The Center for Intellectual Property (CIPU)

June 2022

To Patent or Not to Patent? Understanding the Open-Source Option for Businesses

Key Takeaways from This Report:

Whether or not a company considers itself to be a software business, the Fourth Industrial Revolution will turn businesses that conduct R&D into software developers.

1. Businesses must decide early whether or not they want to use IP rights for their software developments.
2. The choice whether to pursue patents or open source should be based upon a company's products and business model.
3. Software patenting and open-source business models can co-exist successfully.
4. Software patent policy is informed by misguided scholarly research on "software patents," a term that has no clear definition, legal or otherwise.
5. Firms conducting R&D that choose to patent software need to advocate for better eligibility in order to protect their interests.

Introduction

Software is the preferred embodiment for almost every improvement to computing technology today, which is a major issue given the uncertain nature of software patentability in the United States. In the way of illustration, we submit the following hypothetical:

Consider a fictional company that makes toothbrushes. This company sells one of the oldest tools known to human beings, and one that has undergone remarkably few innovative changes over the course of thousands of years. Even if this business innovates some unique curvature or new bristle, no one would consider that company to be a software company.

But what if this toothbrush company decided to sell toothbrushes with electronic components? Now your toothbrush has a motor, gears for different brushing speeds, a rechargeable battery and even a circuit board to control the separate hardware components. This company is still a toothbrush company. In some ways, it's also an electronics company, even if this business is largely assembling toothbrushes from hardware purchased wholesale from components manufacturers.

Now consider that same toothbrush company in 2022, during the advent of the Internet of Things (IoT). Now this business develops software upgrades to collect data from sensors and antenna components, using that data to make the toothbrush smarter. What if this company develops an entirely new algorithm or program that, in an entirely novel and non-obvious way, analyzes user data to personalize the brushing sequence, helping its customers achieve cleaner teeth and healthier gums? This is still a toothbrush company. But, to some extent, this is also a software company, and business survival in large part depends on the company's ability to not only commercialize that software product but also to protect against well-resourced infringers copying that invention.

This hypothetical is meant to drive home the main point of this Center for Intellectual Property Understanding (CIPU) report on software patents: most of today's innovative firms are likely creating inventions that can be embodied in software because those firms are operating during the midst of the Fourth Industrial Revolution, where software is the dominant form of embodiment for today's inventions. Businesses that have any intention to protect inventions that could be embodied in software should become stronger advocates of software patent rights. While large patent holders should be good stewards of patent rights for all, it's small startups who need these rights the most in order to scale up their commercialization activities.

Regardless of which industry the startup considers itself to be a member, software will almost certainly be crucial to those companies who are successful at separating themselves from competitors because software is the main way in which innovations are embodied at this current stage of the computer age.

Despite *Alice*, Anywhere From 25% to 63% of U.S. Patent Issuances in 2021 are Related to Software

Even by conservative estimates, more than one-quarter of all patents being issued by the U.S. Patent and Trademark Office cover an invention that can be embodied in software. The highest estimates say that nearly two-thirds of all U.S. patents can be embodied in software. The U.S. Patent and Trademark Office grants nearly 300,000 patents each year, so according to whichever dataset is trusted, anywhere from 75,000 U.S. patents to 200,000 U.S. patents are granted each year covering software innovations. Deriving a conclusive number of U.S. patents covering software inventions has been difficult mainly because of a definitional problem inherent in the term "software patent," a problem many academics note even as they try to perform more empirical research on software patents to inform policymakers.

The first thing that any business creating a software invention needs to know is that the patentability of software inventions is too uncertain for most startups. This is largely due to the U.S. Supreme Court's 2014 decision in *Alice Corp. v. CLS Bank International*, a decision that threatens subject matter eligibility for software inventions under the judicially-created exception to Section 101 patent eligibility for abstract ideas. This has been a boon for large tech implementers in sectors like automotive and financial institutions, who can implement software innovations with very little fear of large damages awards for patent infringement. However, this uncertainty can be fatal for small businesses who have a software innovation: the expense of obtaining patent rights and enforcing them against infringers, only to have a court declare the patent invalid as a matter of law, is beyond insurmountable for a startup that has no other way to protect its competitive advantage against larger players in the field.

