Debugging Software Patents: Increasing Innovation and Reducing Uncertainty in the Judicial Reform of Software Patent Law

Robert E. Thomas

Follow this and additional works at: http://digitalcommons.law.scu.edu/chtlj

Part of the Law Commons

Recommended Citation
Available at: http://digitalcommons.law.scu.edu/chtlj/vol25/iss1/7
DEBUGGING SOFTWARE PATENTS: INCREASING INNOVATION AND REDUCING UNCERTAINTY IN THE JUDICIAL REFORM OF SOFTWARE PATENT LAW

Robert E. Thomas†

Abstract

Software patents do not promote innovation, they instead reduce it. Early in the development of the computer industry, a U.S. presidential commission opposed the recognition of software patents, and the Supreme Court never approved the degree of recognition that software patents have recently enjoyed. This paper reviews the genesis of the patentability of software patents, analyzes the social welfare implications of recognizing software as patentable subject matter, and examines recommendations for reform. Others have suggested enhancing software patent claim disclosure requirements to address social welfare losses. This paper concludes that altering disclosure requirements would dramatically increase litigation while reducing the value of most software patents. Instead, the only way to optimally reform software patent law, without destroying settled expectations of current software patent holders, is for the Supreme Court to reinstate and prospectively apply its physical-transformation test.

† Associate Professor, Huber Hurst Fellow, Warrington College of Business Administration, University of Florida. I would like to express my appreciation to Robert Bird, Sandra Miller, Phil Nichols, David Orozco, O. Lee Reed, Adam Sulkowski, and participants in the Twelfth Annual Huber Hurst Research Seminar in Legal Studies held at the University of Florida. I also want to acknowledge the excellent editorial support I received from Tonya Bolden who pushed me to use plain English. Any remaining errors are my responsibility.
I. INTRODUCTION

The issue of patentability and patent infringement are among the most contentious areas of intellectual property law.\(^1\) Whereas attempts to reform patent law in Congress face a protracted and questionable fate, recent decisions indicate that the courts may precede Congress on this front. For example, in *eBay, Inc. v. MercExchange, L.L.C.*,\(^2\) Justice Kennedy questioned both the strategic use of patents and the validity of some business method patents as reasons for making injunctive relief more difficult to obtain in patent infringement cases.\(^3\)

Although Justice Kennedy’s concerns are legitimate, the genesis of the problem is more fundamental. Historically, policy makers have recognized that excessive use of patent exclusivity can have a deleterious effect on commercial activities. Therefore, Congress and the courts had traditionally been reticent to expand patent subject matter.\(^4\) However, during the last few decades this cautious approach has subsided, leading to the wholesale recognition of controversial areas of innovation that provide society with little or no benefit.\(^5\)

This paper argues that the expansive recognition of certain types of innovation—namely computer software and business methods—

---


3. *Id.* at 397. (Kennedy, J., concurring).


impedes innovation, and that the courts are in the best position to address this problem. Although, the U.S. Court of Appeals for the Federal Circuit (CAFC) has taken an initial step to resolve these concerns in its recent In re Bilski decision, it may take Supreme Court action to fully address the problems that multiple CAFC decisions have created by recognizing software patents. Software innovation requires little investment, so patents are not needed to promote this type of innovation. Instead, patent-generated software monopolies dramatically increase competitors’ research and development (R&D) costs, resulting in an overall decline in innovation.

The Supreme Court has the ability to dramatically refocus software patent policy in a socially beneficial direction by reinvigorating its existing precedents. These precedents require process claims containing software to transform or reduce "an article to a different state or thing." A strict interpretation and application of this test (the physical-transformation test) would render most pure software and business method patent claims unpatentable. A shift in patent policy away from the CAFC's overly inclusive interpretation of patent subject matter will both promote innovation and reduce software patent infringement litigation. Moreover, if existing patents are grandfathered under this policy shift, the Court can minimize damages to settled expectations and existing patent holders.

The analysis begins with a review of the judicial treatment of software patent law. With its 1981 Diamond v. Diehr decision, the Supreme Court ended its efforts to provide guidelines for patent
coverage of software innovations. The CAFC, revealing a pro-
patent-expansion bias, chipped away at this restraint, eventually
removing all limitations to patent coverage of software and business
methods. To promote innovation and reduce litigation, courts should
adopt policies that not only advance social welfare but also do not
increase legal uncertainty. Using this theoretical framework, I provide
reform recommendations that balance judicial and theoretical
considerations and explain why some previously advanced
recommendations are likely to produce more problems than they
solve.

II. THE SUPREME COURT AND THE DETERMINATION OF STATUTORY
SUBJECT MATTER

A. The Development of Statutory-Subject-Matter Jurisprudence

Although patent policy is within Congress’ domain, the courts
have long played a pivotal role in determining patent policy. The
constitutional authorization for patents (the “Patent Clause”) is
parsimonious. While providing Congress with a single tool–periods
of exclusivity–to promote innovation, the Patent Clause fails to
provide detailed guidance about the limits of innovative activity
entitled to constitutional protection. The Constitution left it to
Congress to create, change, and shape patent policy to meet the
nation’s evolving needs.

Congress, in turn, left to the courts much of the job of
determining subject-matter boundaries. The Patent Act of 1790
identifies statutory subject matter as including, “any useful art,
manufacture, engine, machine, or device, or any improvement
therein.” This definition was retained in 35 U.S.C. § 101, with
minor changes. The courts recognized the need to limit patent
breadth by excluding abstract principles and laws of nature from
statutory subject matter. They also excluded outcomes and results

11. Id.
12. AT&T Corp., 172 F.3d at 1357-58.
14. Id.
15. Robert I. Coulter, The Field of the Statutory Useful Arts (Part II), 34 J. PAT. OFF.
SOC’Y 487, 500 (1952).
process, machine, manufacture, or composition of matter, or any new and useful improvement
thereof”).
from patent protection. The courts reasoned that allowing patent coverage in these areas could create perpetual monopolies over "devices or mechanisms that the patentee might not even be familiar with or understand."

Patent coverage of abstract principles or laws of nature would give individuals control over subject matter they did not create and could prevent others from achieving results that depend on such principles and laws. Similarly, patent coverage of outcomes and results would prevent others from using alternative and innovative methods for achieving comparable or identical outcomes and results. Thus, the courts created the "means plus-function" litmus test by limiting patent subject matter to, "the means (processes or machines) embodying such principles that are used to achieve a result that is useful for human beings."

This reasoning, while relatively straightforward when applied to machines, is problematic when applied to processes. Compared to machines, it is far more difficult to draw a clear line between actions that constitute processes and the actions' results. To counter these problems, courts developed a bar against processes that solely or primarily rely on "mental steps." Courts further distinguished processes from mental steps by recognizing only those processes that produce tangible outcomes. This physical transformation requirement (physical-transformation test) lasted over one hundred years and the Supreme Court has never rejected it.

The physical-transformation test provided a clear delineation between patentable and non-patentable subject matter. It provided, perhaps, as objective a test as exists in patent law. However, there is no compelling statutory basis for said requirement. Prior to the

---

18. Corning v. Burden, 56 U.S. 252, 268 (1854) ("It is for the discovery or invention of some practicable method or means of producing a beneficial result or effect, that a patent is granted, and not for the result or effect itself").
19. See Ford, supra note 4, at 60.
20. Id. at 59.
21. See, e.g., Tilghman v. Proctor, 102 U.S. 707, 728 (1880) (distinguishing between machines and processes by stating: "A machine is a thing. A process is an act, or a mode of acting. The one is visible to the eye, [sic] an object of perpetual observation. The other is a conception of the mind, seen only by its effects when being executed or performed").
22. See Ford, supra note 4, at 61-62 n.66.
23. Cochrane v. Deener, 94 U.S. 780, 788 n.61 (1877) (defining a process to be "an act, or a series of acts, performed upon the subject-matter to be transformed and reduced to a different state or thing").
twentieth century, it was inconceivable that an inventive process would produce a result that was not a physical transformation of matter. Thus, the Court's assertion in Cochrane v. Deener\(^\text{25}\) that a process is "a mode of treatment of certain materials to produce a given result" was accepted without question.\(^\text{26}\) This assertion was never an issue until the advent of computers.

Attacks on the physical-transformation test began in earnest during the 1960s and 1970s. Despite a Presidential Commission report recommending against recognizing computer software as statutory subject matter,\(^\text{27}\) the U.S. Court of Custom and Patent Appeals (CCPA) opened the door to recognizing software patents.\(^\text{28}\) In a series of cases, the CCPA rejected both the physical operation\(^\text{29}\) and mental-step requirements.\(^\text{30}\) It declared that patentable subject matter includes anything in the "technological arts," and thereby rejected nearly one hundred years of Supreme Court jurisprudence.\(^\text{31}\)

It is not surprising that the emergence of computer technology created a dilemma for the courts. While both computer hardware and software are creative and indisputably useful, it is not appropriate to treat them identically. Computers, as electronic machines, fit neatly into the traditional patent rubric. Software, however, does not fit into this rubric; even though computers require software to operate, and cleverly written software is both innovative and non-obvious. On a purely theoretical basis, there is a strong temptation to extend patent protection to software. Yielding to this temptation, however, reduces overall incentives to innovate and increases legal uncertainty.\(^\text{32}\)

All software is not the same. Some software arguably performs a physical transformation of matter that is tangible and perceptible. Arranging pixels on the screen of a computer monitor is one example.\(^\text{33}\) In contrast, there is software that merely performs calculations or that developers use as building blocks for other

\(^{25}\) Cochrane v. Deener, 94 U.S. 780 (1877).

\(^{26}\) Id. at 788.

\(^{27}\) REPORT OF THE PRESIDENT'S COMMISSION ON THE PATENT SYSTEM, TO PROMOTE THE PROGRESS OF USEFUL ARTS IN AN AGE OF EXPLODING TECHNOLOGY (1966) [hereinafter, COMMISSION REPORT].

\(^{28}\) See Ford, supra note 4, at 63-70.

\(^{29}\) Id. at 64.

\(^{30}\) Id. at 63.

\(^{31}\) Id. at 69.

\(^{32}\) See infra Part III.

\(^{33}\) See, e.g., In re Alappat, 33 F.3d 1526, 1544 (Fed. Cir. 1994) (en banc) (arguing that a rasterizer of electrocardiographic data is a machine that produces "a useful, concrete, and tangible result").
programs and computer applications.\textsuperscript{34} Such software clearly does not satisfy the physical transformation test. Yet, without a theoretical rationale for distinguishing between valid and invalid subject matter, it is difficult to argue that the latter type of software is less deserving of patent protection than software that commands other devices to perform physical transformations. As neither Congress nor the Supreme Court addressed this emerging technology, the urge to fill this policy gulf was tremendous.\textsuperscript{35}

Therefore, rather than follow Supreme Court precedent, the CCPA instead adapted the law by revising a policy treatment that they perceived to be deficient. In order to provide patent protection for “deserving” software inventions, the CCPA and the CAFC weakened and eventually eliminated the Supreme Court’s physical-transformation test.\textsuperscript{36} These decisions left the CAFC with no logical basis for distinguishing valid software patent claims from invalid claims. Judges could now either apply obtuse and ambiguous tests, or conclude that all software inventions are statutory, provided they produce a “useful, concrete, [and] tangible result.”\textsuperscript{37}

\textbf{B. The Supreme Court’s Approach to Computer Software}

Unlike the CCPA, the Supreme Court recognized that software was a new class of invention. The Court cautiously considered whether to deem it statutory subject matter. It attempted to apply statutory law using rules of construction and case law and exhorted Congress to develop a coherent policy to address this new class of innovation.\textsuperscript{38} The Court deemed computer programs patentable subject matter by default, provided they were an integral part of a claim that included a physical transformation of matter.\

\textit{Gottschalk v. Benson}\textsuperscript{39} was the first Supreme Court case to address the patentability of computer programs.\textsuperscript{40} The disputed claim

\textsuperscript{34.} BEN KLEMENS, MATH YOU CAN'T USE: PATENTS, COPYRIGHT, AND SOFTWARE 40-42 (2006) (explaining that complex programs can be built from specialized, discrete functions contained in function libraries, which can be used without knowing exactly how each function performs its designated task).

\textsuperscript{35.} \textit{But see} COMMISSION REPORT, \textit{supra} note 27, at 13 (recommending against expanding patent coverage to include computer software).

\textsuperscript{36.} \textit{See supra} note 8.

