Vidit Joy Manglani

Dr. Daniel Tomasevich

Computer Science 300 GWAR

13 December 2017

A Thousand Ideas a Day, Thrown into the San Francisco Bay

Making your idea into a product, without losing the ability to sell it

For someone training to be a software developer, the fact that software is inseparable from its source code, is a saving grace. If there's something you want to learn how to do, you can find where's it's been done, reverse-engineer it and learn it.
The same fact about software, from the point of view of an individual trying to make a fair buck in the open-market is not just a problem but a wicked one. It has many unyielding horns and requires a systemic approach to solve; and cuts and nicks are almost a guarantee.

An independent software developer, especially one living in the San Francisco Bay Area, faces the first of such encounters even before the idea has a chance to develop, fully, in their minds. And the journey from idea to product is long and arduous. And the hard-work begins at the start. The much needed feedback loop, from peers or prospective users, even in the infancy stages, is balanced by the fear of a valuable idea being unscrupulously or unsuspectingly picked up by someone with more resources than oneself. If the idea they are thinking of, is an idea capable of serving a purpose,

the to-be-successful software developer naturally asks himself the following questions, how do I build it, how do I get it to the market that needs it, and how do I protect my self, the idea and my investment in the product, in the process.

Since software is copyrightable, outright reverse engineered imitations are not threats, except for the time lost litigating. But an outright imitation, reverse engineered or not, with sufficient differences, perhaps with a comma here and a comma there, make for two different copyrights, and unhealthy competition.

Although the above situation applies to patents software patents too, they offer more protections than copyrights because – they protect the idea and not just the actual product. To be more precise, they protect the implementation of the idea through its description; the embodiment of the idea. Ideas themselves cannot be protected by patents, for they in the public domain. And rightfully so since ideas are less visceral than creations. The public too does need some sort of protection from business interests and for business interests.

That seems to be the to-be-successful software developers smoking gun, patent the embodiment of your idea (akin to a detailed product plan), before you build the software and you're safe till you go commercial, and for 20 or so years after. Sounds like a dream, because it very well might be. In his paper on the problems with Intellectual Property Law, Andrew Beckerman-Rodau clarifies "patent law insures a lengthy process in which a utility patent will only issue if the software is found to be both novel and sufficiently innovative [1]."

Patent applications, especially when they are authentic, need to be precise, unique in every way [that is part of the patent], and thus take time to make, review and get reviewed, ideally by lawyers. Some capital will be required, obviously; of both time and money. One can't just expect to get a patent for something without having checked if something remotely similar exists. In his paper on avoiding IP problems Thomas G. Field Jr. says "proceeding without [patent searches] is akin to erecting a building or signing a long-term lease without clearing the real estate title." [2] Clearly not something even the most ardent DIY'er would do without professional assistance – it really would not be wise to do so. Perhaps you could ask a lawyer friend to look into it – But really, they're probably not going to take on a something of the required scope and detail without some remuneration.

It is unlikely then for an independent software developer (at least in the startup phase) to have the needed resources to expend in pursuit of a patent for an untested product. One saving grace they may have is the clause in the patent law that allows improvements of past patents to be additionally patented but the time/money requirement to research earlier patents is still required, albeit reduced. Should they then spend the time researching the patent application or risk building out the product, in the hope to attract investors for patenting or perhaps even try putting it out in the market without a patent ?

Self-dependent research will be ultimately useful for future patent applications, although the specifics of one might not carry to the other and much of the research may need to be conducted again anyways. But some amount of research and self-education would

be a given with all three possibilities. Any additional research though, for an independent software developer, would only make sense if they intended to actively pursue patenting software in the future, in which case too getting some funding would be beneficial.

If their main intention was to get into business though, involving themselves in the ins and outs of Intellectual Property Laws in the U.S or otherwise, would not be the wisest use of their time. Unfortunately, though if they wanted to sell their product, they would have bite the bullet and comb through all their code for processes they imagine might already be patented. “Owners can halt the unauthorized practice of covered inventions...Copying isn't needed to recover damages or to stop others from practicing...Deliberate infringements generate larger awards, but ignorance is no defense” [3].

Fortunately, this process of outlining possibly patentable (and patented) processes and services within one’s own planned product and research for possibly similar patented products, itself, would show the developer the likely processes they can patent themselves, and protect their own inventions from others; while protecting their ideas from unwittingly and arguably infringing on another’s intellectual property. And in the process also gaining a market advantage and improving their value as a prospective investment. “The worst possibility is that a current patent or pending application covers part of their design, If so, one needs permission to use it” [2] , or they would have to modify their embodiment to avoid using an existing and active patent.

