To: Andrei Iancu, Director of the United States Patent and Trademark Office
From: Alice Chen
Re: Recommendations for Software Patent Reform
Date: March 22, 2019
Section: 203 (TA-First-Name TA-Last-Name)

This policy memo discusses problems with the current U.S. patent system with a specific focus on software patents and recommends policies for action by the United States Patent and Trademark Office (USPTO). Although the USPTO does not have a definition of or collect information on "software-related patents," since the granting of the first software patent [4] in 1968, software patents have occurred in a variety of technologies containing at least some element of software, covering things like sending messages or conducting business over the Internet (e.g., e-commerce) [17].

The groups with the greatest stake in the patentability, validity, and scope of software patents are innovators (e.g., big tech companies, startups, and individual developers), patent lawyers, and non-practicing entities (i.e., "patent trolls" who buy patents in order to profit by means of licensing or litigation instead of actually creating any new products or coming up with new ideas). Broadly speaking, however, all users of products containing (to-be-)patented software are affected by the current state of software patents. The remainder of the memo is organized as follows.

Section I reviews key cases and the law that shape how the USPTO treats software patents.

Section II identifies three major problems with software patents in the U.S.:

- The slow process in getting a software patent application reviewed and decided upon;
- The low quality of granted software patents in general; and
- The difficulty of challenging bad software patents efficiently.

Section III dives into three policy suggestions of varying degrees of radicalness for implementation at the USPTO:

- Roll out clearer guidelines for granting or rejecting software patents;
- Strengthen collaboration with the tech community, universities, and law schools; and
- Invest in an internal information technology (IT) system revamp in view of the growing power of artificial intelligence (AI).

Finally, Section IV concludes this memo with the limitations of the policies suggested and their implications beyond software patents.

I. Background: Key Cases and the Law

The purpose of this section is to lay the foundation for understanding the current landscape of the software patents and the USPTO's role in it by examining software patent-related judicial and legislative history.

Statutory Patentability. Patent examiners at the USPTO grant or reject patent applications based on the law. According to Sections 101-103 and 112 of the Patent Act, for any invention to be

granted a patent (i.e., to be patentable), the subject matter must be eligible (i.e., it must be a process, machine, manufacture, or composition of matter, or improvement thereof) [7] and the invention must be novel [2], non-obvious [1], and well-disclosed [18].

Subject Matter Eligibility. It is unclear from the Patent Act whether and which software patents would be subject matter eligible, and patent examiners must determine subject matter eligibility in line with judicial precedents. In 1978, in the case of Parker v. Flook, the Supreme Court of the United States (SCOTUS) came close to banning software patents because software is just math, and "an algorithm, or mathematical formula, is like a law of nature. . . a law of nature cannot be the subject of a patent." Flook, the patent applicant, argued that his patent was more than just math because he had combined a computer program with a real-world industrial application, but the SCOTUS was unimpressed: "The notion that post-solution activity, no matter how conventional and obvious in itself, can transform an unpatentable principle into a patentable process exalts form over substance." On the other hand, three years later, the SCOTUS approved a software-related patent on the idea of using a computer to manage a rubber-curing process, dynamically calculating the optimal time to open the press in order to produce perfectly cured rubber parts: "Their process admittedly employs a well-known mathematical equation, but they do not seek to pre-empt the use of that equation." Given those rulings, it was almost impossible to patent software back then [13].

That situation changed in 1982, when Congress created a specialized appeals court for patent cases, the United States Court of Appeals for the Federal Circuit (CAFC). The CAFC loves software patents; the SCOTUS is more skeptical. In the next couple of decades, software patents like Amazon's infamous 1999 patent on the concept of shopping with one click [5] began to proliferate. In the landmark 2014 SCOTUS decision of CLS Bank v. Alice, the justices ruled unanimously that the patent—which claimed a software-based method of making sure two parties to a financial transaction keep their promises by having a third party hold their funds—was too abstract: "The relevant question is whether the claims here do more than simply instruct the practitioner to implement the abstract idea of intermediated settlement on a generic computer. They do not." In the first three years after Alice, the CAFC rejected 92.3 percent of the patents challenged under the Alice precedent [13]. Now a patent claim under examination is rejected at the USPTO or the courts if it (1) contains an abstract idea (which is true for most software patents), and (2) does not add to it something extra that embodies an "inventive concept" (the "two-step test" or the Alice test).

