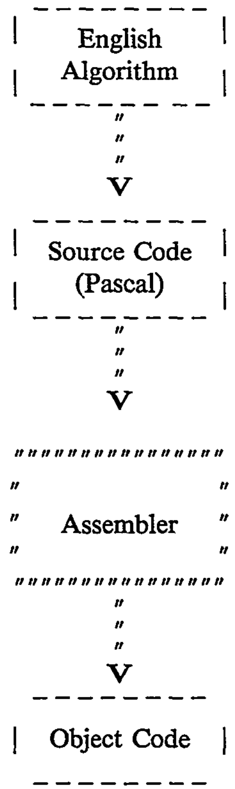
This new scheme of protection would provide economic reward for a limited time of a value commensurate with the creativity involved as measured by the product's usefulness to society. Moreover, this plan promotes further innovation and competition. Realizing that competition will soon follow, first developers would no longer rest on the economic success of their products, but would continually develop new software to stay on top of the marketplace. Finally, this system falls squarely within the guideline set forth in the U.S. Constitution because it promotes "the Progress of Science and useful Arts, by securing for limited Times to [programmers] the exclusive Right to their respective Writings and Discoveries."[195]

VANCE FRANKLIN BROWN

[195] U.S. CONST., art. I, § 8, cl. 8.

APPENDIX

Various Stages of a Computer Program



A programmer first identifies a problem and composes a solution in an English algorithm format. The algorithm is then used as a guide by the programmer to create the program in a high level source code language (such as Pascal). Next, the computer automatically translates the program into object code. There is an intermediate stage recognized between source code and object code known as assembler. Programmers can initially write assembler in lieu of the high level language, but assembler level of code requires much more detail from the programmer.

The computer can automatically reverse engineer (or disassemble) object code into assembler language. However, under the existing state of technology computers cannot recreate high level source from object code. The generated assembler code is slightly more comprehensible than the "1"s and "0"s of object code, but is much less understandable than high level source.

Each of the four stages of a program are demonstrated below. The purpose of the program used for this demonstration is to generate a list of prime numbers. The program was written by the author, taken through the steps, and then reverse engineered.

English Algorithm
(To Generate Prime Numbers)

STEP 1: Start with a prime number candidate of 2.

STEP 2: Process until the requested number of primes is found.

STEP 3: Start with a divisor of 2.

STEP 4: Process until the divisor is determined to be evenly divisible or until the divisor reaches the candidate.
STEP 5: Check this divisor and then loop back to STEP 4.
STEP 6: If the divisor was evenly divisible then display the candidate as a prime number.
STEP 7: Proceed to the next candidate and loop back to STEP 2.

SOURCE CODE
(Pascal Programming Language)

```
    Program PrimeNum (Input, Output);
    {
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*     WRITTEN BY:          Vance F. Brown                                    *
*     DATE:                April 5, 1988                                     *
*     PROGRAM PURPOSE:  This program allows user to enter the                *
*     desired number of prime numbers to be ascertained.  The                *
*     program then lists the prime numbers (starting with the prime          *
*     number 2) in numerical order in five columns.                          *
*   DEFINITION OF PRIME NUMBER:  A number is prime if it is not              *
*   evenly divisible by any number other than itself and 1.                  *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
}
    Var  {for variable function explanation, see below}
      num, count, primes, times, divisor: Integer; check: Boolean;
    Begin {of program}
      {***** initialize variables *****}
    num :=2;          {the prime number candidate}
    count := 0;       {number of prime numbers found}
    times := 0;       {number of columns before a carriage return}
    {***** ask user how many prime numbers to list *****}
    write ('How many prime numbers do you want to find?  ');
    read (primes);
    writeln;
    writeln;    {carriage returns}
    writeln;
    repeat  {until the number of primes is found}
      check := false;  {initialize: assume first that number is not
                          prime}
      divisor := 2;      {first number to be divided into candidate}
      divisor := succ(divisor);  {first number to be divided into candidate}
      {***** loop to determine if number is prime *****}
      while (divisor < num) and (not check) do begin
    check := num mod divisor = 0; {the remainder after dividing divisor
                                   into num}
    divisor := succ(divisor);       {increment divisor}
  end; {of while loop}

  {***** if a prime number, then display it *****}
  If not check then begin
    write (num:10);
    count :=succ(count);   {increment number of primes
                             found}
    times := succ(times);  {increment column number}

{***** column adjustment, once five numbers displayed, carriage
 return *****}
      if times = 5 then begin
        writeln; {carriage return}
        times := 0; {start counter for column number over}
      end; {of column adjustment if statement)}
    end; {of display of if prime number statement}
    num := succ(num); {next candidate for prime number}
  until count = primes; {once the number of primes entered by user is
                         found, the program ends}
end. {of program}
```

