From: Byron Gibson [e-mail redacted]
Sent: Monday, September 27, 2010 9:03 PM
To: Bilski_Guidance
Cc: [e-mail redacted]
Subject: Bilski Guidance

Hello, I am a US citizen and independent software developer and contractor and would like to submit comments.

Software patents hurt individuals by taking away our ability to control the devices that now exert such strong influence on our personal freedoms, including how we interact with each other. Now that computers are near-ubiquitous, it's easier than ever for an individual to create or modify software to perform the specific tasks they want done -- and more important than ever that they be able to do so. But a single software patent can put up an insurmountable, and unjustifiable, legal hurdle for many would-be developers.

The Supreme Court of the United States has never ruled in favor of the patentability of software. Their decision in Bilski v. Kappos further demonstrates that they expect the boundaries of patent eligibility to be drawn more narrowly than they commonly were at the case's outset.
The primary point of the decision is that the machine-or-transformation test should not be the sole test for drawing those boundaries. The USPTO can, and should, exclude software from patent eligibility on other legal grounds: because software consists only of mathematics, which is not patentable, and the combination of such software with a general-purpose computer is obvious.

Further, one framework for thinking about software patents is the Edison Criteria - inspiration vs perspiration. Thomas Edison said:

"None of my inventions came by accident. I see a worthwhile need to be met and I make trial after trial until it comes. What it boils down to is one per cent inspiration and ninety-nine per cent perspiration."

This statement illuminates the fact that there are several domains under which inventions can be categorized:

1. Easy/inexpensive to derive, and easy/inexpensive to implement or produce (ex: Amazon 1-click check out) 2. Easy/inexpensive to derive, and difficult/costly to implement or produce (Google Search) 3. Difficult/costly to derive, easy/inexpensive to implement or produce (pharmaceuticals) 4. Difficult/costly to derive, difficult/costly to implement or produce (Saturn V rocket, Large Hadron Collider)

Easy and difficult are, of course, relative values, and there are certainly grey areas in between, not every invention or innovation falls neatly within one of these criteria. However, if the objective of the patent system is to promote the progress of science and useful arts, then I suggest the PTO allow patents only for #3 and #4, but not for #1 and #2.

#1 is both obvious and inexpensive, and hence completely likely that multiple parties would both derive and implement the idea independent of one another. Allowing one of them to patent it (and not necessarily even the first to create it, just the first to get their paperwork through the PTO), prevents a natural state of competition from arising, in which said parties continually develop and improve their versions of the product, attempting to one-up each other. Such a state of competition can propel investment, hiring, new innovation, and advancement of science and useful arts. Allowing one party to patent such a product does the opposite, chilling advancement and progress in that domain.

#2 does not require patenting, since the difficulty and cost of implementation provides a natural 'moat' - the term Warren Buffet uses to describe natural obstacles protecting a business from potential competitors. This moat generally negates the need for patent protection, since most potential competitors are dissuaded from attempting to replicate the product and encroaching on the market.
The high cost of implementation makes the ROI too uncertain, too unattractive. The only competitors who attempt to do so are ones who believe they can make that product better, create a new moat, and have the financial resources to do so.

An excellent example is Google Search. The fundamental idea of the Page Rank algorithm was had by two graduate students, and

prototyped for less than $100,000 (the first check written by their first venture capitalist investor). But scaling Google Search globally has required massive infrastructure investment and expert engineering teams, along with other investments in areas such as reliable electricity supply and data bandwidth (ref Google's investments in clean energy and fiber optic capacity).

The only search engine that has formed since Google became the dominant one and that has actually managed to make inroads against Google Search is Microsoft Bing, and that is only because it has the massive marketing budget of the world's richest company behind it. There are other capable search engines like DuckDuckGo.com and Gigablast.com, whose results are as good or better than Google's (partly due to algorithm innovation, and partly because they are too small to attract the attention of the Search Engine Optimization industry which games and distorts some of Google's results). But these have so far failed to make inroads due to the competitive moat Google possess - name recognition, good-enough technology, and infrastructure investment. Only Bing has been able make progress, and only by spending hundreds of millions on marketing, infrastructure, and engineering talent.

Allowing software patents in domain #2 would freeze out small upstarts like Gigablast and DuckDuckGo (who may at some point be acquired for their technology or their engineers, a common acquisition strategy of Google and other high-tech companies, which benefits both the acquirer and acquiree), without actually providing substantially more protection from competitors than the implementation moat currently does. The net effect of allowing software patents in domain #2 would be to disincentivize progress in the sciences and useful arts.

The primary example of domain #3 is the pharmaceutical industry. Developing a new drug, proving its effectiveness and eliminating harmful side effects, and getting it through clinical trials costs billions in investment. Yet mass producing it is relatively inexpensive, as proven by the generic drug industry. Furthermore, it is relatively easy and inexpensive for the generic drug makers to reverse-engineer drugs from over-the-counter samples. This is the epitome of an

industry that benefits from patent protection, yet is exactly the opposite in the Edison Criteria as the software industry. Expecting patent protection to encourage advancement in both industries in the same way is a fallacy. In fact, patent protection has the exact opposite effect on Domains #1 and #2 as it does on Domain #3.

Finally, domain #4 consists mainly of 'blue sky' projects whose derivation and implementation are both on the very edge of human understanding and ability. The Space Program, from Saturn V and Apollo to the Space Shuttle and the ISS are one example. Particle Colliders like the defunct SSC and new LHC in Europe are two others. In industry, the development of quantum computing is another. Extremely difficult, time-consuming, and costly to derive, as well as difficult and costly to implement. This domain usually deals with making physical things of great complexity, rather than the applied logic of software products.

Whether such products in domain #4 should be patented or not, I cannot offer an opinion on, and will leave to the courts to figure out. This letter is primarily concerned with domains #1 and #2, where most day-to-day software development exists, and where software patents have the most negative effect on the PTO's mission of promoting progress in science and, most particularly, the useful arts.

Please consider that the net effect of software patents is to chill, rather promote, progress in the science and useful art of software development. Thank you for soliciting feedback on this matter.

Regards,
Byron Gibson
Honolulu, HI