1-1-2006

Rationalizing Software Patents: Suggestions for a Livable System

Shane Glynn

Follow this and additional works at: https://repository.uchastings.edu/hastings_comm_ent_law_journal

Part of the Communications Law Commons, Entertainment, Arts, and Sports Law Commons, and the Intellectual Property Law Commons

Recommended Citation
Available at: https://repository.uchastings.edu/hastings_comm_ent_law_journal/vol28/iss2/4

This Note is brought to you for free and open access by the Law Journals at UC Hastings Scholarship Repository. It has been accepted for inclusion in Hastings Communications and Entertainment Law Journal by an authorized editor of UC Hastings Scholarship Repository. For more information, please contact wangangela@uchastings.edu.
Rationalizing Software Patents: Suggestions for a Livable System

by SHANE GLYNN*

I. The Problem with Software Patents .............................................................. 287
II. Legal Developments In Software Patents .................................................. 289
   A. Constitutional and Statutory Definition of Patentable Material ............... 289
   B. Case Law ................................................................................................. 291
      2. The Diehr Case .................................................................................. 294
      3. From Diehr to Alappat ....................................................................... 296
      4. State Street Bank and AT&T .............................................................. 298
   C. Conclusion ............................................................................................... 300
III. The USPTO Guidelines for Software Patents ............................................. 301
   A. The USPTO Guidelines ......................................................................... 301
   B. Conclusion ............................................................................................... 302
IV. Software Patents in the Wild ..................................................................... 303
   A. Should Software be Patentable? .............................................................. 303
   B. Prosecution and Litigation of “One-Click” .............................................. 304
   C. The Problem is the Process ................................................................... 305
V. Making the System More Efficient ............................................................... 305
   A. Effective Prior Art Searches by Examiners Skilled in the Art ................. 306
   B. Publication and Third Party Interference ................................................. 307
   C. Stevens’ Diehr Dissent ......................................................................... 308
VI. Conclusion .................................................................................................. 308

I. The Problem with Software Patents

In October 2004, a federal jury found that Sun Microsystems' Java programming language violated a patent owned by the Eastman Kodak Corporation.¹ This decision was notable for two reasons: first, it was a high-profile patent case between large corporations that

---

¹ University of California, Hastings College of the Law, Juris Doctor Candidate, 2006. The author would like to thank Julie A. Glynn and Michael Wells for their support during this project.

involved a potential billion-dollar damages award\(^2\) and second, this case involved a software patent, a new and relatively controversial form of intellectual property.\(^3\) Kodak purchased the patent at issue from Wang Laboratories Inc. in 1997. The patent covered a method by which one computer program can "ask for help" from another program.\(^4\) Although Sun Microsystems subsequently settled the case for $92 million\(^5\) before the jury could decide on a damages award, the case can be seen as a warning sign for intellectual property and patent law jurisprudence that there are flaws in the assignment and adjudication of software patents in the United States.

To say that there is a problem with the current US software patent system is relatively uncontroversial.\(^6\) However, many commentators have overlooked that the US patent system is a system, comprised of government agencies, patent agents and attorneys, patent holders, and federal courts. In order to determine where there is a problem with the software patent system, the parts of the system need to be individually examined. As explained below, while most of the focus has been on the role of courts, there are opportunities for significant improvement of the overall system through changes in the prosecution of software patents.

Part II of this note will examine the historical development of software as patentable subject matter in the federal courts, from the "progress of Science and useful Arts" language of the Constitution\(^7\) to modern electronic computing and data storage devices through federal court decisions from *Diamond v. Deihr*\(^8\) to *State Street Bank & Trust Co. v. Signature Financial Group, Inc.*\(^9\) Part III will discuss how the jurisprudence of the Supreme Court and the Court of Appeals for the Federal Circuit (Federal Circuit) has been interpreted at the

---

2. Id.
9. 149 F.3d 1368 (Fed. Cir. 1998).
United States Patent and Trademark Office (USPTO) and incorporated into the "Examination Guidelines for Computer Related Inventions." Part IV will discuss the legal and business application of these rules in the real world. Finally, Part V will discuss the problems associated with the current software patent process and propose modifications to the procedure for grant and review of software patents that will place an emphasis on early detection of prior art in prosecution before the USPTO.

II. Legal Developments in Software Patents

A. Constitutional and Statutory Definition of Patentable Matter

The origin of patent legislation is the United States Constitution, which directs Congress to "Promote the Progress of Science and Useful arts, by securing for limited Times to Authors and Inventors the exclusive Right to their respective Writings and Discoveries." This guarantee of a limited monopoly for inventors followed the established common law tradition of rewarding inventors with exclusive rights to the manufacture and sale of their inventions for limited times in exchange for publishing the details of their inventions. Publication of patents serves two goals: it places new inventions into public knowledge where other inventors can make use of current inventions as inspiration for future inventions, and publication provides notice to other inventors of what is protected. As part of a patent application, the inventor is required to disclose both a general description of the invention and the "best mode" implementation of the invention as conceived by the inventor. In exchange for public disclosure, the inventor is granted a limited monopoly over the manufacture, use, sale, and importation of the invention.

