
Idelle R. Abrams

Follow this and additional works at: http://repository.jmls.edu/jitpl

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

Recommended Citation


http://repository.jmls.edu/jitpl/vol9/iss2/1

This Article is brought to you for free and open access by The John Marshall Institutional Repository. It has been accepted for inclusion in The John Marshall Journal of Information Technology & Privacy Law by an authorized administrator of The John Marshall Institutional Repository.


STATUTORY PROTECTION OF THE ALGORITHM IN A COMPUTER PROGRAM: A COMPARISON OF THE COPYRIGHT AND PATENT LAWS

By Idelle R. Abrams

TABLE OF CONTENTS

I. INTRODUCTION ............................................. 125

II. PATENTS ......................................................... 128
   A. EXCEPTIONS TO PATENTABILITY ............................ 129
      1. The Mental Steps Doctrine .............................. 129
      2. Natural Law Exception ................................. 131
      3. Administrative Concerns ............................... 133
   B. THE CURRENT STATUS OF SOFTWARE PATENTABILITY .... 134

III. COPYRIGHT .................................................. 135
   A. LIMITATIONS ON COPYRIGHTABILITY ......................... 138
      1. The Idea-Expression Dichotomy .......................... 138
      2. Utility ................................................ 139
   B. THE CURRENT STATUS OF COPYRIGHTABILITY FOR SOFTWARE .................................................. 140

IV. CONCLUSION ............................................. 143

I. INTRODUCTION

The copyright and patent laws have been enacted pursuant to the constitutional provision giving Congress the power "to promote the Progress of Science and useful Arts, by securing for limited Times to Authors and Inventors the exclusive Right to their respective Writings and Discoveries."¹ As with other broad powers granted by the Constitution, these intellectual property rights, codified by statute, have easily adapted to the many technological changes that have developed over the past 200 years. The copyright and patent laws have been flexible

---

¹ U.S. CONST., art. I, § 8, cl. 8.
enough to accommodate new technologies without altering or distorting the essential character of these provisions. Patents have been found to be appropriate for such new technologies as organic chemistry, electronics, and biotechnology. The copyright laws have been amended to encompass technological methods of expression such as photography and motion pictures.

The development of the computer and the rise of the software industry, however, has confounded the intellectual property system. The variety of situations in which computer technology can be applied, as well as the multiplicity of levels on which computer programs function, has created a situation that defies easy answers for copyright and patent law. Computer programs do not fit neatly into any existing category of intellectual property law but have been protected by both patent and copyright in different instances. This has created an unstable environment for computer programmers who do not know how best to protect their work.

The question of what is actually protected by a patent or copyright on a computer program is still an open issue. Does copyright cover just the actual code written by the programmer, or is it less literal, encompassing the structure of the program as well? When a machine or process which uses a computer program is granted a patent, what elements of the computer program are protected? Is the algorithm, the heart of the computer program, protected by either of these devices? The ambiguity and uncertainty surrounding these issues may be responsible in part for the problem of piracy that plagues the software industry, as well as for what some perceive as a lag in the development of software applications for computers.

These questions are unique to software because of what has been described as its "dual nature," that is "software is both symbolic and mechanical." Software is mechanical in that it is "used to manipulate computer machinery." It is symbolic because the language used to instruct the machine is a representation or symbol of the actual mechanical elements that comprise the computer. Software therefore has both a utilitarian aspect, which is traditionally the subject matter of patent,

5. Chisum, supra note 2, at 1015.
7. Id. (emphasis in original).
8. Id.
and an expressive element, which usually falls into the domain of copyright.

Software is also distinguished by the fact that it may be either physically embedded in the machine in a ROM (read-only memory), superscript 9 or detached from the machine on a floppy diskette. superscript 10 In addition, the same software program can be expressed in different languages which are not equally intelligible to human beings — (a) higher level languages such as Pascal, FORTRAN or BASIC which use ordinary English language as their basis; (b) lower level languages such as assembly language which consists of highly abbreviated code words; and (c) machine readable "binary" form, known as machine language, which is the form the computer understands. This language "can be represented by ones and zeros, but its actual form is slightly different; electrically, it usually consists of high and low voltages, and magnetically, it consists of different polarities of magnetization." superscript 11 Higher and lower level languages are called source code. Machine language is referred to as object code. superscript 12

Software is also classified according to its different purposes. Operations programs control the internal processes of the machine and are invisible to the user. For example, compiler or interpreter programs translate higher level languages into machine readable form, a process that is not apparent to the user. On the other hand, users interact directly with applications programs which enable a user to instruct the computer to perform a particular function such as word processing or manipulating data. superscript 13

Because software is all these things and more the courts, as well as legislators and bureaucrats, have been inconsistent in their responses to the protection of computer technology. This has resulted in widespread confusion and the lack of a coherent system of protection for computer software. Although software is granted patent protection in some instances, many programmers turn to the copyright system for protection. Trade secret law, though not a statutory intellectual property right, has

9. "A ROM chip is an array of transistor or other switches set permanently to high or low voltage states." Wharton, Use and Expression: The Scope of Copyright Protection for Computer Programs, 5 COMPUTER/L. J. 433, 436 n.13 (1985).