\textsuperscript{37.} AT&T Corp. v. Excel Comme’ns, Inc., 172 F.3d 1352, 1359 (Fed. Cir. 1999).

\textsuperscript{38.} Gottschalk v. Benson, 409 U.S. 63, 73 (1972) ("If these programs are to be patentable, considerable problems are raised which only committees of Congress can manage, for broad powers of investigation are needed, including hearings which canvass the wide variety of views which those operating in this field entertain." (citation omitted)).

was for a device-independent "method for converting binary-coded decimal (BCD) numerals into pure binary numerals." The program's purpose, converting concatenated binary numerals representing decimal numbers, could be done mentally or manually using a chart.

The Court had an easy time disposing of this patent claim because the claim was "so abstract and sweeping" as to forestall the use of a BCD conversion in virtually all conceivable applications. According to the Supreme Court, allowing a patent on such a broad concept as this program, the equivalent of a mathematical formula, would amount to allowing a patent on an idea.

Although the Court could dispose of the case based on the broad breadth of the claimed program, Benson was, nevertheless, a difficult decision due to its policy implications. The Court recognized that the determination of software patentability is more appropriately a matter for Congressional consideration. The Court ruled that, "transformation and reduction of an article 'to a different state or thing' is the clue to the patentability of a process claim that does not include particular machines." However, the Court retreated from this clear delineation by stating, "[w]e do not hold that no process patent could ever qualify if it did not meet the requirements of our prior precedents." To further emphasize their discomfort with creating software patent policy, the Court continued, "[w]e do not hold] that the [Benson] decision precludes a patent for any program servicing a computer." However, the Court was clear that it did not believe that software programs were patentable either directly as programs or indirectly as processes. Extending patent protection to cover software was a job for Congress. In making the case for legislative

40. Id. at 73.
41. Id. at 64.
42. Id. at 66-67.
43. Id. at 68.
44. Id. at 71-72.
45. Id. at 73 ("The technological problems tendered in the many briefs before us indicate to us that considered action by the Congress is needed.").
46. Id. at 70.
47. Id. at 71.
48. Id.
49. Id. at 72 ("Indirect attempts to obtain [software] patents and avoid the rejection, by drafting claims as a process, or a machine or components thereof . . . rather than as a program itself, have confused the issue further and should not be permitted" (quoting COMMISSION REPORT, supra note 27)).
50. Id. at 73.
intervention, Benson left us an ambiguous decision subject to conflicting interpretations.

This ambiguity led the Supreme Court to revisit the software patentability issue six years later for the purpose of correcting the erroneous CCPA interpretation of Benson.\(^5\) Without legislative intervention, the CCPA interpreted Benson as barring software patents only when the process claim pre-empted use of the claimed algorithm.\(^5\) In reviewing the CCPA's action in Parker v. Flook,\(^5\) the Supreme Court found a claim that was not substantially different from the rejected algorithm in Benson. In Flook, the disputed claim was a method for calculating alarm limits for use in catalytic converters.\(^5\) Like the Benson claim, the disputed claim was machine independent and calculated a numerical result (an alarm limit) derived from entered data (temperature readings).\(^5\)

Rather than clarifying the law, Flook increased the ambiguity associated with software patentability. The Court could have resolved the case by applying the Cochrane physical-transformation test or by concluding that the associated process was merely a clever attempt to avoid receiving the same negative treatment as the Benson claim. However, the Court refused to embrace or reject the physical-transformation test. The Flook Court equivocated by acknowledging in a footnote that one could argue that Supreme Court precedent requires processes to change materials to a "different state or thing."\(^5\) However, the decision never revisited the argument raised in the footnote.

Instead, Flook introduced a new test that added to the confusion of the patentability of software. The Court proposed a subject-matter test that conflated the subject matter question with the 35 U.S.C. § 102 novelty requirement.\(^5\) The Court instructed claim reviewers to consider the algorithm to be "well known" in the prior art and then, subject to this constraint, evaluate whether the claim as a whole is statutory subject matter.\(^5\) However, in reality, this approach required a separate patentability review of the non-algorithm component of the claim. Flook unintentionally suggested this approach by declaring

\(^{52}\) In re Flook, 559 F.2d 21, 23 (Fed. Cir. 1977).
\(^{54}\) Id. at 585.
\(^{55}\) Id.
\(^{56}\) Id. at 588 n.9 (citing Cochrane v. Deener, 94 U.S. 780, 787-88 (1877)).
\(^{58}\) Flook, 437 U.S. at 592-94.
that, "the discovery of such a [law of nature or mathematical algorithm] cannot support a patent unless there is some other inventive concept in its application."59

The confusion created by the Supreme Court’s equivocations soon became apparent when the Supreme Court agreed to hear an appeal from a rejected claim in Diehr.60 The claim employed a computer-based algorithm in a process to continually monitor and adjust the curing of synthetic rubber.61 Both the patent examiner and U.S. Patent and Trademark Office (PTO) Board of Appeals rejected the claim.62 The patent examiner, applying Flook, concluded that the algorithm was non-statutory subject matter based on Benson, and that the remaining process elements did not contain an inventive element.63 The CCPA reversed the PTO Board of Appeals by rejecting the Board’s opinion that the inclusion of a computer program in an otherwise statutory claim rendered the claim non-statutory.64

In an effort to reconcile its prior decisions, the Court agreed with the CCPA that it was possible to have a patentable claim that included a computer program or algorithm.65 The Court’s first step was to unequivocally reiterate the continued vitality of its physical-transformation test.66 Applying this test, the Court reasserted that any patent claim that did not include a machine or apparatus had to transform an article "to a different state or thing" in order to be approved.67 The next step was to nullify Flook’s apparent adoption of a point-of-novelty test. The Court emphatically stated that, “[i]t is inappropriate to dissect the claims into old and new elements” and asserted that patent claims must be considered as a whole.68 To distinguish between the rubber-curing claim of Diamond and the alarm-limits claim of Flook, the Court asserted that the Flook claim covered a mathematical algorithm contained within an abstract,

59. Id. at 594.
61. Id. at 177.
62. Id. at 179, 181.
63. Id. at 180-81 ("The remaining steps--installing rubber in the press and the subsequent closing of the press--were 'conventional and necessary to the process and cannot be the basis of patentability.'").
64. In re Diehr, 602 F.2d 892 (1979).
65. Diehr, 450 U.S. at 187.
66. Id. at 184.
67. Id.
68. Id. at 188.
nebulous process. Therefore, even though the patent applicant drafted the claim as an algorithm embedded in a transformative process, the only concrete, substantive element was the description of the mathematical algorithm. Hence, the Flook claim was quite similar to the BCD conversion claim in Benson.

Diamond v. Chakrabarty also influenced the treatment of software's statutory-subject-matter status even though it did not directly consider software. Subsequent courts have misused a famous quote from the case to justify lowering standards for software patentability. In this case, the claimant engineered a bacterium that had the then unique characteristic of being able to break down crude oil. The issue was whether this living organism was statutory subject matter under 35 U.S.C. § 101. A critical question was whether all living organisms were by definition a product of nature and, therefore, excluded from patent protection.

The Supreme Court dispelled this notion with the now famous statement that, “Congress intended statutory subject matter to ‘include anything under the sun that is made by man.’” Living entities could be statutory subject matter provided they are invented and not pre-existing in nature. Inventions have properties that are a direct result of man’s handiwork and for which such properties do not exist already in nature. The non-existence of any crude-oil eating organism in nature was clear evidence that Chakrabarty invented, not discovered, his bacterium. Therefore, he was entitled to receive patent protection for the invention.

For purposes of this discussion, the most notable aspect of this case is the context of the Court’s “anything under the sun made by man” declaration. Inferior courts used this statement to assert that the Supreme Court gave its imprimatur to the expansion of § 101

69. Id. at 187.
71. See supra notes 40-44 and accompanying text.
73. Id. at 305.
75. Chakrabarty, 447 U.S. at 306.
76. Id. at 309 (quoting S. REP. NO. 82-1979 (1952); H.R. REP. NO. 82-1923 (1952)).
77. Id.
78. Id. at 310 (“...the patentee has produced a new bacterium with markedly different characteristics from any found in nature... His [invention] is not nature’s handiwork, but his own; accordingly it is patentable subject matter under 101.”).
statutory subject matter to include computer programs. However, this interpretation is incorrect. Rather, the Court’s objective with this quote was to dispel arbitrary limitations on statutory subject matter. Claims still had to qualify as § 101 processes, machines, manufactures, or compositions of matter. The Court concluded that a living being qualifies as a manufacture if "made by man." The concern with patent claims that include software or mathematical algorithms is whether they qualify as valid patent processes under § 101.

A two-part test was implicit in Chakrabarty. The first part determines whether the claim was an invention or a discovery of a product of nature. If an invention, then the claim must fit into one of the § 101 allowed categories. The Court considered the bacterium either a manufacture or composition of matter. The Diehr decision, which followed Chakrabarty, supports this interpretation. Although software may be an invention, it still must be a valid § 101 process to receive patent protection. Diehr reemphasized that a claim that did not include a machine must be rejected unless the claim involved the physical transformation of an article from one state to another. Thus, whereas a "naked" computer program may meet the invention requirement, it would not satisfy Diehr's physical transformation requirement.

C. The CAFC and the Dismantling of the Physical-Transformation Test

After 1982, the Supreme Court ceased considering software patentability questions, thereby ceding the ability to shape software
patent policy to the federal courts. The newly formed CAFC replaced the CCPA, but continued the CCPA’s generally expansive interpretation of statutory subject matter. While State Street Bank & Trust Co. v. Signature Trust Financial Group, Inc.,89 which removed the common-law business method patent prohibition, may be the CAFC’s most famous software patent decision, two earlier decisions were also pivotal in the federal court’s expansion of statutory subject matter.90

In Arrhythmia Research Technology v. Corazonix Corp.,91 the CAFC reviewed a claim for a process to identify the risk of ventricular tachycardia in heart attack patients by monitoring and analyzing heart activity using an electrocardiograph device (EKG).92 EKGs use electrodes attached to a patient’s body to record the heart’s electrical activity.93 EKGs then display the observed waveform information on a monitor or record it on a chart.94 In the patent claim, the included EKG also fed digital waveform information to a digital computer for analysis.95 The output of this analysis provided physicians with the means to distinguish between high-risk patients who require preventative treatment with potentially dangerous drugs and low-risk patients.96

The biggest question was whether the Arrhythmia Research claim could satisfy the Supreme Court’s physical-transformation test. The Arrhythmia Research claim appeared to be very similar to the rejected alarm-limits claim in Flook.97 As in Flook, the contribution over prior art was limited to the computer-based component of the claim.98 In addition, both claims received input in the form of conventional data—temperatures in Flook99 and EKG measurements in Arrhythmia Research.100 Moreover, in both cases data was the end product. The only significant difference between the two claims is

90. See Arrhythmia Research Tech., Inc. v. Corazonix Corp., 958 F.2d 1053 (Fed. Cir. 1992); In re Alappat, 33 F.3d 1526 (Fed. Cir. 1994) (en banc).
91. Arrhythmia Research Tech., Inc. v. Corazonix Corp., 958 F.2d 1053 (Fed. Cir. 1992)
92. Id. at 1054.
93. Id.
94. Id.
95. Id. at 1055.
96. Id. at 1054.
98. Arrhythmia, 958 F.2d at 1055.
100. Arrhythmia, 958 F.2d at 1054.
that the *Arrhythmia Research* claim included the EKG, the data-measuring device, whereas the *Flook* claim did not include a device to measure process variables. More important, the *Arrhythmia Research* innovation appeared to fail the Supreme Court's physical-transformation test because the claim did not transform an article from one state to another.

The CAFC avoided these obstacles by drawing conclusions that substantially lowered the threshold for software patents. First, the court concluded that the EKG's reading of electrical heart activity "transformed" such readings into digital signals. Second, the computer-based analysis of the EKG generated waveforms constituted a physical transformation of an electrical signal, a heart's electrical impulses, into a different state. In essence, the court concluded that measuring a physical phenomenon, recording the resultant data, and then manipulating that data in a digital computer meets the physical transformation requirement.

To evaluate the validity of this assertion, it is useful to ask whether the *Arrhythmia Research* claim is most analogous to the rejected *Flook* alarm-limits claim or the validated *Diehr* rubber-curing process claim. According to *Diehr*, the *Flook* claim provided a method for computing an alarm-limit. "[A]n 'alarm-limit' is simply a number and the *Flook* Court concluded that the application sought to protect a formula for computing this number." The *Arrhythmia Research* claim provided a method for isolating and filtering "noise" from the portion of a patient's EKG chart that is most likely to exhibit anomalies in the patient's EKG wave patterns. The presence of such anomalies provides an indication of a high risk of a specific heart disease.

