The Inappropriately Large Role Of Copyright Protection In Regards To Computer Programs:

Suggestions for Decreased Reliance on the Copyright System and an Increased Reliance on Patent Protection

FRANKLIN PIERCE
LAW CENTER LIBRARY
CONCORD N. H.

JUL 18 1997

Duncan Y.C. Hsu
May 13, 1994

Introduction

In the mid to late eighties, many articles were written regarding the ability of the United States' IP laws to adequately and efficiently protect computer programs and software. It is possible that this was partly due to the recent passage of the Chip Act of 1984, which provided sui generis protection for mask works fixed in a semiconductor chip product.[1] Since the beginning of the 90s, there have been far fewer articles regarding similar sui generis protection for computer programs. It is beyond the scope of this paper to sufficiently explain this, however I speculate that there has been a realization that it is highly unlikely that Congress can be persuaded in the near future to change our IP laws vis-a-vis the protection of computer programs. In addition, there may be an acknowledgment that however inefficient, the current IP laws can properly protect computer programs if creatively implemented.

The purpose of this paper is to address some of the major criticisms of the United States' IP laws when it comes to protecting computer software. My focus, however, is on

---

1 17 U.S.C. §§ 901-914. See specifically §902 which covers the subject matter of protection. In a nutshell, a mask work that is original, not in the public domain, and not obvious shall be entitled to protection. However, such protection does not extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery. In fact, §102(b) of the Copyright Act of 1976 (17 U.S.C. §102(b)) uses the exact same wording as §902(c) of the Chip Act. In a nutshell, the Chip Act only protects the actual mask image that is embodied on the semiconductor chip.

criticisms of using Copyright law to protect useful computer programs. My approach will be to suggest certain amendments to our IP laws that may aid them in working more efficiently in regards to the protection of computer programs. In no way do I suggest that an entirely sui generis system be developed, since I believe that the current system is working. I simply state that the current system could work better.

Policy considerations behind the protection of software

Why protect computer programs? For that matter, why protect any intellectual creation? Although this question may be rudimentary, it is crucial to keep it in mind when criticizing the current framework for protecting computer programs within the United States, and it is even more crucial when proposing amendments to the current IP laws. At this point, the old Incentive/Dissemination policy argument comes into light. We protect intellectual creations in order to encourage inventors and authors to create, and to disclose that which they have created. By giving such creators certain exclusive rights with regard to their works, they are able to exploit these rights for economic gain. Of course, this is where the other side of the coin shows up. What good does it do society if a creator is allowed to exclude all others from its inventions or works? In order to ensure that

all of these creations eventually enter the public domain within a reasonable amount of time, the creators' exclusive rights are both limited and temporary.

When evaluating the effectiveness of the current IP framework for protecting computer programs, and when making suggestions to improve this framework, one must keep the Incentive/Dissemination balance firmly in mind. The basic questions are: How much protection should be given to a computer program so that enough incentive exists to encourage the production of new programs? How do you keep from over-protecting computer programs so as to enrich the public domain and nurture the growth of the software industry? These questions will be addressed throughout this paper.

The Problematic Nature of Computer Programs

In May of 1992, the Office of Technology Assessment (OTA), a congressional agency, published a report titled, "Finding a Balance: Computer Software, Intellectual Property and the Challenge of Technological Change".[2] OTA listed six objectives in preparing the report, among which are:

1. Understanding the characteristics of software as a technology;

---

2 see generally 40 Fed.B.News & J. 226, available in Westlaw

2. Exploring the relationship between the legal and statutory protection of software and incentives for innovation;

3. Identifying current intellectual property challenges presented by software and computing technologies; and

4. Anticipating future challenges from technological developments in computer software, operations, and architectures.

The OTA freely acknowledged that it is difficult, if not impossible, to fit software into the current framework of copyright and patent laws. In so doing, it listed several factors:

1. The nature of software technology;

2. The swift pace in which technological developments occur in the computer industry;

3. The difficulties in reconciling cultural and definitional differences between the legal and technological communities;

4. The difficulties that functional/utilitarian aspects of computer programs have on determining the propriety of copyright protection and the proper scope of such protection; and

5. Difficulties in determining the proper scope of patent protection for computer programs, and the obstacles facing the PTO in determining the prior art.

In the report, the OTA focused on a main question: whether there should be a sui generis form of protection for software, or should the traditional legal forms of protection be forced to accommodate to the unique characteristics of computer programs? The OTA concluded that given the prior difficulties in incorporating software protection into the existing IP laws, coupled with the dizzying speed of development within the computer industry, "There are questions as to whether this process of accommodation can--or should-- continue indefinitely."[3] The OTA suggested that we may soon come to a point where it will be in our best interest to introduce new legislation for software protection either to compliment or replace the existing legal copyright and patent framework, rather than continuing incremental accommodations within the current system.

As acknowledged by the OTA, most problems that arise from trying to protect computer software under existing intellectual property systems arise from the nature of computer programs:

> The most fundamental difficulty is one of classification. Software is a hybrid, neither pure writing, traditionally protected by a copyright system, nor pure

---

3 OTA report at 8, quoted in 40 Fed.B.News & J. 226

machine, usually protected by patent law.[4]

Most critics of the current framework for protecting computer software share the same argument: by nature, many computer programs are best protected under a patent-like framework due to their utilitarian features. However, due to the Supreme Court decision, Gottschalk v. Benson[5], and the PTO's strict interpretation of that holding, patent protection was not viewed as a viable method of protecting computer programs per se. Given the rapid proliferation of the software industry during the late seventies and eighties, the legislature and courts turned to copyright as the mainstream form of protection for computer programs. The same critics now agree that copyright alone cannot adequately protect programs which derive their value from their utilitarian functions.

