Savvy Software Protection

Excerpt from *Entrepreneurs' Guide to Patents, Trademarks, Copyrights Licensing and Trade Secrets* (Penuguin/Putnam Press 2004)

The PTO has recently become more flexible in extending patent protection to software. This liberalizing trend has met with resistance from a surprising source -- the software industry itself. Many prominent and vocal software developers seem, at first glance, to be taking positions strangely at odds their own financial interests. Chat rooms, bulletin boards and online industry publications are ablaze with the opinions as to whether and how patents should apply to the software industry.

In This Chapter You Find Out:

- When and how you can protect software with patents and copyrights.
- How to protect software as a trade secret.
- When your software screens and icons can be trademarked.
- Which types of legal protections work best for software.
- About the direction of future legal trends for software protections.

Less than two years after the controversial Amazon.com 1-Click patent, the PTO is flooded with objections to Patent No. 6,430,602. The patent covers ActiveBuddy software which creates "bots" (short for robot). Bots, which are also called interactive agents, are software components that work with instant messaging services to allow a user to ask questions in plain text and receive an automatic response from a server. For example, if an employee wants to know how many times a customer

has ordered a certain part, the employee could simply type the question in plain text, rather than running through a series of commands.

Instant messaging is expected to be the workplace successor to traditional e-mail, having already earned the loyalty of a huge (mostly teenage) market. The patent promises to be very lucrative for ActiveBuddy.

The ActiveBuddy Patent is greeted with an industry outcry that bots are prior art; developers cite examples of products already on the market such as "Wirebot" and "Runabot." Many experts concur that a programmer named Aryeh Goldsmith developed a bot technology called the Net::AIM module, which was "time stamped" on the Internet nearly two years prior to the ActiveBuddy patent. [1]

ActiveBuddy's CEO, Tim Kay aggressively maintains that "We invented interactive agents. Anybody using his or her own tools (to make bots) is obviously using our technology without paying us…" However, if ActiveBuddy attempts to enforce its patent with a lawsuit, the defendant can request a re-examination of the patent, with the risk of placing the ActiveBuddy patent among the ranks of several dozen software patents that have recently been invalidated amid claims of prior art.

Meanwhile in India, a bot technology is announced that can make computers to accessible to millions of poor, illiterate Indian residents who speak only Hindi. The technology interacts and responds to the user in their native Hindi language. However, the company introducing the technology fears that the Active Buddy patent may have foreclosed the field. [2]

ActiveBuddy founder Tim Kay continues to defend his company's decision to seek a patent against the spate of negative publicity, arguing "[a]ny company such as

---

[1] Ryan Naraine, *ActiveBuddy Turns to Developers*, September 10, 2002.

[2] Reported at AI Dept, *Chatbot Mania, Patenting & Hindu Speech*, http://ai-depot.com/Applications/665.html (August 28, 2002).

ours that is venture-funded has to protect itself. It's standard procedure to file for patents when you invent something. This simply allows us to build a business."[3]

By definition, the process of issuing patents is law chasing technology. But software technology seems to be moving at a breakneck speed, with the PTO, the courts and the legislature frantically trying to keep up. The software industry is critical of the PTO's evolving understanding of software issue.

Bad Laws or Just Bad Patents?

Neither patent nor copyright protection provides a perfect fit for software. Law school professors and academics have, for several years, suggested that Congress pass legislation to create a unique category of legal protection for software, a sort hybrid with characteristics of both patent and copyright law. (This is what was done for mask protection and semi-conductor technologies.)[4]

But software developers *don't* seem to think that a bad fit between software and patent law is the problem. They suggest the problem is just "bad patents."

San Francisco-based BustPatents, which acts as a legal resource for software/technology patents suggests the PTO is simply undermanned and overworked. "One of the reasons so many bad software patents (are issued) is that patent examiners do not have enough time and library resources to do their jobs." BustPatents adds that "strained conditions under which patent examiners do their jobs lead to many patents of

[3] As quoted by Ryan Naraine, *ActiveBuddy Turns to Developers,* September 10, 2002.
[4] For a terric discussion of sui generic protection for computer technologies, *see* Lemley, et.al, Software and Internet Law (Aspen Law and Business 2000).

questionable quality being issued."[5] The Website www.bustpatents.com lists dozens of software patents that have been invalidated either by the USPTO during a patent re-examination or by courts charged with reviewing them.