Although the method's output was far more complex than the alarm-limit "number" in *Flook*, the *Arrhythmia Research* claim output was still data, whether represented graphically or numerically. Although the *Arrhythmia Research* claim's pre-solution use of an EKG is a difference, the claim contained no significant industrial post-solution activity. There is no conversion of a chemical into a different state and, unlike the *Diehr* claim, the solution does not

---

102. *Arrhythmia*, 958 F.2d at 1059-60.
104. *Arrhythmia*, 958 F.2d at 1055 ("[F]iltering is described as the critical feature of the Simson invention, in that it enables detection of the late potentials by eliminating certain perturbations that obscure these signals.").
105. *Id.* at 1054.
signal the automatic closing and opening of an industrial mold.\textsuperscript{106} One way to make the \textit{Arrhythmia Research} monitoring claim analogous to the rubber-curing claim would be to allow the method to automatically administer an appropriate drug therapy to patients found to be at high risk through the associated monitoring. Without such additional substantive steps, the \textit{Arrhythmia Research} monitoring claim is clearly closer to the \textit{Flook} alarm-limits claim.

In \textit{In re Alappat},\textsuperscript{107} the CAFC paved the way for the patenting of pure software claims by jettisoning the Supreme Court's physical-transformation test.\textsuperscript{108} The claim was a computer program that accepted a waveform data sequence (vector list) as input.\textsuperscript{109} The computer program (a rasterizer), residing in a conventional digital computer, then processed and filtered the resulting data to produce output data amenable to display on a cathode ray tube.\textsuperscript{110} While the rasterizer was at the heart of the claim, Alappat chose to draft the patent application as a machine claim.\textsuperscript{111} However, for meeting § 101 requirements, the PTO Board of Appeals required the claim to meet the same standards as a process that included software.\textsuperscript{112}

The CAFC decided \textit{Alappat} on policy grounds. Whereas \textit{Arrhythmia Research} trivialized the \textit{Diehr} physical-transformation test, \textit{Alappat} completely ignored it. \textit{Alappat} attempted to ascertain the limitations on software patentability by divining the policy bases for such limitations. Rather than rely on Supreme Court precedents dating back to the nineteenth century,\textsuperscript{113} the CAFC asserted that \textit{Diehr} identified just three categories of subject matter that are not entitled to patent protection: laws of nature, natural phenomena, and abstract ideas.\textsuperscript{114} The CAFC then drew the inference that a claim including software is non-statutory only to the extent that the claim as a whole "represent[s] nothing more than abstract ideas."\textsuperscript{115} Thus, \textit{Alappat} switched the burden of proof for claims containing software. Instead of having to meet the burden of proving the claim is § 101 subject

\textsuperscript{106} Diamond v. Diehr, 450 U.S. at 186-87.
\textsuperscript{107} \textit{In re Alappat}, 33 F.3d 1526, 1544-45 (Fed. Cir. 1994) (en banc).
\textsuperscript{108} \textit{Id.} (concluding that the proper test of a claim that contains mathematical elements is to determine whether the claim as a whole falls within one of the designated exclusions).
\textsuperscript{109} \textit{Id.} at 1537.
\textsuperscript{110} \textit{Id.}
\textsuperscript{111} \textit{Id.} at 1539.
\textsuperscript{112} \textit{Id.}
\textsuperscript{113} \textit{See supra} notes 23-26 and accompanying text.
\textsuperscript{114} \textit{Alappat}, 33 F.3d at 1542.
\textsuperscript{115} \textit{Id.} at 1543.
matter, subsequent challengers had to show only that the claim, as a whole, did not fit into one of the three designated patent-subject-matter exclusion categories. Under this standard, the CAFC had no trouble concluding that the rasterizer claim produced "a useful, concrete, and tangible result" that was patentable.

*Alappat* substantially lowered the software patentability hurdle. *Alappat's* "useful, concrete, and tangible" test provided a much lower threshold to patentability than did the *Diehr* physical-transformation requirement. Although it is likely that the BCD claim in *Benson* would still be rejected using this test, the alarm-limits claim in *Parker v. Flook* would be a much closer call. Alarm limits are clearly useful and, using the *Alappat* logic, may well produce concrete and tangible results. *Alappat* further weakened the software patentability barrier by ruling that a software program employed in a conventional digital computer could qualify as a machine claim for § 101 subject matter purposes. Framing a claim as a machine makes it more difficult to argue that the claim does not produce concrete and tangible results.

*State Street Bank* and *AT&T* removed all remaining § 101 impediments to software patents. *State Street Bank* involved a computerized accounting system used to allocate returns for mutual fund shareholders. The *AT&T* patent identified a method for recording useful billing information from long-distance telephone callers. These cases took a step beyond *Alappat*, in that their patent claims consisted entirely of business applications. Prior to *State Street Bank*, methods for conducting business did not generally satisfy the statutory subject matter requirement (the business method exception). *State Street Bank* rejected the business method exception, asserting that reliance on 35 U.S.C. § 101 and patent law in general are sufficient to evaluate such claims. The court employed *Alappat's* useful, concrete, and tangible result test to evaluate the

---

116. *Id.* at 1544 ("[T]he proper inquiry . . . is to see whether the claimed subject matter as a whole is a disembodied mathematical concept . . . which in essence represents nothing more than a 'law of nature,' 'natural phenomenon,' or 'abstract idea.'").

117. *Id.*

118. *Id.* at 1545.


120. *AT&T Corp. v. Excel Commc'ns, Inc.*, 172 F.3d 1352 (Fed. Cir. 1999).

121. *State St. Bank*, 149 F.3d at 1370.

122. *AT&T Corp.*, 172 F.3d at 1353.

123. *See DiMatteo, supra* note 5, at 14 ("Historically, business methods were not patentable under American patent law.").

mutual fund accounting method as constituting statutory subject matter.\textsuperscript{125}

\textit{AT&T} explicitly extinguished \textit{Diehr}'s physical-transformation test. Excel Corporation defended \textit{AT&T}'s infringement claim on the patent's failure to effect a physical transformation.\textsuperscript{126} Ignoring the physical transformation discussion in \textit{Diehr},\textsuperscript{127} the CAFC, instead, focused on the Court's use of "e.g." in \textit{Diehr} to infer that physical transformation was one of multiple ways software could satisfy § 101 subject-matter requirements.\textsuperscript{128} The court then proclaimed that the useful, concrete, and tangible result test had supplanted the physical-transformation test in stating: "[w]hatever may be left of the earlier test, if anything, this type of physical limitations analysis seems of little value."\textsuperscript{129} Thus, software and business methods no longer receive statutory-subject-matter scrutiny. Any software claim drafted in terms of a process, machine, manufacture, or composition of matter overcomes the § 101 hurdle.

III. SOCIAL WELFARE AND LEGAL UNCERTAINTY

A policy enhances social welfare if society overall, by a chosen metric, is better off than under the status quo or an alternative policy. The typical measure is whether the benefits of increasing innovation through a given patent policy exceed losses resulting from concomitant reductions in competition.\textsuperscript{130} Software patent analyses typically employ a social welfare analysis as their primary theoretical framework.\textsuperscript{131} While social welfare is important, any analysis of a judicial-implemented policy must also evaluate how any recommendations affect legal uncertainty. If implementing a reform recommendation increases legal uncertainty, any reform gains may be neutralized by reductions in predictability and increases in litigation.

This paper refutes the hypothesis that patents provide incentives for software innovation. Not only do patents not increase software innovation, they actually reduce innovation incentives. This result is

\textsuperscript{125} \textit{Id.} at 1373.

\textsuperscript{126} \textit{AT&T Corp.}, 172 F.3d at 1358-59.

\textsuperscript{127} \textit{Diehr}, 450 U.S. at 183-84.

\textsuperscript{128} \textit{AT&T Corp.}, 172 F.3d at 1358-59.

\textsuperscript{129} \textit{Id.} at 1359.


\textsuperscript{131} See, e.g., Thomas & DiMatteo, \textit{supra} note 5, at 37-41 (2007) (arguing that software patents produce dead-weight losses); Merges & Nelson, \textit{supra} note 130, at 868-84 (employing welfare economics to determine optimal patent scope by industry type).
not surprising. Even analyses that begrudgingly accept the existence of software patents base their acceptance on pragmatic considerations rather than economic optimality. According to these arguments, software patents are a fait accompli. Therefore, rather than offer a solution that provides optimal treatment of software patent claims, these commentators offer reform recommendations that ameliorate major software patent flaws without considering the seemingly radical step of banning software patents. Although ameliorative measures have an appeal because they address some economics-based criticisms, they do not alleviate welfare losses due to uncertainty. In fact, such measures likely increase uncertainty-based welfare losses by augmenting the discretionary judgments that decision makers in the patent review process make.

A. Economic Analysis of Software Patents

1. Overview

The Constitution grants Congress the power to promote innovation through its patent power. The tradeoff is that the exclusivity gained through patents provides holders with monopoly-like powers over their patents’ subject matter. If the product of a patent is in high demand and there are few or no close substitutes, the patent grant gives the patent holder power to set his prices above a competitive level. This power is enhanced when demand is highly

132. See, e.g., Julie Cohen & Mark Lemley, Patent Scope and Innovation in the Software Industry, 89 Calif. L. Rev. 1, 4 (2001) ("With some eighty thousand software patents already issued, the Federal Circuit endorsing patentability without qualification, and the Supreme Court assiduously avoiding the question, software patentability is a matter for the history books.") (citations omitted)).

133. Id.


135. Commentators are careful to avoid characterizing intellectual property rights as monopolies generally because there is a weak mapping, at best, between patent ownership and monopoly power. See, e.g., Daniel R. Cahoy, Changing the Rules in the Middle of the Game: How the Prospective Application of Judicial Decisions Related to Intellectual Property Can Promote Economic Efficiency, 41 Am. Bus. L.J. 1, n.18 (2003) ("One of the most common errors is in describing intellectual property rights as ‘monopolies.’"); HERBERT HOVENKAMP, ECONOMICS AND FEDERAL ANTITRUST LAW § 8.3, at 219 (1985) ("Many patents confer absolutely no market power on their owners . . . ."); Nat’l Inst. on Indus. & Intell. Prop., The Value of Patents and Other Legally Protected Commercial Rights, 53 Antitrust L.J. 535, 547 (1985) ("Statistical studies suggest that the vast majority of all patents confer very little monopoly power . . . .").
inelastic, as is the case of essential pharmaceuticals.\textsuperscript{136} If the product is one that consumers cannot live—or live well—without, then demand is insensitive to price changes. However, when producers exert monopoly power, society incurs losses due to suboptimal production and non-competitively high prices.\textsuperscript{137}