However, certain programs fit within the copyright scheme quite nicely. In particular, computer games and screen savers -- any programs that derive their sole or main value from their artistic or literary properties. For programs such as these, it would be inappropriate to provide them with patent protection. And of course, then you must deal with the gray area: programs that have both utilitarian and

---

4 Max W. Laun, *Improving the International Framework for the Protection of Computer Software*, 48 U.Pitt.L.Rev. 1151, 1151

5 409 U.S. 63, 175 U.S.P.Q. 673 (1972)

artistic/literary aspects. Examples might be a Windows-like operating system that has an original graphic display or a computerized flight simulator game that has a novel and nonobvious graphics generator for the animated terrain. In these cases, do you simply categorize the program as either utilitarian or artistic and then apply the appropriate protection, thus precluding the other form of protection? Or, do you recognize that the dual natures of the program require independent protection? The current practice in the United States is the latter, and I propose that a more efficient system for protecting computer software should not do away with this feature. Perhaps it would be useful, then, to codify the current practice and acknowledge that a computer program may have separate features that are entitled to separate protection, and in such a case, require that the two features be protected separately.

In fact, most commentators agree that patent and copyright protection for computer programs are not mutually exclusive. Since many programs contain both patentable and copyrightable elements, it is possible to obtain both patent and copyright protection for a given program. As David Bender and Anthony Barkume illustrate:

> ...consider a patent claiming a computer-implemented business method. It might be possible to protect certain aspects of some such methods by copyright -- a screen display here, or a form of document there...But even so, protecting the occasional screen display or form of document would in no way deter others

FRANKLIN PIERCE LAW CENTER LIBRARY CONCORD, N.H.

> from practicing the method. If the method is where the value lies, the proprietor must rely for protection on either trade secret or patent.[6]

Another problem with respect to a computer program is that it is manifested in many ways. First, you have the actual text of the source code and the resulting object code. Secondly, you have the effects of the implementation of this code (screen displays, user interface, not to mention the actual job performed by the program and the manner in which it is performed). Copyright law can be used to protect both the code of the program (source or object), the sequence, structure, and organization of the program, along with the visual displays and user interface. Patent law is generally used to protect the inventive elements within a program, and source/object code disclosures are not necessary in order to obtain patent protection. Even the disclosure requirement of §112 of the Patent Act may be satisfied without the inclusion of the program's code; a flowchart and description of the program can suffice.[7] The question, then, is if we intend to

---

6 David Bender, Anthony Barkume, *Patents for Software-Related Inventions*, 5 Software L.J. 279, 283.

7 see generally *Northern Telecom, Inc. v. Datapoint Corp*, 908 F.2d 931 (Fed.Cir), cert. denied, 111 S.Ct. 296 (1990), where the CAFC looked into several factors to determine whether §112 of the Patent Act was satisfied in regards to claims which dealt with an improved way to enter, verify, and store data with a programmable processor-based batch data entry terminal. The CAFC first noted that in this case, programming was the art that must be considered. Therefore, the important question was whether a skilled programmer, given that disclosed in the application, could carry out the claimed invention in a relatively straightforward way. If this could be accomplished without a disclosure of the source code, such source code disclosure would not be

protect all of the aforementioned manifestations of a computer program, we should clearly indicate this in our IP laws. In addition, we should address the issue as to whether or not registration of only the source code or object code precludes protection of another manifestation, such as the graphic display.

The History of Software Protection in the United States

Patent Protection

The initial resistance towards the patenting of computer programs was based upon the contention that such programs were not the proper subject matter for a patent. The Patent Act states the following regarding the subject matter for a patent:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.[8]

---

necessary. However, the court noted that if the program is unique and is the claimed invention itself, a source code disclosure may be required under §112.

[8] 35 USC §101. For further clarification, see 35 USC §100 which defines "invention" to include discoveries and "process" as a "process, art or method, and includes a new use of a known process, machine, manufacture, composition of matter, or material".

From a plain language reading of this section, one would assume that any useful computer program is proper subject matter for a patent and could therefore be given patent protection provided that the additional requirements of novelty and non-obviousness are met. Furthermore, neither §100 nor §101 expressly require that a process have some form of material embodiment in order to qualify as proper subject matter. The question, then, is why was there initial resistance to the patenting of computer programs?

As Professor Pamela Samuelson explains, the resistance was based on the view that algorithms and related computer programs were not "the sort of "process" that Congress meant to include within the reach of the patent statute"[9]:

> Upon reflection, it is obvious that not everything which is a "process", in the ordinary sense of the word, is meant to be a patentable kind of process. For example, the processes of reading a book, of interpreting a book, and of writing a book are unpatentable processes, [since] Congress meant for the word "process," as it is used in the patent statute, to have a narrower meaning than the ordinary usage of the word might suggest...In determining the patentability of computer programs, the chief doctrine of concern is the "mental Process" or "mental steps" rule.[10]

---

9 Pamela Samuelson, *Benson Revisited: The Case Against Patent Protection For Algorithms And Other Computer Program-Related Inventions*, 39 Emory L.J. 1025, 1033 1990.