Of course, patenting models are not the only means by which startups dealing in innovative software solutions can achieve business success. There are many companies, typically "pure"

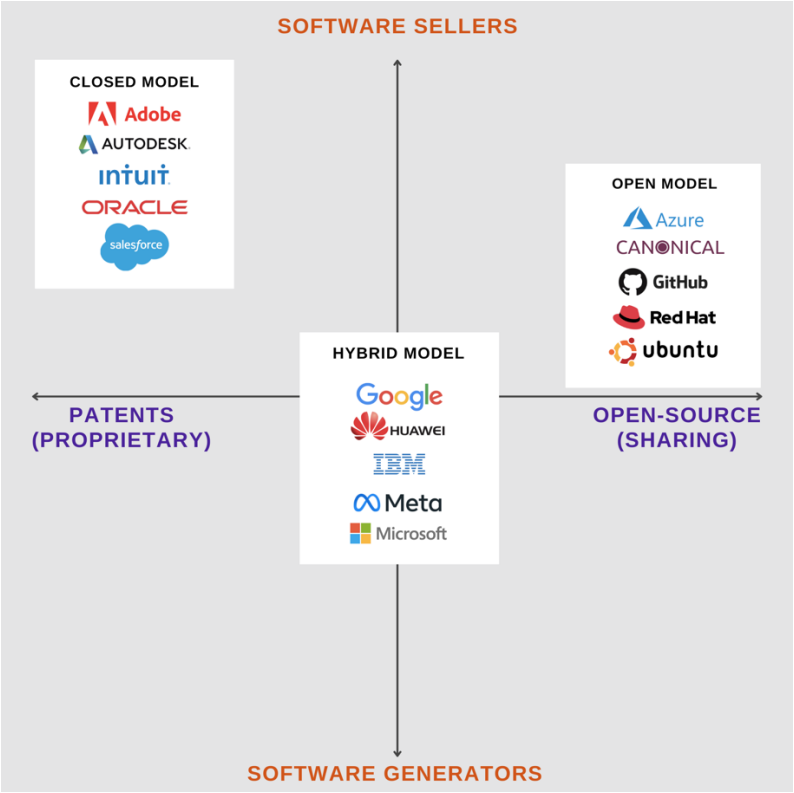
software companies not focused on any hardware product, that enjoy business success even as startups by employing an open-source model to the distribution of their software. The advent of cloud computing platforms has enabled Software-as-a-Service (SaaS) business models that are incredibly profitable by using open-source to increase the size of their customer base. Take for example the Microsoft Office suite of software programs. In the 1990s and through the 2000s, personal computer owners would install those programs via CD-ROM and although the cost of those programs were often bundled into the sale of new computers, those programs were still very expensive.

Today, CD-ROMs are almost as dated as floppy disk drives, and access to Microsoft Office is available on a subscription basis via Internet channels. In a similar way, there are companies that profit from contributing to the Linux open-source computer operating system and employing a subscription-based model with their customers. For these companies, it would make little sense to obtain patent rights that other Linux contributors may view with concern, dissuading many potential collaborators from trying to make improvements to the underlying code of an application or platform.

Distribution Models Should Inform Business Choice to Pursue Patents or Open-Source

Software innovation is incremental, and if the developer of an application for Linux insists that Linux users license a patent on that application, that application won't be included in Linux distributions. When that incremental advance is created for a traditionally non-software industry, whether automobiles or toothbrushes, there's a much better chance that the developing company will see their innovation as one they should protect from competitors. Within the software industry, many firms do obtain patents on inventions that can be embodied in software, and even if it's done for defensive purposes as an asset for counterclaims against patent infringement plaintiffs, the act of obtaining those patents tacitly acknowledges a value to those assets for those businesses. But the choice as to whether a particular business creating software embodiments of inventions should be open-source or should instead pursue patent rights should be premised upon a sober understanding of the company's distribution model.