When making a novel software application, one that would be useful to the community and that could be made profitable, a software developer ensures not to infringe on any copyrights while also making sure that their idea is unique. Not infringing on copyright is easy, not infringing on patents, considering the number of vague patents filed, is not so easy. On a hunch, most software developers, like businessmen, see an unfulfilled demand in the market and see that they have the tools to serve it. They go about building up their idea into a product and hopefully a business, they want to protect their idea, their product and their business from competitors that steal their idea and challenge them in the open market. If the competitor is a fair one, like a competing idea or implementation, then the open market decides. But since nefarious activity is always abound, like any businessman with a novel idea but with an openly visible implementation, one the software developer sweated into existence, they would like to get a patent to protect their right to profit from it. This can be fairly challenging, but an accomplishable, and even a rewarding task in the open market. But how does a software developer working to accomplish this also protect their ideas from unscrupulous individuals disguised by shell corporations who sue based upon lofty ambitions of uninteresting and untrue, but patented, inventions, meet Patent Trolls. When a lawyer demand you pay their client a royalty fee for an inane, fairly important, and probably non-functional feature of your software, that really shouldn't have been patented 'cause everybody's already doing it, and is not something really that you should even be liable for, but according to some absurd law district where the suit is filed you are, for a patent of which you are by now sure is not the invention of the

fictional entity being allegedly represented by this lawyer is the scenario you will find yourself in when you are being hit by a patent troll.

For an aspiring inventor, patentable software holds more importance than copyright alone and that's step one of a software business, to actually start up. And step two is stay up. Staying started is easy, but once the ball gets rolling the hollow Patent Trolls coming knocking on the new office door. In all cases, when the ball is already rolling, for companies big and small, legal roundabouts are kicks in the wrong direction. For everyone! Austin Meyer, in his movie, The Patent Scam, shows that eventually consumers are affected by prices increases to pay legal fees and by stalling innovation because of redirected resources. So like any three-act story, we will take step one to solve step two eventually to discover step three, turning from a product, into a company.

Open source and open documentation, the boilerplate of the software industry is also the bread and butter for Patent Trolls. Well documented software that is legitimately based on open source principles, often seen as a sign of a good product to purchase, or invest in, is also a good one for patent trolls to litigate with. DBL Lawyers offer the following advice, "Protect your documents and your research. Patent trolls comb the internet looking for virtually anything that can be twisted into a patent infringement accusation. Make certain that your research has the maximum protection from computer hackers and from anyone else who wants to snoop around and look at your confidential information. Don't put anything on your public website that might interest a patent troll" [3].

A software developer who thinks that by ensuring that they have followed the licensing instructions to use copyrighted 'code', not infringing on any trademarks and

patenting their own innovations, they would be safe or perhaps immune to future uncalled-for litigation would be highly mistaken. Not only are they liable for infringing upon intentionally vague patents but also patents whose owners didn't invent the idea they patented nor ever intended to. "The only way to discover potentially blocking patents is to search U.S. Patent and Trademark Office (PTO) grants (and published applications)" [2]. As per the PTO statistics website that's 500,000 patents, each year [3]. To the already long list of non-software related tasks the software innovator has to add 'hire a lawyer'. Those 3 words, along with the Trolls out to play, mean that an independent software developer can no longer, under normal circumstances, expect to develop and profit from inventions without litigation, without financial support.

A leading example of the litigation an independent software developer can expect is currently being experienced by Austin Meyer, the developer of a popular, innovative and in comparison to its competitors, an independently developed, flight simulation software. He goes over the details of the lawsuit on his website[4], where he mentions the section of the patent he was being sued for allegedly copying. "code for verifying the license data stored on the licensing medium by communicating with a registration authority having verification data"

He adds "Uniloc is suing myself, and 8 other developers, for distributing Apps on Android, claiming that we infringe on "their" idea."

He also shows the frivolity of the US PTO and the lawsuits by exemplifying multiple software implementing the above mentioned section, from a patent filed in 2001, for years before the patent was filed. And in fact another patent filed in 2011 claiming

ownership over the same idea, in different works. And mentions the 2.1 million patens that are in force now.

With that many patents around, and the frivolity of the claims, a software developer may be inclined to think that it would be unlikely for them to get pulled into litigation let alone lose the right to sell the software they have developed. If they find themselves so careless, it would do them good to consider the number of companies that had to shut shop just to litigate in the east Texas town, where most if not all Patent Troll suits are filed. And to consider that the judges in that town are related to the lawyers that file the suits against the unsuspecting companies. Or the jurors that benefit from the 'tourism' that befalls their town every time a company, big or small, is forced to come litigate in their town, for stealing ideas from their folk.

Going out on to the streets, when there's a gang of murderous motorists doing the rounds of your neighborhood, is obviously a bad idea, but if you do have to get your product out, making sure you hire lawyers to protect you, your customers and your product, is probably the best idea.

Works Cited

1. Beckerman-Rodau, Andrew (2011) "THE PROBLEM WITH INTELLECTUAL PROPERTY RIGHTS: SUBJECT MATTER EXPANSION," Yale Journal of Law and Technology: Vol. 13: Iss.1, Article 2.
Available at: h p://digitalcommons.law.yale.edu/yjolt/vol13/iss1/2

2. Thomas G. Field, Jr., Professor Emeritus. IP Basics: Avoiding Patent, Trademark and Copyright Problems.University of New Hampshire School of Law Franklin Pierce Center for Intellectual Property.

3. Dunlap Bennett & Ludwig PLLC. Protecting Yourself From “Patent Trolls” https://www.dbllawyers.com/intellectual-property/protecting-patent-trolls/

4. U.S. Patent Statistics Chart. Calendar Years 1963 - 2015 https://www.uspto.gov/web/offices/ac/ido/oeip/taf/us_stat.htm

5. Austin Meyer. The Patent Scam. Gravitas Ventures. 2017

6. Austin Meyer. Details of the Uniloc Lawsuit. http://www.x-plane.com/x-world/lawsuit/details/