Hastings Communications and Entertainment Law Journal

Volume 14 | Number 2 Article 7

1-1-1991

The Case for Software Patent Protection

Willis E. Higgins

The Case for Software Patent Protection

by
WILLIS E. HIGGINS*

Table of Contents

* Partner, Flehr, Hohbach, Test, Albritton & Herbert, Palo Alto, California. B.S. in Ch.E., University of Pittsburgh, 1962; J.D., University of Chicago Law School, 1965.

Introduction

Intellectual property protection for computer software in the U.S. is available in a number of forms. Software can now be protected by both patents and copyrights in the U.S., with some software copyrights being given a broad scope of protection against non-literal infringement, such as by a user interface.[1] Does the extent of currently available intellectual property protection for software through patent and copyright represent a disaster for the software community? A segment of that community strongly believes that it does.[2] Those beliefs, however, are not universally accepted there.[3] A review of the arguments advanced for abolishing patent protection for software and copyright protection for user interfaces leads to the conclusion that the case for such abolition remains unproven. Software patents have an important role to play in providing protection against misappropriation of inventive concepts that is more appropriate than an expanded scope for software copyrights.

I
Background

The basis for the U.S. patent and copyright laws is found in the Constitution as follows: "The Congress shall have 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."[4] It should be noted that this provision does not require the existence of such protection. It merely authorizes Congress to provide the protection for the specific purpose of promoting progress in science and useful arts. It therefore cannot validly be maintained that intellectual property protection for software is constitutionally required. However, the presence of this provision in the Constitution shows that there is a strong bias in our fundamental law for intellectual property protection.

This clause in the Constitution was adopted without debate, and there is thus little material explaining it. A comment by James Madison

1. *See generally,* Willis E. Higgins, *Technological Poetry: The Interface Between Copyrights and Patents for Software,* 12 HASTINGS COMM/ENT L.J. 67 (1989).

2. *See* Brian Kahin, *The Software Patent Crisis,* TECH. REV., Apr. 1991, at 52; THE LEAGUE FOR PROGRAMMING FREEDOM, AGAINST USER INTERFACE COPYRIGHT (1990); THE LEAGUE FOR PROGRAMMING FREEDOM, AGAINST SOFTWARE PATENTS (1990). Both of these articles by the League for Programming Freedom are available on-line through Internet at league prep.ai.mit.edu.

3. PAUL HECKEL, THE ELEMENTS OF FRIENDLY SOFTWARE DESIGN 223-94 (2d. ed. Sybex 1991).

4. U.S. CONST. art. I, § 8, cl. 8.

in *The Federalist No. 43* reveals that the public policy behind both copyright and patent protection is the same:

> The utility of this power will be scarcely questioned. The copyright of authors has been solemnly adjudged, in Great Britain, to be a right of common law. The right to useful inventions seems with equal reason to belong to the inventors. The public good fully coincides in both cases with the claims of individuals.[5]

This public policy was explained in more detail by the Supreme Court as follows:

> The economic philosophy behind the clause empowering Congress to grant patents and copyrights is the conviction that encouragement of individual effort by personal gain is the best way to advance public welfare through the talents of authors and inventors to "Science and useful Arts." Sacrificial days devoted to such creative activities deserve rewards commensurate with services rendered.[6]

The development of this country from an agrarian society to the world's preeminent economic power is one indicator of the success of this policy. This constitutional basis for intellectual property laws has resulted in a strong commitment by the federal judiciary to protect the work products of authors and inventors against misappropriation.

Arguments against intellectual property protection for software continue to remain outside of the intellectual mainstream. At issue is the degree and nature of such protection. Any doubt that computer software was covered by the copyright laws was foreclosed by the 1980 amendments to the Copyright Act of 1976,[7] which specifically provides for software as copyrightable subject matter.[8] In contrast, patent protection for software was generally thought not to be available prior to a 1981 Supreme Court decision holding that patent protection was available for a rubber curing process implemented with software.[9] It would be inaccurate to claim that the U.S. Patent and Trademark Office "unilaterally" decided in response to the *Diehr* decision to start issuing large numbers of software patents. This claim ignores the role of the Court of Appeals for the Federal Circuit, which hears patent appeals, both from the Patent and Trademark Office and the U.S. district courts, and its predecessor court, the Court of Customs and Patent Appeals. The *Diehr* decision represented an acceptance by the Supreme Court of positions taken by the Court of Customs and Patent Appeals and subsequently confirmed by the Court of Appeals for the Federal Circuit. In any case, there is no real doubt that both copyright and patent protection are presently avail-

5. THE FEDERALIST NO. 43, at 279 (James Madison) (Mod. Lib. ed., 1941).
6. Mazer v. Stein, 347 U.S. 201, 219 (1953).
7. 17 U.S.C. §§ 101-914 (1982 & Supp. 1987).
8. *Id.* §§ 101-102, 117.
9. Diamond v. Diehr, 450 U.S. 175 (1981).

able for software. It is in light of this background that one should consider not whether or not software is proper subject matter for patent protection, but whether or not it should be.

