WIPO
WORLD **INTELLECTUAL PROPERTY** ORGANIZATION

WIPO | MAGAZINE

Five years after Alice: five lessons learned from the treatment of software patents in litigation

August 2019

*By **Joseph Saltiel**, Marshall Gerstein & Borun LLP, Chicago, USA*

It has been five years since the Supreme Court's landmark decision in *Alice Corp.* v *CLS Bank International*. *Alice* established a two-part test to determine if a software patent was unpatentable under US Patent Law (35 USC Section 101) for claiming ineligible subject matter. Under this two-part test, a court must first consider whether the patent claims are directed to a patent ineligible concept such as an abstract idea, and if so, the court should consider whether the claim's other elements transform the claim into a patent eligible concept. Applying this two-part test, the *Alice* decision held that known ideas are abstract, and reciting the use of a conventional computer in the claims to implement the known idea does not make the claim patentable subject matter. *Alice has* greatly impacted the litigation of software patents. *Alice* also gave defendants a new and highly successful defense that could be asserted early in litigation. In turn, patentees have had to take this new defense into account in their litigation strategy, and companies have questioned the value of software patents. After five years and hundreds of court decisions applying *Alice*, litigation of software patents has changed dramatically. Below are five lessons learned from software patent litigation after *Alice*.

It has been five years since the Supreme Court’s landmark decision in Alice Corp. v CLS Bank International, which established a two-part test to determine the patentability of software patents under US Patent Law. Alice has had a great impact on the litigation of software patents (photo: iStock / Getty Images Plus / © monsitj).

Alice should be considered in every software patent litigation.

Before *Alice*, software patents were rarely challenged as unpatentable. After *Alice,* there were hundreds of patentability challenges per year targeting software patents. Most of these challenges were at least partially successful. The use of *Alice* became ubiquitous in software patent cases. Software patents were being challenged routinely and early in the litigation. Over half of the *Alice*-based challenges were made in early dispositive motions whereby the court decides the claim in favor of one or another party without need for further trial proceedings. Every patentee considering asserting a software patent therefore needs to consider the possibility of a patentability challenge based on *Alice.* Likewise, every defendant accused of infringing a software patent should consider an *Alice* motion.

Legal analysis of software patents under *Alice* differs from other legal analysis.

In litigation, parties must abide by the Federal Rules of Evidence. These rules provide for when evidence can be considered, what kind of evidence is proper, how evidence is introduced and how it should be considered. Words in legal instruments, and especially in patents, are vital. Attorneys spend countless hours debating the meaning of the words used in claims, and cases often turn on the most innocuous phrases. But for software patents facing an *Alice* inquiry, evidence and words are not as significant.

Under *Alice,* a court must first determine if the claim encompasses an abstract idea. Conventional methods of software are abstract. But because this first determination is a question of law, a defendant does not necessarily need to submit evidence that the claim is conventional (and therefore abstract). While the *Alice* decision cited publications to support its position that the concept was conventional, most courts applying *Alice* have not supported their findings with evidence. Attorney argument is sufficient. Moreover, the terminology used in the claim or the length and complexity of the claim do not matter for either part of the *Alice* test. *Alice* did not analyze the words of the claims, but instead characterized the claims as “the use of a third party to mitigate settlement risk” and found that concept to be conventional (i.e. abstract). Following *Alice,* most courts rely on a characterization of the claims instead of the words used in the claims for their analysis. Hence, an *Alice* decision may not be supported by evidence and may not depend on the entirety of the specific language of the claims.

***Alice* allows for quick resolution of litigation involving software patents of questionable validity.**

Software does not exist physically; it is represented by many 1s and 0s. Software can also represent the same functionality in unlimited ways. Software is inherently abstract, but because software is also patentable, abstractness under *Alice* means something else.

Generally, software source code is not publicly available and is difficult to reverse engineer. Software constantly changes, often with little record of the changes or the reasons for them, and software has no standard naming conventions. These attributes make it difficult to determine whether a software patent is valid. For example, it might be difficult to find prior art, make technical comparisons, or determine whether a disclosure is enabling. These are fact-intensive inquiries. To prevail on invalidity, a defendant will typically have to litigate up to or through trial even for highly questionable patents.

Alice makes it easier for a defendant to seek invalidity of a software patent that would otherwise be invalid as lacking novelty, obvious, or not enabled. In *Alice*, the software patent at issue recited a conventional methodology. But because the methodology was conventional, the court found it abstract. To be patentable, the claims needed another element that would transform the unpatentable subject matter into patentable subject matter. *Alice* held that using a conventional computer to perform the methodology did not make these claims patent eligible. That is, combining a conventional element with another conventional element does not make the claimed invention patentable. An obviousness analysis achieves the same result. By using an *Alice* analysis instead of an obviousness analysis, the court reaches a conclusion on invalidity but forgoes obviousness requirements, such as evidence that elements are conventional and the reasons for combining those elements.

