From: jose lorenzo [e-mail redacted]
Sent: Monday, September 27, 2010 7:26 AM
To: Bilski_Guidance
Cc: [e-mail redacted]
Subject: Submission due by Monday, September 27

To whom it may concern,

Thank you for giving the public the opportunity to comment on adjusting the valuation and granting of patents in light of the SCOTUS Bilski decision.

This comment will focus on the restrictions that could take place at the USPTO to limit the granting of software patents so as to bring that domain in line with past Supreme Court rulings, most notably, with Bilski.

The Bilski decision did not say too much that was new. The minority of the Court that seemed more open towards the possibility of a valid patent existing on information processing explicitly stated they did not take a position for or against software patents. Meanwhile, a different and equally sized minority largely rejected software (and business method) patents. Where there was agreement was that the patent in question was abstract (all Justices rejected the patent) and that the machine-or-transformation test for process patents is a very good yet incomplete test.

It's important to mention that the Court appears to reject entirely patents on algorithms and has not cited a single case of an acceptable patent for a software program running on a programmable device if the purpose of that software was merely to manipulate or generate information and perform nothing else of use with that information or merely use that information in very standard ways such as to display the contents of the information. There were frequent appeals to the Flook case, and the references to Diehr focused on the overall transformation being achieved with matter within an industrial process. In contrast, the particular Bilski patent in question which created no new machine and which referred generally to processes humans could carry out with ordinary tools was universally rejected.

Additionally to this, it's important to keep in mind that patent granting potentially involves creating many restrictions on what individuals may do. The section of the US Constitution generally believed to support the Patent Act includes a criteria that the progress be promoted. The First Amendment to the US Constitution supports individual's free speech (especially if fairly original) with few bounds placed on it. For this reason, the USPTO has a great responsibility not to go too far granting exclusivities that would tax too many individuals in ways not supported by our laws.

With all of this in mind, I suggest the USPTO consider adopting the essence of one or more of the following two rules:

*****
R1 -- In light of the fact that the machine-or-transformation test may not cover every single case, but considering that it is a well supported test and exceptions appear lacking, consider placing a soft upper bound to the number of patents awarded annually on inventions that do not pass this test. This soft upper bound should be fairly small, I would presume, and would serve as a check on other USPTO rules. Because granted exclusivities can lead to significant damage to competitors and to consumers (not to mention to individual liberties), the USPTO should not take the attitude of granting when in doubt. Having an upper limit would help contain damage, without removing the granting possibility entirely, and would be a safe position consistent with all prior Court rulings related to "software patents".

R2a -- Do not grant patents (except at most a token sum) for data processing of any kind where the processing is not intimately a part of an industrial process or generally is not intimately a part of a manufacture. A variation of this is to forbid patent grants for inventions composed significantly of software created for and used on consumer devices.

i - The main motivation for this is that such software generally simply manipulate information very much the same way accomplish mentally by humans and rejected by the Court: through pure algorithms entirely founded on mathematics (eg, symbol manipulation) and where the information quantity is all well defined, discrete, and finite (ie, where Mother Nature's secrets have been factored out in the creation of the digitalization processing model (aka, digitalization abstraction)).

ii -- A further motivation is that software creation and distribution is an important part of self-expression. Patents that can hinder such expression of many people are also making moot the protections reserved to the public by copyright law (and which help avoid violations of independent original expression protected by the First Amendment).

iii -- Another important issue is that software creation and use on (programmable) consumer devices is extremely inexpensive to produce and replicate so that patent exclusivities would imply a potentially very significant liability on society.

iv -- Yet another consideration is that vast quantities of open source software are being created, and these provide much to the common good ("promoting the progress" and improving the "general welfare") and usually much more transparency than do patent application details. Currently, open source software is not excluded from the scope of patent grants, and patents are in fact being granted that negatively affect a whole lot of original works created in open source fashion.