The first Congress codified United States patent law in the Patent Act of 1790 which laid the groundwork for all subsequent

---


12. The English Statute of Monopolies of 1623, 21 Jac. 1, c. 3, §6, provided a 14-year monopoly to the inventor of "any manner of new manufactures."

13. Of the two goals, the notice function presents a significant problem with respect to current software patent scheme. See discussion infra Part IV.


Although there are some notable differences between the 1970 Act and current patent law, many features of modern patent law are retained from the Act of 1790. Patents were to be granted on any "useful art, manufacture, engine, machine, or device, or any improvement therein not before known or used," and applicants were required to submit a specification containing an explanation and model of each invention that provides sufficient detail to distinguish the new invention from the prior art and permit one skilled in the art to create the new invention.

Congress has recodified the patent law several times since 1790, but the basic system of granting a limited monopoly to inventors of novel, useful, inventions has remained unchanged. With one minor exception, the definition of patentable subject matter has remained unchanged since 1793—"any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof." In addition to the statutory definition of patentable matter, the other major limitations on patenting a new invention relevant to this Note are that an invention must be novel and non-obvious.

The next section will discuss the application of the above described statutory patent law to software patents. It is worth noting, however, that without any additional information computer software appear to be patentable. A software-controlled machine appears to fit
the definition of a "new and useful . . . machine" and software itself likewise appears to fit the definition of a "new and useful process." The Supreme Court and the Federal Circuit, in a thirty-year progression from *Benson* to *State Street Bank*, eventually arrived at this conclusion.

B. Case Law

1. *Benson, Flook, and the Freeman-Walter-Abele test*

   The Supreme Court has, until relatively recently, been skeptical of claims involving software or computer algorithms. *Benson* was one of the first Supreme Court cases that dealt with an attempt to patent a software algorithm. The respondents in *Benson* attempted to patent a computer algorithm used to convert numbers from binary coded decimal (BCD) notation into pure binary numbers. The USPTO rejected claims directed to the BCD-to-pure-binary computer process, but these claims were reinstated by the Court of Customs and Patent Appeals.

   Justice Douglas began the *Benson* opinion by noting "while a scientific truth, or the mathematical expression of it, is not a patentable invention, a novel and useful structure created with the aid of knowledge of scientific truth may be." Douglas then reviewed the history of process patents and the gradual expansion of patentable subject matter between 1854 and 1972. In rejecting the process claims, Douglas wrote that upholding the process claims would have the "practical effect" of patenting an idea and "the patent would wholly pre-empt the mathematical formula and in practical effect would be a patent on the algorithm itself." The opinion questioned the validity of software patents in general, and noted several practical difficulties with prosecuting software patents.

---

26. *Id.*
29. *Benson*, 409 U.S. at 64.
30. *Id.* at 65-66.
33. *Id.* at 68-71.
34. *Id.* at 71-72.
35. "It may be that the patent laws should be extended to cover these programs, a
After Benson, there was considerable confusion about what types of software patents would be allowed. After Benson, there was considerable confusion about what types of software patents would be allowed. The Benson opinion stated "program" patents were not unpatentable, but did not provide guidance as to what would constitute a patentable software invention. The only firm rule that emerged from Benson was that a patent involving a "mathematical formula" that had the "practical effect" of patenting the algorithm itself was not permitted.

In 1978, the C.C.P.A. used the above interpretation of Benson in its analysis of a claim to a computer process that would automatically update an alarm limit in a chemical hydrocarbon conversion process. The C.C.P.A. phrased the question at issue in Flook as "whether a claim to a process which uses an algorithm to modify a conventional manufacturing system is statutory subject matter." In answering this question in the affirmative, the C.C.P.A. first interpreted its own decision in Christiansen as being limited to cases in which the process of solving the algorithm itself is the claimed invention and nothing is done after solution of the algorithm. The C.C.P.A. then found that Benson applied only to claims that would "wholly preempt a mathematical formula." The claim at issue in Flook did not implicate Benson, according to the C.C.P.A., because it involved the use of an algorithm in a manufacturing process and "solution of the algorithm, per se, would not infringe the claim.

The Supreme Court reversed the C.C.P.A. in Parker v. Flook. The Court rejected the C.C.P.A.'s narrow interpretation of Benson policy matter to which we are not competent to speak." Id. at 72.

36. Douglas noted that the Patent Office "cannot examine applications for programs because of a lack of classification technique and the requisite search files ... reliable searches would not be feasible ... because of the tremendous volume of prior art," and that without a proper prior art search program patents "would be tantamount to mere registration and the presumption of validity would be all but nonexistent." Id. Thirty-three years later, the USPTO is still grappling with these issues.

