NOTE

ABORT, RETRY, FAIL: PROTECTION FOR SOFTWARE-RELATED INVENTIONS IN THE WAKE OF STATE STREET BANK & TRUST CO. V. SIGNATURE FINANCIAL GROUP, INC.

Chad King

INTRODUCTION ............................................................................. 1118
I. BACKGROUND ............................................................................... 1121
   A. Statutory Subject Matter ............................................................... 1126
   B. The Mathematical Algorithm Exception ...................................... 1131
      1. Supreme Court Foundations .................................................... 1131
      2. The Freeman-Walter-Abele Test .............................................. 1135
      3. The 1994 Cases .................................................................... 1139
   C. The Business Method Exception ................................................ 1142
II. STATE STREET BANK: A WATERSHED ...................................... 1145
   A. A New Look at Old Requirements .............................................. 1147
   B. Implications for Software Patents ............................................. 1152
III. PATENT: THE OPTIMAL FORM OF PROTECTION FOR SOFTWARE? ............................................................................... 1156
   A. Alternative Forms of Protection ................................................ 1159
      1. Trade Secret ........................................................................ 1159
      2. Trademark ........................................................................... 1160
      3. Copyright ............................................................................. 1161
      4. Sui Generis Protection Schemes .............................................. 1164
   B. Proposed Amendments to the Patent Act ...................................... 1169
      1. Explicitly Define Software as Patentable Subject Matter ............. 1170
      2. Streamline the Patent Application Process ................................ 1172
CONCLUSION .................................................................................. 1179

INTRODUCTION

Computers and computer software constitute an undisputedly significant part of our economy as we enter the third millennium.¹

¹ For example, the Information Technology Industry Council, a consortium of 30 computer and information-technology companies, accounted for 16% of all industry-funded research in 1996, had revenues in excess of $405 billion during that year, and
Scholars and government agencies alike, however, struggle to determine the proper form of intellectual property protection for computer software. Likewise, software developers themselves are unsure about how to most effectively protect their inventions. Much of the uncertainty stems from the historical ambivalence of the courts and the United States Patent and Trademark Office (PTO) toward software patents. Specifically, two amorphous, judicially created exceptions to patentable subject matter, the business method exception and the mathematical algorithm exception, have prevented consistent and effective enforcement of patent protection for software-related inventions. This Note avers that despite past uncertainty, the patent system remains the best form of protection for software-related inventions. In addition, this Note considers two modifications to the current patent system that might further facilitate the efficient protection of software-related inventions.

In its landmark decision in State Street Bank & Trust Co. v. Signature Financial Group, Inc., the United States Court of Appeals for the Federal Circuit ("Federal Circuit") purported to clarify the confusing status of software patents wrought by longstanding judicial application of the mathematical algorithm and business method exceptions. The opinion emphatically reaffirmed an earlier holding by the court that the mathematical algorithm exception does not apply to software as long as the software produces "a 'useful, concrete and tangible result'" through "the transformation of data . . . by a machine through a series of mathematical calculations." Moreover, the Federal Circuit in State Street Bank elected to "take [the] opportunity to lay [the]
ill-conceived [business method] exception to rest," and held that the exception was born in dicta, had never attained the status of binding precedent, and thus should never have precluded patent protection for any inventions. By abolishing the business method exception and limiting the mathematical algorithm exception, the Federal Circuit sought to promote a new era of certainty in software patentability. State Street Bank established the viability of the current patent system as a vehicle to protect software-related intellectual property. In addition, the decision may serve as a guide for congressional amendment of the Patent Act, which would enhance the patent system's ability to properly protect software-related inventions.

Some commentators argue that a major overhaul of the Patent Act is necessary to provide proper software protection; they assert that the Act's current incarnation is ill-equipped to address the various nuances unique to the field of software patents. Under this view, even the State Street Bank decision fails to harmonize the current patent system with software-related inventions. Still other commentators reject the entire patent system itself as a viable source of protection for software-related inventions and advocate alternative means of software protection. Some suggest copyright and trademark doctrines as potential candidates for this role. Others call for a sui generis system of protection specifically for software.

This Note argues that State Street Bank has firmly established the current patent system as the best method to protect software-related inventions, despite the historical shortcomings of the patent system in

---

13 Id. at 1375.
14 See id. at 1375-77.
16 For amendments to the Patent Act that this Note considers, see infra Part III.B.
17 See, e.g., McDonald, supra note 3, ¶ 123 (noting that "the current patent statute is woefully inadequate for the protection of computer software" and arguing that "[t]he policy decisions inherent in protecting software are simply too complicated to allow adequate solution through the [Federal Circuit's] legislating from the bench").
18 For a discussion of these contentions, see infra Part III.A.
this arena. The Federal Circuit's *State Street Bank* decision created a new paradigm for software patents, in which the requirements for a software patent parallel those for traditional patents. This Note further suggests two amendments to the Patent Act, which would allow the current patent system to more easily embrace software-related inventions. Admittedly, a legislative overhaul of the Patent Act is long overdue; however, the *State Street Bank* decision provides a much needed guide for Congress to use to renovate the Act to accommodate the idiosyncrasies of software-related inventions.