10 Samuelson, at 1033.

The mental steps doctrine was well established before the issue of computer software patentability even existed. The doctrine basically states that any process which consists substantially of a series of mental steps shall not be proper subject matter for a patent[11]. An old Supreme Court case, Cochrane v. Deener[12], defined "process" as "including only those processes that transform matter from one physical state to another." This doctrine was relied upon in 1966 when the President's Commission on the Patent System was appointed to evaluate the patent system's ability to deal with rapidly changing technologies. The Commission concluded that computer programs should not be protected under the patent system, due to their unique nature and the resulting inability to search the prior art for a determination of novelty.[13] The Commission provided additional reasons to support its decision:[14]

1. Computer programs were not though to be the type of process which Congress intended to protect through the patent laws,

2. The art of computer programming, having widely developed without the aid of patent protection, did not need the type of protection that patents offered, and

---

11 Samuelson at 1033.

12 Samuelson, fn. 37.

13 Anthony Miele, Bruce Bernstein, *Drafting Claims For Patent Protection Of Software/Computer-Related Inventions In The U.S. In Order To Maximize The Scope Of Protection*, 1 U. Balt. Intell. Prop. L.J. 41, 42 (1992).

14 Samuelson at 1034.

3. Copyright and trade secret protection was available to computer programs.

In response to this, in 1966 the PTO issued guidelines which stated that computer programs are not patentable subject matter unless they could be claimed as a component of a patentable process in which the nonobvious elements are separate from the program itself.[15] In addition, such a process had to produce a physical result.[16] The Court of Customs and Patent Appeals (precursor to the CAFC) initially held that a computer program was patentable as long as "the claim language limited the patent scope to machine implementations."[17] However, in In re Musgrave[18], the CCPA held that the Mental Steps Doctrine, as far as its applicability to computer programs was concerned, was overreaching and inappropriate.[19] The CCPA based its decision on the constitutional purposes behind the patent laws, which is to promote the progress of the useful arts. The new standard for subject matter patentability was for the process to be part of the "technological arts".[20]

What we see here is the beginning of a tension between the CCPA and the PTO. Despite the holding of Musgrave, the

---

15 Samuelson at 1034.
16 Samuelson at 1034.
17 Samuelson at 1035.
18 431 F.2d 882, 167 U.S.P.Q. 280 (C.C.P.A. 1970).
19 Miele and Bernstein at 42.
20 Samuelson at 1036, citing Musgrave at 893.

PTO continued to adhere to its stance that as a series of mental steps, computer programs were not patentable:

> The Musgrave decision, however, did not end the dispute between the Patent Office and the CCPA over patentability issues. In the four cases following Musgrave (but before the Supreme Court's Benson decision), the Patent Office continued to reassert its own definition of what was a patentable "process" and rejected computer program patent applications on the ground that they failed to claim statutory subject matter...The CCPA opinions in these four cases reflect an increasing impatience with the Patent Office and its stubborn resistance to the CCPA's view of what constituted patentable subject matter.[21]

In Gottschalk v Benson[22], the subject matter of the invention was a computerized algorithm in which binary-coded decimal numerals were converted to pure binary form. The claims, however, were construed by the PTO as being overbroad in that they were not limited to any particular art or technology, nor to any particular machinery, implementation, or end use. The Supreme Court agreed and held that the Benson claims did not fit within the definition of patentable subject matter.[23] In so holding, the Court focused on the fact that the claim covered any computerized use of the conversion process and that such process was merely an algorithm.[24] The rationale behind the Court's decision was

---

21 Samuelson, fn 66.
22 409 U.S. 63, 175 U.S.P.Q. 673 (1972).
23 *Benson*, 409 U.S. 63, 71 (1972).
24 *Benson*, 409 U.S. 63, 71 (1972).

that a mathematical algorithm was akin to a phenomenon of nature and thus wasn't proper subject matter.[25]

Despite Benson, the tension between the PTO and CCPA remained. For example, until the next Supreme Court case which dealt with the patentability of computer programs[26], there were 20 cases regarding such subject matter patentability. Of them, the CCPA reversed patent office rejections in 12 of these cases:[27]

> The predominant theme was the CCPA's effort to find an interpretation of Benson that would greatly confine its application (in contrast to the Patent Office, which read Benson broadly). The CCPA's most common interpretation was that Benson forbade only patents on "mathematical algorithms" (by which it generally meant equations), and then only when granting the patent would "wholly pre-empt" use of the algorithm (i.e., when any use of the equation would infringe the patent).[28]

It is important to note that many computer programs are not subject to the Benson preclusion of mathematical algorithms. This is simply because they generally do not contain mathematical algorithms and therefore could not pre-empt an abstract mathematical formula. Such inventions would

---

25 *Benson*, 409 U.S. 63, 71 (1972).
26 *Diamond v Diehr*, 450 U.S. 175, 209 U.S.P.Q. 1 (1981).
27 Samuelson at 40.
28 Samuelson at 40.

include user interfaces, program algorithms, and database structures.[29]