II
A Response to Mr. Stallman

Mr. Stallman has presented a number of arguments against patent protection for software. Two arguments dominate his analysis. First, he argues that the Patent and Trademark Office has mistakenly issued patents for software on various occasions.

Mr. Stallman refers to U.S. patent number 4,736,308 — dealing with nested scrollable objects — which was the subject of litigation between Quickpoint Systems and Apple Computer Corporation. The argument advanced is that both scrolling and subwindows were known in the art and that it was therefore inappropriate to issue a patent for merely combining these elements. Most inventions, however, are combinations of old elements and such combinations are patentable unless they are suggested by the prior art.[10] The bottom line in this case was that Apple apparently felt there was some merit to Quickpoint's position. Apple settled the litigation by taking a license for a substantial payment, the amount of which is confidential, a common practice for such agreements.[11]

If a patent is believed to have been issued by the Patent and Trademark Office by mistake because the Office lacked the best prior art during the examination process, the patent can be reduced or eliminated by someone coming forth with the better prior art.[12] Concerns about the Patent and Trademark Office issuing patents in ignorance of commonplace software techniques are best addressed by efforts to insure that the Patent and Trademark Office has the best art available. If the position is simply that of disagreeing with the decision of the Patent and Trademark Office to issue the patent over the prior art that was considered, it is hardly a compelling argument for categorizing a patent as absurd. A truly absurd patent will not be honored by the concerned technical community. One need only read the "Letters to the Editor" in any weekly issue of *Science* magazine to see that disagreements on scientific and technical issues are commonplace. Better prior art than that considered by the Patent and Trademark Office can be called to the attention of the

10. 35 U.S.C. § 101 (1988) (defining patentable subject matter); 35 U.S.C. §§ 102-103 (1988) (setting the standards of novelty and non-obviousness).

11. HECKEL, *supra* note 3, at 265.

12. HERBERT F. SCHWARTZ, PATENT LAW AND PRACTICE 25-26 (1988).

Patent and Trademark Office through requests for re-examination[13] or made available to parties who are accused of infringing a patent.

Reference is also made to U.S. patent number 4,555,775, which is owned by AT&T and which deals with the use of "backing store" in a window that lets multiple programs have windows. Mr. Stallman argues that the patent covers a technique that was obvious to those skilled in the art and that the issuance of a patent was therefore inappropriate. Again, this is merely a factual argument. The existence of controversy over patents issued by the Patent and Trademark Office does not mean that patents for software should be abolished.

A second argument is that the burden of software patents will adversely affect the industry. Mr. Stallman argues that software costs less to write, market, and sell than other products of similar complexity, but will contain just as many possibly patented techniques. Due to the considerable transactional cost of obtaining and enforcing a software patent, however, it is unlikely that excessive use of patents will occur. Software patents are among the most expensive patents to obtain since they deal with subject matter that is esoteric in nature. A typical software patent will cost $10,000 or more just for preparation and prosecution. In addition to the monetary cost, the transactional cost includes a substantial amount of time that the inventors must spend with a patent attorney helping the attorney prepare and prosecute the application. Inventors must therefore be selective in their decisions to seek patent prosecution for potentially patentable techniques in software. Claims that these transactional costs unfairly burden the small or entrepreneurial enterprise ignore the fact that patents may be the only way for such an enterprise to adequately protect itself from a larger competitor. Without patent protection, a larger competitor may be free to use whatever new concepts are developed by others and utilize their superior resources to take over the market. This is no less the case with software than it is with other industries.

Ultimately, the difficulty with all of these arguments is that they can be applied to virtually any technology. If taken to their logical conclusion, the patent system should be abolished and all inventions should be in the public domain.

III
The Need For Patent Protection for Software

Both copyright and patent protection should be available for software because, quite simply, both forms of protection are appropriate.