10. "A 'floppy' is a small magnetic disk, resembling a phonograph record, that may be written on, erased, reprogrammed, and removed." Note, Copyright Infringement of Computer Programs: A Modification of the Substantial Similarity Test, 68 MINN. L. REV. 1264, 1267 (1984).

11. Davidson, supra note 6, at 620.


13. Id.
also been used to protect computer programs against infringement and piracy.

Nor is this a trivial issue. The development of advanced technology is quickly becoming the foundation of the American and global economy and changing the way we live our daily lives. Yet this technology, which is very expensive to develop, is very easy and inexpensive to copy. Companies invest hundreds of man-hours and thousands of dollars in developing software programs. Because of the enormity of investment required for development, we want a system of protection that provides sufficient incentive to spur inventors to further creativity but does not grant such exclusive terms that it discourages further development and competitiveness.

In computer technology the key process for most programmers, and the one that takes the most time and money, is the development of the algorithm. The algorithm has been explained in various ways but there is a general consensus among programmers that an algorithm can be broadly defined as "a recipe or specific set of rules or directions for performing a task." One author has isolated five features of an algorithm: finiteness, definition, input, output, and effectiveness. Algorithms in computer programs are not limited to solving mathematical problems, though it is true that they are often used for that purpose. Many algorithms are "devised to solve all sorts of nonmathematical problems." It is not always clear that the courts appreciate this. The confusion stems from the fact that "in one sense all computer programs involve mathematics." Noted one commentator, "[n]o one can question that the notion of an algorithm has given the U.S. Supreme Court no little trouble." In the following Article, I will look at the statutory rights protecting computer programs and survey the developments in patent law and copyright law that affect the protection of the algorithm.

II. PATENTS

Patents may be obtained for "any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof . . . ." Upon the approval of a patent by the Patent and Trademark Office ("PTO"), the inventor receives a seventeen-year

---

16. Chisum, supra note 2, at 976.
grant entitling him or her to "the right to exclude others from making, using, or selling the invention throughout the United States ..."20 By granting a limited monopoly to the inventor, Congress intended to encourage invention and "promote the Progress of Science."21 Since this grant is a valuable and powerful one, an invention has to meet certain stringent standards to qualify for patent protection. An invention, which must qualify as either a process, machine, manufacture or composition of matter, must also be novel,22 useful,23 and non-obvious.24 An inventor asserts his or her invention in a recitation of claims25 set forth in the patent application. These claims are then evaluated in light of the prior art in the field to determine the novelty, usefulness, and non-obviousness of the invention.

Software has often been denied patent protection because it failed to meet one of these criteria or because it fell into one of the exceptions for patentability. Several factors have been important in this development. Software is now conceived of and created independently of the hardware. This has allowed computer hardware to be used more flexibly than "[i]n the earliest days of the industry [when] programs were hard-wired into machines."26 However, the separation of the software and hardware elements of the computer has led to "[t]he failure to recognize the engineering and mechanical nature of software,"27 according to one commentator. "[B]y divorcing programming from the hard-wiring of machines, courts and others have been led to the conclusion that programs are not part of the machinery but are 'mental' or 'mathematical' processes."28

A. EXCEPTIONS TO PATENTABILITY

1. The Mental Steps Doctrine

In defining the terms "art"29 and "process" to determine patentability, a number of cases pointed toward an exception for "mental steps."

20. Id. § 154.
23. Id. § 101.
24. Id. § 103.
25. 35 U.S.C. § 112 (1982) ("The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.").
26. Davidson, supra note 6, at 625.
27. Id.
28. Id.
29. Patent statutes, when they were first enacted in 1793, Act of Feb. 21, 1793, ch. 11, § 1, 1 Stat. 318, until the Patent Act of 1952, 35 U.S.C. § 1 et. seq., used the word "art" rather than "process."
This doctrine claimed that "inventions which require human thought in whole or in part for their practice were judicially considered not to be patentable subject matter."\textsuperscript{30} This doctrine was first articulated in \textit{In re Abrams},\textsuperscript{31} a case involving "[a] method of prospecting for [the presence of] petroliferous deposits."\textsuperscript{32} Claiming that "[c]itation of authority in support of the principle that claims to mental concepts . . . are not patentable is unnecessary,"\textsuperscript{33} the court concluded, "[i]t is self-evident that thought is not patentable."\textsuperscript{34}

The \textit{Abrams} court set out three rules by which to determine patentability of a claim involving mental steps: (1) a claim in which all the steps are purely mental is unpatentable; (2) a claim that includes physical as well as mental steps is not patentable if the novel aspect of the process is contained in the mental steps; and (3) a claim containing both physical and mental steps is patentable if the novelty is in the physical steps and the mental steps are just "incidental parts of the process."\textsuperscript{35}