37. See Id.
39. Id. at 71-72.
40. Id.
42. Id. at 22.
43. In re Christiansen, 478 F.2d 1392, 1394 (C.C.P.A. 1973) (holding that "a method claim in which the point of novelty is a mathematical equation to be solved as the final step of the method" is not statutory subject matter).
44. See Flook, 559 F.2d at 22.
45. Id. at 23 (quoting Benson, 409 U.S. at 72).
46. Id.
and stated a process claim that "implements a principle in some specific fashion" does not "automatically fall[] within the patentable subject matter of § 101." Flook's claim was nonstatutory under section 101, not because it involved a mathematical algorithm, but because "the application, considered as a whole, contains no patentable invention." The specific chemical process at issue, using variable alarm limits to regulate processes in general, and use of computers to calculate alarm limits in general, were all known in the art. The Court held: 1) use of a computer to regulate the alarm limits in this specific process was not novel, and 2) "if a claim is directed essentially to a method of calculating, using a mathematical formula, even if the solution is for a specific purpose, the claimed method is nonstatutory.

As in Benson, the Court noted that its decision was not a complete bar on software patents. Flook was based "to a large extent" on precedent disfavoring software patents. The Court held open the possibility that software may comprise statutory subject matter, but the decision involves "[d]ifficult questions of policy" that were better addressed by Congress than a court.

The Supreme Court's decisions in Benson and Flook caused the C.C.P.A. to develop what eventually became the Freeman-Walter-Abele test to identify unpatentable mathematical algorithms. Freeman-Walter-Abele is a two part test:

"First, the claim is analyzed to determine whether a mathematical algorithm is directly or indirectly recited. Next, if a mathematical algorithm is found, the claim as a whole is further analyzed to determine whether the algorithm is applied in any manner to a physical elements or process steps, and, if it is, it passes muster under § 101."

48. Id. at 593.
49. Id.
50. Id. at 594.
51. Id.
52. Id. at 594-95. However, note that the Court acknowledges that the claim provided "a new and presumably better method for calculating alarm limit values." Id. Such a claim would likely be statutory under 35 U.S.C. § 101 today.
53. Id. at 595 (quoting In re Richman, 563 F.2d 1026, 1030 (C.C.P.A. 1977)).
54. Id.
55. Id.
56. Id.
57. See In re Freeman, 573 F.2d 1237 (C.C.P.A. 1978); In re Walter, 618 F.2d 758 (C.C.P.A. 1980); In re Abele, 684 F.2d 902 (C.C.P.A. 1982).
The test represented an attempt by the C.C.P.A. to adhere to the Supreme Court's decision in *Flook* while, at the same time, leaving the door open for software patents. *Flook* prohibited patents that involved a calculation as nonstatutory; *Freeman-Walter-Abele* acknowledged this limitation but permitted mathematical algorithm patents that involved the application of an algorithm to a physical process. The test has been criticized as misleading, and ultimately had little relevance due to the Supreme Court's abrupt change of heart regarding software patents in *Diehr*.

2. *The Diehr Case*

*Diamond v. Diehr* is the Supreme Court's last decision involving software patents. Before *Diehr*, the Court had clarified its position on the permitted scope of patents under section 101 in *Diamond v. Chakrabarty*. *Chakrabarty* involved patent claims directed to a genetically modified bacterium. In a reversal of earlier Supreme Court jurisprudence, the *Chakrabarty* Court chided lower courts for "read[ing] into the patent laws limitations and conditions which the legislature had not expressed." The Court identified Congressional intent for section 101 to "include anything under the sun that is made by man." The only exceptions to this broad interpretation of section 101 is for "laws of nature, physical phenomena, and abstract ideas."

The Court applied its interpretation of section 101 directly to a patent "which includes in several of its steps the use of a mathematical formula and a programmed digital computer" in *Diehr*. Diehr's patent included process claims for that covered the use of a computer to monitor the curing of rubber in heated molds. The patent examiner rejected these claims "on the sole ground they were

59. *In re Alappat*, 33 F.3d 1526, 1543-44 (Fed. Cir. 1994) (§ 101 patentability analysis should be directed to a claim in its entirety is statutory subject matter and not to whether part of a claim would be nonstatutory by itself).
62. *Id.* at 305-06.
63. *Id.* at 308 (quoting U.S. v. Dubilier Condenser Corp., 289 U.S. 178, 199 (1933)).
64. *Id.* at 309 (citing S. REP. No. 1979, 82d Cong., 2d Sess., 5 (1952); H.R. REP. No. 1923, 82d Cong., 2d Sess., 6 (1952)).
65. *Id.*
66. *Diehr*, 450 U.S. at 177-78 (claims recited the use of a temperature probe, computer, and the Arrhenius equation to precisely calculate cure times for molded rubber products).
67. *Id.*
drawn to nonstatutory subject matter under [section 101].” The C.C.P.A. reversed in In re Diehr, noting that the claims were to a process and that an otherwise allowable claim is not invalidated because it uses a computer. The Commission of Patents and Trademarks appealed to the Supreme Court on the argument that the C.C.P.A.’s decision was inconsistent with Benson and Flook.