DIAGRAM 1: Business Use of Software Patents



Source: The Center for IP Understanding

A company's choice to pursue either patent or copyright protections for their code, or to make their code available to others on an open-source basis, is of critical importance to the success of that company. It's a choice that should be highly informed by the model of distribution which a particular business intends on employ, and not based upon personal beliefs whether software should be patentable or not. For example, if the developer of an enterprise-level management software suite wants to distribute their software as a product that is implemented in-house by business customers, then a patenting or copyright model for protecting those software assets would be a better choice for those firms. If, however, that same business intends to employ its own data engineers who build new applications and perform data analytics on behalf of business customers, then an open-source subscription-based model makes much more sense than filing for patents. It should be noted that open-source companies, like patenting firms, believe they have some proprietary right to the software systems they develop, but those rights are enforced via contractual agreements governing consumer access to those systems among subscribers instead of through patent rights.

Hearkening back to the hypothetical toothbrush example, it's clear that a company in such a position, which is delivering a software innovation embedded within a hardware product sold to consumers, should pursue patents for the software powering the personalized brushing

algorithm employed by the toothbrush. Unfortunately, the state of U.S. patent rights for any invention involving software is uncertain at best. Since *Alice*, an incredible number of patents that cover an invention embodied in software have been invalidated by courts as directed toward an unpatentable abstract idea. A court applying *Alice*/*Mayo* is very likely to look at the personalized brushing algorithm employed in our hypothetical toothbrush and determine as a matter of law that the patent is directed at nothing more than the abstract idea of brushing your own teeth more effectively.

Because software will continue to be the mode by which the most valuable innovations are delivered in the IoT world, especially as 5G networks speeds up device connectivity by an order of magnitude over 4G networks, this abstract idea problem is very concrete indeed for the current generation of startups across the globe.

Survey of Academic Research Shows Foundational Problems with Empirical Research on "Software Patents"

Unfortunately, the policy debate surrounding the patentability of software has been greatly misinformed by a large amount of academic research that purports to be empirical but suffers from foundational issues in how software patents are defined. For this CIPU report, a total of 50 articles, mainly from law review and scientific journals that were highly relevant to the search term "software patent." One of the most salient findings from this research is that a definitional issue inherent to the term "software patent" creates doubt in the empirical nature of the research being conducted. Of the 50 articles reviewed, 12 purported to give empirical research on some aspect of software patents (how many have been issued by the USPTO, how many are asserted in U.S. district courts, etc.).

Some studies focus on a selection of U.S. patent classification, especially classes identified in a 2013 U.S. Government Accountability Office report on patent infringement litigation. The majority of these studies acknowledged that resulting numbers of software patents can be either under- or over-inclusive, meaning they don't reflect the precise universe of U.S. software patents. A few of the articles surveyed considered any patent invalidated in U.S. courts under the *Alice*/*Mayo* two-step patent eligibility test to be a software patent, despite the fact that this framework has invalidated patents that clearly don't cover software, such as is the case in *American Axle v. Neapco*. However, many of these articles were notes or comments offering policy recommendations based on legal theories instead of purporting to offer new empirical evidence on software patents.

Empirical research on software patents can only truly be useful if there is a way to define that term such that an accurate and complete universe of U.S. software patents can be determined.

That is one of the main points of a 2006 field research guide on the definitions employed for the term "software patent" authored by Dr. Anne Layne-Farrar, currently a Vice President in the Antitrust & Competition Economics Practice of the global consulting firm Charles River Associates. "The policy debate over software patents has been quite strident in recent years," Layne-Farrar noted more than 15 years ago, and that debate has grown even less harmonious since that time. Her work identified that, although there was no perfect definition for software patents that could create an authoritative dataset comprising all U.S. software patents, surveying U.S. patent classes covering computer-related inventions and filtering those results based on judiciously selected keywords would create the most highly accurate datasets for software patents. None of the research articles surveyed by the Center for IP Understanding for this report went beyond a survey of U.S. patent classes in determining the universe of U.S. software patents.