R2b -- A softening of R2a would forbid patents for consumer devices only to the extent the software instructions are loaded in a reprogrammable fashion.
****

I would like to add some more discussion in support of these rules.

First, and mostly in support of R2a ii, from Eldred v. Ashcroft, 537 U.S. 186, 219-20 (2003):

>> In addition to spurring the creation and publication of new expression, copyright law contains built-in First Amendment accommodations. See id., at 560, 105 S.Ct. 2218. First, it distinguishes between ideas and expression and makes only the latter eligible for copyright protection. Specifically, 17 U.S.C. § 102(b) provides: "In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work." As we said in Harper & Row, this "idea/expression dichotomy strike[s] a definitional balance between the First Amendment and the Copyright Act by permitting free communication of facts while still protecting an author's expression." 471 U.S., at 556, 105 S.Ct. 2218 (internal quotation marks omitted). Due to this distinction, every idea, theory, and fact in a copyrighted work becomes instantly available for public exploitation at the moment of publication. See Feist, 499 U.S., at 349-350, 111 S.Ct. 1282.

Software patents would appear to run afoul of our First Amendment if the indication above is correct that ideas and processes should not be given exclusivity as a matter of free speech. This is a major reason why patents should not apply to processes and machines that are relatively inexpensive and/or within the reach of most individuals.

[For completeness, I'd like to point out that, in addition to the actual software source code write-up, a *software binary* driving a computing machine is an important way to communicate, as it creates an important unique expression through ordinary output devices. Software source code per se does not substitute for the machine display or machine driven effect any more than a set of written instructions on paper substitutes for a visual work of art generated from those instructions.]

Specific to open source software, such software is designed primarily to be sharable by all. It is free speech in both major senses of the word "free", and this speech most certainly includes the software running on a computing device.

Second, quoted via "Justice Breyer, with whom Justice Scalia joins as to Part II, concurring in the judgment:"

>> “[T]he underlying policy of the patent system [is] that ‘the things which are worth to the public the embarrassment of an exclusive patent,’ ... must outweigh the restrictive effect of the limited patent monopoly.” Graham v. John Deere Co. of Kansas City.

Third,

>> the Court has long held that “[p]henomena of nature, though just discovered, mental processes, and abstract intellectual concepts are not patentable” under §101

Software's entire essence is the precise step by step application of a model. This is made possible by the abstraction referred to as "digitalization". It is completely removed from all tangible realities except to the extent it is executed by a real thing (eg, by a human mind or by a computer) and to the extent it is intended to model a part of reality. In one important use, it creates virtual realities (eg, as is also the essence of fictional works). In other uses, it's little but a glorified calculator carrying out the supplied mathematical algorithms.

For these reason, I believe software patents should not be awarded unless, as the Court has ruled and as was reinforced strongly in Bilski through several agreeing minority opinions covering the majority of Justices, it is but a part of a larger process involved in the transformation of matter.

Exceptions might exist, but these would likely be very limited in number if existing at all (and, hence, the recommendation for R1 given above).

Fourth, we can find lots of support by the Court for rejecting computer applications.

From the majority opinion in Bilski:

>> §101. 409 U. S., at 64–67. The Court first explained that “ ‘[a] principle, in the abstract, is a fundamental truth; an original cause; a motive; these cannot be patented, as no one can claim in either of them an exclusive right.’ ” Id., at 67 (quoting Le Roy, 14 How., at 175). The Court then held the application at issue was not a “process,” but an unpatentable abstract idea. “It is conceded that one may not patent an idea. But in practical effect that would be the result if the formula for converting ... numerals to pure binary numerals were patented in this case.” 409 U.S., at 71. A contrary holding “would wholly pre-empt the mathematical formula and in practical effect would be a patent on the algorithm itself.” Id., at 72.

>> The application’s only innovation was reliance on a mathematical algorithm. 437 U.S., at 585–586.

>> Nevertheless, Flook rejected “[t]he notion that post-solution activity, no matter how conventional or obvious in itself, can transform an unpatentable principle into a patentable process.” Id., at 590.