The computer, it was discovered, was able to perform these "thought" functions much more quickly and accurately. However, since these functions, such as comparing, determining, registering, calculating, etc., were characterized as thought functions, the Patent Office applied the mental steps doctrine to software applications. The result was that patents on software were denied as a matter of policy.\textsuperscript{36}

The Court of Customs and Patent Appeals (CCPA), in reviewing applications denied by the Patent Office, "dismantled the mental steps doctrine"\textsuperscript{37} in a series of decisions. In its first case, \textit{In re Prater},\textsuperscript{38} the CCPA declared programs to be patentable subject matter. The court held that patent protection could not be denied a process that could be accomplished without human intervention just because it could also be performed through mental steps.\textsuperscript{39} In \textit{In re Bernhart},\textsuperscript{40} the CCPA rejected the anthropomorphic equation of the computer's electrical and magnetic processes with the thought process of the human mind.\textsuperscript{41} The

\textsuperscript{31.} 188 F.2d 165 (C.C.P.A. 1951).
\textsuperscript{32.} \textit{Id}.
\textsuperscript{33.} \textit{Id} at 168.
\textsuperscript{34.} \textit{Id}.
\textsuperscript{35.} \textit{Id} at 166.
\textsuperscript{36.} Chisum, \textit{supra} note 2, at 969.
\textsuperscript{37.} \textit{Id}.
\textsuperscript{38.} 415 F.2d 1378 (C.C.P.A. 1968), \textit{rehg and revd on other grounds}, 415 F.2d 1393 (C.C.P.A. 1969).
\textsuperscript{39.} 415 F.2d at 1389.
\textsuperscript{40.} 417 F.2d 1395 (C.C.P.A. 1969).
\textsuperscript{41.} \textit{Id} at 1401.
CCPA expressly rejected the mental steps doctrine in *In re Musgrave* asserting that the doctrine was logically unsound and lacked any statutory basis. The court found nothing to indicate that mental steps could not be part of a patentable process. "All that is necessary, in our view," said the CCPA, "to make a sequence of operational steps a statutory 'process' within 35 U.S.C. § 101 is that it be in the technological arts" so that it is within the ambit of constitutional purpose. The CCPA clearly articulated this technological aspect of software in its opinion in *In re Benson*, which was later overturned by the Supreme Court. The CCPA stated, "Realistically, the process of [the] claim . . . has no practical use other than the more effective operation and utilization of a machine known as a digital computer." 

2. *Natural Law Exception*

Another limitation on patentability is the long-accepted view that a patent will not be granted for the discovery of a "law of nature" or a "principle in natural philosophy or physical science." The rationale for this exclusion is that patents on these laws or principles would "effectively preempt the natural laws on which science and technology are built," and would therefore "inhibit rather than promote the development of science and technology." Patents will, however, be granted for specific applications of a natural law. This was noted by the Court in *Funk Brothers Seed Co. v. Kalo Co.*, where the Court stated that "[i]f there is to be invention from such a discovery, it must come from the application of the law of nature to a new and useful end."

This exception has been applied to the field of computer programming. The intersection of computer technology and mathematics has prompted people to see the computer as a purely mathematical entity. In *Gottschalk v. Benson*, the first Supreme Court decision on software patentability, the Court denied the patent sought by the inventors of "a method of programming a general-purpose digital computer to convert signals from binary-coded decimal form into pure binary form." The
Court found that the claims would protect the algorithm, which the Court found to be akin to a law of nature because of its mathematical component. "Phenomena of nature . . . mental processes, and abstract intellectual concepts are not patentable . . . the patent would wholly preempt the mathematical formula and . . . would be a patent on the algorithm itself."54

The Court in Benson defined an algorithm as "a procedure for solving a given type of mathematical problem. . . ."55 The Court did not acknowledge that the term "algorithm" is "used in differing ways in mathematics (as a formula) and in programming (a structured sequence to solve a problem)."56 This has had a long term effect on the protection of computer software. In defining an algorithm as a method for the solution of mathematical problems, the Court seemed to equate an algorithm with a mathematical formula, and thereby subject all computer programs to the natural law exception. The Court rejected this conclusion,57 yet the practical effect of this decision was to dissuade programmers from seeking patents for their software.

The Benson decision has provoked much criticism. "The Court erred both in implying that algorithms relate only to mathematical problems and in characterizing the method involved in Benson as directed to 'mathematical' problems."58 Benson's conversion programs would be more accurately described as a translation program than as a solution to a mathematical problem.59 Its purpose is simply to take numbers represented by one type of symbol (binary codes) and present them in another symbolic form (binary). This algorithm does not recite any law of nature (such as the nature of radio waves) or describe any mathematical knowledge (such as the length of a hypotenuse in relation to the other sides of a triangle).

