ASPEN PUBLISHERS

The Computer & Internet Lawyer

Volume 27 ▲ Number 6 ▲ JUNE 2010

Ronald L. Johnston, Arnold & Porter, LLP Editor-in-Chief*

Features

New Federal Circuit Decision Spurs Wave of False Patent Marking *Qui Tam* Actions 1
By James Blackburn and Matthew Bathon

How Software Developers Can Protect Their Rights in the Aftermath of *In re Bilski* 4
By Pam Fulmer, Ilham Hosseini, and Laurie Charrington

Cost-Effective Enforcement Strategies for a Challenging Economy 10
By Lisa Greenwald-Swire

Trademark Fair Use Online 13
By Peter Brown

Current Developments

D.C. Circuit Rules Against FCC on Net Neutrality 20

FTC Warns That Peer-to-Peer File-Sharing Networks May Expose Personal Information 20

Dave & Buster's Settles Claims That It Did Not Protect Customers' Information 21

Second Circuit Rules eBay Not Liable for Trademark Infringement in Sales of Tiffany-Labeled Goods 21

US Patent & Trademark Office Considering Extension to Provisional Patent Applications 23

Events of Note ..... back page

Wolters Kluwer
Law & Business

How Software Developers Can Protect Their Rights in the Aftermath of *In re Bilski*

By Pam Fulmer, Ilham Hosseini, and Laurie Charrington

Recently, the US Supreme Court heard oral arguments in a case that may dramatically affect the way that the software industry protects its intellectual property. The invention at the heart of *In re Bilski*, a method of conducting a business, will likely test the limit to which the Supreme Court permits process patents, including software. It has been nearly 30 years since the Supreme Court last touched on the patent eligibility of software in *Diamond v. Diehr*, and the only justice remaining from that era, Justice John Paul Stevens, wrote the dissenting opinion that "no program-related invention is a patentable process."[1] Society's growing reliance on technology and the software industry's integral role in our modern economy may sway the current justices to chart a course more permissible for software protection.

The technology sector is keenly watching the *Bilski* case, apprehensive of where the Supreme Court might draw the line on patent eligible technology. If the Supreme Court finds that business methods, such as the one in *Bilski*, are not patentable subject matter, this may impact the patent eligibility of software patents, which are often linked to implementing a business method. If *Bilski* ends up limiting patent protection of software, weakening the validity of thousands of existing software patents, other vehicles for protecting intellectual property rights still exist. This article addresses how intellectual property rights apply to software technology and different avenues to protect them in the aftermath of *In re Bilski*, regardless of its holding.

Pam Fulmer is a partner in Jones Day's San Francisco and Silicon Valley offices. Fulmer is an experienced litigator representing clients in patent, copyright, trademark, trade secret, unfair competition, and other intellectual property and complex commercial disputes. **Ilham Hosseini** is an associate in Jones Day's San Francisco office. Her practice focuses on complex civil litigation in both the United States and abroad, including international arbitration. **Laurie Charrington** is a senior associate in Jones Day's Silicon Valley office. Her practice focuses on intellectual property litigation, and her experience includes litigating copyright, patent, trade secret, unfair competition, and general commercial matters. For more information, please contact *pkfulmer@jonesday.com* or go to *www.jonesday.com*.

Different Methods of Protection

Intellectual property rights, which refer to intangible rights of ownership in an asset such as a software program, are important to the software industry. There are four basic types of intellectual property rights: patents, copyrights, trade secrets, and trademarks. Each provides a different type of legal protection. Patents, copyrights, and trade secrets may be used to protect the technology itself and are therefore the focus of this article. Trademarks do not protect technology, but rather the brand names or designs used to identify and distinguish products in the marketplace.

Patents

A patent is a limited monopoly that lasts for 20 years after filing a patent application. It grants exclusionary rights; the patent holder has the right to stop others from making, using, or selling the patented invention.[2] This legal monopoly rewards an inventor, but only if the inventor files a patent application. The application, in turn, requires the inventor to clearly describe the invention and how it works. An inventor must demonstrate that the invention is new, useful, and non-obvious. The US Patent and Trademark Office (PTO) publishes the application upon grant of the registration, thus stimulating the flow of scientific and technological knowledge available to the public.

Software patents can be incredibly valuable economic tools. Unlike copyright protection, a software patent protects the functional aspects of a product. Novel editing functions, user-interface features, compiling techniques, operating system techniques, menu arrangements, display arrangements, and program language translation methods have all found protection under software patents. Since patent rights are exclusive, making, using, or selling the patented invention without the patent owner's authorization constitutes infringement, which may lead to stiff financial penalties. Penalties for infringement do not require knowledge of infringement; a manufacturer of an infringing product is liable regardless of whether he knows about the infringed patent. Independent development of the invention still constitutes infringement.