The Court began by noting that the process of converting raw rubber into a cured, finished part is an industrial process “which have historically been eligible to receive the protection of our patent laws.” Agreeing with the Federal Circuit, the Court noted that a process claim is not rendered nonstatutory by “the fact that in several steps of the process a mathematical equation and a programmed digital computer are used.” Turning to its decisions in Benson and Flook, the Court stated that they did not represent a bar on software patents but stood for the “long established principles” that laws of nature, natural phenomena, and abstract ideas are not patentable.

Benson involved a computer algorithm used to solve a type of mathematical equation that “is like a law of nature, which cannot be subject to a patent.” In Flook the court concluded that the claims were drawn only to the formula for calculating an “alarm limit” and not to the use of that limit in a process.

The Court reasoned that Diehr was different, “the respondents here do not seek to patent a mathematical formula ... they seek patent protection for a process of curing synthetic rubber.” The fact that the claimed process used a well-known mathematical equation and a computer did not affect the section 101 analysis. Of particular importance was that Diehr did not seek to patent every use of the Arrhenius equation in a computer, only its use as part of a larger process of curing rubber.

Justices Stevens, Brennan, Marhsall, and Blackmun disputed this conclusion. They argued in dissent that the claims at issue were not a

68. Id. at 179.
70. Id. at 984.
71. Diehr, 450 U.S. at 184.
72. Id. at 185.
73. Id.; see also Diamond v. Chakrabarty, 447 U.S. 303 (1980).
74. Diehr, 450 U.S. at 186.
75. Id. at 186-87.
76. Id. at 187.
77. Id.
78. “[Respondents] seek only to foreclose from others the use of that equation in conjunction with all the other steps in the claimed process.” Id.
process but a computer algorithm for measuring cure time, similar to the algorithm for calculating “alarm limits” rejected in Flook as nonstatutory under section 101.\(^7\) Because the only novel portion of the process, use of the computer, was nonstatutory, the entire claim is invalid under section 101.\(^8\)

3. *From Diehr to Alappat*

Although the implications of *Diehr* were broad, in that for the first time the Supreme Court upheld a patent that included a computer program, the actual holding was rather narrow. *Diehr* did not hold that software was patentable subject matter, only that the inclusion of software in a process claim was not fatal.\(^8\) The precise boundaries of how, and if, software was to be afforded patent protection was left to later cases.

Between 1981 and 1994, the C.C.P.A. (and its successor, the Federal Circuit) applied the *Freeman-Walter-Abele* test\(^2\) to determine the validity of computer-related claims. As a result, patent practitioners developed what one commentator called “the doctrine of magic words,” phrasing any computer- or software-related invention in terms of a tangible object to satisfy the second prong of *Freeman-Walter-Abele*. Although the standard was not entirely clear, the C.C.P.A. and Federal Circuit generally upheld claims involving computer software that had a tangible relation to physical elements or tangible steps.\(^8\)

The Federal Circuit significantly changed the software patent analysis in 1994.\(^8\) Alappat and his co-inventors developed a computerized method of smoothing out the waveform display of an oscilloscope.\(^8\) Claim 15, the key claim in Alappat's application, claimed the new method (“a rasterizer”) in means-plus-function

---

79. *Id.* at 209-10.
80. *Id.* at 216. The dissent also argued that a computer program would be nonstatutory under the disfavored “mental steps” and “function of a machine” tests, *id.* at 195-96, and that the courts should not be involved in deciding whether software is statutory subject matter. *Id.* at 215-16.
81. *Id.* at 192-93.
82. See *In re Pardo*, 684 F.2d 912, 915 (C.C.P.A. 1982).
84. See *Arrhythmia Research Tech., Inc. v. Corazonix Corp.*, 958 F.2d 1053, 1061 (Fed. Cir. 1992) (holding a computer that analyzes electrocardiograph signals was statutory subject matter).
85. *In re Alappat*, 33 F.3d 1526 (Fed. Cir. 1994).
86. *Id.* at 1537.
An eight member panel of the Board of Patent Appeals and Interferences ("the Board") rejected this claim for two reasons: 1) the claim should be interpreted as a mathematical algorithm alone, and 2) because Claim 15 recited a mathematical algorithm and not an application of a mathematical algorithm, it was nonstatutory under section 101.

The Federal Circuit, en banc, reversed the Board. Citing precedent, the Federal Circuit stated that a means-plus-function claim must be interpreted under 35 U.S.C. § 112 as "cover[ing] the corresponding structure ... described in the specification." The specification clearly describes the rasterizer as a machine comprised of computerized units programmed to perform specific functions. Turning to the section 101 issue, the Federal Circuit explicitly rejected the idea that claims which embodied "mathematical algorithm[s]" were excepted from patent coverage. Although a mathematical algorithm itself is not patentable, Claim 15 involved "a specific machine to produce a useful, concrete, and tangible result."

In conclusion, the majority opinion explicitly stated, "a computer operating pursuant to software may represent patentable subject matter, provided, of course, that the claimed subject matter meets all of the other requirements of Title 35." Software patents, although not fully endorsed by Alappat, were now one step closer to express validity under section 101.