Despite these criticisms, the Court reaffirmed its Benson decision in Parker v. Flook.60 In Flook, the inventor had devised a method for updating alarm limits during catalytic conversion processes. Using the definition of an algorithm proposed in Benson, the court recognized that a patent for the algorithm in Flook would not "wholly preempt the mathematical formula,"61 as it would have in Benson. However, the Court, in reasoning reminiscent of the Abrams opinion, held that the

54. Id. at 65.
55. Id. at 71-72.
57. Benson, 409 U.S. at 71 ("It is said that the decision precludes a patent for any program servicing a computer. We do not so hold.")
58. Chisum, supra note 2, at 976.
59. See id. at 977.
60. 437 U.S. 584 (1978).
process is not patentable if the only novel element is the algorithm. "The process itself, not merely the mathematical algorithm, must be new and useful. Indeed, the novelty of the mathematical algorithm is not a determining factor at all."\textsuperscript{62}

The rationale for this statement is that a natural law or algorithm, though newly discovered, "reveals a relationship that has always existed,"\textsuperscript{63} and therefore is considered within the prior art in the field. The Court concludes that Flook's process is not only unpatentable, but that it is unpatentable subject matter. This is true "not because it contains a mathematical algorithm as one component, but because once that algorithm is assumed to be within the prior art, the application, considered as a whole, contains no patentable invention."\textsuperscript{64}

3. Administrative Concerns

The third factor that has contributed to the denial of patents for computer software has no doctrinal rationale at all. In 1964, when the patentability of computer software was first considered, the "Patent and Trademark Office (PTO) favored patents on computer programs"\textsuperscript{65} which were considered to transform existing hardware into new and useful machines.\textsuperscript{66} This would place software within the traditional scope of patentable material. However, in its 1966 report, the President's Commission on the Patent System "recommended patent protection not be extended to computer programs primarily because of the anticipated administrative burden on the Patent Office."\textsuperscript{67} The Supreme Court quoted the Commission's report in its decision in Benson:

The Patent Office now cannot examine applications for programs because of a lack of a classification technique and the requisite search files. Even if these were available, reliable searches would not be feasible or economic because of the tremendous volume of prior art being generated. Without this search, the patenting of programs would be tantamount to mere registration and the presumption of validity would be all but nonexistent.\textsuperscript{68}

One commentator concluded, after a thorough analysis of the Benson decision, that it is "[i]n this quotation, [that] we find the only com-

\textsuperscript{62.} Flook, 437 U.S. at 591.
\textsuperscript{63.} Id. at 593 n.15.
\textsuperscript{64.} Id. at 594.
\textsuperscript{66.} See, e.g., Ex parte King, 146 U.S.P.Q. 590 (1964).
\textsuperscript{67.} Wharton, supra note 9, at 434 (citing PRESIDENT'S COMMISSION ON THE PATENT SYSTEM, "TO PROMOTE THE PROGRESS OF . . . USEFUL ARTS" IN AN AGE OF EXPLODING TECHNOLOGY 14 (1966)).
\textsuperscript{68.} Benson, 409 U.S. at 72.
prehensible reasoning in Benson for denying patent protection."  
Another commentator echoed this sentiment, claiming that the only sensible explanation for the holding is that bureaucratic concerns of the PTO rather than "sound principles of intellectual property law" guided the Court. The Court in Flook also recognized the concern of the Acting Commissioner of Patents and Trademarks that a finding of patentability would produce a flood of patent applications. The effect of these decisions has been to distort the theoretical analysis of the patentability or copyrightability of computer software.

B. The Current Status of Software Patentability

Following several decisions of the CCPA expanding the forms of software programming that would be deemed patentable subject matter, the Supreme Court held a computer program to be patentable subject matter for the first time in Diamond v. Diehr. Diehr involved a process for curing rubber in which a computer would be constantly fed data on the temperature of the rubber inside the mold and, using a known formula, calculate the point at which the rubber was properly cured. The computer would then signal the opening of the mold. This process solved the problem of undercured or overcured rubber, a problem the industry had been unable to resolve. The Court found that "[i]ndustrial processes such as this are the types which have historically been eligible to receive the protection of our patent laws." Without overruling Benson or Flook the Court stated that its conclusion that respondent's claims fall within the category of statutory subject matter "is not altered by the fact that in several steps of the process a mathematical equation and a programmed digital computer are used." In other words, "a claim drawn to subject matter otherwise statutory does not become nonstatutory simply because it uses a mathematical formula, computer program, or digital computer."