Copyrights

While a patent can protect the novel functionality of software, a copyright cannot. A copyright protects

the expression of an idea or information as it exists in a fixed medium.[3] The copyright grants an author the exclusive rights to reproduce, distribute, perform, and create derivative works of an original work. Just as with patents, the lifetime of a copyright is limited. The copyright term, however, depends on several factors and is significantly longer than the patent term. Just as patents are not available to protect scientific facts or laws of nature, copyrights cannot protect ideas or information; instead, they cover the expressive elements in the presentation or expression of an idea.

In the case of software, copyright law protects the right to copy the software, create derivative or modified versions of it, and distribute copies to the public by license or sale. Software copyright protection generally covers the source and object code, as well as certain unique original elements of the user interface. Unlike patents, independent development of a copyrighted work does not constitute infringement. Generally, copyright law does not prevent a competitor from writing new code to perform the same functionality or using the same ideas encompassed by the original code.

The exclusive rights afforded under copyright law, just as with patents, are intended to reward the creative efforts of the author of the copyrighted work and punish infringers with fines. The exclusive right to control duplication protects the owner of copyrighted software against the competition that would result from simple verbatim copying of code. Copyright law also protects against indirect copying, such as unauthorized translation of the code into a different programming language. Copyright protection of software's "nonliteral elements," such as a program's hierarchical structure and design, however, is fraught with its seemingly contradictory court decisions and little practical guidance on what might constitute infringement.

Trade Secrets

A trade secret can be any form of information, such as a formula, recipe, technique, customer list, or software program, that (1) derives its economic value from not being generally known to others and (2) is the subject of reasonable efforts to maintain its secrecy.[4] Unlike patents, copyrights, and trademarks, which are primarily protected under federal laws, trade secrets are governed by state laws. Trade secret law therefore varies by state. California and many other states have adopted the Uniform Trade Secrets Act (UTSA).

As the name suggests, a trade secret maintains its value because it is kept secret, thereby providing its owner with a competitive edge. The formula to Coca-Cola is a classic example of a trade secret. Trade secret law allows a perpetual monopoly in secret information. Because trade secret protection can extend indefinitely, it offers an advantage over patent protection, which lasts only for a limited time. A third party, however, is not prevented from independently developing the secret information. Unlike patents and trademarks, a trade secret is protected only when the secret is *not* disclosed. Moreover, unlike patents and copyrights, trade secrets are not subject to being infringed, but they are subject to theft.

To maximize the economic value of a software asset, software developers are advised to analyze how best to use the available forms of legal protection to safeguard their intellectual property rights.

Various features of software, such as code and the ideas and concepts embodied in it, can be protected as trade secrets. This protection lasts as long as the protected information maintains its trade secret status, but such protection is relatively easy to lose. For example, unlike patents, trade secret protection may be lost due to reverse engineering or independent development and provides no minimum guaranteed period of years. As long as the owner can prove that the trade secret was not known and that reasonable measures were taken to maintain its secrecy, the trade secret will be protected as an intellectual property right.

To maximize the economic value of a software asset, software developers are advised to analyze how best to use the available forms of legal protection to safeguard their intellectual property rights. This article provides guidance to software developers in choosing which form of intellectual property protection is best suited for their needs. For example, an invention may be protected under trade secret law or patent law. Under patent protection, the patent holder must disclose the invention, but under trade secret law, the trade secret holder can and must withhold critical information and maintain its competitive edge. Of course, this means that these options are mutually exclusive because once a patent is published, trade secret protection is no longer available for the published claims. On the other hand, as discussed below, both copyright law and trade secret law can protect the same information.

A Brief History on Software Patents

Development of Pre-*Bilski* Case Law

The Supreme Court first addressed the patent eligibility of software in 1972 with *Gottschalk v. Benson*.[5] The

Benson court ruled that an algorithm, or mathematical formula, is like a law of nature, which cannot be the subject of a patent.[6] Therefore, the process for converting numbers from one numerical representation into another was unanimously declared ineligible for patent protection. The Court reasoned that such a patent, were it to be granted, would give the patentee an unfair control over any imaginable, practical use of that mathematical formula. The *Benson* court left open the future of software patents by noting that "[i]t is said that the decision precludes a patent for any program servicing a computer. We do not so hold."[7]

The Federal Circuit found an opportunity to re-clarify its position on patent eligibility with *In re Bilski*.