Nine years after the Benson decision, the Supreme Court made its second and last decision regarding the subject matter patentability of computer programs. In Diamond v. Diehr, 450 U.S. 175, 209 U.S.P.Q. 1 (1981), the PTO rejected a method claim for curing rubber with the aid of a computer program. The program utilized a well-known mathematical equation to calculate the total time necessary to cure the rubber, and would automatically open the mold upon the conclusion of that time. In holding that the claims were statutory, the Supreme Court noted that Diehr made no attempt to preempt the use of the equation, but sought "only to foreclose from others the use of that equation in conjunction with all of the other steps in their claimed process."[30] Furthermore, the Court provided a much broader interpretation of §101 of the Patent Act:

> ...when 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 (e.g., transforming or reducing an article to a different state or thing), then the claim satisfies the requirements of section 101.[31]

---

29 Lance L. Vietzke, *Software Patent Protection: A Problem-Solution Theory for Harmonizing the Precedent*, 12 Computer/L.J. 25, 50.

30 Samuelson, fn. 271.

31 *Diamond v. Diehr* at 193.

This statement seems to have been the basis behind the PTO's current acceptance of computer programs as patentable subject matter. For example, in 1987 3200 computer patent applications were filed with the Patent and Trademark Office; in 1990, 6500 were filed.[32]

Copyright Protection

Critics of copyright protection for computer software are quick to point out how much the element of timing may have affected the decision of Congress to amend the copyright laws to include computer programs as copyrightable subject matter. In 1975, Congress authorized the formation of CONTU (National Commission on New Technological Uses of Copyrighted Works). CONTU's first job was to study the legal issues which arose from computer software and other new, emerging technologies.[33] After three years, CONTU recommended amending the Copyright Act (which was recently amended in 1976). As a result, §101 of the Copyright Act was amended to include computer programs as copyrightable subject matter and defines a computer program as "a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result."[34] It is important to note that during this time, patent protection for computer software was believed to be unavailable, since Diehr (which held that a

---

32 Busse, *Patents Gain Favor with Software Firms; Vendors Slow to Adopt Old Weapon*, Infoworld, Aug. 26, 1991 at 82.

33 Max W. Laun, *Improving the International Framework for the Protection of Computer Software*, 48 U.Pitt.L.Rev. 1151, 1151.

34 17USC§101.

process claim that includes the use of a computer algorithm does not necessarily preclude patent protection) had not yet been decided by the Supreme Court:

> In practice, from 1964 until well after the Diehr decision in 1981, most of the software community viewed copyright as the only form of intellectual property protection available for software. Given this situation, it is not surprising that courts broadened the scope of software copyright protection in order to protect against misappropriation.[35]

Bender and Barkume agree:

> We do note yet know the full scope of copyright protection. The cases have extended copyright father in the case of programs than for other types of subject matter and perhaps there is more elasticity. But so long as 17 USC §102(b)[36] remains in the copyright statute, and so long as we are governed by decades of case law on the appropriate scope of copyright, there is a limit to how far the copyright fabric can stretch. And when it is through stretching, it will still be nowhere near the full ambit

---

35 Willis E. Higgins, *Technological Poetry: The Interface Between Copyright and Patents for Software*, 12 Hastings Comm/Ent L.J. 67, 71. In particular, Mr. Higgins refers to Whelan Associates v. Jaslow Dental Laboratories, 797 F.2d 1222 (3rd Cir. 1986), cert. denied, 479 U.S. 1031 (1987), where "the court held that anything more specific than the general purpose of a software program is protected expression." In other words, the court determined that the copied elements of the program were to be protected and they could only be protected if they were arbitrarily defined as an expression. Higgins notes that the Whelan test has been widely criticized as being overbroad. For example, in Computer Associates International Inc. v. Altai Inc., 775 F.Supp. 544 (E.D.N.Y.1991), the court noted that the Whelan structure, sequence, and organization test is "inadequate and inaccurate." The Altai court stated that it is inappropriate to simplify a computer program into one big idea, while categorizing all methods used to accomplish that idea as protectable expression.

36 "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." [underline added]

> of patent protection, because patents were meant to protect such things as processes, systems, etc.[37]

An example of this stretching of the copyright fabric can be seen in Whelan Associates v. Jaslow Dental Laboratories[38], where the 3rd Circuit held that anything more specific than the general purpose of a software program is the protected expression. This overbroad and highly criticized test reflects the inability of the copyright system in developing a suitable procedure for defining the protectable scope of technical subject matter.[39]

It is quite possible that had the PTO and the courts viewed computer programs as patentable subject matter during the CONTU study, there would have been no recommendation that the Copyright Act be amended to include computer programs.

---

37 David Bender, Anthony Barkume, *Patents for Software-Related Inventions*, 5 Software L.J. 279, 283.

38 797 F.2d 1222 (3rd Cir. 1986), cert. denied, 479 U.S. 1031 (1987).

39 Willis E. Higgins, *Technological Poetry: The Interface Between Copyright and Patents for Software*, 12 Hastings Comm/Ent L.J. 67, 80. Mr. Higgins goes as far as to state: "In effect, a judge or jury in a software copyright infringement action in federal court must exercise the role of a patent Examiner and determine de novo the scope of a software copyright." He also acknowledges that IP counsel find it very difficult to give their clients accurate opinions on copyright infringement, due to the current state of the law.