That Parker and Flook stood for no more than the unpatentability of laws of nature, natural phenomena, and abstract ideas was not altered by the decision in Diehr. The Court does not discuss the algorithm because, though respondents use a mathematical formula, the

69. Chisum, supra note 2, at 989.
71. Flook, 437 U.S. at 587-88.
72. See In re Freeman, 573 F.2d 1237 (C.C.P.A. 1978); In re Walder, 618 F.2d 758 (C.C.P.A. 1980).
73. 450 U.S. 175 (1981).
74. Id. at 184.
75. Id. at 185.
76. Id. at 187.
Court finds that they do not seek to preempt its use. "[T]hey seek only to foreclose from others the use of that equation in conjunction with all of the other steps in their claimed process."\textsuperscript{77}

It is widely held that the Court's decision in \textit{Diehr} opened the door to the patentability of computer software. However, the Court did not redefine the algorithm in a way that would make it more compatible with the definition used by programmers. Nor did the Court address the status of nonmathematical algorithms. However, the PTO, implementing the \textit{Diehr} decision, issued new guidelines\textsuperscript{78} which exhibited "an administrative policy in favor of patents on most types of programs."\textsuperscript{79} In addition, the PTO did not limit patentability to those programs which applied mathematical algorithms, as in \textit{Diehr}, but extended patentability to software that contained nonmathematical algorithms as well.\textsuperscript{80}

"Software patentability is a \textit{de facto} reality today, as the PTO now commonly issues patents for software inventions."\textsuperscript{81} Patentability, however, is still limited to claims which meet the "process" definition as it has been articulated by the Court. "Transformation and reduction of an article 'to a different state or thing' is the clue to the patentability of a process claim that does not include particular machines."\textsuperscript{82} Therefore, it is still not clear to what extent an algorithm which is embedded in a patentable process is protected by the patent.

\section*{III. COPYRIGHT}

Until the Supreme Court's decision in \textit{Diehr}, patent protection for software was widely regarded as being unavailable because of the PTO's "adhere[nce] to the view that all computer programs and program-related inventions were unpatentable subject matter."\textsuperscript{83} In an effort to protect their work, programmers turned to the provisions of the copyright law.

The copyright statute has its roots in the same constitutional clause\textsuperscript{84} as the patent laws, however, the purpose of the copyright laws is somewhat different from the objective of the patent process. The patent scheme seeks to provide an incentive for technological innovation. Copyright's purpose, on the other hand, is to foster "the dissemination

\begin{itemize}
\item \textsuperscript{77} \textit{Id}.
\item \textsuperscript{78} \textit{Manual of Patent Examining Procedure, Section 2110: Patentable Subject Matter—Mathematical Algorithms or Computer Programs 538-38.3 (Oct. 1981).}
\item \textsuperscript{79} \textit{Stout, supra note 30, at 222.}
\item \textsuperscript{80} \textit{Id.}
\item \textsuperscript{81} \textit{Maier, supra note 70, at 157.}
\item \textsuperscript{82} \textit{Benson, 409 U.S. at 70.}
\item \textsuperscript{83} \textit{Rodau, supra note 12, at 529.}
\item \textsuperscript{84} \textit{U.S. Const., art. I., § 8. See supra note 1 and accompanying text.}
\end{itemize}
The free exchange of ideas, it is hoped, would have a salutary effect on the development of society's collective knowledge.

Copyright registration for computer programs first became an issue in 1961 when a copy of a computer tape was deposited with the Copyright Office. In 1964, the Copyright Office announced that it would accept computer programs for registration under its "rule of doubt" policy, "giving the applicant the benefit of the doubt" that the material was copyrightable. The program still had to meet the rather minimal copyright standards: it had to contain sufficient original authorship, it had to have been published bearing a copyright notice, and the deposited copy had to be in human-readable form.

Two general questions about the propriety of copyright for computer programs disturbed the Copyright Office. First, the case of White-Smith Music Publishing Co. v. Apollo Co. had held that a piano roll was not a "copy" within the meaning of the statute because it was intelligible only when it was operated on a player piano. The statute limited its protection to copies which were perceptible to humans in either written or printed form. Computer programs, at least those in machine-readable form, seemed to be excluded from copyright protection by this holding. Second, the Constitution specified that protection was to extend to "writings" of an author, yet computer programs are not writings in the conventional sense.

The effort to revise the copyright laws, which began in 1955, culminated in the enactment of the Copyright Act of 1976 ("1976 Act"). The 1976 Act revised the copyright statute in several areas that were very significant to the copyrightability of software. The definition of "copies" was revised to include any work that is "fixed" so that it "can be perceived, reproduced, or otherwise communicated, either directly or with the aid of a machine or device." This definition makes it unnecessary for copies to be directly intelligible to humans, effectively overruling White-Smith Music Publishing Co., and eliminating
one reservation of the Copyright Office in copyrighting software.

The 1976 Act also eliminated the publication requirement of the prior copyright act, providing protection to a work as long as it was "fixed in any tangible medium of expression..." Congress, by eliminating this publication requirement, which was conceptually tied to traditional forms of expression such as books and magazines, and endorsing copyrightability for any "fixed" expression, expanded the permissible forms of expression that would be copyrightable.