87. Id. at 1538.
88. For an interesting discussion of the PTO Director's "expansion" of the Board of Patent Appeals and Interferences to change the outcome of Alappat, see id. at 1530-36.
89. This interpretation ran counter to Federal Circuit precedent that limitations in the specification should be imputed to claims in means-plus-function language. Id. at 1539.
90. Id. at 1540.
91. Id. at 1530.
92. See Arrhythmia Research Tech., 958 F.2d at 1060.
94. Id. at 1541.
95. Id. at 1542.
97. Alappat, 33 F.3d at 1544.
98. Id. at 1545.
99. The opinion states that the computer, and not the software itself, may be patented; Id.
100. At least one commentator has noted that Alappat was not the revolution it appeared to be, as the USPTO had already granted 10,000 software-related patents by
4. State Street Bank and AT&T

The State Street Bank patent dealt with a computerized system for pooling and tracking mutual fund assets. Multiple independent mutual funds could use the system to pool their assets, permitting them to lower transaction costs, while a computer kept track of each fund's daily income, expenses, and trading history. Each of the six claims at issue was written as a "machine" claim using means-plus-function language. The Federal Circuit addressed three issues in State Street Bank: 1) whether the claims were directed to statutory subject matter; 2) whether the claims fall under the "mathematical algorithm" exception to statutory subject matter; and 3) whether the claims fall under the "business method" exception to statutory subject matter.

The Federal Circuit held that all of the claims were drawn to statutory subject matter under section 101. Following the Alappat analysis, the court held that each claim was directed to a machine when construed in connection with the limitations in the specification. The Federal Circuit rejected the district court's contention that the claims were directed to a process, and noted that this distinction was academic for a section 101 analysis because both processes and machines are statutory subject matter under section 101.

The court's application of the "mathematical algorithm" exception began by noting that the exception applies to "abstract ideas constituting disembodied concepts or truths that are not useful." The application of a mathematical concept, such as the oscilloscope in Alappat, or the heartbeat monitor in Arrhythmia

102. Id.
103. Id. at 1371. For example, claim 1 was directed to a data processing system that included a "computer processor means," computer storage, and several "arithmetic logic circuit[s]" that calculated share price, profit, and loss for each mutual fund. Id.
104. "[T]he [Supreme] Court has held that mathematical algorithms are not patentable subject matter to the extent that they are merely abstract ideas." Id. at 1373.
105. Id. Several previous cases had suggested that a method of doing business can not be afforded patent protection. See In re Schrader, 22 F.3d 290 (Fed. Cir. 1994).
106. State Street Bank & Trust Co., 149 F.3d at 1371.
107. Id.
108. Id. at 1372.
109. Id. at 1373 (internal quotation marks omitted).
110. In re Alappat, 33 F.3d 1526, 1544 (Fed. Cir. 1994).
Research Technology, is statutory subject matter because it applies a mathematical concept in a useful way.

The court held that:
the transformation of data, representing discrete dollar amounts, by a machine through a series of mathematical calculations into a final share price, constitutes a practical application of a mathematical algorithm, formula, or calculation because it produces 'a useful, concrete and tangible result'—a final share price momentarily fixed for recording and reporting purposes and even accepted and relied upon by regulatory authorities in subsequent trades.

This sentence removed the requirement that a software claim be tethered to a physical result, be it a cured rubber product or an output on an oscilloscope. State Street Bank held that the transformation of numbers, though mathematical calculations, is itself a statutory "practical application" under section 101. At the same time, the court repudiated the relevance of Freeman-Walter-Abele, stating that it has "little, if any, applicability to determining the presence of statutory subject matter."

Turning to the "business method" exception, the court took State Street Bank as an opportunity to "lay this ill-conceived exception to rest." The court embraced Judge Newman's assertion that patentability depends on "whether the method... meets the requirements of patentability as set forth in Sections 101, 102, 103, and 112" regardless of whether the method relates to a business. Noting that the Federal Circuit had never invalidated a claim for the business method exception, the exception was subject to scholarly criticism, and that the USPTO removed the exception from the Manual of Patent Examining Procedures, the court concluded that business methods are statutory subject matter under section 101.

112. State Street Bank & Trust Co., 149 F.3d at 1373.
113. Id.
114. Id. at 1374.
115. Id. at 1375.
116. Id. at 1376.
117. Id. at n. 10 (citing In re Schrader, 22 F.3d 290, 298 (Fed. Cir. 1994)).
118. Id. at 1375-76.
119. Id. at 1375.
120. Id. at 1376.
121. Id. at 1377.
The Federal Circuit relied on State Street Bank in subsequent decisions involving the section 101 validity of software patents. One example is AT&T Corp. v. Excel Communications.\textsuperscript{122} A district court invalidated AT&T's patent claims, covering the addition and use of a primary interexchange carrier ("PIC") indicator\textsuperscript{123} in telecommunications, as a nonstatutory mathematical algorithm.\textsuperscript{124} The Federal Circuit reversed, using the analysis put forth in State Street Bank.\textsuperscript{125} Because AT&T's claims were written to a "process [that] applies the Boolean principle to produce a useful, concrete, tangible, result . . . the claimed process comfortably falls within the scope of section 101."\textsuperscript{126}