Object Code

```
11101001 01111001 00101100 10010000 10010000 11001101 10101011
01000011 01101111 01110000 01111001 01110010 01101001 01100111
01101000 01110100 00100000 00101000 01000011 00101001 00100000
00110001 00111001 00111000 00110101 00100000 01000010 01001111
01010010 01001100 01000001 01001110 01000100 00100000 01001001
01101110 01100011 00000010 00000100 00000000 10110001 01010111
00000000 00111100 00110011 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00000000 00000000 00000000 00000000 00000000 00000000
00000000 00010100 01000100 01100101 01100110 01100001 01110101
01101100 01110100 00100000 01100100 01101001 01110011 01110000
01101100 01100001 01111001 00100000 01101101 01101111 01100100
01100101 01010000 00011001 00000001 11111111 11111111 00001111
00000111 00000111 01110000 00001111 00000111 00000111 01110000
00001110 00000111 00000111 01001111 00101110 10001010 00100111
00001010 11100100 11111001 01110100 00001110 01000011 00101110
10001010 00000111 01010000 11101000 11011000 00001000 01011000
11111110 11001100 01110101 11110011 11111000 11000011 01111010
00000000 11111111 01111011 00001000 00000000 00011111 11000111
00000110 00010010 00000000 01101110 00000000 00101110 11000110
00000110 10010100 00000001 00000000 10111110 00100000 00000000
```

The above object code is only the first 24 of 1,674 printed lines of the prime number program.

Reverse Engineered Assembler

```
L05CF:  PUSH      AX            ;05CF  50
        PUSH      CX            ;05DO  51
        MOV       CL,4          ;05D1  B1 04
        SHR       AX,CL         ;05D3  D3 E8
        ADD       BX,AX         ;05D5  03 D8
        POP       CX            ;05D7  59
        POP       AX            ;05D8  58
        AND       AX,0F         ;05D9  25 0F 00
        RET_NEAR                ;05DC  C3
                                ;L05DD       L062A CC
L05DD:  CMP       BX,DX         ;05DD  3B DA
        JNZ       L05E3         ;05DF  75 02
        CMP       AX,CX         ;05E1  3B C1
                                L05E3        L05DF CJ
L05E3:  RET_NEAR                ;05E3  C3
```

```
                                      ;L05E4          L0656 CC
L05E4:    ADD        AX,CX            ;05E4   03 C1
          ADD        BX,CX            ;05E6   03 DA
          JMP        SHORT L05CF      ;05E8   EB E5
                                      L05EA           L0625 CC
L05EA:    MOV        AX,ES:[DI+4]     ;05EA   26 8B 45 04
          MOV        BX,ES:[DI+6]     ;05EE   26 8B 5D 06
          PUSH       AX               ;05F2   50
          OR         AX,BX            ;05F3   0B C3
          POP        AX               ;05F5   58
          RET_NEAR                    ;05F6   C3
```

The above reverse engineered assembler code is an excerpt of 29 out of 7,330 total lines. This code was automatically generated from the prime number program object code. Using this code to reconstruct the original 66-line high level Pascal source would take much longer than it originally took to write the high level program.