Paradoxically, an additional implication of monopoly power is that the monopolist has less incentive to innovate than does a firm operating in a competitive market.\textsuperscript{138} The monopolist, receiving an extra-competitive return from the status quo, has less need to replace his dominant product than does a producer in a competitive market receiving zero economic profits.\textsuperscript{139} Instead, the monopolist may invest in blocking competitors' efforts in order to maintain the profit-producing monopoly.\textsuperscript{140} Seeking and enforcing patent rights is consistent with this strategy. Historically, patent and copyright holders have prevented or delayed market entry by obtaining extensions to periods of exclusivity and by erecting greater legal barriers to unauthorized use of their protected intellectual property.\textsuperscript{141}

\begin{itemize}
\item \textsuperscript{136} Patricia M. Danzon, Price Comparisons for Pharmaceuticals: A Review of U.S. and Cross-National Studies 25 (1999) ("[T]he demand for drugs is likely to be more inelastic than the demand for other consumer goods . . . .").
\item \textsuperscript{137} See generally William D. Nordhaus, Invention, Growth, and Economic Welfare (1969) (analyzing the tradeoff between patents' enhanced incentives and the reduction in competition due to patent exclusivity).
\item \textsuperscript{139} See William J. Baumol & Alan S. Blinder, Economics: Principles and Policy 260 (10th ed. 2006) ("[T]here is little incentive for management [in a monopoly] to make the effort to produce efficiently with a minimum of waste or to undertake the expense and risks of innovation.").
\item \textsuperscript{140} See Cecil D. Quillen, Jr., Innovation and the U.S. Patent System, 1 Va. L. & Bus. Rev. 207, 210 (2006) ("The paramount motivation for [obtaining patents] is not the expectation of a patent monopoly, but rather the hope of preempting and blocking others from obtaining patents that might impede their own innovations . . . ."); Michael J. Malinowski & Radhika Rao, Legal Limitations on Genetic Research and the Commercialization of Its Results, 54 Am. J. Comp. L. 45, 49 (2006) ("[A gene patent gives] its owner the power to prevent others from conducting research, performing tests, or developing therapies for that gene . . . . As a result, gene patents may actually hinder innovation . . . ."); Merges & Nelson, supra note 130, at 865 ("[T]he original patentee may use her patent as a ‘holdup’ right, in an attempt to garner as much of the value of the improvement as possible."). But see Howard A. Shelanski & J. Gregory Sidak, Antitrust Divestiture in Network Industries, 68 U. Chi. L. Rev. 1, 12 (2001) (countering Paul Romer's antitrust testimony that Microsoft had used its market power to limit competition).
\item \textsuperscript{141} 17 U.S.C. § 302 (2000) (establishing the copyright terms for anonymous works, pseudonymous works, and works for hire, as the shorter of 95 years from publication or 120 years from creation); 35 U.S.C. § 154(a)(2) (2000) (reflecting Congress' change of the calculation of patent terms from 17 years from issuance to 20 years from the date of filing to bring the United States into compliance with the Agreement On Trade-Related Aspects of
Some scholars believe that the gains from issuing patents do not exceed the associated economic losses. Empirically, software technology developed satisfactorily prior to the recognition of software patents, and there is little to suggest that continued software development requires patent protection. In addition, the assertion of patent rights impedes the attempts of others to innovate by blocking their efforts or significantly raising their costs. In some industries, innovation may be nearly impossible for anyone other than companies with large patent portfolios because new products often require obtaining rights to use dozens or even hundreds of patent-protected technologies. Companies with large patent portfolios often cross-license patents in order to avoid litigation and gain needed patent rights. This practice benefits large companies at the expense of smaller companies. Without patents to trade, smaller companies and not-for-profit entities must negotiate license fees without leverage or risk litigation in order to obtain access to technology required for competition. These costs of acquiring technology rights have a pernicious impact on small companies' competitiveness. This

---


142. See Edwin Mansfield, Patents and Innovations, 32 MGMT. SCI. 173 (1986) (asserting that the lack of patents does not deter innovation in many industries).

143. See ADAM B. JAFFE & JOSH LERNER, INNOVATION AND ITS DISCONTENTS: HOW OUR BROKEN PATENT SYSTEM IS ENDANGERING INNOVATION AND PROGRESS, AND WHAT TO DO ABOUT IT 201 (2004) ("[S]oftware innovation was flourishing before the 1980s, when the CAFC clarified and broadened the patentability of software. This seems to show that patents for software are not necessary."). See also COMMISSION REPORT, supra note 27, at 13 ("It is noted that the creation of programs has undergone substantial and satisfactory growth in the absence of patent protection.").

144. See FED'L TRADE COMM'N, TO PROMOTE INNOVATION: THE PROPER BALANCE OF COMPETITION AND PATENT LAW AND POLICY, EXECUTIVE SUMMARY pt. II, § A, at 5 (2003), available at http://www.ftc.gov/os/2003/10/innovationrptsummary.pdf ("One firm's questionable patent may lead its competitor to forgo R&D in the areas that the patent improperly covers.").


146. See Rochelle Dreyfuss, Protecting the Public Domain of Science: Has the Time for an Experimental Use Defense Arrived? 46 ARIZ. L. REV. 457, 468 (2004) ("[I]f every competitor in a field knows that the others are also obtaining patents, there is less of a tendency to engage in patent warfare. Assertions may not be made at all; when they do occur, disputes are settled by cross licensing."); Steven C. Carlson, Note, Patent Pools and the Antitrust Dilemma, 16 YALE J. ON REG. 359, 363-70 (1999).

147. See Dreyfuss, supra note 146.
reduction in competition adds to the social welfare losses attributable to patents.

2. Prospect Theory

Other economists believe patents play an important role in promoting innovation. Edmund Kitch speculated that the absence of patent rights produces a tragedy of the commons in that failing to assign exclusive rights leads to inadequate development of innovations.148 In the tragedy of the commons, the ability of multiple private parties to consume a good in common supply for less than their consumption’s social costs leads to the depletion of such goods due to overuse or over consumption.149 Analogously, under Kitch’s prospect theory, failing to assign property rights to innovative technology leads to inadequate or rushed development of technology because multiple private parties will exploit the innovation without incurring the costs of that development. As a result, there is inadequate investment in innovation.150 If, however, an innovator has a property interest, he has the breathing room to continue developing the technology without fear of appropriation by a competitor.151 The property right also allows the innovator to coordinate licenses to other parties in order to avoid inefficient duplication of development effort.152

Others reject Kitch’s prospect theory. Robert Merges and Richard Nelson argue that, “the real problem is not controlling overfishing [as in the tragedy of the commons], but preventing underfishing after exclusive rights have been granted.”153 The two contrasting views both have validity. Merges and Nelson observed that technological advances are not homogeneous across industries.154 They identified four types of inventions: discrete, cumulative, chemical, and science-based.155 The cumulative and discrete categories are most relevant to this inquiry. Discrete inventions are autonomous and have limited impact on subsequent innovations.156

---

150. Kitch, supra note 148, at 276-77.
151. Id. at 276.
152. Id. at 279.
153. Merges & Nelson, supra note 130, at 873.
154. Id. at 843.
155. Id. at 880.
156. Id.
The ballpoint pen and many pharmaceutical products are discrete inventions. Cumulative inventions, by contrast, build on earlier efforts and interact with other technologies. Generally, they have little intrinsic efficacy and provide value only through interactions with existing technology and as mileposts in the development of new technology. Manufactured products, such as automobiles and semiconductors, as well as software, are all products that comprise numerous cumulative inventions.

Prospect theory does a good job of describing the development of pharmaceutical products, which do face an overfishing problem. Pharmaceutical chemical entities often require millions of dollars to develop products. Without the ability to at least recover development costs through patent protection, few firms would invest in developing pharmaceuticals. Without patent protection, followers would quickly enter the market for valuable inventions, driving the price for such products down to the zero-profit point. This situation would leave the inventor unable to recoup development costs. In addition, redundant research on a chemical entity is socially wasteful if a firm has a sufficiently high probability of successfully developing the efficacious chemical entity. In such cases, society would be better off if only one firm focused on the targeted product and other firms reposition their investments. Therefore, once a firm creates a new chemical entity, society benefits if that firm is able to manage or curtail outside development of the chemical entity. Granting patent rights to the first inventor of a discrete invention is unlikely to impede development efforts of other firms because subsequent innovations can and do generally proceed independently of the earlier invention. Therefore, blocking patents are not a concern for owners of discrete inventions. By insuring that the original creator has the ability to fully exploit the commercial value of an invention, granting pharmaceutical patents provides incentives for firms to continue developing new chemical entities.

157. Id.
158. Id. at 881.
159. Id.
160. Id.
161. Henry Grabowski, Patents and New Product Development in the Pharmaceutical and Biotechnology Industries, 8 GEO. PUBLIC POL’Y REV. 7, 9 (2003) (“It takes several hundred million dollars to discover, develop and gain regulatory approval for a new medicine.”).
162. Merges & Nelson, supra note 130, at 881 (“[P]ossession by [a patent holding] firm of a proprietary lock on the invention is not a serious hindrance to inventive work by many other firms.”).
3. Economic Taxonomy of Industrial Innovation

Burk and Lemley, in their analysis of judicial policy levers, identify four other highly related types of innovation: competitive innovation, cumulative-innovation, anti-commons, and patent thicket. In competitive innovation industries, companies have incentives to innovate even in the absence of patent protection. R&D is light or limited and companies gain sufficient incentive to innovate from first-mover advantages or support from government funding. In cumulative-innovation industries, inventors progress in small, incremental steps building off the work of their predecessors. In such industries, optimal innovation occurs when patent rights are narrow, thereby giving inventors incentives to create without blocking the path of subsequent creators. In an anti-commons industry, innovators must assemble rights to advance or secure the value of their innovations. If rights are sufficiently dispersed, or an uncooperative firm holds a critical right, then either high transaction costs or inadequate patent coverage can turn the net value of an innovation to the negative. The final innovation type involves "patent thickets." Patent thicket conditions are similar to anti-commons conditions in that patent rights are highly complementary. Under these conditions, multiple patent rights overlap, making it extremely difficult to exercise the rights granted under a patent without receiving licenses for all overlapping patents.

Burk and Lemley identify industries that they believe are archetypical of each of these innovation types. Competitive innovation characterizes business methods and the early development of software. In such industries, patents are largely unnecessary for innovation. According to their analysis, cumulative innovation effectively describes modern software development.
Lemley believe patent policy should narrow the scope of software patents in order to minimize the harmful impact of software patents on subsequent innovation. Their analysis suggests that the anticommons condition describes biotechnology because of the relative ease of developing close substitutes for efficacious chemical entities. In such cases, the creator of the original product may not be able to recover its high research and development costs, leaving it with inadequate incentives to continue investing in innovation. Therefore, Burk and Lemley recommend that patent scope for biotechnology and other anti-commons industries be broad enough to cover substitutes. The patent-thicket condition describes the semiconductor industry. In designing semiconductors, producers must marshal a multitude of technologies. To address the patent thicket condition, Burke and Lemley recommend that examiners construe patent scope narrowly in order to reduce the multitude of overlapping patent rights that deter innovation.

4. Why Patents Do Not Promote Software Innovation

Although I find Burk and Lemley's policy recommendations for anti-commons and patent thicket industries persuasive, I disagree with their recommendations for modern software patents. The objective of their analysis is to identify policies that create optimal incentives for inventors and firms to innovate. Thus, for competitive-innovation industries, patent protection is unnecessary because inventors receive adequate rewards outside the patent system. In cumulative-innovation industry, inventors may require some inducement, so patent policy must be narrowly applied to provide this incentive without giving the inventor so much protection that the patent impedes others from innovating. However, rather than focus on innovation incentives at the individual level, the focus should instead be on the impact of patent policy on market-wide innovation levels.

The argument, advanced by Ben Klemens, that patent policy should maximize the level of overall market innovation, is consistent

175. Id. at 1624-25.
176. Id.
177. Id. at 1624-27.
178. Id. at 1681.
179. Id. at 1627-29.
180. Id. at 1627-28.
181. Id. at 1694-95.
182. Id. at 1578-79.
183. Id. at 1689.
with the Patent Clause’s mandate, “to promote the useful arts and sciences.”  

If reducing individual patent rights increases market-wide innovation, then society is better off. More importantly, scientific progress is advanced more when overall innovation increases than by increasing protection of individual patent rights. If patent policy, instead, induces a few firms to increase their innovation while obstructing industry-wide innovation, then society is worst off. Therefore, the objective of this analysis is to determine how market-level innovation varies with patent policy.

Market-wide software innovation is maximized when patent protection is non-existent. Although software development has some characteristics of a cumulative-innovation industry, competitive-innovation is a closer fit. The cost, both in terms of capital investment and risk of failure, of software development is relatively low. Moreover, first-mover advantages, together with lock-in effects from switching costs and network externalities, are sufficient incentives to insure continued software innovation. However, the existence of software patents has a chilling effect on market-wide innovation, resulting in a loss to society. The following discussion illustrates these claims through a series of simple observations.

\[ \text{a. Observation 1: When R&D Costs are High There are Strong Disincentives to Investing in Innovation in the Absence of Patent Protection} \]

When R&D costs are high, the incumbent inventor faces much greater costs than rivals. If the rival is able to imitate the incumbent’s innovation inexpensively, then competitors are free to duplicate the incumbent’s innovation without incurring substantial research costs. Their rapid market entry drives prices to the zero-profit level, leaving the incumbent unable to recoup its initial R&D investment. As a result, there is no innovation, because a profit-maximizing firm refuses to invest when it faces a negative return. Thus, in this situation, there is no innovation without patent protection.