Six years later, in *Parker v. Flook*, the Supreme Court revisited patenting mathematical algorithms.[8] Unlike *Benson*, in which an algorithm had been claimed over all potential fields of use, *Flook*'s algorithm was tied to a specific application: the catalytic conversion of hydrocarbons. Except for the incorporation of his algorithm, however, the remainder of Flook's process was identical to other conventional systems. The Supreme Court ruled against Flook, finding that a mathematical equation cannot support patent eligibility "unless there is some other inventive concept in its application."[9] Simply tying a generic "post-solution activity," such as catalytic conversion of hydrocarbons, to a mathematical formula was inadequate. The Supreme Court noted that, had the implementation of Flook's abstract principle been novel, the invention would have been patent eligible.

The last of the great software trilogy, *Diamond v. Diehr*, was decided by the Supreme Court in 1981. At issue in *Diehr* was a claim to a process for curing synthetic rubber. The underlying process, however, was defined by an equation. The patent examiner's rejection under 35 U.S.C. § 101[10] cited *Benson* in arguing that steps performed by the computer were unpatentable. In reversing the rejection, the Supreme Court ruled that a process that involves "transforming or reducing an article to a different state or thing" remains patent eligible even if it includes software limitations.[11] Such a transformation, however, had to be more than mere "insignificant post-solution activity."[12]

The Supreme Court's reticence in allowing broad patent protection for software is contrasted by the more recent, pro-software decisions issued by the US Court of Appeals for the Federal Circuit. Created by congressional act in 1982, the Federal Circuit was given jurisdiction over all patent appeals and quickly set out to "clarify" certain problem areas in patent law. By 1994, the Federal Circuit in *In re Alappat* had held that a general purpose computer running a software program was considered a new machine, thereby eligible for patent protection.[13] At issue was a software-based system for smoothing waveform data to be displayed on a computer screen. In essence, the court permitted patent eligibility of software so long as it was related to use within a computer and produced a useful, concrete, and tangible result. While the Supreme Court in *Diehr* had earlier proscribed patent eligibility for "insignificant post-solution activity," the Federal Circuit chose to accept the display of a result on a screen as sufficient.

This trend toward greater patent protection for software by the Federal Circuit culminated in 1998 with *State Street Bank & Trust Co. v. Signature Financial Group, Inc.*,[14] which also opened the door to business method patents. The Federal Circuit ruled that any invention, including a mathematical calculation, was eligible for patent protection if "it produces a useful, concrete and tangible result."[15] Thus, a patent covering software to allocate the profits and losses among a pool of different mutual funds was upheld because "the transformation of data . . . into a final share price, constitutes a practical application of a mathematical algorithm."[16] The permissive bar for patent eligibility set by *State Street* yielded a deluge in software and business method patents and, as a result, an increase in patent litigation.

As software patents and patent litigation increased post-*State Street* and greater concern arose that patents may be covering abstract concepts, the Federal Circuit found an opportunity to re-clarify its position on patent eligibility with *In re Bilski*.

The *In Re Bilski* Decision

At issue in *In re Bilski* is the patent eligibility for a method of hedging risks in commodities trading under 35 U.S.C. § 101. In affirming the PTO's rejection, the Federal Circuit established a new two-prong test for patentable subject matter under 35 U.S.C. § 101, "[a] claimed process is surely patent-eligible under § 101 if: (1) it is tied to a particular machine or apparatus, or (2) it transforms a particular article into a different state or thing."[17] The court cited to the Supreme Court's patent eligibility trilogy as the basis for its "machine-or-transformation test" and abrogated its earlier, permissive "useful, concrete, and tangible result" formulation in *State Street*.[18] This ruling is now before the Supreme Court and has attracted a great deal of attention because of its implications for the viability of software patents. If upheld, it could have the effect of invalidating many thousands of business methods and software patents and

will set a precedent for the evaluation of all future patents. Notably, during oral arguments on November 9, 2009, several justices questioned the inflexibility of this test and whether it would effectively foreclose meaningful patent protection to many business methods and software.

Protection Post-*Bilski*

Patents

Tip: Draft future software patents in terms of a computer-related process or a machine and consider alternate methods of protection.