Opposition Procedures. The USPTO currently offers several ways of challenging a patent's validity, but it was not always the case. It used to be the case that patents can only be invalidated by the courts [14], thereby allowing patent trolls to force their targets to pay hefty settlement demands in order to avoid the even heftier litigation fees. With the passage of the Leahy-Smith America Invents Act (AIA) in 2011, post-grant opposition procedures are expanded. In particular, the AIA expanded inter partes reexamination to become inter partes review (IPR) [11]. The IPR process allows the Patent Trial and Appeal Board (PTAB) to invalidate a patent, contributing to the sharp fall of cost of patent infringement litigation in general [15].

II. Problems with Software Patents

In this section I flesh out three major issues with software patents.

Slow Process. It takes nearly 2 years on average before a patent application is granted (if at all) [8], but software and software-related products usually have a life cycle as short as a few months to a couple of years. The situation is much better now than ten years ago, when the average pendency at the USPTO was over 3 years [16]. The USPTO currently has a backlog of over 540 thousand pending applications [8]. There is simply not enough patent examiners to review the patent applications, let alone carefully and thoroughly research previous inventions or publications, known as "prior art." The slowness in getting a patent granted alone defeats one of the two main purposes of the patent system, namely, to reward and incentivize the inventors. As a result, in the interest of innovators, patent attorneys tend to draft software patent claims that are as wide-ranging as possible to protect the basic technical concept used in a string of future products or to cover potential improvements and innovations by their clients' competitors.

Low Quality. Granted software patents often contain vague and overbroad claims, and the key ideas are usually obvious "to a person having ordinary skill in the art to which the claimed invention pertains" [1]. Even though patent examiners try very hard to find grounds to reject each patent, their time, energy, and tools are limited. The general low quality of software patents defeats the other of the two main purposes of the patent system, which is to disclose technical advances to the public [3]. We end up with software patents that are not useful either because the language is so vague and broad that a developer eager to learn the technical advances would have to tirelessly code and debug on their own before there is any viable product (since actual code is optional in a software patent), or because the implementation is so obvious that any programmer would be able to independently "invent" the same thing without knowledge of the patents. In addition, vague and overbroad patents prevent other people from patenting specific but truly original patents in a similar field, further stifling innovation [9].

Costly Opposition. If it is hard to ensure that granted patents are generally of high quality, it does not help if it is also hard to challenge the validity of bad patents. Introduction of procedures such as IPR made it easier to invalidate bad patents, but the USPTO's recent changes to how claims are interpreted in trial proceedings (including IPR) before the PTAB made it harder again [20]. Instead of the broadest reasonable interpretation (BRI) standard previously used before the PTAB, "ordinary and customary meaning" is in place now instead. Consequently, challenged patents would be interpreted more narrowly and thus less likely to be invalidated over prior art. Even without the changes, the cost to file an IPR easily goes up to six digits, which is admittedly much cheaper than going to court but still extortionate for most small companies and individual developers who may seek to invalidate bad software patents owned by patent trolls or large companies so they could develop their own products [15].

III. Policy Proposals for the USPTO

In this section I propose three policies that the USPTO can implement to alleviate the problems above. All of them are based on past or ongoing initiatives at the USPTO. The goal of these policies is to help the patent system better serve its intended purposes of encouraging innovation and making useful information public.