Criticisms of US Patent Law

Inability to conduct a proper prior art search

One of the major criticism of using Patent law to protect computer programs is more administrative than it is substantive. In a nutshell, it is argued that patent protection, although best suited for computer programs, fails to adequately apply the standards of novelty and non-obviousness in a determination of software patentability. The main reason for this is the fact that it is virtually impossible to perform a proper prior art search:

> The search for prior art in examining software-related claims is also in its initial stages; institutional difficulties challenge the establishment of a repository of prior art and administration of patent prosecution."[40]
>
> ...prior art libraries are limited, the search classification system was designed for hardware patents, few computer scientists are examiners.[41]

However, it should be mentioned that the PTO is taking measures to improve the system, such as improving its prior art search facilities in software, publication of a new software classification system, and the active recruitment of computer scientists.[42]

---

40 Sinn, 40 Fed.B.News & J. 226, referring to OTA report

41 Paul Heckel, *The Software-Patent Controversy*, 9 No. 12 Computer Law. 13,18.

42 Paul Heckel, *The Software-Patent Controversy*, 9 No. 12 Computer Law. 13,18.

Granting of overbroad patents

One of the real dangers resulting from the inability to perform a proper prior art search is the granting of overbroad patents:

> under fire from industry executives for giving the green light to questionable software patent applications, the U.S. Patent Office is implementing dramatic reforms in the way that it handles technology patents. At the heart of the changes taking place this summer is a move to make patent examiners into software experts, so that they can more easily spot shoddy patent applications and handle a greater work load.[43]

Criticisms of US Copyright Law

Copyright law should only protect literary and artistic works

The main criticism of copyright law is its use to protect utilitarian and functional aspects of a computer program. There is a plethora of copyright cases which uniformly hold that copyright protection only extends to the original expression of a work--not to the functional aspects of the work.[44] A great majority of computer programs, ranging

---

43 S. Burke, *U.S. Agency Revamps Software Patent System; U.S. Patent Office*, P.C. Week, May 27, 1991 at 145.

44 See generally *Mazer v. Stein*, 347 U.S. 201, 74 S.Ct. 460, 98 L.Ed. 630 (1954). In *Mazer v. Stein*, it was acknowledged, however, that a copyrighted work of art does not lose its protected status merely because it subsequently is put to a functional use. See also *Sega v. Accolade*, 977 F.2d 1510 (9th Cir, 1992), where it was held that disassembly of object code in order to gain access to the unprotected ideas and functional elements embodied in a copyrighted

from operating systems to word processors have copyright protection. However, as argued by Peter Menell,

> In order to encourage socially desirable technological innovation, legal protection should be tailored to protect the socially valuable aspect of the intellectual work. Unlike traditional subjects of copyright protection--literary and artistic works--computer operating systems are not valued for their expression per se. Operating systems create value through their utilitarian functions--their ability to direct the inner workings of computer systems. Yet copyright law protects only the expression of an idea rather than the idea itself. Therefore copyright protection does not protect the valuable part of operating systems. Consequently, copyright protection does not in general greatly encourage software developers to invent better operating systems.[45]

Look and Feel

A line of copyright cases has developed over the years, and it involves the protection of the "look and feel" of computer programs. "Look and feel" refers to how the program interacts with the user, from the selection of menu bars to the way information is displayed on the screen. The basis behind copyright protection is that "Traditional copyright law has recognized that nonliteral as well as literal aspects of a literary work are protected."[46] This means that

---

computer program constitutes fair use if there is a legitimate reason behind such disassembly.

45 Peter Menell, *Tailoring Legal Protection For Computer Software*, 39 Stan.L.Rev. 1329, 1348.

46 Peter Hohenhaus, *An Introductory Perspective on Computer Law: Is it, Should it be, and how do we best Develop it as, a Separate Discipline?*, 1991 WL 330761.

infringement may occur even though the underlying code of the two software packages are not at all similar. As long as the "look and feel" of the two packages are similar, this may be a basis for a finding of infringement. The arguments for and against copyright protection of "look and feel" are presented as follows:

> At one extreme, some might argue that all screen displays of a software program, and all their component parts and arrangements, are and should be protected by the copyright in the program. Developers put a lot of work into creation of user interfaces and screens; these aspects of a program are no less important than the literal code underlying it. On the other extreme, some might argue that nothing about the screen displays of a software program, or their component parts, should be protected. As long as the underlying code is different, no one should be able to "monopolize" or control use of visual devices, screen and menu arrangements, and menu command word choices. It then follows that there may be a small number of, a finite limit to, original approaches to the kinds and arrangements of visual outputs from programs, and that therefore ideas and expressions merge and copyright protection should not apply.[47]

In addition to this "merger doctrine", there is the "conceptual separability doctrine" which states that "If design elements reflect merger of aesthetic and functional considerations, artistic aspects of work cannot be said to be conceptually separable from utilitarian elements, and thus

---

47 Peter Hohenhaus, *An Introductory Perspective on Computer Law: Is it, Should it be, and how do we best Develop it as, a Separate Discipline?*, 1991 WL 330761.

work is not copyrightable, but where design elements can be identified as reflecting designer's artistic judgment exercised independently of functional influences, conceptual separability exists, and the work is copyrightable."[48] Under this doctrine, it may be argued that many seemingly aesthetic elements of a program, such as "look and feel" are not copyrightable because they are dependent upon the utilitarian function of an efficient user interface. Most programmers would probably agree that when designing a pull-down menu or interface system, the prime consideration is not one of artistic aesthetics, but one of efficiency and ease of use. As argued later, it seems that such considerations are substantially functional in nature. Since "the true test of functionality is not whether the feature in question performs a function, but whether the feature 'is dictated by the functions to be performed,'"[49] it seems that most "look and feel" features should not be protectable under copyright law.

