From: Brad Allen [e-mail redacted]
Sent: Sunday, September 26, 2010 10:11 PM
To: Bilski_Guidance
Cc: [e-mail redacted]
Subject: feedback on software patent guidelines

To whom it may concern,

As a professional software developer, I'd like to contribute feedback about the USPTO's planned new guidelines for reviewing software patent applications.

I strongly prefer that software patent applications be rejected now and retroactively. Failing that, software patent applications should have stricter requirements in order to discourage frivolous software patents and legalize innovation.

The following points argue for rejecting all software patents outright:

* PRIOR ART: I don't believe the USPTO can or will ever have the resources to determine whether a software design is an example of prior art, any more than a person could read every book ever written. Due to the intractability of determining prior art, software patents should never be granted.

* MATHEMATICAL: Software is mathematics (per the Church-Turing Thesis) and therefore not patentable, according to the precedent set forth in the case of Parker v. Flook (1978, USA). For this reason, software patents should never be granted.

* FREE EXPRESSION: Software is a form of written expression, and should therefore be protected as free speech. Software patents infringe on the free speech of software developers to write down their ideas in the form of source code. Compiling and running source code is nothing more than a sophisticated way to interpret that free expression, like reading a book containing specific instructions. For this reason, software patents should never be granted.

* ECONOMICS: The sheer volume of software created, the increasing rapidity of development, and the fluidity of software

integration renders software patent checking and enforcement impractical and counterproductive. Software developers write code for work, recreation, and community service; most lack the resources to check their code against the system of patent entitlements. Fear and doubt about unrecognized patents hinders innovation and progress. Meanwhile lawsuits abound to enforce protections on absurdly broad and obvious software applications. The United States Constitution established the patent system to "promote the Progress of Science and useful Arts", but prominent studies have shown the opposite effect in the case of software patents. The USPTO has a responsibility to uphold the intent of patent law, and should examine the research showing how software patents hinder innovation and progress.

The following arguments suggest an approach toward stricter requirements for software patents:

* IMPLEMENTATION REQUIREMENT: An invention is not a real invention if it is only a plan or idea. The invention must be functional in a testable way; automated tests must be presented to validate described functionality.

* ALLOW FOR "A BETTER MOUSETRAP": The patent should apply to the implementation, not the high level functionality. Third parties who wish to implement the same idea should not be restricted from pursuing the same functionality as long as the implementation (source code) is publicly disclosed. This requirement puts software patents back in the game of encouraging innovation, by discouraging closed source software. It also discourages frivolously broad patent applications such as "one click purchase."

* SOURCE CODE REQUIREMENT: Human readable source code is a pre-requisite for making a determination of prior art. If the code is obfuscated in any way, then the application should be considered too sloppy to accept.

* DEPENDENCY REVIEW: If a particular implementation is not self contained, but depends upon another piece of software, serious questions should be raised about whether the software is truly original.

* PEER REVIEW: Review of software functionality and implementation is highly subjective. A public peer review process should be established to balance the subjectivity of patent examiners.

This may not be the forum to argue about the societal harms and benefits of software patents; regardless, I hope the USPTO patent officials clearly understand that by accepting software patents they contribute to an accumulating societal cost with no redeeming benefits to society at large. After all, the original purpose of patents was protecting the efforts of those who invested in innovation; today the reality of the software marketplace shows that innovation happens not because of software patents, but in spite of software patents.

The views I have expressed here are common among software developers and in industry news publications. Frequently I hear that many corporations bear heavy costs to patent their software "defensively", and might breathe a collective sigh of relief if they could somehow achieve multilateral software patent disarmament.

Please review my comments with careful consideration, and consider establishing a public forum to further explore these considerations.

Thank you!

Brad Allen
5524 Baker Dr.
The Colony, TX 75056