>> The Court concluded that the process at issue there was “unpatentable under §101, not because it contain[ed] a mathematical algorithm as one component, but because once that algorithm [wa]s assumed to be within the prior art, the application, considered as a whole, contain[ed] no patentable invention.” Id., at 594.

>> As the Court later explained, Flook stands for the proposition that the prohibition against patenting abstract ideas “cannot be circumvented by attempting to limit the use of the formula to a particular technological environment” or adding “insignificant postsolution activity.” Diehr, 450 U.S., at 191–192.

We note carefully as well:

>> Diehr explained that while an abstract idea, law of nature, or mathematical formula could not be patented, “an application of a law of nature or mathematical formula to a known structure or process may well be deserving of patent protection.”

>> Finally, the Court concluded that because the claim was not “an attempt to patent a mathematical formula, but rather [was] an industrial process for the molding of rubber products,” it fell within §101’s patentable subject matter. Id., at 192–193.

From a quote via "Justice Breyer, with whom Justice Scalia joins as to Part II, concurring in the judgment:"

>> “[t]ransformation and reduction of an article to a different state or thing is the clue to the patentability of a process claim that does not include particular machines.”

>> Rather, the Court has emphasized that a process claim meets the requirements of §101 when, “considered as a whole,” it “is performing a function which the patent laws were designed to protect (e.g., transforming or reducing an article to a different state or thing).”

The machine running software is assumed not to be a new machine from the machine as it existed beforehand, any more than a human becomes a new human when carrying out a different set of steps or following a different newly learned mental algorithm. I believe "process" patents were intended to cover the cases of the same machine working in a different fashion.

All processing that exists in software are mental steps. In order to be eligible for patents (according to the Court), these can be a part of a machine that performs something otherwise patentable, much as if the software were factored out and a human were sitting at the controls of such a patentable machine performing the calculations speedily and then adjusting levers accordingly.

To look at an example: A general purpose computing device with an additional display screen, printer, scanner, keyboard, mouse, etc, as used by ordinary consumers, has no extra patentable component when the hardware components are taken as a whole (each individually patentable perhaps) and when software is run on these machines where the result of the software processing is merely used to produce information eventually to be presented. The processing of the information is achieved purely through digital abstraction mental algorithms designed for a well-defined abstract machine and mimicked on an existing machine that adheres to such a model. This abstraction of data processing itself is not patentable, and the conversion of the resulting data into, eg, light waves is but obvious and conventional post-solution activity provided by a standard display screen (and is not creating a newly patentable process or machine). There is, notably, no new transformation of matter beyond what is already anticipated by the individual components.

Note in the prior example that all processing done by any software on the general purpose machine (no matter its input source) merely produced as an end goal a form of ordinary data input (eg, digital values) to the machine components (like the printer or display screen or hard drive or memory) that then carry out transformations of matter in the expected ways (ie, in no longer novel ways). This is true no matter how novel are the algorithms (mental steps) that process the data. There is no novel transformation occurring of matter.

Fifth, a word on abstract.

Information processing is an abstraction. It is the equivalent of mental steps no matter which calculator does the processing and which display shows the results.

The Bilski patent was rejected for being abstract. It too provided little beyond steps people could undertake using any of many existing suitable tools (hardware+software combinations) at their disposal.

Conclusion:

I consider software patents a large threat to personal liberties and to progress. They are particularly threatening (a) because independent creation through software is accessible readily to any and all yet would arguably be trumped by patent exclusivity grants; (b) because of the very large number of progress promoters whose hands would be hand-cuffed; and (c) because of

the very low bar set for granting patents, which, by definition, would curtail the efforts of the many people having above average skills in the art.

Again, thank you for allowing me to express how I think the Bilski ruling should help the USPTO revert patent granting to the form when few to no inventions based just about entirely on the novelty of effects carried out through software would be granted patents.

Jose Lorenzo
hozelda@yahoo.com
[e-mail redacted]