Perhaps it would be helpful to go back to basic property principles, as well as the goals behind our IP laws. If an element within a computer program has value, its owner will take measures to protect it. Given the attention that "look and feel" has received in the courts, it is safe to conclude

---

48 *Brandier International, Inc., v. Cascade Pacific Lumber Co.*, 834 F.2d 1142, 1145 (1987). In this case, the Second Circuit affirmed the lower court's decision that the shape of a bicycle rack was not copyrightable because it was dependent upon functional concerns, not aesthetic ones.

49 *Brandier International, Inc., v. Cascade Pacific Lumber Co.*, 834 F.2d at 1148.

that in certain computer applications, "look and feel" is a valuable component within the program. However, just because an intellectual creation has value does not mean that it should automatically be protected under the existing laws. It is important to remember that copyright law is designed to protect literary and artistic creations. Within this context, it makes a lot more sense to only require a modicum of creativity in order to receive copyright protection. In addition, a long duration of protection is more appropriate for such artistic creations. Under Patent law, a much higher standard of novelty and non-obviousness is required in order to obtain protection. Again, this makes sense because Patent Law is concerned with the protection of useful inventions. Since the protection is an absolute right to exclude for 17 years, it is important to ensure that only inventions that are not in the public domain and not obvious to those skilled in the art are given such strong protection. The much shorter duration of protection reflects the need to quickly place these new inventions within the public domain. The question, then, is whether "look and feel" is artistic (and thus properly protected under copyright law), or utilitarian (and thus properly protected under patent law). I argue that it is primarily utilitarian because it is basically a tool which enables a user to efficiently communicate with its computer. Yes, it may use certain screen images which seem primarily artistic, but the real value behind the "look and feel" of a computer program is the quality of the user

interface. Having established that it is primarily utilitarian (and thus useful), I ask whether or not it is appropriate to subject this useful tool to the lower originality requirements of copyright law, along with the extended duration of copyright protection. Wouldn't it be more appropriate to subject the "look and feel" to a patent analysis? If this were done, then it would be much more difficult for a developer to monopolize screen and menu arrangements, along with menu command word choices. This is important since it is quite probable that there are a limited number of ways to present a user interface for a given application. On the other hand, if we elevate the requirements for "look and feel" protection, would this discourage developers from putting in the time and effort required for a "better" user interface? Perhaps the solution is to develop a hybrid form a protection for "look and feel" with its own standard of originality.

Object code disclosure alone does not meet the Constitutional requirements of disclosure

Early critics against copyright protection for computer software argued that in order for a work to obtain copyright protection, it must be intelligible to a human being in the

form in which it is protected.[50] This would have meant that programs only in machine-readable form should not be granted copyright protection. The rationale behind this argument is that the Constitution requires that intellectual property law "promote the progress of Science and the Useful Arts." This doctrine is reflected in our IP laws by the disclosure requirement. If one merely discloses the object code of a computer program in return for a copyright, is that disclosure the type that would promote the progress of science and the useful arts?

> ...it was argued that machine-readable versions of computer programs should not be copyrightable because 'copyright registration of object code discloses almost nothing in return for the protection of the law.'[51] While a book, of necessity, conveys the knowledge it contains, a computer program may not...because it reveals nothing to the human observer.[52]

Of course, virtually all courts have held that computer programs in machine-readable format are proper subject matter for copyright.[53] It may also be argued that the constitutional requirement of disclosure need not only be fulfilled by introducing human-readable source code into the

---

50 Max W. Laun, *Improving the International Framework for the Protection of Computer Software*, 48 U.Pitt.L.Rev. 1151, (36)

51 R. Saltman, *Copyright in Computer-Readable Works*, 62 (1977).

52 Max W. Laun, *Improving the International Framework for the Protection of Computer Software*, 48 U.Pitt.L.Rev. 1151, (37)

53 See, e.g., Apple Computer, Inc. v. Franklin Computer Corp., 714 F.2d 1240, 1247-48 (3d Cir. 1983).

public domain. By providing the public with a useful product (such as a word processor, spreadsheet, etc.) it may be argued that the useful arts are being promoted. This is especially the case where a particular aspect of a computer program is emulated by other programmers in the industry. In most cases, this type of dissemination does not require an examination of the source code.