Clearer Guidelines. [12] As can be seen from the convoluted judicial history of software patents and the tug-of-war between the SCOTUS and the CAFC, patent examiners and administrative law judges at the USPTO need clearer rules to guide them when they decide whether software patents meet the requirements of subject matter eligibility, among other things. In January 2019, the USPTO announced revised guidance for determining subject matter eligibility. Abstract ideas are now divided into concrete categories such as mathematical concepts, certain methods of organizing human activity, and mental processes. That helps eliminate some uncertainty in the application of the first step in the Alice test, namely, whether the patent claims are directed to an abstract idea. The guidance also "explains that a patent claim or patent application claim that recites a judicial exception is not 'directed to' the judicial exception if the judicial exception is integrated into a practical application of the judicial exception" [19]. That aims to make the second step in the Alice test – whether the patent contains an inventive concept – easier to apply. The revised guidance is laudable progress in making the patent system more consistent and easier to navigate for staff at the USPTO, but I think we need clearer, more comprehensive guidelines that aim to cover issues related to software patents beyond subject matter eligibility for the staff as well as innovators and attorneys. Ideally, the guidelines will cover the recommended claim terms, what counts as enough disclosure, and what counts as non-obvious over prior art. As a consequence, there will be higher quality software patent applications and more efficient and predictable examinations and reviews. Implementation of this proposal would likely involve a committee reviewing areas of software patent-related law and policy that is unclear, drafting guidelines, and soliciting public feedback before rolling it out to the USPTO and the public. Although the USPTO does not have general substantive rulemaking power, its guidelines will likely have a significant impact on the landscape of software patents since any legislative effort takes a long time and the result is uncertain [6].

Stronger Collaboration. [12] More and more students are majoring in computer science, more and more people are getting involved in open-source projects, and more and more law schools are welcoming applicants with a technical background (by, for example, accepting the Graduate Record Examinations in addition to the Law School Admission Test). Now is the perfect time for the USPTO to consider untraditional ways of collaborating with the broad tech community, universities, and law schools to improve the overall quality of software patents and facilitate patent examination and review. For example, the USPTO could consider restarting and actively promoting its Peer-to-Patent program [10] across different schools and open-source communities so that interested people could submit potential prior art for patent examiners' consideration. Previous pilot projects have had support from schools, technology partners, and corporate sponsors. To avoid potential conflict of interest, the USPTO could consider open-sourcing the entire Peer-to-Patent system, reducing the cost of the program and making it better known in the tech community. In addition, the USPTO could hire interested computer science majors and law students with technical backgrounds as interns to help review and shed light on software patent applications. If no budget can be diverted to hire additional interns with a tech background, the USPTO may ask universities and law schools to fund their students during the internship since what they do is beneficial to their studies and promotes public welfare.

Smarter System. Modernizing the IT system has been an ongoing effort at the USPTO [10] but with the advent of drastically improved performance of AI these years, perhaps AI may be incorporated into the patent examination and review system to help expedite various procedures and

improve the consistency and predictability of decisions at the USPTO. AI would be especially useful in prior art search and may even be able to recommend decisions based on judicial precedents and past examiners' decisions using machine learning techniques. Initial investment in AI technology may be costly, but the long-term benefits far outweigh the costs.

IV. Conclusion: Limitations and Implications

What the USPTO can do is limited. A comprehensive and continuous software patent reform would also require joint efforts from Congress, the courts, and tech companies. Nonetheless, I hope this memo gives you additional inspiration. Beyond software patents, my proposal for a smarter system naturally affects the patent system in general, and those for clearer guidelines and stronger collaboration with the public can be easily adapted to other sectors. The proposals require significant time and resources to implement, but they all aim to bring an ideal patent system – one that stimulates innovation and promotes disclosure of inventions – closer to reality.

References

[1] "Conditions for patentability; non-obvious subject matter". In: *35 U.S. Code §103* (Sept. 2011). URL: https://www.law.cornell.edu/uscode/text/35/103 (visited on 03/11/2019).

[2] "Conditions for patentability; novelty". In: *35 U.S. Code §102* (Dec. 2012). URL: https://www.law.cornell.edu/uscode/text/35/102 (visited on 03/11/2019).

[3] Nuno Pires de Carvalho. "The Primary Function of Patents". In: *Journal of Law, Technology & Policy* 1 (2001), pp. 25–74.

[4] Martin A. Goetz. "Sorting system". US3380029A. Apr. 1965.

[5] Peri Hartman et al. "Method and system for placing a purchase order via a communications network". US5960411A. Sept. 1997.