13. *Id.*

Software is both literary and technological, expressive and inventive. As discussed above, the U.S. legal system has a strong bias against misappropriation of intellectual property. When the courts are convinced that misappropriation has taken place, they will fashion a remedy. Since patent protection was perceived not to be available for much software until at least 1981 when the Supreme Court ruled in *Diamond v. Diehr,* the courts have had to look to copyright for preventing misappropriation. The result has been such controversial decisions as *Whelan Associates v. Jaslow Dental Laboratories*[14] and *Lotus v. Paperback Software.*[15] The problem with such cases is that deciding if copyrighted expression has been misappropriated in a given situation is very difficult. As stated in the *Lotus* opinion:

> It seems the better part of wisdom, if not valor, not to press the search for a suitable bright-line test of copyrightability where Learned Hand, even after decades of experience, found none.
>
> By its nature, a legal test that requires weighing of factors or elements such as these is not a bright-line or an either-or test. It requires of the decision maker, instead, an evaluative or "judgmental" weighing of all relevant characteristics of the work in which a copyright is claimed, all relevant characteristics of the allegedly infringing work, and all of the relevant circumstances of their development and use. It requires, also, not a step-by-step decision making process, but a simultaneous weighing of all the factors or elements that the legal test identifies as relevant.[16]

The defendants in the *Lotus* case had argued for a bright line rule that only source and object codes of a computer program could be covered by a copyright, never non-literal elements, such as a user interface.[17] In support of their proposed bright line rule, the defendants asserted the same kind of arguments that are advanced against software patents — that software was a different kind of work and that the usual intellectual property protection rules therefore should not apply to it.[18] After a review of the congressional intent expressed in the 1980 revisions to the Copyright Act specifically extending copyright protection for software, Judge Keeton refused to accept the position that software should be treated differently than other copyrighted works.[19]

The *Lotus* decision is in accordance with well-established copyright principles. The conflicting policy issues argued to the court were largely considered in the 1976 and 1980 revisions to the Copyright Act, and the

14. 797 F.2d 1222 (3d Cir. 1986), *cert. denied,* 479 U.S. 1031 (1987).
15. 740 F. Supp. 37 (D. Mass. 1990).
16. *Id.* at 60-61.
17. *Id.* at 45.
18. *Id.*
19. *Id.* at 54.

resulting legislative history shows a congressional intent to treat software in essentially the same manner as other copyrighted works.[20]

The use of a three part evaluative test, as in *Lotus*,[21] for determining non-literal infringement in software cases means that it will continue to be very difficult for intellectual property counsel to give definitive infringement opinions to their software clients. Trying software infringement cases will continue to be demanding for counsel and the courts. Determining an appropriate scope of protection for a software copyright, whether for an infringement opinion or in litigation, requires a detailed examination of the scope and content of prior software to the copyrighted work and a similar detailed comparison of the copyrighted and alleged infringing works. In most other forms of copyrighted works, judges and juries have a frame of reference to carry out such evaluations. In contrast, the highly technical nature of much software is strange and incomprehensible to those not skilled in the art. The absence of any significant prosecution history, administrative record in copyright registration proceedings, or any definition of the copyright protection comparable to those available with patent claims makes such decision-making more difficult than in comparable patent infringement opinions and litigation. In the patent application process, the claims defining the scope of the invention asserted by the applicant are evaluated by the Patent and Trademark Office examiner both for their definiteness and for stating a patentable distinction over the prior art, and the claims in the issued patent are the result of this examination process.[22]

Because copyright protection is inexpensive and easy to obtain, it has been the dominant form of intellectual property protection for software. Essentially every program is copyrighted. If software patents are abolished, the software industry, the courts and intellectual property counsel will continue to be faced with the difficulties of determining the scope of copyright protection in non-literal infringement cases. The use of software patents to provide coverage for processes and systems embodied in software will reduce the need to broaden copyright protection for preventing misappropriation. Both copyright and patent protection for software are needed to provide a balanced system of intellectual property protection for the literary and the technological components of software.

20. In 1980, Congress amended the Copyright Act of 1976, incorporating a definition of "computer program," thereby including software as a copyrightable work, Act of Dec. 12, 1980, Pub. L. No. 96-517, § 10(a), 94 Stat. 3015, 3028 (Copyright Act of 1980)(codified as amended at 17 U.S.C. § 101 (1982)).

21. *Id.*

22. SCHWARTZ, *supra* note 12, at 13.

In short, the perceived unavailability of patent protection for software has caused courts to provide protection through principles of copyright instead. The resulting protection, however, is problematic due to the inappropriateness of addressing issues of invention and technology under a system intended to address issues of expression. The availability of patent protection for software allows courts to provide the appropriate means of protection.

IV
Conclusion

The copyright system has provided literary works with strong intellectual property protection. Likewise, the patent system has provided technologies with strong intellectual property protection. Since software is both literary and technological, both forms of protection are appropriate for it. At this point, the case against software patents remains unproven.