However, in Sega v. Accolade, the Ninth Circuit believed that as far as computer games are concerned, mere dissemination of the game itself was not enough.[54] The Court reasoned that since the plaintiff's cartridge games were distributed to the public in only object code format (and thus generally not examinable by the public), this precluded "public access to the ideas and functional concepts contained in those programs, and thus confers on the copyright owner a de facto monopoly over those ideas and functional concepts."[55] An interesting feature of this case is that despite the fact that the copied text only reflected the functional aspects of the program (initialization sequences which rendered the

---

54 See generally, *Sega Enterprises, LTD., v. Accolade, INC.*, 977 F.2d 1510 (9th Cir, 1992), where the court held that defendant's reverse-engineering and copying of plaintiff's copyrighted object code constituted fair use. The defendant only copied functional elements of the code in order to develop its own games for plaintiff's "Genesis" game console. The court noted that since the defendant had no other way to make Genesis-compatible games, and the copied code was solely functional, it did not infringe upon plaintiff's copyrights in its games. The court was also influenced by public policy, stating that the ultimate aim of IP laws is to "stimulate artistic creativity for the general public good. When technological change has rendered an aspect or application of the Copyright Act ambiguous, 'the Copyright Act must be construed in light of this basic purpose.' "

55 *Sega Enterprises, LTD., v. Accolade, INC.*, 977 F.2d at 1527.

program compatible with the Sega Genesis system), the fact remained that the text itself was copyrighted and the defendant copied it. In order to find a loophole for the defendant, the court was forced to use the theory of fair use (which is a defense to copying) instead of finding that the copied text wasn't protectable under copyright since it represented these functional elements.

Focus on authors may be detrimental to the dissemination and improvement of computer programs

In his article on the definition of "authorship" and its influence on our copyright laws[56], Peter Jaszi points to certain elements within the copyright laws which may be harmful to the dissemination and improvement of computer programs. Professor Jaszi states:

> The "authorship" concept, with its roots in notions of individual self-proprietorship, provided the rationale for thinking of literary productions as personal property with various associated attributes including alienability. But the emphasis on "authorship" left questions about the scope of "authorship" rights unanswered. In effect, "authorship" reproduced the fundamental contradiction between control and access. A stress on the interests of past "authors" could generate arguments for broad copyright protection, while an emphasis on the interests of future "authors" could generate equally compelling arguments for strict

---

56 See generally, Peter Jaszi, *Toward A Theory of Copyright: The Metamorphoses of "Authorship"*, 1991 Duke L.J. 455.

limitations on the scope of copyright protection.[57]

Professor Jaszi then reveals that early infringement cases under the Statute of Anne focused not on what the defendant copied, but on what he added or contributed. At that time, infringement would only have been found upon a showing of verbatim copying. However, due to the Romanticized concept of "author", and his rights in his work as a whole, "No plagiarist can excuse the wrong by showing how much of his work he did not pirate."[58] The question, then, is whether this attitude towards authorship will eventually be harmful to the software industry. As far as useful computer programs are concerned, is the notion of "author" an appropriate one when determining the scope of his rights in the work? If the current focus of courts is not on the improvements, but on the amount copied, this could eventually become problematic by prohibiting the entire software industry (aside from the copyright owner) from making small, incremental improvements to the protected software.[59] One can argue that copyright law limits the scope of the author's rights by preventing the protection of

---

57 Jaszi, at 462.

58 Jaszi at 462, quoting Benjamin Kaplan.

59 This problem is compounded by the fact that §106 of the Copyright Act provides the right to make derivative works as an Exclusive right. It seems that this right is heavily influenced by the notion of "author". Compare this to §154 of the Patent Act, which provides that a patentee only has the right to exclude others from praticing his invention. This limitation of rights reflects the need to freely allow others to make improvements upon the patented invention. Given the give and take nature of the software industry, it seems that this aspect of Patent law is much better suited to the industry's needs.

utilitarian works, but as far as software copyrights go, this "limitation" has not been adequately observed.

Suggestions for amending the current framework in the United States

Of course, there are many practicing attorneys who are justified in their contempt of a sui generis system. Their basic argument is that the given framework may not work with 100% efficiency, but their training permits them to best utilize a combination of patent, copyright, and trade secret laws which more than adequately protect their clients' interests. Their argument does have its merits: Indeed, a sui generis system, even if in the form of an amendment of the current IP laws, will cost money to implement. In addition, there is no guarantee that such a system will work better than the current one. Finally, isn't it the job of an attorney to know the law and to best utilize the law for his clients' interests? In other words, shouldn't the work be left to the lawyers and not the legislators?

With this in mind, I admit that the creation of an entirely independent sui generis system for the protection of software is not justifiable, given the present realities. First of all, an independent system would require that all current software patents and copyrights be refiled under the new system. Understandably, many currently protected items

of software will not be protected under the new system and we will be faced with a takings problem. Secondly, there isn't much evidence to indicate that the US software industry is suffering from the current system.[60] My proposal, therefore, is to suggest ways in which our current IP laws may be amended in order to better accomodate computer programs.

In General

1. We should clarify that a computer program's utilitarian aspects may only be protected under Patent or Trade Secret Law, and its literary or artistic aspects may only be protected under Copyright law. This may be facilitated by expressly specifying that §102(b) of the Copyright Act applies in full force to computer programs. In addition, §102(a) should expressly include the category of computer programs as independent works of authorship, instead of having them categorized as "literary works". Since many copyrightable programs of today are most akin to audiovisual works, it doesn't make much sense to categorize them as literary works.

---

60 See generally, Burton, *Can U.S. Software Industry Hold Its Lead?; Piracy and Patent Issues Worry Domestic Firms Despite 60% Share of Total Market,* Investor's Daily, Apr. 17, 1990. Fn. 1 of this article states that the United States Commerce Department predicted in 1990 that the software industry will continue to grow by 20 to 35% each year throughout the 1990s. In addition, the article states that U.S. software companies held 60% of the global software market in 1990 and it is estimated that U.S. companies develop 70% of the software sold in Europe and 50% of the software sold in Japan.