Some of these " ex- patents" include:

- Patent No. 5960411 - Method and system for placing a purchase order via a communications network.
- Patent No. 5333184 - Call message recording for telephone systems.
- Patent No. 4205780 - Document processing system and method.
- Patent No. 4169290 - Utility meter date recording method and apparatus.
- Patent No. 4057829 -Communications TV monitoring and control system.

One software CEO, Jeff Bone, whose company holds a number of software patents, argues: "The problem with patents like the ActiveBuddy patent is [they]… undermine the legitimacy of the entire patent system. They turn the patent system into an incredible waste of time, money, and effort on the part of anyone who seeks to obtain protection on true inventions." Bone contends "since patents are granted without apparently any substantial due diligence [by the PTO], no inventor who receives a patent on something is in any way assured that his invention is protected." Bone advocates a total overhaul of the software patent system.[6]

---

[5] As quoted by Ryan Naraine, *ActiveBuddy's Patent Win Riles IM Bot Developers*, August 15, 2002.

[6] Jeff Bone, Bad PTO: Activerse, ActiveBuddy, and Prior Art, (August 15, 2002). *http://www.oreillynet.com/pub/wlg/1836*[WHAT IS THIS REFERENCE FOR????]

The Chronology of Software Patent Cases

A software developer has an interest in protecting *more* than computer code. As an individual, the developer has a keen economic interest in protecting as much of the functionality of the software as possible. But over the long term, the software industry as a whole, is hurt by an over-proliferation of patents that can be roadblocks to future innovations in the field. Innovators may be pre-empted from using or improving upon software that has already been deemed as proprietary. Ultimately, overly broad protections in the industry can stifle, rather than foster innovation.

In the courts, software patent cases reflect the tension between the need encourage innovation by individual developers and the need to foster growth in the industry as a whole. Both the PTO and federal courts have been attempting to achieve balance in this area for nearly three decades.

The 1970's: A Dismal Outlook for Software

Prior to the early 1980s, the prospects of patent protection for the infant software industry looked pretty dismal. In 1972, in *Gottschalk v. Benson*,[7] the Supreme Court considered the novel issue of whether a program converting decimal numerals into binary numerals was patentable. (This type of conversion made it possible for a digital computer to perform a lot of high level tasks and functions.) The Court denied patent protection to the program, reasoning that the patent "would wholly preempt [a] mathematical formula and in practical effect would be a patent on the algorithm itself."

---

[7] 409 U.S. 63 (1972).

An algorithm is a set of steps to achieve a desired result. Since virtually all software programs contain some sort of algorithm, after *Gottschalk*, the outlook for extending patent protection to software looked grim indeed.

Subsequently, in 1978, in *Parker v. Flook,*[8] the Supreme Court issued another blow to the software industry, holding that a method for updating alarm limits during catalytic conversion processes could not qualify for patent protection. The Court reiterated that a patent could not issue for any patent claim based on an algorithm. This certainly seemed to spell doom for software patents.

The Door Opens With the Dot.com Boom

In 1981, coinciding with the start of the dot.com business boom, the judicial tide turned, and the sun began to shine on software patents. In *Diamond v. Diehr,*[9] the U.S. Supreme Court specifically held that a software program was patentable. The Court limited the earlier prohibition on patenting algorithms of *Gottschalk* case to *mathematical* algorithms. More importantly, for the first time, the Supreme Court had actually provided developers with a road map for patenting software!

A few months later, in *Diamond v. Bradley,*[10] the Supreme Court affirmed the distinction between mathematical and other types of algorithms. The court upheld a claim for a program containing an algorithm directing data transfers between registers and memory. *In re Pardo,* decided one year later,[11] the Court stated that that the

[8] 437 U.S. 584 (1978).
[9] 450 U.S. 175 (1981).
[10] *Diamond v. Bradley,* 450 U.S. 381 (1981).
[11] *In re* Pardo, 684 F.2d 912, 916 n.6 (C.C.P.A. 1982).

applicants' use of the term "algorithm" to describe the invention is not an admission of non-patentable subject matter.

In 1982, the Court of Appeals for the Federal Circuit (CAFC) was recognized as the sole appellate court authorized to hear all patent cases. The CAFC continued on the course of liberalizing patent protection.

The effect of these cases was to convey a sense of security to software developers as to their ability to obtain patent protection. Subsequent to the *Diamond* cases, it appeared that claims for software patents would by allowed unless they simply involved the use of mathematical formula to calculate and display numbers.

