From: Noah Mendelsohn [e-mail redacted]
Sent: Monday, September 27, 2010 5:07 PM
To: Bilski_Guidance
Subject: Comments concerning Interim Bilski Guidance

This email is in response to the request for comments on the Interim Bilski Guidance.

I have been a professional developer of software for approximately 40 years, and although I have no formal legal training, I have since approximately 1993 paid particular attention to the ways in which software patents both promote and inhibit the development software. I have also played a significant role in the development of Web and Internet software standards, and I have observed the effects of software patents on those as well.

Based on my experience, software patents have had a dramatic negative effect on the vitality of the software field in the United States, and I strongly urge the USPTO to either eliminate or else minimize the applicability of patents to software.

My 40 years of experience developing software included much of the long period when software was not patentable, as well as the more recent time when it has been. Few areas experienced the level of creative innovation that I observed in the software field before software became patentable.
Tremendous investments were made, and many of the software technologies on which we all depend were developed during that period. These include, to pick a few examples from thousands: the fundamental structures of assembler and higher level languages; compilers and associated optimizations; hypertext systems (precursors to the Web); packet-switched networking (fundamental to the Internet); relational databases and associated transaction management technology (key to banking data storage and electronic commerce); etc. All of this occurred without the “incentive” of patentability. While there are no doubt particular cases in which availability of patents has been an incentive for particular software investments, in my experience it’s been rare.

On the contrary, I have observed numerous cases in which the patentability of software has had a significant detrimental effect on the level of investment in, and the efficiency of, software development. It would be impractical for me to list all of the reasons here, so I will select a few with which I have first-hand experience:

** Perhaps the most important point is that the nature and the rate of innovation in the software field is fundamentally different from that in more traditional technology fields. Consider, for example, a patent on an automatic transmission for a car: there will be over a few decades perhaps a few dozen organizations in the world that might need such technology. If they do, they will likely spend years developing each new generation of the technology; in such a situation, discovering and arranging for licensing of the necessary patents may be a modest additional burden. If patents are granted for 17 years, then a few generations of transmission technology may be encumbered before the patent expires.

Now consider instead an innovation in software. With open source in particular, a piece of software may be made public in the morning, and picked up by someone else and improved later that afternoon. The process may be repeated many times over a periods of days, weeks, months and years.

Even if reliable databases of prior art did exist (and I've noted that they don't), the effort to search them would in many cases dwarf the rest of the process of creating the innovation. Furthermore, a patent for an extended period of 17 years might affect the development of many more generations of the technology. In short, if one wanted to have a patent system that would be in proportion, it would (speaking figuratively) have to be reliably searchable in seconds, and paid for in pennies. I have observed repeatedly that when the need to deal with patents arises, these communities dramatically reduce their level of investment. Most good software developers would rather invent great software, than spend their time researching patents.

** Indeed, there has also been great damage caused by the poor historical records of prior art: stated differently, even if a developer wants to to diligently search, it may not be practical to do so. The lack of good records is not an accident, and for the most part it cannot be remedied.

As noted above, many of the most important innovations were done before software was patentable, at a time when nobody had the need or incentive to leave tracks as to which innovations had been embodied in which products or systems. While working for a large computer manufacturer I spent many weeks helping to dig up prior art documentation for technologies that were widely known in the field prior to the issuance of a patent on the same ideas. Society would have been better served if I had spent that time working on new software. One formative experience for me came in 1993, when a patent attorney presented to a group of perhaps 30 programmers a recently issued patent that he thought interesting: of those 30 people, 20 were personally aware of and indeed had used systems that clearly embodied the prior art. The group were not specialists in the field; the system in question was widely used at many universities, but apparently the patent office had not noted it when considering the filing. Nevertheless, with the patent having issued, there was than a presumption of its validity.
Nearly 20 years have passed, and I've seen few improvements in this state of affairs.

** Licensing may be practical when technology is to be used in commercial products built by large organizations. Unfortunately, if you are reading this note in a Web browser, you are almost surely using software and associated standards technologies that were developed in other ways. It's impractical to ask a graduate student to conduct a patent search before including either an innovation or a widely-used technology in a software product or proposed draft standard. Yet, most of the Internet and Web standards on which we all depend were developed, to a significant degree, by students and others working in non-commercial settings.

The above are not just theoretical points. I have seen the real costs of all of them in my many years of work on software and standards. Indeed, as one who has worked on Internet-related standards for over a decade, it's my belief that the patentability of software is one of the most significant threats to the successful evolution of the Internet, the Web, and many of the other systems on which we all depend.

Thank you very much for your consideration of these concerns. I apologize that this note was written in some haste, as I just became aware of the call for comments, and they are due this evening – I hope you will excuse any lack of polish in the presentation. I sincerely believe that these concerns are of fundamental importance to the economies of the US and of other countries in the world. They are also of great concern to those of us who have devoted our working lives to driving innovation in computer software. Again, thank you very much.

Sincerely yours,
Noah Mendelsohn
Software Developer
Lexington, MA