Pharmaceutical and biotechnology (biotech) are the archetype industries for this condition. In biotech industries, R&D costs are extremely high. It is not unusual for firms to spend millions of dollars

\[ 184. \quad \text{Klemens proposes to maximize the market size. However, this paper assumes that increased market size is synonymous with increased innovation. KLEEMENS, supra note 34, at 16 (2006).} \]

\[ 185. \quad \text{See Burk & Lemley, supra note 138, at 1687-88.} \]

\[ 186. \quad \text{Id. at 1676.} \]
in research on a single chemical entity. In addition, there is a significant probability that the research will fail either in the laboratory or during clinic trials. Competitors face much lower R&D costs because they can learn from the incumbent’s research and public disclosures, so the cost of imitation is low. A competitor who produces an efficacious chemical entity without permission from its creator can drive the price down to a point where the creator cannot recover its research costs. Thus, without patent protection, innovation declines precipitously in biotech industries.

b. Observation 2: Patent Rights Provide Socially Beneficial Innovation Incentives when R&D Costs are High

Optimal patent protection can address the market failure described above. In this condition, patents provide incumbents with sufficient exclusivity to recover their investments, including a risk premium. Optimal patent policy, defined by scope and term, can increase the competitor’s cost to a point where the incumbent can be assured of receiving a positive expected return when the competitor exploits the patent with a direct copy or close substitute. Society is better off because the incumbent has the incentive to produce the initial innovation, and the competitor has the incentive to improve the invention or develop an efficacious substitute. Therefore, society can end up with multiple efficacious products.

c. Observation 3: Patents do not Protect Patent Holders from Competitors’ Attempts to Expropriate the Value of Patents when Patent Scope is Sub-Optimally Narrow

In this condition, the competitor cannot duplicate the patent, but the competitor’s costs for developing and producing a substitute are relatively low. The competitor has little difficulty producing a non-infringing substitute with similar properties as the patent-protected product. In the case of pharmaceuticals, the competitor may be able to replace a key chemical in the patent with a chemical or compound with similar properties without infringing the incumbent’s patent.

187. Id. at 1616.
188. Id. at 1676.
189. See WILLIAM M. LANDES & RICHARD A. POSNER, THE ECONOMIC STRUCTURE OF INTELLECTUAL PROPERTY LAW 299 (2003) (identifying the ability of competitors to gain from innovators’ public disclosure as “incomplete appropriability”).
190. See Burk & Lemley, supra note 138, at 1677.
Costs of producing a substitute are low. Therefore, if patent law does not exclude production of the substitute through the doctrine of equivalents, the patent has little value.\textsuperscript{191} This condition corresponds closely to the no-patent-protection condition.\textsuperscript{192} As in that condition, the incumbent under-invests in innovation because it is unlikely to recover its R&D costs.

d. Observation 4: Rivals Face Disincentives to Producing Socially Beneficial Innovations when Patent Scope is Sub-Optimally Broad

This condition describes a situation for which the incumbent produces a socially beneficial software innovation that nets positive profits. If the patent scope on this innovation is overly broad, then competitors cannot profitably create market substitutes or improvements without infringing the patent. Therefore, with competitors unable to obtain a non-negative return in this market segment, the incumbent innovator faces no competition. The patent is an effective barrier to entry and the incumbent reaps supra-competitive profits. There are significant social welfare losses because not only is the incumbent able to charge prices above the competitive level, but there also are no substitutes or improvements because patent policy does not allow competitors to obtain a non-negative profit. Thus, society loses innovations that it would have had with a less exclusive patent.

Burk and Lemley agree with Klemens that the PTO treatment of software patents results in an overly broad patent scope.\textsuperscript{193} The wide breadth of software patent coverage makes it difficult for competitors to produce improvements in areas for which the PTO has granted software patents.\textsuperscript{194} U.S. software patent policy exacerbates the problem. Software patent applications do not include source code (the program written in a human language) and the PTO often approves claims consisting of little more than a rudimentary flow chart.\textsuperscript{195} Thus, software patent holders lay claim to broad areas of software practice without well-identified claim boundaries, with virtually no implementation details, and with few clues about the quality of claim

\begin{itemize}
  \item \textsuperscript{191} See infra note 280 and accompanying text (discussing the doctrine of equivalents).
  \item \textsuperscript{192} See supra notes 186-190 and accompanying text.
  \item \textsuperscript{193} Burk & Lemley, supra note 138, at 1594 ("[software patents that the CAFC] approves will be entitled to broad protection"); KLEMENS, supra note 34, at 73.
  \item \textsuperscript{194} Burk & Lemley, supra note 138, at 1622-23.
  \item \textsuperscript{195} See KLEMENS, supra note 34, at 21-22.
\end{itemize}
Burk and Lemley suggest that the appropriate way to address these issues is by limiting software patent scope. The feasibility of this recommendation is discussed later in this analysis. However, as discussed below, this recommendation does not produce an optimal outcome.

e. Observation 5: The Cost of Producing Patentable Software Innovation is Very Low Relative to the Costs of Producing R&D-Intensive Innovations

It is accepted that software development is far less expensive than innovation in other industries. However, Burk and Lemley suggest that software development costs have increased due to the complexity of modern end-user applications and time consuming debugging cycles. Although this claim is certainly correct, it does not follow that the cost of patentable software innovation has increased with the cost of developing final, polished consumer products. To understand this difference, a distinction must be drawn between software programs and software functions. Computer and software consumers work with application programs such as word processors, graphics programs, and customized applications. Complex application programs, such as operating systems, may require years of expensive development and debugging. However, complex end-user programs are rarely the subject of patents. Just as a car contains many components and subsystems that are not entitled to patent protection, complex application programs contain multiple technologies for which only a subset are entitled to patent protection. The software counterparts to these automobile subsystems are known as "functions."

196. See id.; Cohen & Lemley, supra note 132, at 24-25.
197. See Burk & Lemley, supra note 138, at 1619.
198. See infra notes 327-349 and accompanying text.
199. See Burk & Lemley, supra note 138, at 1618-19.
200. Id. at 1582-83.
202. See Cohen & Lemley, supra note 132, at 28 ("Many software inventions are internal to the program, and their use cannot be detected without parsing the code.").
203. Id. at 29 (advocating a reverse-engineering exception for software patents to allow developers to gain access to the unprotected components of application programs).
204. See WALTER SAVITCH, ABSOLUTE C++ 92 (1st ed. 2002) ("Most programming languages have functions or something similar to functions, although they are not always called by that name in other languages.").
Modern programming employs a modular approach to manage the complexity of large programs. Again, the car analogy is instructive. Cars contain tires, engines, brakes, and other components that perform specific discrete tasks. Like cars, application programs are built from multiple functions that produce specific program behaviors. Software functions may specify how the program handles data input-output or may provide methods for performing mathematical and statistical operations. Typically, common and related functions are assembled into function libraries, many of which are generally available. These libraries can be used to provide capabilities for any program that requires the included tasks. Programmers just need to know what the library functions do and their syntax. They do not need to know how the functions achieve their results. In fact, it is possible to create a word processor and other common computer applications using function libraries and just enough programming code to integrate the functions into a cohesive whole.

Therefore, it is at the function level—the building block level—where patentable software subject matter is found. Software patents are more valuable to patent holders when patent breadth is maximized. Patents on critical functions that must be used in dozens or hundreds of programs can provide licensing fees or significant leverage in cross-licensing negotiations for patent rights. Such patents also provide substantial benefit to the holder when embedded in complex programs. Competitors are likely to incur high costs and risks in creating program substitutes due to the difficulty of ascertaining how basic patents are used in complex programs.

Therefore, development of patentable software differs from R&D for other types of patentable innovation in two significant ways. The R&D required to obtain software patents is both less expensive and much less risky than other types of innovation. Biotech R&D

205. See KLEMENS, supra note 34, at 40.
206. Id.
207. Id. at 41; see also SAVITCH, supra note 204, at 92 (“C++ comes with libraries of predefined functions that you can use in your programs”).
208. See KLEMENS, supra note 34, at 41-42.
209. Id.
210. Id. at 40.
211. Id. at 42.
212. See Dreyfuss, supra note 146, at 468.
213. See Cohen & Lemley, supra note 132, at 27-29 (recommending the creation of a reverse-engineering exemption for software due to the difficulty of identifying patentable subject matter when developing software and engaged in infringement litigation).
requires great expense and a significant probability of failure.\textsuperscript{214} By contrast, the software R&D required to obtain a patent can literally be conducted with paper and pencil due to the PTO's treatment of software patent claims.\textsuperscript{215} The PTO, following the CAFC's guidance, does not require the submission of written programs or other detailed information about software patent claims.\textsuperscript{216} Therefore, a developer can submit and win approval for a software patent claim after conceiving of and outlining the basic capabilities of its software patent claim and submitting a flowchart to the PTO.\textsuperscript{217} There is no need to have a working program or to even have started programming the described functions.\textsuperscript{218}

The cost is further reduced by the very low risk of failure. If a computer-based-task can be described, then a competent programmer can write the code to execute the task.\textsuperscript{219} Although there may be a significant risk of failure for implementing the full functionality of complex programs, very little risk exists in implementing software functions. Klemens suggests that if a programming task proves difficult, programmers break the task down into smaller component sub-functions until they are able to write code that implements the designated task.\textsuperscript{220} Repetition of these steps and subsequent aggregation of such sub-functions yields the desired program.\textsuperscript{221} Essentially, patentable software innovation is entirely embodied in conception of the new, useful, and non-obvious computer task. Once the developer has devised an innovative software idea, there is very little risk of failure.\textsuperscript{222} Hence, whereas the cost of developing complex software programs may be high, the cost in time, money, and risk of developing patentable software innovations is not high.

\begin{itemize}
\item \textsuperscript{214} See supra notes 186-188 and accompanying text.
\item \textsuperscript{215} See Burk & Lemley, supra note 138, at 1582 ("[I]t has long been possible for two programmers working in a garage to develop a commercial software program.").
\item \textsuperscript{216} See Cohen & Lemley, supra note 132, at 18.
\item \textsuperscript{217} See KLEMENS, supra note 34, at 21-23.
\item \textsuperscript{218} See id.
\item \textsuperscript{219} Id. at 43.
\item \textsuperscript{220} Id. See also Samuelson et al., supra note 5, at 2326 ("Programmers routinely work by decomposing large tasks into smaller sub-tasks and sub-sub-tasks. They write a sub-program to accomplish each of the smaller tasks, then orchestrate the sub-programs' interaction so that the combination works together to accomplish an overall task.").
\item \textsuperscript{221} See, e.g., Samuelson et al., supra note 5, at 2326
\item \textsuperscript{222} See Burk & Lemley, supra note 138, at 1687-88. See also KLEMENS, supra note 34, at 43 ("Given a computable task, any two competent programmers could write a program to perform the task.").
\end{itemize}
Observation 6: Optimal Software Innovation Does Not Require Patent Protection

Patent protection is unnecessary to encourage software development due to its low cost and risk. If software is indeed a competitive innovation industry, then no further arguments are needed. Rewards from other sources would be sufficient to stimulate innovation. However, software development also has cumulative innovation characteristics. In such industries, narrowly tailored patents are useful to induce optimal innovation levels. Therefore, the question is whether, for purposes of motivating innovation, software development is predominately a competitive or cumulative innovation industry.

Competitive innovation most accurately characterizes software innovation incentives due to the upstream, component nature of software innovation. Complex software applications, like complex machines, are composed of many different components and patentable innovation occurs at the component level. Thus, software developers are motivated to solve problems that arise in building complex applications. Application programs sold to end-users protect software innovations without patents due to network effects, switching costs, and the incumbent’s ability to “hide” patent-protected software in complex programs.

Network effects are present when the value of using a product increases with the total number of product users. For example, a video game console with a large market share is more beneficial because it is easier to find other people with whom to play or exchange games. Switching costs tend to lock people into a particular standard due to the high cost of learning new skills, converting data, and buying new equipment and applications. As an illustration, few users of a specific computer operating system change operating

---

223. See Burk & Lemley, supra note 138, at 1618-19 (discussing the success of software in times when software was not patentable); Robert P. Merges, Uncertainty and the Standard of Patentability, 7 HIGH TECH. L.J. 1, 32 (1992) (asserting that patent protection for highly certain projects does not matter because such projects "are likely to be pursued anyway").