Recent Cases

Today, the patentability of software is a fact of life. In 1994, in *In re Alappat,*[12] the CAFC decided that the software patent application presented proper "useful" statutory subject matter, even though the patent claims were merely for a series of physical "elements" within a machine to perform the functions.

More recently, in 1998, in *State Street Bank v. Signature Financial Group,*[13] the CAFC ruled that a patent should issue for a data-processing system that allowed related mutual funds to pool their assets into an investment portfolio with numerous similar funds. The software “invention” made several calculations on a daily basis related to each fund's percentage share of the investment's assets and expenses. The *State Street*

[12] 33 F.2d 1326 (Fed. Cir. 1994).
[13] 149 F.3d 1368 (Fed. Cir. 1998).

case is often quoted by attorneys and courts for the proposition that software algorithms are patentable when applied to accomplish a specific task.

Current PTO Guidelines

The U.S. Patent and Trademark Office (PTO) takes great care to educate its own examiners as to when claims for patenting software should and should not be allowed. In 1996 the PTO prepared a manual titled "Examination Guidelines for Computer-Implemented Inventions" to assist its patent examiners and the public in prosecuting software patents.

Examples given in the current PTO guidelines as to when computer programs are considered “practical applications in the technological arts" under the Patent Act include the following:

- Methods controlling transfer, storage and retrieval of data between cache and hard disk storage devices so that the most frequently used data is readily available.

- Methods for controlling parallel processors so that the processors can simultaneously perform several computing tasks to maximize computing efficiency. (This is commonly called multi-tasking.)

- A method of making a word processor by changing the state of the computer's arithmetic logic unit.

- A digital filtering process for removing “noise” from a digital signal.

Has Patent Protection Gone Too Far?

Has the patent pendulum swung too far? Critics of the Amazon.com 1-Click patent case (discussed in Chapter 11, *What You Can Patent*) and of the ActiveBuddy patent argue that it has. Many key players in the software industry claim that the PTO is now far too liberal in granting patents to pre-existing or obvious software technologies, foreclosing the field to future innovation. Others argue that aggressive patents are necessary for their businesses to thrive.

Regardless of their position on the correct level of innovation that should be required for a patent, software industry leaders seem in agreement that patents will play a major role in the future of the software industry. The PTO currently has thousands of software patent applications pending.

Critical Software Copyright Cases

Copyright never treads on the toes of patent law. You cannot copyright ideas that are functional, as opposed to merely expressive, because that is the purview of patent law.

Copyright doesn't cover procedures, processes, systems, methods of operation, concepts, principles, or new discoveries embodied in the program. Copyright *does* protect all forms of original *expression*, economically and expeditiously. Copyright protects developers from copying of their code, prevents the distribution of copies, and prevents the preparation of derivative works. These are all very valuable rights, since software is so easy to copy and distribute on a disk or over the Internet.

The Watershed Whelan Case

The first significant case extending copyright protection to software was *Whelan Associates, Inc. v. Jaslow Dental Lab.*[14] In this case a consultant created a functionally similar program to a proprietary one she had been hired to create for someone else. At issue was whether the structure, sequence, and organization of the two programs were so similar that the second one was a violation of the proprietary patent rights. The Court of Appeals for the Third Circuit held that "the manner in which the program operates, controls and regulates the computer in receiving, assembling, calculating, retaining, correlating, and producing information either on a screen, print-out or by audio communication." was protected.

After Whelan: Which Features are Copyrightable?

Since the *Whelan* case, courts have cautiously extended copyright protection to "look and feel" as well as software code. The standard of originality from protecting the distinctive appearance and feel is high. Words used to describe software actions or features that are commonly used or intuitive terms are not protected. In *Lotus Development Co. v. Boreland International, Inc.*,[15] a federal district court held that menu structures and the appearance of the menus were not capable of copyright protection. Copyright protection only extends to elements of a program that are dictated by expressive rather than functional considerations.

[14] *Whelan Associates, Inc. v. Jaslow Dental Laboratory, Inc.*, 797 F. 2d 1222 (3d Cir. 1986).
[15] *Lotus Development Co v. Boreland International, Inc.*, 799 F. Supp 203 (D. Mass 1992).

The Computer Associates Test

In 1992, in *Computer Associates v. Altai Associates International, Inc.,* the Court of Appeals for the Second Circuit developed a rather complicated three-part test for determining whether software is infringed under the copyright laws.[16] This test is known as the "abstraction/filtration/comparison" test, and frequently used by other courts.