In another move to update the copyright laws, Congress had established the National Commission on New Technological Uses of Copyrighted Works ("CONTU") to investigate, among other things, the copyrightability of computer programs and to propose any revisions that would be appropriate in light of their findings. In its Final Report, CONTU recommended that:

The new copyright law should be amended: (1) to make it explicit that computer programs, to the extent that they embody an author's original creation, are proper subject matter of copyright; (2) to apply to all computer uses of copyrighted programs by the deletion of the present section 117; and (3) to ensure that rightful possessors of copies of computer programs may use or adapt these copies for their use.

Congress adopted CONTU's recommendations and amended the copyright law in the Software Act of 1980. The amendments incorporated the definition of "computer program" proposed by CONTU into § 101 of the 1976 Act and replaced § 117 with the version recommended by CONTU. From these two revisions, "courts have inferred the broad grant of protection recommended by CONTU." Though CONTU's grant of protection appeared to apply across the board to all forms of computer programs, a lingering doubt remained. Operations programs, programs embedded in ROM chips, and programs in machine-readable form seemed to some to lack the necessary characteristics for copyrightability. However, the decision in *Apple Computer, Inc. v. Franklin Computer Corp.* is generally considered to have resolved this issue and to have extended copyright protection to all forms

---

97. NATIONAL COMMISSION ON NEW TECHNOLOGICAL USES OF COPYRIGHTED WORKS, FINAL REPORT (hereinafter CONTU FINAL REPORT) (1978).
98. Id. at 1.
A. LIMITATIONS ON COPYRIGHTABILITY

1. The Idea-Expression Dichotomy

The copyright laws, like the patent laws, came to be subject to certain limitations as the courts reviewed claims for copyright protection. The copyright law is aimed at encouraging the exchange of ideas; a copyright on the idea embedded in a work would frustrate this goal. This was recognized by the Supreme Court in *Baker v. Selden*, the first case to articulate the idea-expression dichotomy. That case involved the copyright on a book describing a bookkeeping system and included examples of the blank forms on which the bookkeeping entries were to be made. Selden sought to enforce his copyright on the book, including the use of the forms, against Baker who used a system which, though slightly modified, was substantially similar to Selden's system. The Court acknowledged that if Selden "had the exclusive right to the use of the system explained in his book, it would be difficult to contend that the defendant does not infringe it." The question, therefore, was whether Selden did have that "exclusive right" by virtue of his copyright.

The Court concluded that copyright protection covered the author's expression but did not extend to the system itself. The ideas that were described in the book were placed in the public domain. "The very object of publishing a book on science or the useful arts is to communicate to the world useful knowledge which it contains." An author is granted copyright protection on his or her expression in exchange for the ideas that he or she adds to the store of knowledge. Clearly "this object would be frustrated if the knowledge could not be used without incurring the guilt of piracy of the book." People, then, are free to use the ideas in a copyrighted work to create works of their own, but are prohibited from copying the expression of another's copyrighted work.

The Court reiterated this dichotomy in the more recent case of *Mazer v. Stein*. The Court went on to distinguish the rights granted under the patent and copyright laws. "Unlike a patent, a copyright gives no exclusive right to the art disclosed; protection is given only to

---

102. See Rodau, supra note 12, at 552.
103. 101 U.S. 99 (1879).
104. Id. at 100.
105. Id. at 103.
106. Id.
the expression of the idea — not the idea itself."  

This dichotomy is now so entrenched in our understanding of the copyright grant that it was codified in the 1976 Act, which says:

In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work.\(^{109}\)

An author's statutory rights, therefore, extend only to his or her expression.

However, in certain situations, even the expression may be unprotected. When an idea can only be expressed in a limited number of ways, a copyright on the expression would preclude anyone else from using the idea. It would then amount to a copyright on the idea itself, which is impermissible.\(^{110}\) The rules for a contest or a jeweled pin in the shape of a bee are examples of "idea-expression identity" and are not protected by copyright.

Where does one draw the line between idea and expression? This distinction, never an easy one to make, is made even more difficult in the area of computer software. CONTU recognized that "the distinction between copyrightable computer programs and uncopyrightable processes or methods of operation does not always seem to 'shimmer with clarity'."\(^{111}\) The Commission declined the opportunity to establish a standard to distinguish idea from expression, citing the variety of computer software applications and the rapidly advancing technology. "To attempt to establish such a line in this report written in 1978 would be futile."\(^{112}\) Instead, the Commission left the responsibility for such demarcation to "the institution designed to make fine distinctions—the federal judiciary."\(^{113}\)

2. Utility

*Baker v. Selden* also gave rise to another exception to copyright protection. The Court, in the following passage, implied that the security of rights in a useful object is not a feature of the copyright laws; utilitarian objects can only be protected through the patent system.

The description of the art in a book, though entitled to the benefit of copyright, lays no foundation for the exclusive claim to the art itself.