Companies with diverse patent portfolios are concerned that the Federal Circuit's seemingly more rigid machine-or-transformation test applies to all processes, including software. This poses a major concern for the software industry as to whether patent protection for software will now face a stricter standard than the "useful, concrete and tangible result" test provided by the now-defunct test in *State Street*. The first prong of the Federal Circuit's test, set forth in *In re Bilski*, seems to maintain patent eligibility for software under 35 U.S.C. § 101 so long as the software is physically tied to the operation of a computer, such that the computer may be considered to be a patentable different machine. Accordingly, practitioners can employ certain measures in drafting future software patents to improve their chances for patent protection. Software patent applications should include claims directed to the implementation of the invention by a computer as a computer-related *process*. Similarly, in line with *State Street*, the software could also be claimed indirectly as a process that is tied to a computer. In the latter scenario, software would ostensibly be protected as a *machine*.

Many practitioners are currently hedging against a 35 U.S.C. § 101 rejection by including at least one element in their method and computer apparatus claims that includes either one or more processors or a suitably programmed computer, and this practice is condoned by the PTO at this time. Ultimately, the Supreme Court will decide the extent to which such claims that tie computer components to software operations will comprise patent-eligible subject matter, but hedging between these current uncertainties can provide for a stronger issued patent. Therefore, many practitioners have been advising that a software patent application should include claims that seek to satisfy the first prong of the current *Bilski* test.

In addition to the "machine" prong, future software patent applications should include claims directed to the second prong of the *Bilski* test (the "transformation" prong), emphasizing any transformation of physical objects or substances (or representations of such items) that may be occurring. The case law on what constitutes sufficient transformation is not fully developed.

The Supreme Court may take this opportunity to reject the machine-or-transformation test, however, and return to an even stricter eligibility regime. During oral argument, Chief Justice Roberts appeared troubled by the practice of simply tying an algorithm to a computer, and thereby claiming patentability as being a matter of form over substance "that involves the most tangential and insignificant use of a machine."[19] The Court may instead demand more than mere association with a computer to satisfy *Diehr*'s sanction against "insignificant post-solution activity." If that is the case, software developers may look to other means to protect their intellectual property and/or new claiming strategies such as greater incorporation of hardware components into the software claims. Regardless of what happens with *Bilski*, it makes sense to consider the following alternative ways to seek software protection in addition to a patent filing regimen.

Copyrights

Tip: Register copyrightable work.

If patent protection is significantly restricted, then software developers may wish to rely more on copyright protection. Copyright protection arises automatically upon the creation of an original work; there is no need to apply for a copyright or register the copyrighted material in order for protection to exist. There are significant benefits for copyright owners to register their works, however. First, copyright registration is a prerequisite for bringing a copyright infringement lawsuit.[20]

Many practitioners have been advising that a software patent application should include claims that seek to satisfy the first prong of the current *Bilski* test.

Second, registration entitles the prevailing copyright owner to recover statutory damages, legal costs, and attorneys' fees from a copyright infringer.[21] A copyright owner who has registered a copyright may elect to recover statutory damages instead of actual damages and profits.[22] This is particularly significant in cases when proving actual damages is very difficult or the profits of the infringer are very small or even non-existent.

Third, the certificate of registration serves as *prima facie* evidence of the validity of the copyright.[23] This

serves as an effective tool in cases in which the copyright owner wishes to seek a preliminary injunction against a copyright infringer at the outset of the case in order to stop the distribution of the infringing work. The presumption of validity will apply only if the work has been registered within five years from the date of the work's first publication.[24]

Registration can also have a deterrent effect on infringement and provide needed leverage to avoid costly litigation. For example, an infringer receiving a cease-and-desist letter from an owner of a registered copyright may think twice before committing further acts of infringement because the infringer knows the copyright holder can immediately file a lawsuit.

Although several factors seem to question the viability of this strategy, there is typically no harm in registering. Notably, given the narrow scope of a copyright, the reality that many software companies change the code frequently, the fact that independent development is an absolute defense, and that the copyright *prima facie* case requires proving the defendant had access to the source code, it is uncertain how much protection registration offers. There is no harm, however; it costs hardly anything, it provides the litigation benefits noted earlier, and it allows an applicant to maintain trade secrets.

While there are risks associated with the disclosure of software in the copyright registration context, these risks may be mitigated by taking protective measures. Although the filing requirement involves disclosure of the copyrighted work, the disclosure is satisfied if only portions of the source code (the first 25 pages and the last 25 pages of the source code) are filed.[25] It is therefore possible to register the code for copyright protection, yet keep a considerable portion of a computer software program confidential. In fact, even in the portions submitted to the Copyright Office, an applicant can block out portions of the code that contain trade secrets. Thus, in many circumstances software developers can reap the benefits of registering their work and still maintain their trade secrets.