In the first step, the computer program is divided into its various *levels of abstraction.* Abstraction means that the details about how the program works are hidden from the user. Levels of abstraction is a term that describes how a program user sees various levels of functioning; the user doesn't have to know about much of what the computer does internally. Thus, the first part of the test looks at the functions carried out by the program as seen by the user.

The second part of the *Computer Associates* test is the "filtration step." In this step the court examines the software at each level of abstraction to determine whether each level involves protectable ideas, as opposed to prior technology.

The third and final step is the "comparison" step. This step compares the software which is accused of being an infringement to the pre-existing software to determine whether there is substantial similarity between the two. If substantial similarity exists and it's clear that the developer of the accused software had access to the original software, then a court may issue a finding of copyright infringement using the *Computer Associates* test.

---

[16] *Computer Associates Int'l v. Altai, Inc.*, 982 F.2d 693 (2d. Cir. 1992).

Lawyer's Note: There is no centrally designated appeals court for copyrights as there is for patents. Thus, the scope of copyright law protection may vary among the jurisdictional boundaries of various circuit courts of appeals.

Patents vs. Copyrights: Which One Works Best?

The commercial life of even the most useful software program can be cruelly short, and eighteen months to get a patent can seem like an economically ruinous eternity. Copyright protection has the advantage of being automatic, since it applies to any work in fixed, tangible form, whether registered or not. However, other considerations may make patent protection well worth the wait.

Patents Protect Against "Reverse Engineering"

Suppose one of your competitors hires experts who don't have access to your computer code to analyze how your software works and independently write code. This is called "reverse engineering."

Patents protect you against reverse engineering. If you hold a patent, your competitor cannot make a functionally identical work. Copyrights protect only against actual copying. In *Stac Electronics v Microsoft Corp.*, Stac Electronics[17] received a $120 million patent infringement award against Microsoft based when Microsoft created a functionally equivalent a data compression program.

[17] *Stac Electronics v. Microsoft Corp.*, No. C-93-0413-ER (C.D. Cal.2000)

The Ease of Obtaining Copyright Protection

Patents are generally viewed as the Cadillac of software protection. Although they generally provide much broader legal protection than copyrights, patents always cost upwards of $10,000 and take an average of eighteen months to two years to obtain. During this time you can't assert any patent rights, and the software may even be technologically surpassed by other products introduced into the market place.

In contrast, copyright protection automatically exists in software upon its expression in a tangible form. This means that the protection is immediate and free. Registration of your copyright is necessary to bring a suit to obtain statutory damages and attorneys fees. (See Chapter 7, *The Core Concepts of Copyright.*)

Business Tip: Including some totally useless code in your program can help detect and prove that your program has been copied. For example, include some code that does nothing other than an unnecessary loop, or code with a precondition for execution that can never be satisfied.

Claiming Both Copyright and Patent Protection for Software

Patent and copyright protections aren't mutually exclusive. In fact, the two can mesh very nicely. Copyright protects expressive elements and precludes literal copying. Patent protects against functionally similar uses of the technology you've patented, including derivation, independent development, or incorporation of protected functional aspects of your program. The fact that copyright protection is immediate can be invaluable to you while awaiting you're waiting out the lengthy approval process for your patent application.

Trade Secret Protections for Software

Unlike copyright and patent protections, which can readily co-exist, there's any uneasy tension between trade secrets and the need to disclose information in patent and copyright applications.

Lawyer's Note: Trade Secret law works best when a developer anticipates limited distribution of the software. If custom software is developed for sale on a limited basis, the contract parties can include provisions to maintain trade secret protection.

Trade Secrets and Patent Applications

Patent law requires that the inventor disclose the invention "in such full, clear, concise and exact terms as to enable any person skilled in the art to which it pertains . . . to make and use" it.[18] The *best mode*, or most efficient way of carrying out the invention, must be disclosed in the application.

Is it necessary to include code in a patent application? Recent cases indicate you may be able to get away without it. *In re Sherwood,* [19] the a federal court found that it was unnecessary to disclose code to satisfy the best mode requirement because the developer had provided a very detailed outline of the methodology used. On the other hand, in *White Consolidated Industries vs. Vega Servo-Control, Inc.,*[20] a patent was invalidated because key software considered a trade secret was not included in the

[18] 35 U.S.C. § 112 (first paragraph).
[19] *In re Sherwood*, 613 F.2d 809 (C.C.P.A. 1980), *cert. denied*, 450 U.S. 994 (1981).
[20] 713 F 2d. 788 (Fed. Cir. 1983).

application. In the *White* case, the developers did not attempt to comply by including a detailed outline, as the developers had done in the *Sherwood* case.