Alice also explained that combining a conventional methodology with a conventional computer was an improper attempt to monopolize an abstract idea. In other words, if a claim is broad enough to encompass (or preempt) all embodiments of an idea, that is an indicator that the claim is abstract. Such a broad claim would also likely be invalid as not enabled because it is doubtful that a patent specification could provide adequate support to enable every possible variation of an idea. But rather than task a defendant with the more difficult chore of identifying embodiments and proving they are not enabled by the specification, *Alice* simplifies the analysis by allowing a defendant to argue that a claim is too broad, and thus, abstract and unpatentable.

Alice frames the issue by asking whether a claim is abstract. But abstractness under *Alice* is a means to eliminate software patents that are overtly obvious or too broad to be enabled. By using an *Alice* analysis instead of an anticipation, obviousness, or enablement analysis, *Alice* allows defendants to bypass many of the complexities associated with litigation discovery and proving invalidity, which in turn allows defendants to file early dispositive motions and thereby attempt to avoid further trial proceedings.

Alice decisions are not predictable.

While courts have consistently applied the two-part test set forth in *Alice,* the results of that application are unpredictable. One court may find a software patent unpatentable, but another court may find a similar software patent patentable. For many software patents, it is too difficult to make reliable predictions. As Paul Michel, former Chief Judge of the Federal Circuit, recently testified before Congress, the application of *Alice* has been "excessively incoherent, inconsistent and chaotic."

Alice has conflated patentability, obviousness and enablement. Patent law is complicated as it is, but with *Alice*, courts are forced to cobble together three distinct and complicated legal concepts and rely on generic characterizations of the claims without evidence or a developed record. It is a difficult task, which has caused unpredictability when courts apply *Alice* to software patents.

Both the United States Patent and Trademark Office (USPTO) and the Court of Appeals for the Federal Circuit have tried to provide consistency, but neither has been effective. Andrei Iancu, Director of the USPTO, recognized this problem and recently issued USPTO guidelines on applying *Alice* to "keep rejections in their own distinct lanes [*e.g.,* 101, 102, 103, and 112] and to stop commingling the categories of invention on one hand with the conditions for patentability on the other." While these guidelines are helpful, the USPTO is still limited by *Alice.* Moreover, courts are not bound by the USPTO guidelines, and in some instances, have chosen not to follow them.

Similarly, the Federal Circuit has tried to bring some consistency to *Alice.* For example, the Federal Circuit has held that the second part of the *Alice* analysis may require a factual inquiry. The effect of this holding is limited because it does not apply to the first step of an *Alice* inquiry. Furthermore, some courts have determined that no factual inquiry is necessary for their particular case nullifying the effectiveness of the holding. Regardless, the Federal Circuit is limited in what it can do because it also must confine itself to *Alice.* Indeed, one Federal Circuit Judge, in acknowledging the lack of clarity in evaluating patentability, advised practitioners that "[y]our only hope lies with the Supreme Court or Congress" to receive clarification. *See Athena Diagnostic, Inc.* v *Mayo Collaborative Services, LLC.* In that case, the Federal Circuit issued seven different opinions disputing how to apply US Supreme Court decisions on patentability. So, five years after *Alice*, if the Federal Circuit cannot agree on how to evaluate patentability, no one should expect to predict the outcome of a patentability challenge to software patents.

Going forward, more software patents should survive an *Alice* challenge.

In 2015, over 60 percent of the software patents challenged under *Alice* were found to have at least one claim unpatentable. Since 2015, however, the percentage of successful *Alice* challenges to software patents has dropped each year. Year-to-date in 2019, the percentage of successful or partially successful *Alice* challenges is less than 50 percent. The trend indicates that the number of successful *Alice* challenges will continue to drop. As noted above, the Federal Circuit has acknowledged, at least in some circumstances, that a factual inquiry may be necessary, making it harder to prevail on some early *Alice* motions, delaying a decision on *Alice*, and increasing the odds that the case can be resolved on other grounds. Also, some plaintiffs are no longer seeking to obtain and/or assert questionable software patents (or are seeking such a low settlement that an *Alice* challenge is not economically viable). In addition, as a result of *Alice*, patentees are drafting better claims, and the USPTO has done a better job of scrutinizing claims for patentability. Recently issued software patents are therefore more likely to survive an *Alice* challenge in litigation. Moreover, courts may defer to the USPTO's determination on *Alice* if the issue of patent eligibility was considered during prosecution. It seems likely, therefore, that the rate of successful challenges of software patents under *Alice* will continue to drop.

There is no doubt that *Alice* has disrupted and will continue to disrupt software patent litigation. While the US Supreme Court is unlikely to overrule its unanimous *Alice* opinion, Congress has been actively considering legislation to overrule *Alice*. If passed, such legislation will significantly impact litigation of software patents and likely reverse many of the trends noted above. Until a new law is passed and used in litigation cases, it will be difficult to gauge the impact of that new legislation.

Joseph Saltiel is special counsel at Marshall, Gerstein & Borun LLP. He is an IP litigator with a long track record of successfully representing clients in IP matters in courts across the country, the USPTO and the International Trade Commission*. Mr. Saltiel also regularly counsels clients on IP issues such as licensing, opinions of counsel, NDAs, due diligence and related IP matters. He can be reached at* jsaltiel@marshallip.com*.*