Firms that File Patents Should Advocate for Greater Clarity on Section 101

Part of the reason why it's problematic to define a patent as a software patent is that patents for computer-related inventions are capable of being embodied as either hardware or software. Software is the virtualization of computer hardware environments, and while software is a preferred embodiment for many computer inventions due to the existence of powerful microprocessor platforms for running software, it is somewhat prejudicial to look at a patent covering a computer-related invention and assume that software is the only possible embodiment for that invention. Such an assumption could change over time if certain technological fields, such as additive manufacturing or 3D printing, advance to the point that distributions of hardware embodiments of computer-related inventions become feasible. As a practical matter for businesses, there are a couple of takeaways. One is, if you plan on patenting to protect your research and development, there is a good chance that your patent will be considered to be software-related.

With the most conservative estimates showing that 25 percent of U.S. patents are software-related, companies have at least a one-in-four chance of filing a patent application that appears to be a software patent. While this percentage must vary from industry to industry, the nature of software bringing productivity gains to almost every industry means that industries traditionally seen as reliant on protections for physically-engineered innovations, such as automotive and manufacturing, are increasingly reliant on software innovation for products like self-driving cars and services like plant production management. This reliance on software innovations among non-traditional software companies will only increase as the Internet of Things (IoT) and 5G mobile networks enable data communications with a host of everyday items.

Businesses need to be aware of this slippery slope towards software because these mischaracterizations are also causing misunderstandings regarding the business models employed by these tech developers. Fourteen articles surveyed by CIPU spoke about software patents as being closely associated with nefarious actors in the tech marketplace, often called "patent assertion entities" or "non-practicing entities."

In 22 articles surveyed, researchers used the pejorative term "patent troll," which immediately connotes some form of malicious intent in monetizing patent assets against infringers. Further, these terms suffer from a similar problem to the "software patent" definition as they are easily thrown about to describe certain actors but are vague and difficult to define.

Conclusion

Innovation during the Fourth Industrial Revolution is increasingly embodied as software, even as much software is unpatentable as a matter of law under Supreme Court cases like *Alice*. This conundrum creates a challenging situation for startups who need certainty in patent rights to operate effectively and attract investment. At the same time, open-source models can be profitable for startups. The choice whether to pursue either patents for software innovations or an open-source model should be based on a firm's products and distribution model: if software is packaged and distributed to end users, a firm may want to choose a patenting model, whereas if a startup plans to offer Software-as-a-Service, an open-source model may be more appropriate. Unfortunately, academic literature surrounding "software patents" is likely creating more confusion than clarity due to a definitional problem inherent to the term "software patent." Therefore, patenting firms of all sizes should recognize the importance of software to the current generation of patent rights and advocate for more certainty for software patentability.

Perspectives: Insights on Patenting Software vs. Open-Source from Industry Experts

In conjunction with this report, CIPU distributed a series of questions to various insiders from the software industry as well as patent law, each of whom were interviewed to develop background for the preceding report. As a follow-up to those interviews, we distributed the following questions to those insiders:

1. Can software patents and open-source platforms co-exist in the business world?
2. What should a startup consider before choosing to either patent or open-source their software innovations?

3. Why have software patent filings been increasing despite Section 101 issues with software patentability?

Answers provided in response to these questions are included below.

Ray Millien, CEO, Harness IP; Former Chief IP Counsel, Volvo Cars Group; Former Associate Counsel, General Electric

1. Yes, they can co-exist. First, a going concern may choose an open-source business model to distribute its product(s), or a proprietary model. The two models do not have to be mutually exclusive. That is, some products in their ecosystem may be distributed via an OSS license, while others may be distributed under a proprietary licensing model. In both instances, software patents may protect the underlying code for those who are unlicensed or choose to breach the license under which they received the software product.
2. It is really a matter of what business (i.e., revenue) model they choose to implement. In simple terms, they can choose to distribute their software product for "free" under an OSS model and just charge for implementation and/or customization services. Alternatively, they can choose to distribute their software product under a proprietary licensing model and charge for the license and any associated services. In either case, filing for software patents is still possible. In the former case it is a "belts and suspenders" approach to guard against those who choose to breach the OSS license under which they received the software product.
3. Software patent filings are increasing because software is still eating the world. No matter the difficulties of obtaining a software patent, more and more technologies involve more and more software and thus the attempt is still a sound business and legal decision.