C. Conclusion

The case history from Benson through Alappat shows several trends in software patent jurisprudence: The Supreme Court has been generally suspicious of permitting software-related patents,\textsuperscript{127} and the Federal Circuit has generally been more lenient.\textsuperscript{128} Within both courts, there has been a gradual trend towards allowing software as patentable material. This trend can be seen in the shift from exclusion of software-related patents,\textsuperscript{129} to the Freeman-Walter-Abele analysis based on physical elements,\textsuperscript{130} to an implicit recognition that a programmed computer may be patentable.\textsuperscript{131} State Street Bank is the last step in this trend, permitting the section 101 patentability of software programmed into a digital computer that performs financial calculations.\textsuperscript{132} The current state of software patent jurisprudence is represented by AT&T, with courts holding that software claims are presumptively valid as either processes or machines.

\textsuperscript{122} 172 F.3d 1352 (Fed. Cir. 1999).
\textsuperscript{123} A PIC is part of the record that a telephone company makes of a phone call. The PIC field stores the name of the caller's long distance provider, which is used by a phone company to determine billing charges. \textit{Id.} at 1353-54.
\textsuperscript{124} \textit{Id.} at 1354.
\textsuperscript{125} \textit{Id.} at 1357.
\textsuperscript{126} \textit{Id.}
\textsuperscript{127} See discussion infra Part III.
\textsuperscript{130} \textit{In re} Pardo, 684 F.2d 912, 914 (C.C.P.A. 1982).
\textsuperscript{131} \textit{In re} Alappat, 33 F.3d 1526, 1545 (Fed. Cir. 1994).
\textsuperscript{132} State St. Bank & Trust Co. v. Signature Fin. Group, 149 F.3d 1368, 1374 (Fed. Cir. 1998).
III. The USPTO Guidelines for Software Patents

A. The USPTO Guidelines

In 1995, after Alappat and before State Street Bank, the USPTO issued its Examination Guidelines for Computer-Related Inventions ("Examination Guidelines"). The Examination Guidelines lay out a five step process for determining the validity of a computer-related invention. The steps are:

1. Determine what the applicant is seeking to patent.
2. Conduct a thorough search of the prior art.

The Examination Guidelines on statutory subject matter track Supreme Court and Federal Circuit jurisprudence. The Examination Guidelines stress that a computer program in the abstract is non-statutory, but a program running on a computer is statutory as a machine, and a program on a computer as part of a larger process is statutory as a process. According to the Examination Guidelines, "a process that merely manipulates an abstract idea or performs a purely mathematical algorithm is non-statutory." The Examination Guidelines review several types of claim language, but they do not place a great emphasis on the form of the claims. "What is determinative is not how the computer performs the process, but what the computer does to achieve a practical application." A practical application must do more than

134. Id. §i-iii.
135. The 41 page guide devotes a single paragraph to prior art, arguably the most important part of the process. Id. at 6.
136. See id. at 6.
137. Id. at 8.
138. Id. at 18.
139. See id. at 16, 19, and 21.
140. Id. at 18.
Once an examiner has determined that the claims are drawn to statutory subject matter, they are then directed to ensure that the applicant's claims state the subject matter of the invention and "particularly point out and distinctly claim the invention" according to 35 U.S.C. § 112, second paragraph. Next, the examiner determines if the application contains an adequate written description, enablement, and discloses the inventors best mode according to 35 U.S.C. § 112, first paragraph.

The last step in evaluation under the Examination Guidelines is to determine if the claimed invention is novel in light of the prior art, and if it would have been obvious at the time of invention by one skilled in the art. Although the Examination Guidelines do not go into great depth on this point, the examiner is supposed to apply his or her search of the prior art to make the novelty determination and his or her knowledge of computer science to make an obviousness determination.

B. Conclusion

The Examination Guidelines tend to track the case law, at least up to its 1995 publication date. A majority of the Examination Guidelines is devoted to the section 101 statutory subject matter, as is a majority of the jurisprudence dealing with software patents. Relatively little space is devoted to searching the prior art and making an obviousness determination. As discussed below, this is the main source of discontent with the current software patent system.

141. A "significant use" is "any activity which is more than merely outputting the direct result of [a] mathematical operation." Id. at 21.
142. Id. at 23. The analysis is not significantly different for software and non-software patents.
143. Id. The section 112 analysis is not significantly different between software and non-software patents.
144. Id. at 27; see 35 U.S.C. § 102 (2003).
146. 18 of the 28 substantive pages of the Examination Guidelines are devoted to the section 101 statutory subject matter analysis.
147. See discussion supra Part II.
IV. Software Patents in the Wild

A. Should Software be Patentable?

Having examined the evolution of the federal courts' treatment of software patents, it is illustrative to return to the question presented in Part I of this note—should software be patentable? Although the issue has been decided in the courts, and software patents have caused a great deal of difficulty in the business world, few would argue that software patents in the abstract are not proper subject matter for patents. Patent protection is or has been available for nearly every processed product in existence, from the ballpoint pen to the industrial process used to manufacture the latest drugs. Facially, there appears no reason why a process confined to a computer should not be patentable in the same way that a process confined to a beaker or an assembly line is patentable.

