The Workplace Hazards of Software Patents

Who does current patent law really protect in the realm of software?

The number of companies receiving patents and filing patent applications for software has increased dramatically in the last couple of years. Although more companies are now filing applications, a small handful have held almost all software patents, with IBM and Hitachi topping the list. Patenting software may seem to be more of a legal issue than a workplace issue, but this trend increasingly impacts the everyday lives of staffers, managers and independent developers.

Patents should not be confused with copyrights; there are critical differences between the two concepts. *Copyrights* cover "works," a particular literal expression of an idea, such as a song or book. *Patents* originally were intended to cover "inventions" and to grant a 17-year monopoly on the production and use of such "devices." The social justification for this policy is to encourage progress in science and the "useful arts"; it provides an incentive for inventors to make the precise nature of their inventions public so they will eventually become available to all humanity. It should be noted that—in practice—the U.S. Patent and Trademark Office (PTO) and the courts have come to define *invention* as including relatively abstract techniques, and now interpret *devices* as encompassing the actual ideas underlying the devices.

Problems with Software Patents

At first, software patents might seem like a great idea. In 1994, mighty Microsoft was forced to bow to tiny Stac Electronics when it was determined that Microsoft's DoubleSpace product infringed upon the patented compression techniques used in Stac's Stacker product. But, just as Microsoft could be stung by a (presumably) accidental re-creation of a patented software technique, so can any other software company or independent developer.

However, consider for a moment how many lines of code have been written in the last few decades; then consider the fact that there is a dearth of computer science graduates among the PTO's examiners. Obviously, there are serious questions as to whether the PTO can reasonably determine if a specific claim has already been invented or may be unreasonably broad.

More importantly, the patenting of software discourages the innate creative process that is software development. For example, an essential element in the current Internet revolution has been free software like the Unix-based HTTPD Web server and the DOS/Windows-based Pegasus e-mail program, whose distribution cuts costs for providers and allows users to explore the Net for little initial cost. Ironically, this kind of software is precisely the type of innovation that is already being stifled by software patents.

Many independent developers have laid the foundations of their careers on freeware and shareware, and many managers of small firms have successfully leveraged these marketing methods into effective competition against "the big boys." The rapidly increasing number of software patents creates a much greater opportunity for accidental infringement and the costly legal proceedings and product derailments that often go along with it. By and large, software patents empower the big guys and disadvantage the little guys.

The case of Derek Noonburg is a good example of the hazards facing independent developers. Noonburg is a graduate student in Electrical and Computer Engineering at Carnegie-Mellon University in Pittsburgh, PA. For fun, he created an X Window System-based application for reading PDF (a.k.a. Adobe Acrobat) files and was giving it away for free on the Internet. Unfortunately, the PDF standard includes LZW data compression, a technique for which Unisys Corp. holds a patent. In December 1995, he says, Unisys told him there was "no way" he could "legally distribute a free PDF viewer" that uses LZW. Noonburg was forced to pull his product from the Net until he could develop a workaround that doesn't use LZW.

Small companies are also vulnerable; from an economic standpoint, perhaps even more so than independent freeware developers. Some unscrupulous attorneys deliberately seek out overly broad patents and then demand "one-time licensing fees" from companies offering related products. Large companies may have the relative luxury of choosing to pay off or fight it out, but smaller firms often can't afford to do either and are forced to take a severe financial blow. (For a more thorough exploration of the history and legal aspects of software patents, see "High Technology and the Law" on page 54.)

By Jim Johnson

Fighting Back

As important as it is to protect oneself against the dangers of accidental patent infringement, it's not an easy task. The first and most important thing is to raise awareness; staffers should make sure that their managers and coworkers know of the danger, and managers should push their company to develop fluency in patent concepts and follow developments in the field. Both organizations and independent developers can do their part in keeping the problem from getting worse by refraining from patenting their own products (of course, they should not hesitate to copyright them, if they wish).

In the design stages of a product, a company could conduct its own patent search to determine if any similar techniques already exist, but this is costly, time-consuming and frequently unreliable. However, if one is facing litigation or other threats from patent-holders or their agents, a "prior art" search may turn up evidence that could challenge the validity of the patent in question.

Of course, opinions about how to react vary with individuals and firms. Some might choose to praise and patronize companies that have voiced opposition to software patents, such as Adobe, Autodesk and Oracle. Conversely, they may also choose to petition or not patronize companies that patent software as a matter of policy; top software patent holders include AT&T, DEC, Hitachi, IBM, Sharp, Toshiba and Xerox.

The real problem, of course, is PTO policy, which probably will require a legislative remedy. Completely exempting software from patent coverage is one option; other proposals include shortening the span of time that software patents may apply and providing for mandatory licensing. It might seem naive to ponder the prospect of getting any such legislation through Congress, but the economy is one thing that politicians seem to react to. While a handful of large corporations may be defending software patents, a league of small and medium-size firms actively opposing them would be a substantial political force.

The primary organizational resource in the protection from and opposition to software patents is the League for Programming Freedom (LPF) in Cambridge, MA; opposing software patents is a primary reason that the LPF was founded. It acts as a focal point for lobbying, as well as a resource for individuals and firms being directly threatened by patents. One of the best ways for corporations and individuals to help out is to join LPF and propagate its materials. For further information, send e-mail to lpf@uunet.uu.net. The LPF Web site is http://www.lpf.org/, and its FTP site can found at prep.ai.mit.edu in the directory /pub/lpf.

Let's act now to protect our companies, our own careers and software innovation itself. IT

Jim Johnson *is a certified personnel consultant and the principal of Options Unlimited, specializing in the placement of Unix professionals in the Washington, DC, area. He can be reached at jim@uujobs.com.*