Dr. Anne Layne-Farrar, Senior Consultant, Charles River Associates; Adjunct Professor, Northwestern University Pritzker School of Law

1. I think the last twenty years has proven that they can co-exist. They serve different needs and are thus often complimentary to one another, even when they are substitutes in other situations. Look at the example of IBM, which employs both models side by side.
2. What are their business goals? If my business creates an ecosystem for others (e.g., Google) or offers a plug-in technology that can increase the sales of related products (e.g., Bluetooth used in lots of different hardware), then open-source may maximize my commercial success. If my technology creates a competitive advantage for my business, however, a patent is likely a better option to protect that advantage.
3. Software patent filings are increasing because software is where innovation is now. Cars aren't that different now versus decades ago but for the software innovations in them. Thermostats are software platforms, as are most household appliances, and the list goes

on. It's anachronistic to think of software as just programs for computers, as it was back in the 1990s.

Ron Katznelson, Ph.D., President, Bi-Level Technologies; Chairman of the Intellectual Property Committee. IEEE-USA

1. This is an indefinite question that cannot be answered because the terms "software patents" and "open-source" are undefined. First, software may be copyrighted but not patented; there are, however, patents for computer-implemented inventions. Second, it is unclear what is "open" in "open-source." Some "open-source" licenses convey only a royalty-free copyright without any mention of patent rights. To the extent the question is "can patents for computer-implemented inventions co-exist in the business world with open-source that subject to royalty-free patent license," the answer is yes; this is so just as royalty-bearing patents can co-exist in the business world with royalty-free patents.
2. Do not select the royalty-free open-source option if your business model is centered on the patentable technology. If, however, you have a business model where the major commercial benefit to your company is the operation of a business and sales of products that are not the software itself and that such collateral business will be enhanced by the large number of users of the software, then you might benefit from keeping it open-source and letting as many users as possible implement it so that your core business can benefit from the larger number of users/customers.
3. Again, "software patent" is an undefined term. Software is a tool for implementing all kinds of inventions, that has now become more ubiquitous. What used to be implemented with discrete hardware is now implemented in software on a general purpose computer hardware. That is why patentable computer-implemented inventions are growing in number as the preferred implementation modality. Many of these filed as patent applications do not encounter the Section 101 barriers and therefore impediments due to Section 101 uncertainties do not slow such filings too much.

Kate Gaudry, Partner, Kilpatrick Townsend & Stockton LLP; Ph.D., Computational Neurobiology; J.D. Harvard Law

3. The eligibility analysis of software patent claims changed rather dramatically since the *Alice v. CLS Bank* decision in 2014. Some software claims that likely would have been found by examiners or judges to be patent eligible before the decision would likely be found to be patent ineligible after the decision. One potential strategy for reacting to the changed reality would be to forego patenting software inventions. Another potential strategy would be to adapt approaches for defining and claiming an invention. Indeed, various courts and the USPTO have provided frameworks and explanations that can be used to

help identify what type of claim may be allowable and valid for a given software technology.

Jonathan Stroud, Adjunct Professor of Law, American University Washington College of Law; General Counsel, Unified Patents

(Excerpted from *Debugging Software Patents after Alice*, South Carolina Law Review, Autumn 2017)

> "Software patent applications are among the most complex patents the USPTO has to examine. The above proposals do not directly confront the ambiguous two-prong test in the Alice decision—nor should that be the USPTO's job—but they may serve as tools for improving patent quality. Higher-quality patents will help the courts to pay deference to the USPTO's decisions, avoid post-grant review, and later bodies and courts will be more likely to uphold the validity of patent claims. Furthermore, fewer patents of higher quality would decrease patent litigation lawsuits, because litigants would be deterred from fighting over validity and patentability. These suggestions will not end the patent troll problem. These entities will continue to collect patents to sue and license others for patent infringement. But the problem does not lie in the fact that patent trolls sue or threat to sue others to collect licensing fees; rather, it is the ready availability of both issued and purchased broad and ambiguous software patents which can be asserted for more exclusive rights than the actual invention covers. If the USPTO could issue higher-quality software patents that the PTAB and courts would likely uphold as valid, more companies will likely turn to licensing those higher-quality patents rather than taking their chances in courts—conversely, making the jobs of NPEs with high-quality patents much easier and settling rights across the board."

