From: Steve Holden [e-mail redacted]
Sent: Sunday, September 26, 2010 11:43 PM
To: Bilski_Guidance
Cc: [e-mail redacted]
Subject: The Future of Software Patents

Dear Sirs:

I am writing to you to assist the USPTO in its determination of its future approach to software patents in the light of the recent Supreme Court ruling in Bilski v. Kappos (2010, USA). I am aware that many of the representations you receive will be from the legal profession, and I
wanted to counteract those opinions with a view from a professional software designer and implementer.

Firstly, I am forced to question the validity of many extant patents on software. Many appear to fail when tested for obviousness or prior art -
the corpus of work is huge, and I do not seriously believe that the examiners can be aware of all relevant literature. Unfortunately this places a significant advantage in the hands of big business, since they have the financial resources available to obtain the patents, whereas many of those who could successfully oppose them have neither the time nor the financial resources to do so.

The patent system as it currently operates with regard to the software industry is, far from encouraging innovation, stifling the free flow of ideas. Further, many software engineers are loath to even familiarize themselves with even the principles of patented software lest the knowledge "taint" their work. This is directly contrary to the intention
that patents will give a temporary commercial advantage to their holders
in return for advancing the state of the art.

It seems to me that the Bilski decision is potentially of much broader applicability that the USPTO appears to believe. It implies that there are some serious objections to software patents already issued, as well as giving guidance for future patent-ability decisions:

1. Software is mathematics. According to the Church-Turing thesis, all physically computable functions are Turing-computable. Parker v. Flook (1978, USA) established that mathematics is not patentable, and this precedent should be applied to software.

2. Free speech. As a written expression of the ideas that an algorithm embodies, software should be protected as free speech. The appropriate form of intellectual property protection should therefore be copyright, not patent, law. The fact that the written expression can be executed (after suitable mechanical transformations) by a machine is neither relevant nor significant.

3. Impracticality of avoidance. The body of software patents that have now been granted is so large that only the largest organizations stand the remotest chance of being able to know whether a particular

algorithm, when expressed as computer code, violates any extant patents.
Such application of the patent system does not match Congress's intention that patents should promote the progress of science and useful
arts", and therefore the USPTO is arguably hindering innovation and progress with its current approach.

Should the USPTO determine that is wishes to continue issuing patents to
software despite the arguments above then I trust that the following principles could be incorporated into examiners' practices.

1. A "working model"must be provided in the form of readable, demonstrable, testable, code which includes tests to validate the claims
made for the software in the patent. The source code must not be obfuscated in any way, since this contravenes the need for patents to inform the state of the art.

2. It should be the implementation, rather than the idea, that receives the patent grant. In that way others will be able to read the patent and
improve upon the methods it uses, thereby meeting the original requirement that patents should stimulate innovation rather than hindering it.

3. It would be helpful to open the patent examination process up by requiring peer review; this would help the examiners, not all of whom can be software experts, to avoid the grant of patents which are obvious, or which fail to take into account prior art.

I have worked in the computer industry for 43 years, and I feel that it is long past time to end the granting of software patents, or at least restrict their scope significantly as outlined above. Many of my peers in the industry share these opinions, but alas most are so cynical about
the current process that they do not feel there will be any benefit in communicating their feelings. Perhaps it would be appropriate to take wider soundings in the software industry (the IEEE, the ACM and the various open source foundations would all be able to offer useful guidance) before determining how to proceed.

Thank you for taking the time to read these comments, which are made solely with the intention of trying to restore the operation of the patent system to fulfil the true intent of the legislation that established and enforces it.

Yours sincerely
Steve Holden FBCS, CITP, MIEE
--
Steve Holden +1 571 484 6266 +1 800 494 3119
DjangoCon US September 7-9, 2010 http://djangocon.us/
See Python Video! http://python.mirocommunity.org/
Holden Web LLC http://www.holdenweb.com/