---

108. *Id.* at 217.
110. See, e.g., Morrissey v. Proctor & Gamble Co., 379 F.2d 675 (1st Cir. 1967); Herbert Rosenthal Jewelry Corp. v. Kalpakian, 446 F.2d 738 (9th Cir. 1971); Sid & Marty Kroft Television Prods. Inc. v. McDonald's Corp, 562 F.2d 1157 (9th Cir. 1977).
111. CONTU FINAL REPORT, *supra* note 97, at 18.
112. *Id*.
113. *Id.* at 19.
The object of the one is explanation; the object of the other is use. The former may be secured by copyright. The latter can only be secured, if it can be secured at all, by letters-patent.\textsuperscript{114}

A computer program is defined in the 1976 Act as a "set of statements or instructions to be used . . . to bring about a certain result."\textsuperscript{115} This definition seems to bring computer programs into direct conflict with the prohibition against copyright for utilitarian objects. The Commission, however, was not disturbed by this problem and recommended that copyright protection be granted to all computer programs regardless of their form or function.

\textit{Mazer v. Stein} recognized the exception for utilitarian objects but found that a utilitarian work was not precluded from copyright protection if it also exhibited copyrightable features. For those utilitarian objects that qualified for copyright protection, the copyright would only apply to its expressive elements. Computer programs satisfy the definition of protectible works which are "original works of authorship fixed in any tangible medium of expression." The fact that they may also perform useful activities is not, following \textit{Mazer v. Stein}, a bar to their copyrightability. Some commentators, nevertheless, still assert that computer programs which in their effects are wholly utilitarian, such as operating programs and programs that control fuel injection engines or monitor traffic lights, should not be copyrightable.\textsuperscript{116}

\section*{B. The Current Status of Copyrightability for Software}

The first case heard by any court on the copyrightability of a computer program challenged the copyright on the basis of the idea-expression dichotomy. In \textit{Synercom Technology Inc. v. University Computing Co.},\textsuperscript{117} the matter in controversy, as in \textit{Baker v. Selden}, was forms. In \textit{Synercom} the forms were input formats designed to structure the sequence in which data was input into the computer by the user. The court recognized "the reality that these input formats \textit{express} to the user the sequencing of data for simplified access to the computer programs."\textsuperscript{118} The question the court faced was whether this expression was separable from the ideas that were expressed. The court concluded that it was not.

Subsequent cases brought to enforce copyright interests in computer programs typically involved infringers who had directly copied the

\begin{flushleft}
\textsuperscript{114} 101 U.S. at 105.  \\
\textsuperscript{115} 17 U.S.C. § 101 (as amended) (1982).  \\
\textsuperscript{116} See, e.g., Concurring Opinion of Commissioner Nimmer, CONTU FINAL REPORT, supra note 97, at 26-27; Samuelson, supra note 4, at 727-49; Goldstein, Infringement of Copyright in Computer Programs, 47 U. PIT. L. REV. 1119 (1986).  \\
\textsuperscript{117} 462 F.Supp. 1003 (N.D. Tex. 1978).  \\
\textsuperscript{118} Id. at 1012 (emphasis added).
\end{flushleft}
source code of the copyrighted program. In these cases, the inescapable evidence of impermissible copying did not provoke an inquiry into the idea-expression dichotomy or an analysis of what constitutes “expression” in the software context.

Expression in the area of fictional works has long been understood to include more than just the literal string of words as set down by the author. Since Judge Learned Hand’s opinion in Nichols v. Universal Pictures Corp., courts have recognized that the expression in literary works extends to the structure of the work, to the pattern that is displayed when one takes “an abstract of the whole.” If copyright only protected the literal text “a plagiarist would escape by immaterial variations.”

Judge Hand recognized that you can take the abstraction process too far. “[T]here is a point in this series of abstractions where they are no longer protected.” The further away the abstraction process gets from the actual language of the work, the closer you come to the uncopyrightable ideas the work describes.

Two recent cases adopt a similar approach to isolating the expression in a computer program. These cases may have an impact on the protectibility of the algorithm in a computer program. In SAS Institute v. S & H Computer Systems, Inc. the court identified the key issue as “whether S & H appropriated from SAS only ideas and concepts or whether it also appropriated expression.” S & H had taken the SAS program, which operated on an IBM, and adapted it so that it would run on a VAX computer.

The analysis the court applied asked whether the two programs were substantially similar. The court recognized that “[s]ubstantial similarity, of course, does not require literal identity; ‘a play may be pirated without using the dialogue.’” Though the court found evidence of literal copying, it also found that “the copying proven at trial does not affect only the specific lines of code . . . Rather, to the extent that it

