Why I don't call software 'technology'

Ben Klemens

9 June 2017

[Though I work in policy, and used to be the tech policy person for a notable think tank, I try to not talk about politics on this site—I have Bureauphile[1] for that. But this is my 200th post on this blog, so it's like my birthday and I'll post what I feel like posting about.]

Here's an amusing anomaly that has grown popular: software people have started calling the software community the *tech community*, and referring to things like Java or structured query language as *Java technology*[2] or *SQL technology*[3].

We might just take this as puffery, like how people used to add *engineer* to their job title, leading to the joke about trash collectors being *sanitary engineers*. I suppose this is socially harmless.

But there are serious policy implications.

Patents Until a few days ago[4], the Patent Office was run by the former Vice President for Intellectual Property at Google, Michelle Lee. She replaced the former Vice President for Intellectual Property at IBM, Dave Kappos. To the best of my knowledge, both of these people support a *technical effect* standard for patenting of intangibles like software or medical facts.

What is a technical effect? Who knows. A technical effect standard has never existed under US patent law, but as things like medical tests and software were ruled to be unpatentable because they were too close to unpatentable abstractions like facts about biology and pure math, people started talking about the technical effect standard as an effort to get more things classed as an invention instead of a discovery.

One proposed manifestation of the technical effect rule would be that software per se should not be patentable, but an algorithm that improves the functioning of a computer[5] should be.

I see that face you're making, and I agree—the distinction, if any, is too hair-splitting to be tenable in the real world. The intent is that *an algorithm to schedule pet food deliveries online* should not be patentable but *an algorithm to select schedules for optimization via simulated annealing* should be, because the second is just so much more...technical.

[1] http://bureauphile.com

[2] https://duckduckgo.com/?q="java+technology"&t=vivaldi&ia=web

[3] https://duckduckgo.com/?q="sql+technology"&t=vivaldi&ia=web

[4] https://www.washingtonpost.com/news/the-switch/wp/2017/06/06/the-director-of-the-u-s-patent-office-just-abruptly-resigned/

[5] https://www.b2ipreport.com/swip-report/federal-circuit-clearly-says-software-can-be-patentable/

This is a distinction made of pure squish, and lawyers who don't know how to write code might see something there. But I expect that if you're reading this blog you can easily think of situations from your own work where you solved serious technical problems in assembling the pet food delivery systems in your life. You can probably also think of Greek-laden math you could do to wow friends and judges but which you know to be a little trivial.

But under a technical effect rule, language clearly matters. If I were in a team of lawyers working hard to change the opinions of judges by any means possible, so we could make our software patentable under a technical effect rule, I would be sending out memos all day long: *from now on, we will refer to all software as "technology." Remember, technology is patentable but software isn't.*

Tax and funding As a general rule, things associated with technological development will always get a better deal from the government—who doesn't like technological advancement? So there are some carve-outs in tax law for different kinds of intellectual property. Should software be included?

There are research credits and a preferential expensing scheme for research costs. These are broadly defined, and would include even the development of a pet food delivery system. So those seem to cover software whether it is technology or mere development work. But there are other points of open debate.

In the 2013–2014 Congressional session, Congressperson Schwartz (D–PA) introduced a bill[6] proposing a *patent box* that would tax income associated with patents at a 10% rate, rather than the typical 35% corporate rate. This bill was entitled "The Innovation Promotion Act of 2015" just to drive in how this is ostensibly about revolutionary new technology.

The Boustany-Neal proposal (PDF[7]) had a broader intellectual property box, which included in its list "Any program designed to cause a computer to perform a desired function." This was a win for a lobbyist somewhere, because if software is largely not patentable in the present day, the Schwartz proposal doesn't give Microsoft et al the tax break they would get in a bill that explicitly includes software.

I won't belabor the analysis of the IP box proposals here—I wrote a 45-page working paper to do that[8]. But setting aside whether it's a good idea or not, these bills already show that the decision of whether software is innovative technology or a mere clerical function has a real policy effect, the difference between a 35% and 10% tax on domestic revenue.

Maybe you want software to be patentable; maybe you are excited by the thought that Apple will finally get the domestic tax break it deserves. Maybe you work in any field but software, or lean toward free and open source code, and wonder why government should give for-profit software vendors a financial leg up over your work, and potentially the right to sue you. Maybe you want Google to pay something more

[6] https://www.congress.gov/bill/113th-congress/house-bill/2605/text

[7] http://waysandmeans.house.gov/wp-content/uploads/2015/07/Innovation-Box-2015-Bill-Text.pdf

[8] https://www.cepweb.org/intellectual-property-boxes-and-the-paradox-of-price-discrimination/

than a 17% effective tax rate in the USA[9] [and 0.16% in the EU![10]].

This gentleman[11] can dictate to you that *data* is not a Latin plural, but I can't dictate to you whether to call a repeating block of code *for loop technology*. I can point out that there are constant efforts to push software into a box where firms can package it into patents and are given credits and discounts for doing so. There's no grand conspiracy (as far as I can tell), but this little quirk of calling software *tech* directly supports those efforts, which is worth being aware of should you choose to use it.

[9] https://www.sec.gov/Archives/edgar/data/1288776/000165204416000012/goog10-k2015.htm

[10] https://arstechnica.com/tech-policy/2014/07/after-moving-money-around-google-paid-tiny-amount-in

[11] http://nxg.me.uk/note/2005/singular-data/