As noted before, the Constitution provides protection to "Promote the Progress of Science and Useful arts" and software can be viewed as both an application of science as well as a useful art. There is ample evidence from the record of the 1952 Patent Act that patent protection was intended to cover "anything under the sun that is made by man." After a period of initial reservation, federal courts have followed this intent in finding that software may be patented. If Congress objected to the courts' interpretation of the 1952 Patent Act to cover software, it could enact legislation to prevent such patents. Notably, in the more than twenty years since Diehr Congress has not acted to exclude software from the scope of patentable material.

149. See Regan, supra note 1, at 1.
152. U.S. CONST., art. 1, §8.
153. Depending on what one is seeking to patent.
B. Prosecution and Litigation of “One-Click”

Assuming, arguendo, that software patents are here to stay, the next question is how should the USPTO and the courts properly handle them? The real world example of the infamous Amazon.com “one-click” patent is illustrative in showing that, given enough time and money, the present-day patent system does work. Amazon.com’s patent covered the now well-known process of storing an online shopper’s billing and shipping information after a first purchase. The shopper can complete his or her transaction by clicking a single button on subsequent visits.

Amazon.com received its “one-click” patent on September 28, 1999, and immediately sued BarnesandNoble.com for infringement. A district court granted a preliminary injunction after a five day bench trial preventing BarnesandNoble.com from using its “Express Lane” automated checkout process. In holding that Amazon had a reasonable likelihood of success at a trial on the merits, the district court agreed with Amazon.com’s contention that the patent was nonobvious, and rejected Barnesandnoble.com's assertion that the patent was covered by the prior art.

Thirteen months later, the Federal Circuit reversed the district court. Noting that “we have several references that were urged upon the court as invalidating the asserted claims,” the Federal Circuit opinion held BarnesandNoble.com mounted a “serious challenge” to the validity of the patent. Although the Federal Circuit agreed that Amazon.com met its burden to show a likelihood of success, the court denied the preliminary injunction and remanded.

158. Eventually.
160. Id.
161. For a discussion of the turmoil caused by the grant of the patent, see Gnu Project, Boycott Amazon!, http://www.gnu.org/philosophy/amazon.html.
164. Id. at 1237.
165. Id. at 1233.
166. Amazon.com, Inc., 239 F.3d at 1347.
167. Id. at 1359.
168. Id. at 1360.
the case to the district court for a trial on the merits because BarnesandNoble.com made a serious challenge to the validity of the patent.\textsuperscript{169}

The case was remanded to the District Court and opening statements were scheduled for March of 2002, roughly 2\frac{1}{2} years after Amazon.com filed suit and 4\frac{1}{2} years after Amazon.com filed for the original "One Click" patent. On March 7, 2002, the suit was settled on undisclosed terms.\textsuperscript{170} Although the parameters of the settlement are unknown, presumably the prior art uncovered during litigation served as a deterrent against Amazon.com's asserting its patent against other companies. As of May 21, 2004, Amazon.com has not attempted to enforce its "One Click" patent,\textsuperscript{171} despite the use of similar technology by many web sites.

C. The Problem is the Process

The "One Click" litigation history shows that, given enough time and money, a patently invalid patent will eventually be nullified.\textsuperscript{172} However, many companies do not have the resources to fight a protracted legal battle. Moreover, as seen in the Kodak-Sun Microsystems litigation above,\textsuperscript{173} companies that are not likely to settle might rather make a "bet the company" wager on an appellate court reversal. The least costly place to invalidate a software patent, or any patent for that matter, is in the USPTO before a patent issues. For a variety of reasons, software patents place a unique burden on the USPTO, and the next section of this Note will discuss modification to USPTO procedure that will resolve the problem at its source.

V. Making the System More Efficient

Although it appears that, subject to the investment of significant time and money, software patent disputes can be settled in the federal courts, a more efficient solution would be to catch invalid patents at the USPTO. Below are three proposals for altering existing USPTO procedure to increase the efficacy of software patent prosecution. The first two proposals involve technical aspects of prosecution; the third

\begin{footnotesize}
\textsuperscript{169} Id. at 1366.
\textsuperscript{171} Gnu Project, supra note 161.
\textsuperscript{172} Or, as in this case, effectively nullified.
\textsuperscript{173} See Regan, supra note 1.
\end{footnotesize}
addresses the impact of implementing Justice Stevens' dissent in Diehr.\footnote{Diamond v. Diehr, 450 U.S. 175, 206 (1981) (Stevens, J., dissenting).} These proposals predominantly address section 102 novelty and section 103 non-obviousness.