120. 45 F.2d 119 (2d Cir. 1930).
121. Id. at 121.
122. Id.
123. Id.
125. Id. at 829.
126. In the absence of absolute evidence of copying the court will find copying if the plaintiff shows that (a) the defendant had access to the copyrighted program and (b) the defendant's program is substantially similar to the plaintiff's. See Whelan Assocs. Inc. v. Jaslow Dental Lab. Inc., 797 F.2d 1222, 1231-32 (3d Cir. 1986).
127. 605 F.Supp. at 829 (quoting Sheldon v. Metro-Goldwyn Pictures Corp., 81 F.2d 49, 55 (2d Cir.), cert. denied, 298 U.S. 669 (1936)).
COMPUTER/LAW JOURNAL

represents copying of the organization and structural details of SAS, such copying pervades the entire S & H product."\textsuperscript{128} The court relied, in part, on Meredith Corp. v. Harper & Row Publishers Inc.,\textsuperscript{129} which found that a textbook abstracted into outline form and then rewritten "independently" following this outline "resulted in duplication of expression, and not merely of ideas."\textsuperscript{130} Whelan Associates Inc. v. Jaslow Dental Laboratory Inc.,\textsuperscript{131} brought before the Third Circuit, was the first court of appeals decision to rule on the scope of copyright protection afforded computer programs. The court determined that the question at issue was "whether the structure (or sequence or organization) of a computer program is protectible by copyright, or whether the protection of the copyright law extends only as far as the literal computer code."\textsuperscript{132} The District Court found that the overall structure of Whelan's program, designed to handle the administrative procedures of dental laboratories, had been copied and that the protection of this structure was contemplated by the copyright laws.\textsuperscript{133} The Court of Appeals agreed and dismissed the defendant's argument "that what is true of other literary works is not true of computer programs."\textsuperscript{134} Instead, the Court of Appeals held computer programs to the same standards of expressiveness as other literary works.

The Third Circuit devised a test, derived from Baker v. Selden and particularly appropriate for utilitarian objects, to determine when expression and idea are identical:

\textit{[T]he line between idea and expression may be drawn with reference to the end sought to be achieved by the work in question. 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... Where there are various means of achieving the desired purpose, then the particular means chosen is not necessary to the purpose; hence, there is expression, not idea.}\textsuperscript{135}

As long as the idea and the expression of that idea are not of necessity identical, a programmer can obtain a copyright on his or her particular expression.

The \textit{Whelan} court found support for its conclusion that the struc-

\begin{itemize}
\item \textsuperscript{128} Id. at 830.
\item \textsuperscript{130} SAS, 605 F.Supp. at 826.
\item \textsuperscript{131} 797 F.2d 1222 (3d Cir. 1986).
\item \textsuperscript{132} 797 F.2d at 1224.
\item \textsuperscript{133} 609 F.Supp. 1307, 1321-22.
\item \textsuperscript{134} 797 F.2d at 1234.
\item \textsuperscript{135} Id. at 1236.
\end{itemize}
ture of a program is protected by copyright in the SAS case. The court also found the 1976 Act to be another source of support.\textsuperscript{136} Compilations and derivative works are given copyright protection under § 103. A compilation is defined in § 101 as material or data that is assembled, coordinated, or arranged. The particular assemblage is the protected work. Similarly a derivative work is one which abridges, condenses, or otherwise recasts a preexisting work. The court concludes from this “that Congress was aware of the fact that the sequencing and ordering of materials could be copyrighted.”\textsuperscript{137} One commentator has argued that these provisions could be used to protect algorithms as well.\textsuperscript{138}

Neither the SAS court nor the Third Circuit in Whelan discussed the copyrightability of algorithms. However, their grant of copyright protection to the structure, organization, or sequence of a computer program would seem to encompass protection of the algorithm as well, for an algorithm is nothing more than a particular sequence or structure.\textsuperscript{139} Whether the courts will continue to follow the Whelan line of reasoning is unclear at this point. Whether, in addition, they would extend this reasoning to include algorithms cannot be predicted.

IV. CONCLUSION

The Commissioners who comprised the CONTU panel agreed unanimously with the general sentiment that computer software should be provided legal protection.\textsuperscript{140} They did not, however, agree on the best method by which to provide this protection or on how far the protection should extend. Almost ten years later there is still no consensus on this issue and software protection continues to be haphazard. The courts, recognizing the need for such protection and operating in the absence of any clear legislative mandate, have begun shaping policy.

Protection of the algorithm is hampered in part by an incomplete understanding of the concept by the courts and in part by the traditional bars to copyrightability and patentability. Nevertheless, both patent and copyright law seem to be moving toward some approximate form of protection for algorithms. The question that must next be asked is whether this movement on both fronts is a desired situation. It has been said that “[i]n effect, copyright protection has been stretched in Whelan to fill the gap left when the courts denied software inven-

\textsuperscript{136} Id. at 1239.
\textsuperscript{137} Id.
\textsuperscript{139} See supra notes 14-18 and accompanying text.
\textsuperscript{140} CONTU FINAL REPORT, supra note 97, at 18.
tions patent protection." On the other hand, Whelan can be seen as the application of copyright's long accepted abstraction analysis to a new form of literary work. This ambiguity demonstrates that a reevaluation of the basic principles of copyright and patent law and their application to computer software is needed to bring some coherence to the statutory protection of computer software.

141. Maier, supra note 70, at 161.