Generally, you can assume that detailed code is not required to satisfy statutory disclosure requirements in software patent applications. Algorithms and techniques need only be described in general terms. These descriptions may include summaries, diagrams or flow charts.

> **Business Tip:** Patent, rather than trade secret protection, is more suitable for software that can easily be reverse-engineered using computers called *decompilers*. Patents protect against reverse-engineering, but trade secret laws do not.

Confidentiality for Copyrighted Software

Unlike pending patent applications which can be perused on the Internet, copyrighted code need not be registered immediately. You must register your copyright prior to bringing an infringement action, but you have several different options for depositing materials with the Copyright Office, depending upon your need for confidentiality. These options include:

- Filing the first twenty five and last twenty five pages of source code with portions containing trade secrets blocked out;
- Filing the first ten and last ten pages of source code alone, with no blocked out portions;
- Filing the first twenty five and last twenty five pages of object code plus any ten or more consecutive pages of source code, with no blocked out portion; or

- For programs fifty pages or less in length, filing the entire source code with trade secret portions blocked out.

Business Tip: For copyright applications, the goal is to include enough information to be able to prove copying, while giving away as little trade secret information as possible.

Trademark Protection for Software

Trademark registration can be used to protect screens generated by computer code, even though these screens may be separately and automatically protected under copyright law. If you use icons that are sufficiently unique, these too may entitled to trademark protection. For more information about what visual elements of your program may be entitled to trademark protection, see Chapter 3, *The Basics of Trademark Law, What Makes a Mark*?

Software Licensing Agreements

When you purchase some new software and click, click, click through the contractual terms, you may not realize you're actually entering into a licensing agreement with the manufacturer of the software. What are the implications of these legal agreements that are so commonplace that many computer users barely pay attention to the fact they're entering into one?

Are "Shrink-Wrap" Agreements Valid?

When you click the “I Accept” button, most states now recognize the existence of a legally binding agreement with the software vendor. These agreements generally address the issues of making copies of the software, or using it in a manner that infringes on the intellectual property rights of the developer. They may give the vendors special remedies such as the right to sue you in a particular jurisdiction.

Regardless of the extent to which a shrink-wrap agreement is recognized in a particular state, they offer some benefit to the manufacturer under federal law. They make it more difficult for an infringer to claim that their infringement was “innocent” or inadvertent under federal copyright law.

Agreements for Customized Software

Customized software is now critical to the functioning of many businesses. It represents a major investment for the business. Review of software development agreements has turned into a profitable sideline for many attorneys.

Customizations need to be spelled out in an agreement. At a minimum, a software customization agreement should contain provisions specifying the following:

- Customizations, interfaces or integration of the software
- A description of the final software functionality deadlines
- Testing
- Conditions for acceptance
- Installation
- Any ongoing maintenance and technical support to be offered by the developer

- Whether the licensee or licensor owns the customized version of the software
- Conditions for termination of the agreement
- Conditions and milestone dates that must precede payment
- Conditions for terminating the agreement
- Liability limits
- Updates and upgrades
- Indemnification
- Any duty of confidentiality by the licensee
- Whether the licensee is permitted to make backup copies

Software Piracy Issues

The Business Software Association's "NAIL YOUR BOSS" mail campaign encourages employees to blow the whistle on employers that use unauthorized copies of software. The U.S. Naval Academy recently seized 100 computers from U.S. midshipmen in a crackdown on online piracy.

Non-compliance with copyright law is an easy trap for most businesses, particularly where software is concerned. To protect against civil and criminal liability, companies need to adopt specific policies and procedures to ensure employees are not violating copyrights.

A company should take two common sense precautions insulate itself from potential liability for software copyright infringement. First, the business should maintain a software inventory which tracks what software each computer is running with

what it's licensed to have running. Second, a company should clearly communicate its policy about unauthorized software copying to its employees in writing. If you are a developer seeking damages for infringement, it's likely that a jury will be swayed by an infringing company's failure to undertake these reasonable measures.

Business Tip: CD's and other storage media containing software that can be copied should be secured. Programs are available that can help your organization figure out what software is running on your network, and who is running it.