\section{Effective Prior Art Searches by Examiners Skilled in the Art}

One of the important aspects of the novelty and obviousness determinations made by the USPTO is that these determinations are not made from the perspective of a layperson, but from the perspective of one “having ordinary skill in the art to which said subject matter pertains.”\footnote{35 U.S.C. § 103(a) (2003).} The USPTO has historically had difficulty hiring experienced computer scientists as patent examiners. This problem has been known for fifteen years,\footnote{See Advisory Commission on Patent Law Reform, A Report to the Secretary of Commerce, Background and Mandate of the Advisory Commission on Patent Law Reform, at 149-51 (1992).} yet it has only been addressed in the past three to four years. One simple expedient to hiring and retaining qualified software patent examiners is to recruit examiners from universities with strong computer science departments and to pay examiners in the software technology group salaries commensurate with entry-level positions in the industry. The USPTO recently began offering salaries in key technology centers, including computer science, that are commensurate with those offered in the private sector and hopefully will be able to retain more qualified software patent examiners in the future.

Commentators have suggested that the problem of non-obvious software patents issuing from the USPTO is due to the lack of a suitable prior art database for software inventions.\footnote{See David Syrowik, Software Patents – Just Make a Good Thing Better, 2 Mich. Telecomm. & Tech. L. Rev. 113, 122-24 (1996).} This author contends that a prior art database is not the solution. The majority of objectionable software patents are similar to the “One Click” Patent discussed above\footnote{See supra Part IV.C.} in that they are not entirely described in the prior art, but instead represent an assemblage of pre-existing parts obvious to one skilled in the field. Although a more extensive prior art database would be helpful, even a perfect prior art database requires skilled examiners to make the all-important obviousness determination.

Another important consideration for the examination of software patents through prior art searches is the nature of technical
publication in the industry. Unlike the biological and material sciences, there are relatively few academic publications devoted to the dissemination of information about new techniques in computer science. Creating effective prior art databases of advances in computer science is correspondingly more difficult than in other fields because the underlying source material for creating such a database is not readily available. This is not to say that a prior art database would not be useful, or that the construction of a prior art database for computer science should not be undertaken by the USPTO. Rather, it is an acknowledgement of the fact that computer science prior art databases are only part of the solution and will require examiners who are knowledgeable in the field to effectively use them.

B. Publication and Third Party Interference

Presently, applicants filing for patent protection solely in the United States are kept secret until a patent issues. From a business standpoint, this is a good thing, because it permits a business to apply and prosecute patents free from the eyes of the competition. However, the competition is the most motivated player in finding prior art for novelty and obviousness challenges to new patents. Because the USPTO may never be able to keep up with the rapid pace of change in the software industry, this proposal would effectively outsource the job of finding prior art to the software industry as a whole.

Although the theory of this proposal is simple, it would be complex to implement. First, Congress would have to pass legislation to modify 35 U.S.C. § 122 to eliminate the option of privacy in domestic patent applications. Second, the procedure for third party reexamination of published patents would have to be adopted to apply to the submission of prior art in the patent prosecution process. Although this process would require a significant overhaul of the patent system, and would create additional overhead for the individual patent examiners, by engaging the marketplace it would have the benefit of making a company's software patents self-enforcing relative to the competition.

180. Id.
C. Stevens' Diehr Dissent

In Diehr, Justice Stevens' dissent noted two key problems with software patents: the lack of standards for examiners regarding patentability and the difficulty in determining exactly what an unpatentable "algorithm" is relative to a patentable "process." Justice Stevens' solution to this problem was twofold; he recommended that no patent should be unique merely because it involves a computer, and an explicit determination that "algorithms," defined as "computer programs," are not statutory subject matter for patents.

Justice Stevens' first point provides valuable insight into the debate surrounding the validity of software patents, but it does not add much to the prosecution process. Presumably, adding a computer to an already-known process would fail for a lack of section 103 non-obviousness. Stevens' second assertion is likewise unworkable because the majority of software-patents are not defined in terms of algorithms. Rather, the majority of software patents are described in terms of machines or processes.

VI. Conclusion

For better or worse, software patents appear to be here for good. As shown in this Note, there is nothing inherently bad about software patents; they merely exist to protect an inventor's time and effort of invention in exchange for disclosure of the new invention as patents do in a variety of fields. As with other types of patents, the ultimate utility of software patents as a class depends largely on the care taken in the prosecution of patent applications. Although the current process is imperfect, the modifications discussed in this Note will add to the utility and overall effectiveness of software patents in general.

182. Diehr, 450 U.S. at 218 (Stevens, J., dissenting).
183. Id. See also State St. Bank & Trust Co., 149 F.3d at 1373-74 (providing a discussion of algorithms versus processes).
184. Diehr, 450 U.S. at 218 (Stevens, J., dissenting).
185. This proposal would also single out software as a separate class of patents, something that Congress and the USPTO have avoided in the 200 plus year history of American Patents. For a critique on defining software as a separate class of patentable material, see R. Polk Wagner, Of Patents and Path Dependency: A Comment on Burk and Lemley, 18 BERKELEY TECH. L.J. 1341 (2003).