Bibliography

Acharya, Athul K. "Abstraction in Software Patents (and How to Fix It)." John Marshall Review of Intellectual Property Law, vol. 18, no. 4, Summer 2019, p. vi-381. HeinOnline.

Anderson, Michael H., and Daniel Cislo. "Venue Selection and the Rise of Super Patent Courts in the Post-AIA Era." Business & Bankruptcy Law Journal, vol. 3, no. 2, Spring 2016, p. 279-294. HeinOnline.

Anderson, Shane D. "Software, Abstractness, and Soft Physicality Requirements." Harvard Journal of Law & Technology, vol. 29, no. 2, Spring 2016, p. 567-594. HeinOnline.

Armstrong, Timothy K. "Symbols, Systems, and Software as Intellectual Property: Time for Contu, Part II." Michigan Telecommunications and Technology Law Review, vol. 24, no. 2, Spring 2018, p. 131-180. HeinOnline.

Asay, Clark D. "Patent Pacifism." George Washington Law Review, vol. 85, no. 3, May 2017, p. 645-711. HeinOnline.

Bervik, Trevor. "Roots to Bits: How the History of Plant Patents Can Shape Software's Future." Colorado Technology Law Journal, vol. 17, no. 1, 2018, p. 187-212. HeinOnline.

Bradley, Joel. "Patent Law and Means-Plus-Function Claim Language: Where It Was, Where It Is (Post Williamson v. Citrix), and Where It Should Go in the Future." Georgia Law Review, vol. 52, no. 3, Spring 2018, p. 897-912. HeinOnline

Chiang, Jennifer. "A Nonobvious Approach to Functional Claiming in Software Patents." American University Intellectual Property Brief, vol. 8, no. 1, November 2016, p. 1-62. HeinOnline.

Chien, Colleen V. "Software Patents as a Currency, Not Tax, on Innovation." Berkeley Technology Law Journal, vol. 31, no. 3, 2016, p. 1669-1724. HeinOnline.

Craig, Joseph Allen. "Deconstructing Wonderland: Making Sense of Software Patents in a Post-Alice World." Berkeley Technology Law Journal, vol. 32, no. Annual Review Issue, 2017, p. 359-378. HeinOnline.

Daily, James E. "Alice's Aftermath: Changes in Patentee Behavior since Alice v. CLS Bank." Boston University Journal of Science and Technology Law, vol. 23, no. 2, Summer 2017, p. 284-303. HeinOnline.

Dobias, Nicole L. "Protecting Software Intellectual Property after the Lexmark Decision." Indiana Law Review, vol. 52, no. 2, 2019, p. 305-328. HeinOnline

Estall, Peter. "Venerunt, Viderunt, Vicerunt Venue: How TC Heartland and In re Cray Have Conquered Patent Value for Corporate Defendants and How Congress Can Balance the Scales of Patent Value Justice." Minnesota Law Review, vol. 103, no. 3, February 2019, p. 1523-1566. HeinOnline.

Franklyn, David, and Adam Kuhn. "The Problem of Mop Heads in the Era of Apps: Toward More Rigorous Standards of Value Apportionment in Contemporary Patent Law." Journal of the Patent and Trademark Office Society, vol. 98, no. 2, 2016, p. 182-222. HeinOnline.

Ford, Laura R. "Patenting the Social: Alice, Abstraction, & Functionalism in Software Patent Claims." Cardozo Public Law, Policy and Ethics Journal, vol. 14, no. 2, Spring 2016, p. 259-342. HeinOnline.

Fusco, Stefania. "TRIPS Non-Discrimination Principle: Are Alice and Bilski Really the End of NPEs." Texas Intellectual Property Law Journal, vol. 24, no. 2, 2016, p. 131-160. HeinOnline.