225. Id. at 1623.

226. See supra notes 202-213 and accompanying text. See also Burk & Lemley, supra note 138, at 1590 ("Machines of even moderate complexity are composed of many different pieces, and each of these components can itself be the subject of one or more patents.").


systems because it would require purchasing new computers, new software applications, and learning how to work in the new computer environment. Therefore, it has been very difficult for Unix and Apple Inc. operating systems to increase their market share even though Microsoft's operating systems are frequently the subject of criticism.\textsuperscript{229}

Finally, innovative functions are effectively hidden in application programs.\textsuperscript{230} Although it may be easier to reverse engineer software without patent protection, the most effective method of reverse engineering software, decompilation, is sufficiently time consuming to provide software innovators with sufficient lead-time to develop a first-mover advantage.\textsuperscript{231} These three effects secure rewards from marketable innovation while reducing the risk of rivals appropriating the value of an incumbent's patent. Network externalities and switching costs "lock in" users. Therefore, the incumbent who produces a product with a positive revenue stream is likely to maintain those rewards without patent protection. Thus, patent protection is not required for software innovation.

g. Observation 7: Software Patents Reduce the Overall Level of Software Innovation

The dynamics described above raise competitors' innovation costs without providing an offsetting increase in the incumbent's incentives. In addition, unlike other types of innovation, competitors do not benefit from published software patents because the threshold for meeting disclosure requirements for software patent claims is low.\textsuperscript{232} Thus, Landes and Posner's claim that patent public disclosure requirements lower competitors' development costs does not apply to software patents.\textsuperscript{233} Instead, the limited disclosure requirement allows software patent holders to carve out large areas of software practice that competitors must avoid.\textsuperscript{234} Limited disclosure, broad patent scope, and the prohibition against reverse engineering increase the

\textsuperscript{229} See Thomas A. Piraino, Jr., \textit{An Antitrust Remedy or Monopoly Leveraging by Electronic Networks}, 93 NW. U. L. REV. 1, 18-19 (1998) ("Apple's Macintosh [and other] operating systems have, at certain times, been superior to the operating system offered by Microsoft. Nevertheless, Microsoft's Windows system has gained an unassailable dominance . . . ").

\textsuperscript{230} See Cohen & Lemley, supra note 132, at 28.

\textsuperscript{231} Samuelson et al., supra note 5, at 2336 ("[D]ecompilation is a painstaking and time-consuming process.").

\textsuperscript{232} See supra notes 195-196 and accompanying text.

\textsuperscript{233} See LANDES & POSNER, supra note 189, at 299.

\textsuperscript{234} See supra notes 195-196 and accompanying text.
costs of developing substitutes as well as identifying areas of practice that must be avoided. These increased costs act as a disincentive to rival innovation. In addition, the patent-holding incumbent, as long as it receives positive profits from its patent, has little incentive to improve its product. Rather than innovate, the incumbent has a strong incentive to maintain the positive profit flows from existing products. Thus, both incumbent and competitors reduce their level of software innovation when software patents are available. Hence, when software patents cover a market, there is less software innovation and less competition than in a patent-less regime.

5. Conclusion: Software Patents Do Not Enhance Social Welfare

The economic model shows that software patents do not provide incentives to innovate and reduce the aggregate level of software innovation. This conclusion follows from the relatively low cost and low risk associated with software innovation. Innovation in other industries requires higher investments with lower probabilities of recovering investments. In these industries, firms devote considerable resources to research with a significant probability that the innovative firm will not recover its investment. Given the size of the investment, the innovation return must be at least as large as ongoing costs, plus R&D costs. Therefore, profit-maximizing firms would not innovate in high-investment industries absent patent protection because such firms could not prevent others from expropriating the value of a successful innovation. Without effective intellectual property protection, competitors appropriate valuable innovations. Such appropriation leaves the innovator without the means to recoup its investment. This inability to recover innovation investments is a market failure, and patents effectively correct this failure.

However, software innovation is not subject to this problem. Rather, the application of patent policy to the software industry results in social welfare losses with no gain because software innovators do not risk losing sizable research investments and benefit from market rewards. Patentable software innovation occurs during idea conception and does not require extensive research and experimentation like biotech and other industries. Once a software producer conceives an innovative idea, code creation is a trivial and deterministic exercise. After idea conception, the idea creator can

235. See supra notes 138-141 and accompanying text.
236. See KLEMENS, supra note 34, at 43.
develop products using the little risk that rivals will expropriate the value of his creation. Any application of patent protection at this point raises competitors' innovation costs and reduces competition without increasing the incumbent's incentives to innovate. Thus, rewarding software producers with patents fails to satisfy the legislative mandate of promoting science.

### B. Legal uncertainty and Patent Law

1. Overview of Legal Uncertainty

The remainder of this analysis considers the appropriate norms and rules courts should adopt in effectuating these principles. Desired rule characteristics include resolving disputes accurately and expeditiously. A rule is accurate when its application is calculated to achieve clearly identified policy results. Measuring normative accuracy is difficult. However, useful proxies include consistency and predictability. Failure to accurately decide cases injects inconsistency and lack of predictability into the rule application. Inconsistency in decision-making reduces confidence in the process, increases transactions costs, and raises questions about the appropriate allocation of rights. It can also have a stifling impact on business because companies may be reticent to develop new products or enter new markets when they cannot accurately predict whether their interests will be protected. When decision rules are applied in an inconsistent manner and there is a lack of predictability, high legal uncertainty arises.

For these reasons, legal uncertainty inhibits innovation. Inventors are deterred from investing in innovation when there is significant uncertainty about whether the judiciary will protect

---


238. *Id.* at 1149.

239. *Id.*

240. *Id.*

241. *Id.* at 1158-59 (arguing that consistency and predictability flow from accuracy).

innovations.\textsuperscript{243} Similarly, uncertainty about patent coverage deters firms from innovating and significantly increases costs associated with innovation due to the threat of infringement allegations.\textsuperscript{244} In patent law, uncertainty can increase litigation and appellate activity because parties are unable to resolve disputes without judicial intervention.\textsuperscript{245} Uncertainty in litigation increases appellate activity and reduces faith in the judicial process when parties consider trial outcomes unreliable and unpredictable.\textsuperscript{246}

Uncertainty in patent litigation may also encourage opportunistic behavior. High reversal rates on appeal, in addition to diminishing judicial credibility, may encourage parties with low-merit cases to exploit high trial error rates. Thus, if a multi-million-dollar award is at stake, a plaintiff can have a positive expected return without a strongly meritorious case. Moreover, because the threat to litigate is credible due to lack of predictability, defendants without deep pockets may settle such claims rather than incur litigation expenses and risk an inaccurate ruling. Many high-technology companies believe that pejoratively labeled "patent trolls" are responsible for many patent infringement cases they face.\textsuperscript{247} If such strategic behavior exists, then uncertainty, high judicial error rates, and high potential awards are major catalysts.

The relationship between uncertainty and litigation rates has been the subject of multiple studies.\textsuperscript{248} With symmetric interests and in the absence of uncertainty, all disputes between rational parties settle. Under these conditions, both sides agree on the expected outcome of trial, so it is less costly for the parties to settle and avoid litigation costs. In this situation, if both sides face positive costs, the


\textsuperscript{244} See Thomas, \textit{Vanquishing}, supra note 1, at 721-22.

\textsuperscript{245} See Wallace, supra note 243, at 1383.

\textsuperscript{246} See Cybor Corp. v. FAS Techs., Inc., 138 F.3d 1448, 1478 (Fed. Cir. 1998) (en banc) (Rader, J., dissenting).

\textsuperscript{247} See Thomas, \textit{Vanquishing}, supra note 1, at 721-22.

plaintiff's net expected gain is always greater than the defendant’s net expected loss. Therefore, both sides are better off without going to trial. However, when uncertainty causes the two sides to disagree about the expected outcome of litigation, no “positive settlement zone” exists, leaving trial as the only outcome.

In their seminal work on litigation, George Priest and Benjamin Klein provided a theory that identifies the characteristics of disputes selected for trial. Their theory identifies the cases that are likely to produce the greatest disagreement between disputing sides, thereby resulting in trial. In these circumstances, the plaintiff’s case is very close to the decision standard. The decision standard is the point at which the facts and evidence is just enough for the plaintiff to prevail at trial. When cases strongly favor or disfavor the plaintiff, the parties may disagree about how strong or how weak the case is, but they do not disagree about who is going to prevail. By contrast, when a case is close to the decision standard, even small differences in the parties’ respective evaluations of the case could result in radically different beliefs about the case prospects. Therefore, if disputes are evenly distributed, then the Priest-Klein theory predicts that the vast majority of cases settle, and that non-settling cases are “close” cases.

The Priest-Klein theory has been controversial and is most famous for its fifty percent result. This theory predicts that as the uncertainty about the likely trial result approaches zero, plaintiffs win fifty percent of fully adjudicated cases. The apparent failure of this prediction has obscured the strength of the Priest-Klein theory. Several studies have examined trial data for evidence of a fifty percent result. The data did not support said result. Instead, the data suggested that plaintiff win rates vary significantly by case type. However, this data does not invalidate the Priest-Klein theory. Rather,
the data accentuates the importance of uncertainty in determining trial rates. Researchers have shown that when there is uncertainty, a non-zero standard deviation exists around each side's beliefs, resulting in win rates that are inconsistent with the Priest-Klein selection hypothesis. With positive uncertainty, there can be systematic biases due to case type or asymmetric information that causes win rates to diverge from fifty percent.

Researchers have identified an optimism bias for which the sum of each side’s probabilistic beliefs about their respective prospects exceeds one. In such cases, one or both sides may be overly optimistic about the trial outcome, preventing the parties from identifying a mutually acceptable settlement. Uncertainty about trial prospects exacerbates these biases.

There are several types of litigation-related uncertainty, all of which result in welfare losses. With increased uncertainty, the precision of the parties’ beliefs around the decision standard declines, thereby increasing the probability that the parties disagree significantly about the plaintiff’s prospects. The implication of this increased probability is that a greater proportion of disputes go to trial than when the parties’ beliefs are more precise. Factual uncertainty produces a situation of asymmetric information. For example, Hylton suggests that the defendant in certain tort actions may have superior knowledge about his level of care. Therefore, said defendant has a more precise estimate of his prospects than does the plaintiff. There may also be legal uncertainty when there is an uncertain legal standard. Uncertainty about the degree of proof required, the nature of the legal standard, or how the legal standard applies to known facts are sources of legal uncertainty.


260. See Thomas Selection, supra note 259, at 225.
261. See Hylton, supra note 259, at 188; Shavell, supra note 259, at 500.
263. See Thomas Selection, supra note 259, at 225.
264. Hylton, supra note 259, at 188-89.
265. See id. at 199.
266. See Siegelman & Waldfogel, supra note 248, at 101.
2. Legal Uncertainty and Patent Law

Legal uncertainty is prevalent in patent disputes due to the highly technical subject matter and the complex decisions patent law requires of examiners and judges. In patent claims and patent validity disputes, the parties typically agree about relevant facts, but may disagree about whether the accepted facts support the plaintiff's claims. For example, in *Markman v. Westview Instruments, Inc.*, the Supreme Court allocated the responsibility of patent claim construction to judges because, "construction of written instruments is one of those things that judges often do and are likely to do better than jurors unburdened by training in exegesis." Theoretically, the expertise of judges in interpreting technical documents would increase the accuracy of claim construction. However, empirical studies have shown that appellate reversal rates for claim-construction disputes are quite high, ranging between twenty-nine percent and thirty-three percent. Moreover, if we include claim-construction judgments modified on appeal among the set of imperfectly decided cases, then appellate courts changed a whopping forty-four percent of patent claim-construction appeals. Thus, data does not support the hypothesis that judges have an advantage in interpreting technical documents.

These high reversal and modification rates show that even with judge-rendered decisions, error rates and uncertainty are unusually high. Empirical research suggests that judges and juries tend to agree on fact situations 68% of the time. In addition, Clermont and Eisenberg show that federal courts affirm district courts 80% of the time. By contrast, claim-construction-affirmance rates were roughly 60%, a substantial deviation from the overall 80% federal court reversal rate.

---


268. *Id.* at 388.

269. See *Wallace, supra* note 243, at 1390.


271. See *Chu, supra* note 270, at 1104.


273. *Id.* at 152 (attributing the deviation from the expected fifty percent rate to the low cost of appealing a trial judgment, thereby indicating that appellate cases are drawn randomly from all fully adjudicated trial cases rather than drawing solely from the subset of closely decided cases).

274. See *Wallace, supra* note 243, at 1391.
affirmance rate that Clermont and Eisenberg observed. This deviation of claim-construction-affirmance rates from overall federal court affirmance rates can be attributed to uncertainty, high judicial discretion, and complex subject matter.

