

## Give Software Patents a Break

Vonage settlement of three software patents suits has critics up in arms again. But calls to restrict software patents as anticompetitive remain pointless and futile

by John Kheit

Why is it that software patents get no respect? After almost every high-profile lawsuit involving software patents, there are cries of dismay and derision over how the intellectual property system is being abused to con juries and extract corporate ransom.

Vonage (VG), after belittling the software patents it was accused of infringing, has now opted to dole out at least \$200 million to settle those headaches with Verizon (VZ), Sprint (S) and AT&T (T). With Vonage's Internet telephone service imperilled by the three suits, some view these settlements as sheer pragmatism rather than as validation of its rivals' patents—even though juries had already found Vonage guilty of infringement in two of the cases. The reaction was similar last year when BlackBerry maker Research in Motion (RIMM) shelled out \$612.5 million to a no-name company to do away with a software-related suit that threatened to shut down RIM's mobile e-mail service. The plaintiff in that case was maligned by many as a "patent troll."

### ATTEMPTS TO WEAKEN PATENTS WILL FAIL

Such legal victories regularly spur an outcry by the anti-software-patent camp. Unlike the relatively high regard given hardware patents, the view of software as substandard intellectual property is misguided and potentially harmful to inventors and entrepreneurs. By weakening this alleged barrier to innovation and competition, these detractors are hurting the very underdogs they aim to empower. In reality, smart patent lawyers will always find ways to protect software innovations for well-heeled corporations. But startups that can't afford the legal expertise will have less incentive to invest in intellectual property if it can't be guarded with a strong patent.

The bad news is that this drive against software patents may make them more expensive to obtain. The good news is that this effort is destined to fail.

One needs to understand that there is fundamentally no difference between software and hardware; each is frequently expressed in terms of the other, interchangeably describing the same thing. For example, many microprocessors are conceptualized as software through the use of hardware description languages (HDL) such as Bluespec System Verilog and VHDL. The resulting HDL software code is downloaded to special microprocessors known as FPGAs (field programmable gate arrays), which can mimic a prospective chip's design and functions for testing. Eventually, the HDL code may be physically etched into silicon. Voilà! The software becomes hardware.

### PATENT NEUROSIS

Nevertheless, anti-patent sentiment has become such a common theme in tech circles like Slashdot.com that the mere mention of the word "patent" sends readers into a Fred Flintstone fit ("bet, Bet, BET!") extolling their evils. This is a neurotic reaction to patent law.

To better understand the neurosis, a little more background may be in order. To obtain a patent, your invention must be one of two things: Either it's completely novel (that is, nothing like it has existed before) or it's a "non-obvious" combination of things that existed before. While many people like to think of invention as 1% inspiration and 99% perspiration, that view isn't the law. Loosely speaking, "non-obvious" means an invention was not readily foreseeable or predictable. Hard work may or may not have been involved—but it isn't enough to qualify an invention for a patent.

However many people tend to focus on how obvious an invention is, denying credit to those innovations that in hindsight seem to be no-brainers. One of the reasons why software patent debates are endless is that some people don't like how easily implemented some inventions are, and they try to express this displeasure by disqualifying them under the non-obvious legal requirement. After all, it used to require lots of money, people, and effort to take an idea and turn it into a piece of technology. Now the same thing can be done by a couple of teenagers in the proverbial garage. Software has allowed inventors to greatly reduce the gap between concept and a viable invention.

For example, Amazon's 1-Click checkout patent for speeding online purchases has been challenged in both the courts and by detractors as being incredibly obvious. No evidence of a similar innovation predating the Amazon patent was found despite a large bounty offered for its discovery and the ensuing widespread search. The real problem people have with 1-Click is that once you conceive of it, it's relatively trivial to implement. This ease doesn't sit well with traditional, sweat-of-the-brow sensibilities.

But if people truly have a problem with "easy" inventions, they should lobby to change U.S. law to require sweat and struggle, also known as "non-triviality-of-implementation," as prerequisites for patents. Instead, we get ridiculous debates and ill-conceived policies that are easily surmounted by well-heeled companies with good lawyers.

## SOFTWARE PATENTS GROW UP

There is a long history of such ill-conceived policies. In 1972, in *Gottschalk v. Benson*, the Supreme Court cited a dubious Presidential commission in railing against patents for bare algorithms with no application outside a computer.

In response to such rulings, software patents have been obfuscated to look like hardware for decades. For example, instead of patenting a software algorithm to better time how long to cure rubber, one might patent an "apparatus" to cure rubber with a unique timer component—which just happens to be software expressed as hardware. This cat-and-mouse game went on until 1995, when the U.S. Patent & Trademark Office relented and issued guidelines for software patents.

But recent signals from regulators, politicians, and judges are troubling. In July, the European Patent Office essentially declined to further debate its stance on software patents—which is that they're generally unwelcome, but not necessarily impermissible. And the outcome of a recent U.S. case, *In re Nuijten*, though heralded by Slashdot as limiting software patents, really just reaffirmed that bare algorithms are not patentable.

The reaction to both developments will likely be similar: Lawyers will keep crafting patent applications for software technologies to look like they do something physical. Nothing short of making the electrical engineering arts non-patentable subject matter will prevent attorneys from couching and describing software in terms of hardware.

## AND THE RICH SHALL INHERIT EVERYTHING

The only real effect of efforts to devalue software as patent-worthy will be to create more work for lawyers, making them more difficult and expensive to obtain. Because patents are such a crucial competitive tool, Big Business will continue to spend money to obtain them. Notably, patent filings by technology companies have grown hand-in-hand with both their revenues and research and development budgets over the past two decades. Apple ([AAPL](#)) reportedly filed over 200 patent applications on the iPhone alone. When companies invest huge amounts to develop the next iPhone, they will want

to make sure that such products are not instantly ripped off.

But if the filing, analyses and appeals involved make patents harder to obtain, small startups may be priced out of the market.

In the end, making patents harder to obtain makes the few that are issued even more desirable to those that can afford them. Large companies know that if you are the only one with a big club, it makes it easier to squash would-be competitors. Another case of "them that got are them that gets."

*John Kheit is a senior associate specializing in intellectual property at the international law firm Chadbourne & Parke. He is reachable at [jkheit@chadbourne.com](mailto:jkheit@chadbourne.com) .*