Gabison, Garry A. "Spotting Software Innovation in a Patent Assertion Entity World." Hastings Science and Technology Law Journal, vol. 8, no. 1, Winter 2016, p. 97-[ix]. HeinOnline.

Gan, Magnus. "Before Mayo & after Alice: The Changing Concept of Abstract Ideas." Michigan Telecommunications and Technology Law Review, vol. 22, no. 2, Spring 2016, p. 287-316. HeinOnline.

Garza, Robert Daniel. "Software Patents and Pretrial Dismissal Based on Ineligibility." Richmond Journal of Law & Technology, vol. 24, no. 2, 2018, p. 1-80. HeinOnline.

Gershoni, Michael. "An Argument against Reinventing the Wheel: Using an Obviousness Analysis to Bring Consistency and Clarity to Patent Eligibility Determinations of Software Patents after Alice Corp." AIPLA Quarterly Journal, vol. 44, no. 2, Spring 2016, p. 295-326. HeinOnline.

Greenberg, Anastasia. "Protecting Virtual Things: Patentability of Artificial Intelligence Technology for the Internet of Things." IDEA: The Law Review of the Franklin Pierce Center for Intellectual Property, vol. 60, no. 2, 2020, p. 328-351. HeinOnline.

Hecker, Peter. "How an Old Non-Statutory Doctrine Got Worked into the Sec. 101 Test for Patent Eligibility." Journal of the Patent and Trademark Office Society, vol. 99, no. 1, 2017, p. 4-19. HeinOnline.

Heedy, David B. "Has Alice Brought Us to Patent Wonderland?: Can the Supreme Court's New Analysis of Abstract Ideas Affect the Current Problems Associated with Business-Method and Software Patents." Florida State University Business Review, 15, 2016, p. 57-80. HeinOnline.

Hylton, Keith N. "Patent Uncertainty: Toward a Framework with Applications." Boston University Law Review, vol. 96, no. 3, May 2016, p. 1117-1148. HeinOnline.

Lin, Yu-Kai and Rai, Arun, Patent Protection and Software Innovation: Evidence from Alice (September 9, 2020). Available at SSRN: https://ssrn.com/abstract=3703055 or http://dx.doi.org/10.2139/ssrn.3703055

Liu, Michael Xun. "Subject Matter Eligibility and Functional Claiming in Software Patents." North Carolina Journal of Law & Technology, vol. 20, no. 2, December 2018, p. 227-285. HeinOnline.

Massarotto, Giovanna. "Open-Source Paradigm: Beyond the Solution to the Software Patentability Debate." John Marshall Review of Intellectual Property Law, vol. 15, no. 4, 2016, p. [i]-[i]. HeinOnline.

Mennell, Peter. "Patent Showdown at the N.D. C[orr]al." Chicago-Kent Journal of Intellectual Property, vol. 18, no. 3, 2019, p. 101-147.

Miller, Shawn P. "Who's Suing Us: Decoding Patent Plaintiffs since 2000 with the Stanford NPE Litigation Dataset." Stanford Technology Law Review, vol. 21, no. 2, Spring 2018, p. 235-275. HeinOnline.

Millien, Raymond. "Alice Who? Over Half the U.S. Utility Patents Issued Annually are Software Related!" IPWatchdog, May 21, 2017. https://www.ipwatchdog.com/2017/05/21/alice-over-half-u-s-utility-patents-issued-annually-software/id=83367/

Millien, Raymond and Yi Chen. "In First Half of 2021, 63% of U.S. Patents, 48.9% at EPO and 40.1% in China Were Software-Related." IPWatchdog, Aug. 6, 2021. https://www.ipwatchdog.com/2021/08/26/first-half-2021-63-u-s-patents-48-9-epo-40-1-china-software-related/id=137100/

Missirian, David E. "Artificial Intelligence: Is It a Good under the UCC or a Service under the Common Law or Is There a Need for a New Category?." Corporate and Business Law Journal, vol. 1, no. 2, June 2020, p. 44-54. HeinOnline.