Judges as well as commentators have recognized the connection between high judicial discretion, case complexity, and uncertainty. For example, in Festo Corp. v. Shoketsu Kinzoku Kogyo Kabushika Co. (Festo I), the CAFC attempted to balance the tradeoff between judicial discretion and certainty in ruling on the complex question of the application of prosecution history estoppel. Specifically, the CAFC determined how the PTO should apply prosecution history estoppel in infringement cases based on the doctrine of equivalents.

Traditionally, the application of prosecution history estoppel in cases alleging infringement by equivalent products has required courts to make complex, subjective determinations. The doctrine of equivalents allows patent holders to obtain recovery for infringement when the defendant's product deviates from the patent's exact claim specifications. To avoid injustice, courts allow recovery under the doctrine of equivalents when the deviation is insignificant because, for all intents and purposes, the defendant is practicing the patent with his product. In such situations, the defendant should not escape infringement liability simply by replacing a chemical with one possessing similar properties or by changing materials in a mechanical device when this change does not improve or alter the device's performance.

However, patent holders, in negotiating with examiners, sometimes alter or limit claims in order to obtain approval for their patents. Prosecution history estoppel prevents a patent holder from asserting that a defendant's product is equivalent to that of the patent holder when the allegedly infringing characteristics fall within the scope of claims that the patent holder voluntarily surrendered during

---

275. Clermont & Eisenberg, supra note 272, at 151-52.
276. See Cybor Corp., 138 F.3d 1448, 1476 (Fed. Cir. 1998) (Rader, J., dissenting); Wallace, supra note 243, at 1396.
278. Id.
279. Id. at 563.
280. Id. at 564 (explaining that "[t]he doctrine of equivalents prevents an accused infringer from avoiding liability for infringement by changing only minor or insubstantial details of a claimed invention while retaining the invention's essential identity.").
patent prosecution. Determining when the prosecution history estops a plaintiff from asserting infringement is a highly unpredictable task that courts must perform.

The question that the CAFC addressed in *Festo I* was whether the existence of any patentability-related prosecution-history concessions should completely bar infringement allegations based on equivalents. The pre-existing rule required courts to determine whether the scope of patent prosecution concessions encompassed the subject matter of an allegedly infringing product (flexible bar). Under the flexible bar, the court determines whether prosecution history provides evidence that the patent holder surrendered the right to claim exclusivity over related creations that are the subject matter of a patent infringement claim. Frequently, the prosecution record does not reveal the rationale for amendments or whether the patent examiner's concerns would have been allayed with a more limited concession. The absence of a complete record clouds this determination with uncertainty. Without being able to ascertain specifically what subject matter the patent holder may have yielded, the court's decision about the scope of surrender is anything but deterministic. As a result, the decisions in such cases are highly unpredictable.

In *Festo*, the CAFC recognized the uncertainty inherent in such subjective determinations and therefore opted to remove judicial discretion by imposing a complete bar in prosecution history estoppel cases. The CAFC noted that, "it is virtually impossible to predict before the decision on appeal where the line of surrender is drawn." The CAFC considered the degree of uncertainty created under the flexible bar to be so great that consistent results and marketplace guidance were highly limited. Defining consistency and guidance

---

282. *See* Pharmacia & Upjohn Co. v. Mylan Pharms., Inc., 170 F.3d 1373, 1376 (Fed. Cir. 1999) (explaining that "[p]rosecution history estoppel precludes a patentee from obtaining under the doctrine of equivalents coverage of subject matter that has been relinquished during the prosecution of its patent application.").


284. *Festo*, 234 F.3d at 572.

285. *Id.* at 573.

286. *Id.* at 574 (asserting that "[t]here is uncertainty as to the extent of the surrender that will be held to exist.").

287. *Id.* at 577.

288. *Id.*

289. *Id.* at 574.

290. *Id.* at 575.

291. *Id.*
to be critical characteristics of “workable” rules, the court concluded that, “[the flexible bar rule’s] ‘workability’ is flawed.”\textsuperscript{292} The CAFC feared that the flexible bar created a disincentive to innovation because,

\[][I]t is impossible \ldots for the public or the patentee to determine the precise range of equivalents available under the flexible bar approach. This creates a “zone of uncertainty which enterprise and experimentation may enter only at the risk of infringement claims \ldots [and which] discourages invention only a little less than unequivocal foreclosure of the field.”\textsuperscript{293}\]

The CAFC believed that eliminating this uncertainty would not only reduce excessive litigation, but would significantly reduce the disincentives to innovation associated with the flexible bar rule.\textsuperscript{294} The complete bar would give the public notice of the patent’s boundaries, thereby freeing innovators to improve existing technologies without fearing allegations of infringement under the doctrine of equivalents.\textsuperscript{295} The CAFC believed that removing this uncertainty would also stimulate investment in innovation.\textsuperscript{296} A complete bar eliminates the implicit tax associated with the uncertainty of not knowing whether the courts would consider a potential innovation infringing under the doctrine of equivalents.\textsuperscript{297} Thus, developers would find certain projects that have a negative expected return under the flexible bar to have a positive expected return under a complete bar.\textsuperscript{298} Concluding that the benefits to patent holders of a flexible bar did not offset the substantial benefits to society of a complete bar, the CAFC made a utilitarian decision in favor of a complete bar.\textsuperscript{299}

The CAFC’s \textit{Festo} rejection of the flexible bar rule did not last long. On appeal, the Supreme Court reversed \textit{Festo I} in \textit{Festo II}.\textsuperscript{300} The Court rejected the CAFC’s conclusion that reducing uncertainty

\begin{footnotesize}
\begin{enumerate}
\item \textit{Id.}
\item \textit{Id.} at 577 (alteration in original).
\item \textit{Id.} at 576. (asserting that “[t]echnological advances that would have lain in the unknown, undefined zone around the literal terms of a narrowed claim under the flexible bar approach will not go wasted and undeveloped due to fear of litigation.”).
\item \textit{Id.} at 576.
\item \textit{Id.} at 577.
\item \textit{Id.}
\item \textit{See id.}
\item \textit{Id.} at 578.
\item \textit{Id.}
\end{enumerate}
\end{footnotesize}
outweighs the benefits to patent holders of applying a flexible bar.\textsuperscript{301} While acknowledging the CAFC's concerns about the negative impact of uncertainty, the Court stressed that elevated uncertainty is the cost of "ensuring the appropriate incentives for innovation."\textsuperscript{302} In particular, the Court was concerned that the abrupt rejection of the flexible bar rule unacceptably disrupted "the settled expectations of the inventing community."\textsuperscript{303} The Court considered the CAFC's decision to be a gross violation of stare decisis. The switch to a complete bar would "risk destroying the legitimate expectations of inventors in their property" based on multiple years of patent activity under the flexible bar rule.\textsuperscript{304} The switch would disrupt business investments and patent strategies created in reliance on this rule.\textsuperscript{305} The Court was unwilling to allow such a profound change in patent policy solely for the purpose of increasing certainty.\textsuperscript{306} Thus, in the Court's calculus, protecting settled expectations and patent investment decisions trumped any gains arising from increased certainty associated with the CAFC's dramatic rule change.

The Supreme Court's \textit{Festo II} decision illustrates that patent reforms must be reviewed for their impact on efficiency, legal uncertainty, and settled expectations. Any gains from implementing efficient patent policy may be nullified by losses resulting from increased litigation due to uncertainty. In addition, the Court made it clear that stare decisis provides an additional constraint to policy alternatives.\textsuperscript{307} The Court is unlikely to validate any judicial-based reforms that deviate dramatically from settled law. Therefore, before evaluating the efficacy of software patent reform recommendations, this paper reviews the current software patent case law to determine exactly what settled law allows.

IV. ADDRESSING THE SOFTWARE PATENT PROBLEM

The objective in reforming software patent policy must include enhancing both social welfare and legal certainty. Any gains in social welfare due to policy changes may be offset by increased legal uncertainty in the handling of software patent disputes. In addition,
gains due to policy changes must not severely violate the principle of stare decisis. Changing the rules after firms have invested in reliance on current law can result in substantial economic losses as well as a loss of faith in the legal system.\textsuperscript{308}

The following section considers the effectiveness of judicial policy in addressing the problem of software patents. Therefore, popular recommendations that require legislative action such as reducing software patents terms and \textit{sui generis} solutions are not a consideration.\textsuperscript{309}

\textbf{A. Are Policy Levers the Answer to Software Patent Problems?}

The judiciary is in the best position to effect change. Some commentators have recommended the use of judicial “policy levers” to enhance the efficiency of various patent types.\textsuperscript{310} Policy levers are essentially approaches that courts apply to certain cases or fact situations to advance policy objectives, such as enhancing efficiency.\textsuperscript{311} Policy levers fall within the bounds of a court’s discretion and can be used to tailor the application of precedent and legislation to specific industries.\textsuperscript{312} Burk and Lemley believe that the policy lever approach is particularly appropriate for ameliorating patent approval and infringement issues because innovation is heterogeneous.\textsuperscript{313} According to Burk and Lemley, the CAFC is in the best position to shape patent law to most effectively deal with the issues that arise from the diversity of modern innovation.\textsuperscript{314} However, this recommendation by no means precludes the Supreme Court from exerting its authority in customizing judicial patent policy.\textsuperscript{315}

Burk and Lemley criticize the CAFC for incorrectly applying industry-specific patent policy levers.\textsuperscript{316} Software innovation is incremental, relatively certain, and low in cost. Therefore, patent scope should be limited and bounded both generationally and by subject matter.\textsuperscript{317} To effectuate these policy objectives, Burk and

\begin{itemize}
  \item \textsuperscript{308} See \textit{id.} at 730-31.
  \item \textsuperscript{309} See Thomas & DiMatteo, \textit{supra} note 5, at 42-44; Samuelson, \textit{supra} note 5, at 1135.
  \item \textsuperscript{310} Burk & Lemley, \textit{supra} note 138.
  \item \textsuperscript{311} \textit{id.} at 1579.
  \item \textsuperscript{312} \textit{id.}
  \item \textsuperscript{313} Id.
  \item \textsuperscript{314} Id.
  \item \textsuperscript{315} See, e.g., \textit{KSR Int’l Co. v. Teleflex Inc.}, 127 S.Ct. 1727 (2007) (requiring the application of a stricter standard in making obviousness determinations).
  \item \textsuperscript{316} Burk & Lemley, \textit{supra} note 138, at 1579.
  \item \textsuperscript{317} \textit{id.} at 1688-89.
\end{itemize}
Lemley recommend the CAFC: (i) tailor the obviousness requirement to facilitate patent awards for limited incremental improvements, (ii) bolster the software disclosure requirements for best mode and enablement purposes, (iii) limit the expansive coverage of the doctrine of equivalents, and (iv) recognize a software reverse engineering exception.318

1. Increased Disclosure for Software Patents is Problematic

Burk and Lemley are critical of the CAFC’s failure to apply available policy levers in the manner they recommend.319 However, given the constraints the CAFC faces, the nature of software innovation, and the CAFC’s pro-software-patent bias,320 the CAFC’s approach is completely logical. First, the Supreme Court, in *KSR International Co. v. Teleflex Inc.*,321 raised the bar for satisfying the non-obviousness requirement, making it more difficult to use the non-obviousness test as a policy lever to increase software patent claim approval rates.322 Secondly, increasing software disclosure requirements will reduce—perhaps dramatically—the number of successful patent claims and introduce unintended consequences. Specifically, increasing disclosure requirements for software claims reduces the value of software patents and increases legal uncertainty.