2. We should clearly delineate what manifestations of a computer program are to be protected under Copyright law, and what manifestations should be protected under Patent Law. Looking at the disclosure requirements of the Copyright law can provide a key here:

> Disclosure is not an integral part of the policy of copyright protection. The focus of the Copyright Act is the regulation of the reproduction, display, performance, and distribution of protected matter; access to the protected material and its dissemination to the public is not a primary requirement under the statute.[61]

On the other hand, Patent law is primarily concerned with the disclosure of an invention, since the main thrust of the act is to enable others to practice the invention.[62] Noting that copyright law and patent law have different goals, these differences should be used to delineate which manifestations of a program are to be protected under which system. In other words, the way something is disclosed to the public might determine the type of protection it receives. Under this analysis, source/object code (or the sequence, structure and organization of the code) should be protected, if at all, under patent law, while actual displays

---

61 Leo J. Raskind, *The Uncertain Case For Special Legislation Protecting Computer Software*, 47 U. Pitt. L. Rev. 1131, 1134.

62 35 U.S.C. § 112, Specification, states in part: "The specification shall contain a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art to which it pertains, or with which it is most nearly connected, to make and use the same, and shall set forth the best mode contemplated by the inventor of carrying out his invention."

and presentations, if at all protectable, should be done so under copyright law. My rationale is that source code is akin to the inner workings of a machine. If it has any value in and of itself, it is because it teaches a new invention, a new way to do something. For example, if a computer game utilizes a new way to quickly process graphic information, this method will be disclosed by the source code and should be considered an invention subject to a patentability analysis. If any of the game's displays and visuals are to be protected, it seems that copyright law is more appropriate.

Of course, when exposed to a much more complicated program my analysis may suffer. For example, if you have a new graphic interface (similar to Windows or the Macintosh interface) which displays information in an extremely easy to decipher way, how do you draw the line? First of all, the value of the display lies in the way it presents information, or its function. Unfortunately, the way the display works can be disclosed by either the source code (patent) or the actual screen shots (copyright). Under a conceptual separability analysis, one must ask what visual aspects of the display are dictated by its function. Those which are dictated by this function should not be protected by copyright. What remains will be subjected to a novelty and non-obviousness analysis under Patent law.

Procedural Requirements

1. You get to protect only that which you disclose. Under copyright law, if you only disclose certain screen shots, storylines, and other elements of expression, those should be the only elements within a program that you get to protect. If you wish to obtain more substantial, patent-like protection of certain processes or methods of operation, you should disclose enough to enable one skilled in the art to practice your process or invention.

2. In order to better determine the scope of protection, it has been suggested that a much more extensive filing system and prior art library be developed. The purpose behind this is two-fold:

> A comprehensive collection of the existing body of software knowledge, if available to the public, would provide an invaluable resource in the research and development of new software inventions...by requiring an extensive package describing the software to be registered. For example, those seeking protection would be required to provide a complete copy of the source code. In addition, other items such as flow diagrams, interface descriptions, and a video tape of an actual run-time session would help to define the system.[63]

Since many courts have had difficulty in determining the scope of protection for a copyrighted computer program, perhaps it would be helpful to require, as in a patent

---

63 John C. Phillips, *Sui Generis Intellectual Property Protection for Computer Programs*, 60 Geo.Wash. L. Rev. 997, 1020.

application, that the applicant and copyright examiner agree upon the scope of protection. This would provide the applicant with incentive to disclose that which he/she believes is valuable within the program, and would reduce the amount of confusion that the courts have been faced with in determining the scope of protection for a given piece of software.

3. The duration of protection for computer programs should be three to five years. This is because (1) given the rapid development of the software industry, a one-year lead time should be enough to provide a developer with the advantage of a significant market share, and (2) within a short time after marketing their products, most developers realize a return to research and development.[64]

4. A mandatory reasonable royalty license should be available in regards to computer programs. In addition, incentives should be provided to encourage parties to enter licensing agreements before any constructive arrangements are created through the courts. Examples of such incentives might be a lower license rate for licensees that initiate their own agreements, or the imposition of costs and/or

---

64 John C. Phillips, *Sui Generis Intellectual Property Protection for Computer Programs*, 60 Geo.Wash. L. Rev. 997, 1020.

attorney's fees to parties which choose a wait and see attitude in regards to copying others' protected software.[65]

Conclusion

In all reality, if the current framework for protecting computer programs is not changed by Congress, the courts will eventually find ways to ensure that misappropriation does not occur. This has been the practice in many copyright cases although many inconsistent tests have been applied in determining the scope of copyright protection for computer programs. The value of a statutory amendment to our current intellectual property laws is that the courts will be given a clearer picture as to the scope of protection and will become more efficient in preventing misappropriation. In addition, potential innocent infringers will have a better idea as to what activities are permissible.

---

65 John C. Phillips, *Sui Generis Intellectual Property Protection for Computer Programs*, 60 Geo.Wash. L. Rev. 997, 1021. Other advantages of such a system would be to reduce the number of potential infringement actions, and to "provide competitors in the industry to have access to, and improve upon, each other's ideas."