Moldovanyi, Matthew. "Alice: The Status Quo or Total Chaos." Case Western Reserve Journal of Law, Technology and the Internet, vol. 7, no. 1, 2016, p. 121-152. HeinOnline.

Osenga, Kristen. "Patent-Eligible Subject Matter.. Still Wielding the Wrong Weapon - 12 Years Later." IDEA: The Law Review of the Franklin Pierce Center for Intellectual Property, vol. 60, no. 1, 2020, p. 104-133. HeinOnline.

Prange, Kurt. "Blockchain & Business Methods: How Business Method Patents May Be Redeemed by Furthering Blockchain Innovation." Colorado Technology Law Journal, vol. 18, no. 1, 2020, p. 185-226. HeinOnline.

Raffiee, Joseph and Teodoridis, Florenta, Does the Political Ideology of Patent Examiners Matter? An Empirical Investigation (June 4, 2020). Available at SSRN: https://ssrn.com/abstract=3619474 or http://dx.doi.org/10.2139/ssrn.3619474

de Rassenfosse, Gaétan and Palangkaraya, Alfons and Palangkaraya, Alfons, Do patent pledges accelerate innovation? (July 30, 2021). Available at SSRN: https://ssrn.com/abstract=3897360 or http://dx.doi.org/10.2139/ssrn.3897360

Reinecke, Jason D. "Is the Supreme Court's Patentable Subject Matter Test Overly Ambiguous: An Empirical Test." Utah Law Review, vol. 2019, no. 3, 2019, p. 581-606. HeinOnline.

Ro, Myungjin, and Neil Davey. "Software in the Wake of Alice: Intended Target or Inadvertent Casualty." Managing Intellectual Property, 267, 2017, p. 88-93. HeinOnline.

Sala, Vinicius. "Can an Improved Disclosure Mechanism Moderate Algorithm-Based Software Patentability in the Public Interest?." Cybaris: An Intellectual Property Law Review, vol. 11, no. 1, 2020, p. 1-33. HeinOnline.

Schreiber, Andrew. "Go (En)Fish: Drawing CAD Files from the Patent Eligibility Pool." IDEA: The Journal of the Franklin Pierce Center for Intellectual Property, vol. 58, no. 1, 2017, p. 1-64. HeinOnline.

Smith, Autumn. "From IP Goals to 3D Holes: Does Intellectual Property Law Provide a Map or Gap in the Era of 3D Printing." Journal of Intellectual Property Law, vol. 25, no. 1, Fall 2017, p. 85-108. HeinOnline.

"Software Patent." Court Uncourt, vol. 8, no. 5, 2021, p. 34-37. HeinOnline.

Stroud, Jonathan, and Derek M. Kim. "Debugging Software Patents after Alice." South Carolina Law Review, vol. 69, no. 1, Autumn 2017, p. 177-220. HeinOnline.

Taylor, Daniel. "Down the Rabbit Hole: Who Will Stand up for Software Patents after Alice." Maine Law Review, vol. 68, no. 1, 2016, p. 217-262. HeinOnline.

Teska, Kirk. "(The Unfortunate) Future of Software Patents under 35 USC Sec. 101 and Sec. 112." Journal of High Technology Law, vol. 16, no. 2, 2016, p. 394-408. HeinOnline.

Train, Tyler, and John Burke. "Heads or Thales: Comparative Mathematical Analysis for the Head-Mounted Display Technology behind the Federal Circuit's Thales Decision." Albany Law Journal of Science & Technology, vol. 29, no. 1, 2019, p. 1-15. HeinOnline.

Tran, Jasper and Benevento, J. Sean, Alice at Five (2019). 2019 Patently-O Patent Law Journal 25, Available at SSRN: https://ssrn.com/abstract=3454922

Wentzel, Douglas B. "Uber & Alice: Could One Patent Really Take down This Ridesharing Giant." Journal of the Patent and Trademark Office Society, vol. 98, no. 4, 2016, p. 856-895. HeinOnli

---

This report was researched and prepared by Steven Brachmann, an intellectual property journalist and a graduate of the University of Buffalo School of Law.