[6] Andrei Iancu. "Remarks by Director Iancu at the Intellectual Property Owners Association 46th Annual Meeting". In: *U.S. Patent and Trademark Office News & Updates* (Sept. 2018). URL: https://www.uspto.gov/about-us/news-updates/remarks-director-iancu-intellectual-property-owners-46th-annual-meeting (visited on 03/11/2019).

[7] "Inventions patentable". In: *35 U.S. Code §101* (July 1952). URL: https://www.law.cornell.edu/uscode/text/35/101 (visited on 03/11/2019).

[8] *January 2019 Patents Data, at a Glance*. URL: https://www.uspto.gov/dashboards/patents/main.dashxml (visited on 03/11/2019).

[9] Adi Kamdar, Daniel Nazer, and Vera Ranieri. "Defend Innovation: How to Fix Our Broken Patent System". In: *Electronic Frontier Foundation* (Feb. 2015). URL: https://www.eff.org/document/defend-innovation-how-fix-our-broken-patent-system (visited on 03/11/2019).

[10] David Kappos. "Modernizing a 21st Century Patent Office". In: *U.S. Patent and Trademark Office News & Updates* (Apr. 2011). URL: https://www.uspto.gov/about-us/news-updates/modernizing-21st-century-patent-office (visited on 03/11/2019).

[11] *Leahy-Smith America Invents Act*. Sept. 2011. URL: https://www.uspto.gov/sites/default/files/aia_implementation/20110916-pub-l112-29.pdf (visited on 03/11/2019).

[12] Michelle Lee. "Building a Better Patent System". In: *U.S. Patent and Trademark Office News & Updates* (Feb. 2014). URL: https://www.uspto.gov/about-us/news-updates/building-better-patent-system (visited on 03/11/2019).

[13] Timothy B. Lee. "Tug-of-War—Why a 40-year-old SCOTUS ruling against software patents still matters today". In: *Ars Technica* (June 2018). URL: https://arstechnica.com/features/2018/06/why-the-supreme-courts-software-patent-ban-didnt-last (visited on 03/11/2019).

[14] Orlando Lopez. "Inter Partes Review: After Five Years, What Will 2018 Bring?" In: *Burns & Levinson News* (Apr. 2018). URL: https://www.burnslev.com/news/inter-partes-review-after-5-years-what-will-2018-bring (visited on 03/11/2019).

[15] Malathi Nayak. "Cost of Patent Infringement Litigation Falling Sharply". In: *Bloomberg BNA* (Aug. 2017). URL: https://www.bna.com/cost-patent-infringement-n73014463011 (visited on 03/11/2019).

[16] *Pendency of Patent Applications (2 visuals)*. URL: https://developer.uspto.gov/visualization/pendency-patent-applications-2-visuals (visited on 03/11/2019).

[17] Frank Rusco. "Patent Office Should Define Quality, Reassess Incentives, and Improve Clarity". In: *U.S. Government Accountability Office Report to the Chairman, Committee on the Judiciary, House of Representatives* (June 2016). URL: https://www.gao.gov/assets/680/678113.pdf (visited on 03/11/2019).

[18] "Specification". In: *35 U.S. Code §103* (Sept. 2011). URL: https://www.law.cornell.edu/uscode/text/35/112 (visited on 03/11/2019).

[19] U.S. Patent and Trademark Office. "2019 Revised Patent Subject Matter Eligibility Guidance". In: *Federal Register* (Jan. 2019). URL: https://www.federalregister.gov/documents/2019/01/07/2018-28282/2019-revised-patent-subject-matter-eligibility-guidance (visited on 03/11/2019).

[20] U.S. Patent and Trademark Office. "Changes to the Claim Construction Standard for Interpreting Claims in Trial Proceedings Before the Patent Trial and Appeal Board". In: *Federal Register* (Oct. 2018). URL: https://www.federalregister.gov/documents/2018/10/11/2018-22006/changes-to-the-claim-construction-standard-for-interpreting-claims-in-trial-proceedings-before-the (visited on 03/11/2019).