For software patent claims, the CAFC relaxed the Patent Act disclosure requirements. Patent law requires a patent application to contain sufficient detail so that a person of ordinary skill in the relevant art can replicate the invention.323 For software, providing source code or detailed outlines of how the code performs its functions would satisfy the literal meaning of the enablement and best mode requirements.324 However, the CAFC has ruled that software claims do not have to provide this level of detail.325 Instead, disclosing functions alone is enough to satisfy §112 demands.326 For

318. *Id.* at 1689-91.
319. *See id.* at 1650.
320. *See supra* notes 90-129 and accompanying text.
322. *Id.* at 1739 (2007).
324. The term “function” here refers to an invention’s capabilities. The usage here should not be confused with previous reference to functions as components of a software program. *See supra* notes 204-211 and accompanying text.
326. *See Cohen & Lemley, supra* note 132, at 24 n.87.
most patent claims, patent law provides protection solely for the means of achieving identified functions (the "means-plus-function" test).\textsuperscript{327} Previously, courts considered functions to be unpatentable subject matter like ideas and algorithms.\textsuperscript{328} Courts agreed that patents were appropriate for protecting innovative ways of accomplishing a task, but patent claims based solely on functions were inappropriate.\textsuperscript{329} Patents protected the identified means, leaving other innovators the freedom to develop new methods of accomplishing the same tasks and functions without infringing related patents.\textsuperscript{330} By contrast, the lax disclosure requirements for software patents allow a software patent holder to gain coverage over technology that extends beyond the patent boundaries.\textsuperscript{331}

However, requiring software patents to meet the means-plus-function criteria is problematic. Source code appears to be the quintessential way of meeting this patentability test. However, such submissions would have limited efficacy.\textsuperscript{332} There may be thousands of ways to achieve the same programming result. Programmers may use different routines, approaches and languages to accomplish the same programming task.\textsuperscript{333} Asking multiple programmers to write a program to accomplish a stated task will certainly produce multiple programs, many of which differ in significant ways.\textsuperscript{334} Many of these programs will achieve the desired functionality using different means. Therefore, instituting a means-plus-function test for software by limiting software patent scope to the four corners of source code or a

\begin{footnotes}
\item[327] See Fidel D. Nwamu, Does Your Claim Conform to Means-Plus-Function Format Under Section 112, Paragraph Six?: O.I. Corp. v. Tekmar Co., 6 J. INTELL. PROP. L. 189, 194-95 (1999) (observing that means-plus-function claims while satisfying § 112 requirements do so at the expense of limiting patent claim scope to the designated structures or acts identified in the claim).
\item[328] See Corning v. Burden, 56 U.S. 252, 268 (1854) ("[I]t is well settled that a man cannot have a patent for the function or abstract effect of a machine, but only for the machine which produces it.").
\item[329] See Ford, supra note 4, at 60 n.59 (citing cases that find purely functional claims deficient).
\item[330] See id.
\item[331] See Cohen & Lemley, supra note 132, at 25-26.
\item[332] See Fonar Corp. v. Gen. Elec. Co., 107 F.3d 1543, 1548-49 (Fed. Cir. 1997) (quoting an expert witness testifying that source code "wouldn't help someone... anyway because [the code is machine specific]. What's much more important is to have a description of what the software has to do.").
\item[333] See KLEMENS, supra note 34, at 43.
\item[334] Id. ("In view of the astounding number of choices available [when completing the same computer task], the two programmers' solutions could be vastly different.").
\end{footnotes}
detailed description of the program gives the patent holder very little value.

In addition, the application of a means-plus-function test to software source code would dramatically elevate legal uncertainty by requiring examiners and judges to make difficult discretionary judgments. Source code can be written in thousands of different programming languages. It is unlikely that PTO examiners would be conversant in more than a handful of these programming languages. Examiners would have to make subjective judgments in comparing source code written in different languages using different programming styles. The PTO and courts would have to either accept small variations in source code as non-infringing or engage in the arduous exercise of judging whether such code variations are sufficiently different to be non-infringing.

Even allowing submission of information other than source code to meet the increased disclosure requirements would fuel legal uncertainty. Patent examiners and judges would still have to use submitted information to distinguish between patentable software means and unpatentable functions. This complex calculus would introduce ambiguity that would make the determination of software patentability subjective and highly uncertain. This ambiguity would result in increased litigation due to uncertainty about software patent validity. Therefore, the error rates associated with software patent prosecutions would increase. The increased error rates, combined with the ambiguity and uncertainty accompanying examiners’ subjective determinations, would fuel conflict and litigation. Thus, increased disclosure would exacerbate rather than reduce welfare losses.

While dealing with the software disclosure issue, the CAFC opted to grant software patents for entire functions, independent of any identified means. It follows that comparatively fewer software


336. There are two types of error: rejecting a patent claim covering a patentable innovation (type I error), and approving a patent claim covering an innovation just should not be patented (type II error). See Richard J. Larsen & Morris L. Marx, An Introduction to Mathematical Statistics and Its Applications 299 (2d ed. 1986).

337. See Wallace, supra note 243, at 1384 ("[U]ncertainty at the trial level is inefficient because it stimulates appeals rather than settlements.").

338. See Fonar Corp. v. Gen. Elec. Co., 107 F.3d 1543, 1549 (Fed. Cir. 1997) ("[W]here software constitutes part of a best mode of carrying out an invention, description of such a best mode is satisfied by a disclosure of the functions of the software.").
patents are granted because such grants of exclusivity are by definition broad. Software patents cover all means of achieving a function, therefore no room exists for multiple patents covering differing methodologies for a given functional outcome. The CAFC had little choice in radically departing from disclosure and enablement requirements. The CAFC's dilemma is that there is no middle ground. Policy makers and courts can either grant minimal patent protection for specific methods that convey no valuable exclusivity or grant patents that essentially cover all ways of accomplishing identified tasks.

In dealing with this dilemma, the CAFC opted to provide software developers with valuable patents. This choice is not surprising. Since dismantling the physical-transformation test, the CCPA and the CAFC cases have been highly supportive of, and responsible for, the expanded scope of software patentability. Once committed to the proposition that software innovation is valid patent subject matter, the relaxation of software patent claim disclosure requirements was inevitable. More germane to this discussion, the negative impact of increased software patent claim disclosure on legal uncertainty makes increasing disclosure for such claims an inferior policy lever.

2. The Doctrine of Equivalents and Reverse Engineering

Using the doctrine of equivalents as a policy lever for software innovation is likely to yield more positive results. However, it is unlikely to fully address the software disclosure dilemma. Cohen and Lemley urge the courts to use the doctrine of equivalents to modify software patent scope. Effectively employing this policy lever would increase efficiency by reducing software patent scope. However, narrowing patent scope in this way requires judges to employ substantial subjectivity in rendering decisions. Uncertainty associated with this high level of discretion is likely to spawn significant litigation. Nevertheless, judges already wield significant discretion in

339. See Burk & Lemley, supra note 138, at 1688.
340. Fonar Corp. v. Gen. Elec. Co., 107 F.3d at 1549 ("As a general rule, where software constitutes part of a best mode of carrying out an invention, description of such a best mode is satisfied by a disclosure of the functions of the software."); In re Hayes Microcomputer Prods., Inc., 982 F.2d 1527, 1537-39 (Fed.Cir. 1992) (concluding that there was no best mode violation where detailed specifications of microprocessor firmware were excluded but provided information was sufficient to allow one skilled in the art to write firmware with the functionality of the missing firmware).
341. See supra notes 90-129 and accompanying text.
deciding doctrine of equivalents cases. Hence, it is unclear whether narrowing software patent scope for these cases increases uncertainty. Yet, it is possible that the transition to the new, stricter standard is a disequilibrium condition. Once the courts have adopted a new, stricter standard, uncertainty may decline to pre-transition levels or lower. Thus, narrowing software patent scope in doctrine-of-equivalents cases is likely a beneficial step.

Carving out a reverse-engineering exception would enhance efficiency as well. The analysis in Part III A demonstrated that software patents reduce innovation by raising the costs of producing similar non-infringing products. Cohen and Lemley argue that reverse engineering can provide a competitor with knowledge needed to produce non-infringing products. It may be difficult for software developers to determine the limits of patent-protected functions that are incorporated into larger, complex software programs. Developers need to deconstruct software programs to gain access to “unprotected elements.” Such knowledge would allow developers to avoid producing infringing products, reducing both development costs and legal uncertainty because there would be less ambiguity about whether competitive products infringe.

The Burk and Lemley policy-lever recommendations are a mixed bag. Using non-obviousness and software patents claim disclosure as recommended would likely diminish, rather than increase, social welfare. The doctrine of equivalents and reverse-engineering policy levers have a moderating effect, but only address a subset of software patent issues. The doctrine of equivalents reform would not impact literal infringement claims. In addition, if after reverse engineering a program, a developer discovers that the functionality it needs to create a new product is subject to a patent, then his path is still barred. Therefore, policy levers are only partially ameliorative.

B. Reinstating the Physical-Transformation Test

The Supreme Court’s physical-transformation test is the best of the judicial branch’s limited alternatives. The physical-transformation test enhances social welfare and reduces legal uncertainty, but at the

---

343. See supra notes 232-235 and accompanying text.
344. Cohen & Lemley, supra note 132, at 29.
345. Id. at 6.
expense, perhaps, of settled expectations. Strict application of the test will dramatically reduce the issuance of weak software and business method patents, which, rather than spur innovation, deter competitors from innovating.\textsuperscript{346}

The test reduces legal uncertainty because it requires the PTO to approve patent claims only when they include a physical, tangible result and rejects patent claims that solely manipulate data or that solely produce visual or audio output. Reiteration of the principles advanced in the Supreme Court software precedents would provide clear identification of the boundaries of software patent subject matter.\textsuperscript{347} Process patent claims that accept data input from an EKG or output a data analysis to a computer monitor or chart, as in Arrhythmia Research, would not satisfy the physical-transformation test. Similarly, programs that manipulate digital audio, photographic, and video data would not qualify as patent subject matter under the physical-transformation test.

The most serious challenge to reinstating the physical-transformation test is the potential harm to the settled expectations of market players. In Festo II, the Supreme Court was clear that the benefits from increased certainty did not offset the damage to existing patent holders' expectations of how courts would handle their infringement claims.\textsuperscript{348} Changing the rule would be unfair because patent claimants may have chosen not to concede examiner claim challenges had they known that such concessions would result in a subsequent bar.\textsuperscript{349} In essence, Festo I could result in a taking of patent rights without such holders having an opportunity to contest their losses.

There are significant differences that distinguish software patents from the main issue decided in Festo II. There, the Supreme Court expressed agitation that the CAFC had revisited an issue and upset the balance that the Court previously had struck in Warner-Jenkinson Co. v. Hilton Davis Chemical Co.\textsuperscript{350} The CAFC's embrace of a complete bar amounted to a repudiation of the Supreme Court's flexible bar standard in Warner-Jenkinson. For software patents, the reverse occurred. The CAFC's embrace of an unrestricted standard for

\textsuperscript{346} See supra notes 232-235 and accompanying text.

\textsuperscript{347} See supra Part II.


\textsuperscript{349} Id.

software patents is a repudiation of the balance embodied in the Court's physical-transformation test. Thus, reinstating the physical-transformation test would restore the balance of interests that the Court struck in its software patent cases.

In changing the treatment of software patents, the Court can avoid disturbing "settled expectations" by differentiating between existing patents and prospective claims. In *Festo II*, the expectations issue concerned preserving the prospective value of patents. Many patent holders who had made seemingly minor concessions during patent prosecution would lose all rights to challenge equivalents under the *Festo I* complete bar. Judge Linn's dissent in *Festo I* raised strong concerns that adoption of a complete bar would have a retroactive impact rather than solely a prospective toll.

Reinstating the Supreme Court's physical-transformation test for process claims need not produce an analogous harm to existing patent holders. If the Court chooses to apply the physical-transformation test prospectively, then only future software patent claims would need to satisfy that standard. Existing software and business method patents would be grandfathered, leaving the expectations of existing patent holders intact. Although there arguably could be losses from software investments in anticipation of receiving a patent, such losses would not be comparable to those anticipated in *Festo II*. As argued in Part III, software patent R&D is much less resource exhaustive and less dependent on patent protection than other types of innovation. Therefore, disturbing settled expectations due to reinstatement of the physical-transformation test for software patent claims is not comparable to the corresponding issue in *Festo II*.

V. CONCLUSIONS

Cohen and Lemley's comment that the Court has "assiduously avoided" addressing software patents for over twenty-five years is accurate. However, there is evidence that the Court is awake and aware of the morass that engulfs software and business method patents. Innovation is not homogeneous, and therefore, a single patent policy is not appropriate for all innovation types. Although strong

353. *Id.* at 638 (Linn, J., dissenting).
354. *See supra* notes 199-210 and accompanying text.
patent protection may be necessary for high-cost innovation such as pharmaceuticals, applying such policy to low-cost software and business method inventions impedes rather than promotes innovation.

With Congress unlikely to address this issue and the CAFC biased in favor of software and business method patents, it is imperative for the Supreme Court to fine-tune patent policy for software and business method patents. If the Court does intervene, the only effective policy lever is reinstatement of its physical-transformation test. The Burk and Lemley policy levers do not go far enough in ameliorating this patent malaise. In addition, increasing software patent disclosure requirements would likely aggravate the problem. The advantage of revitalizing the physical-transformation test and applying it prospectively is that it would promote software innovation and reduce software patent litigation, while simultaneously protecting the interests of existing software patent holders. Without intervention of this type, the problems associated with software and business method patents will continue to plague the patent system for many years to come.