Trade Secrets

Tip: Identify trade secrets and maintain their secrecy by restricting access to such information, and enter into contracts, confidentiality agreements, and nondisclosure agreements to prevent improper disclosure.

Software developers should also consider protecting some aspects of their intellectual property embodied in software as trade secrets. Technical information such as a software product's computer code (including both source and object code), program logic and algorithms used in software products, and software development tools can all qualify for trade secret protection.[26] First, the company should classify information of value as a trade secret and then take reasonable measures to maintain its secrecy so that it will be classified as a trade secret by a court. As noted earlier, trade secrets are preserved as long as their secrecy is maintained over time. Particularly good candidates for trade secret protection are precise formulas that took a lot of effort to develop but are narrow in scope. For example, the exact weights used in a particular regression equation would be a good candidate for trade secret protection, in part because the cost of obtaining patent protection for a precise set of weights may be costly and risks public disclosure (if the patent application is prosecuted internationally) prior to obtaining a patent.

The trade secret holder must show that it took reasonable steps to maintain the secrecy of its trade secret. "Reasonable" efforts do not require that any and every known effort be undertaken; rather, the measures must be reasonable under the circumstances. Reasonable efforts can include developing security procedures that will reduce the risk that the trade secret will be disclosed; for example, marking confidential information, communicating the status of such information to employees, limiting access to confidential information, keeping electronic information secure, destroying/shredding documents, making the confidential information available to authorized personnel only, and reviewing and updating the trade secret list to keep the information current.[27]

Moreover, companies can enter into contracts to protect their trade secrets. Often software products are licensed instead of sold. Trade secret law does not protect against reverse engineering, and so a contractual obligation not to reverse engineer the product should be added to the license agreement. The trade secret holder can then pursue a breach of contract claim as well as trade secret misappropriation if the product is reverse engineered. Although litigation is costly, at times it may be necessary to protect trade secrets.

A company should also protect its confidential business methods and software applications by entering into nondisclosure and confidentiality agreements with its employees. It is good practice to execute similar agreements with other companies that transact business with the trade secret holder, such as vendors or entities involved in licensing talks or negotiations. As long as the agreement is in effect, the confidential information is protected.

Conclusion

In sum, *In re Bilski* is viewed by many as likely to change the legal landscape for software patents. For example, if the Supreme Court upholds the machine-or-transformation test, then software developers can pursue patent protection by tying their software applications

to a computer and directing claims to satisfy the transformation prong of the test. If, however, the Supreme Court rejects the test and returns to a stricter eligibility regime, new claiming strategies may be required for patent protection as well as greater reliance on copyright and trade secret law.

Notes

1. Diamond v. Diehr, 450 U.S. 175, 219 (1981).
2. 35 U.S.C. § 271(a).
3. 17 U.S.C. § 102(a).
4. *See* Uniform Trade Secrets Act, Cal. Civ. Code § 3426.1(d).
5. Gottschalk v. Benson, 409 U.S. 63 (1972).
6. *Id.* at 67-68.
7. *Id.* at 71.
8. Parker v. Flook, 437 U.S. 584 (1978).
9. *Id.* at 594.
10. In relevant part, § 101 states: "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."
11. *Diehr,* 450 U.S. at 192.
12. *Id.* at 191-192.
13. In re Alappat, 33 F.3d 1526, 1545 (1994).
14. State Street Bank & Trust Co. v. Signature Financial Group, Inc., 149 F.3d 1368 (Fed. Cir. 1998).
15. *Id.* at 1373 (citing *In re Alappat,* 33 F.3d at 1544) (internal quotation marks omitted).
16. *Id.*
17. In re Bilski, 545 F.3d 943, 954 (Fed. Cir. 2008).
18. *Id.* at 959-960 (referring to the "useful, concrete, and tangible result" as "inadequate" and "insufficient to determine whether a claim is patent-eligible under § 101").
19. Transcript of Oral Argument at 35, available at *http://www.supremecourtus.gov/oral_arguments/argument_transcripts/08-964.pdf.*
20. *See* 17 U.S.C. § 411(a).
21. *See id.* at §§ 504-505.
22. *See id.* at § 504(c).
23. *See id.* at § 410(c).
24. *See id.*
25. *See* US Copyright Office, Copyright Registration for Computer Programs (Circular 61: May 2009), available at *http://www.copyright.gov/circs/circ61.pdf.*
26. Randall E. Kay & Rebecca Edelson, Trade Secret Litigation and Protection in California, 72 (2d ed., 2009).
27. *Id.* at 77-88.