From: Patrick Simmons [e-mail redacted]
Sent: Monday, September 27, 2010 2:13 AM
To: Bilski_Guidance
Subject: Software Patents after Bilski

As a software developer who also has a keen interest in law, I hope you will interpret the Supreme Court's recent Bilski decision as pulling back from the expansive interpretation of patentable subject matter endorsed by such cases as State Street. In particular, I hope that you will draft guidelines excluding inventions claiming software executed on a general-purpose computer as unpatentable, abstract ideas. In the first part of this email, I will discuss the legal and technical concerns supporting such guidelines. In the second part of this email, I will discuss the public policy concerns motivating such guidelines.

Software source code is fundamentally nothing more or less than an expression of mathematical theorems. Some of my colleagues' work concerns analyzing and proving theorems regarding source code written in languages such as Java. These theorems generally state things like, "this code will not allow unauthorized access to the system," or, "this code will not dereference a null reference." It is only possible for logical proofs of these properties to exist because Java source code is a mathematical expression. The proper way to consider software is as an alternate notation for first-order logic.

Though mathematics is provably software's fundamental nature, software's applications may sometimes seem far removed from this, and I am not suggesting that any patent merely mentioning software should be unpatentable. A novel assembly line method using software only tangentially may very well be patentable, as software was not part of the inventive step. I would suggest, as a rough test in considering patent applications, replacing a software system with an opaque machine deemed unpatentable (due to the abstract nature of mathematics and the obviousness of its execution on a general-purpose computer), and considering a patent application in that light. **In terms of the Interim Guidelines, I would suggest adding clarification that a general-purpose computer is not to be considered a "particular machine" within the meaning of the guidelines.**

Very strong public policy motivations exist for excluding software in this way. Software is easy to modify, easy to distribute, and often created by individuals and non-profit entities. It is routine for academics and others to post packets of source code on a website as a form of discourse. Even the threat of a patent lawsuit is often enough to have a significant chilling effect on this type of technical discourse. Moreover, the pace of innovation in software, because it is easy to modify and distribute, is extremely quick when compared to the pace of innovation in other fields. 20 years is far too long to hold a software idea hostage. Software startups need to be able to deliver innovative products without licensing the basics from 15 years ago from their competitors, and individuals and academics need the freedom to discuss ideas without the chilling threat of lawsuits. Society is best served by protecting software with copyright and trade secrets. From both a legal and societal standpoint, patents should not extend in scope to cover this form of mathematics.

Thank you for your time.

Patrick Simmons
Ph.D. Candidate, Computer Science
University of Illinois at Urbana-Champaign