Part I of this Note traces the background of software patentability jurisprudence, including the mathematical algorithm and business method exceptions. Part II analyzes the Federal Circuit's holding and rationale in *State Street Bank* and discusses the likely impact of the decision on the field of software patents. Part III explains why, in light of *State Street Bank*, the patent system has emerged as the optimum form of protection for software. Part III also discusses two amendments to the Patent Act for more efficient protection of software-related inventions by the patent system.

### I

**BACKGROUND**

Although patents have been part of the American legal landscape since the nation's inception, at least one area of patent law remains highly unsettled: whether and under what conditions computer software is patentable. Computer software and related technology factor heavily into today's economy; one could convincingly argue that...

---

21 The current Patent Act has not substantially changed since 1952. Congress could not have foreseen the range of technologies that have emerged since then, nor could it have anticipated the issues now facing the Patent Office.

22 For a discussion of *State Street Bank*'s impact on software patent jurisprudence, see infra Part II.B.

23 The Constitution authorizes Congress "[t]o 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." U.S. Const. art. I, § 8, cl. 8. Congress passed the first patent statute in 1790, before the passage of the Bill of Rights. See Act of Apr. 10, 1790, ch. 7, 1 Stat. 109 (repealed 1793). In fact, the rationale underlying the U.S. patent system dates back to before the 1790s. The United States adopted many of the principles of English patent laws, which stemmed from the 1623 Statute of Monopolies. See 1 DONALD S. CHISUM, CHISUM ON PATENTS: A TREATISE ON THE LAW OF PATENTABILITY, VALIDITY AND INFRINGEMENT at OV-3 n.2 (1999). The Statute of Monopolies allowed a fourteen-year grant of "letters patents" for "new manufactures," a practice that forms the basis of the current patent monopoly. Id. (citation omitted). The modern patent term in the United States lasts 20 years from the date of application. See 35 U.S.C § 154(a)(2) (1994). For a historical overview of the development of U.S. patent law, see 1 CHISUM, supra, at OV-2 to OV-15.

24 See Brief of Amicus Curiae Information Technology Industry Council in Support of Defendant-Appellant Signature Financial Group, Inc. at 13-14, State St. Bank & Trust Co. v. Signature Fin. Group, Inc., 149 F.3d 1368 (Fed. Cir. 1998) (No. 96-1327) [hereinafter ITIC
information technology drives modern society. Thus, it is in society's best interest to encourage the continued growth and vitality of the computer software industry. A large software developer typically spends millions of dollars annually on research and development and must rely on a form of market advantage in order to recoup this investment. Congress envisioned the patent system as a tool to provide just this sort of incentive to inventors. Amicus Brief] (noting that, as of 1996, 1331 patents covering business-related software had been issued to "companies that provide a significant percentage of the world gross international product," as well as "independent inventor[s]," and arguing that invalidating business-related software patents "potentially pulls the rug out from under [these companies and inventors] and throws an entire industry into chaos"); see also Mark A. Lemley & David W. O'Brien, Encouraging Software Reuse, 49 STAN. L. REV. 255, 262 (1997) (discussing the importance of software in today's society and noting that "[t]elevision sets may incorporate 25,000 lines of software, and even a simple electric shaver can embed 100 lines" and "[p]ower trains in new General Motors cars contain roughly 30,000 lines of code" (footnote omitted)).

The growth of the Internet reflects the increased prominence of computers and related technology in modern American society. According to a recent Nielsen survey, from January 1997 through June 1998, "the number of Internet users increased at a robust compounded rate of 2.5 percent a month ... [and] the number of on-line buyers increased even more dramatically." Nielsen Media Research, Number of Internet Users and Shoppers Surges in United States and Canada (Aug. 24, 1998) <http://www.nielsenmedia.com/newsreleases/coramnet2.html>. "From September through June 1998, the growth in on-line buyers accelerated to 8 percent a month." Id. The same study found that, for the first time in history, over fifty percent of all Americans aged 16-34 use the Internet. See id. Viewed over a broader span of time, the growth is even more remarkable. For example, in June 1993, 130 web sites existed on the Internet; by February 1999, the total had grown to 4,301,512, an increase of over three million percent in less than six years. See Robert H'obbes' Zakon, Hobbes' Internet Timeline v.0 (visited Jan. 19, 2000) <http://info.isoc.org/guest/zakon/Internet/History/HIT.html>.

See ITIC Amicus Brief, supra note 24, at 7-8 ("[P]rudent investors demand patent protection for new computer-related inventions before placing their risk capital into a new venture." (footnote omitted)); Lemley & O'Brien, supra note 24, at 267 (comparing the construction of a 12-meter yacht to compete in the America's Cup, the construction of a Formula One race car, and the development of a "software order-entry system," and observing that all three endeavors carry price tags of over $1 million, "for the same reason: Each is a custom product designed and constructed manually by skilled and highly paid craftsmen" (quoting Capers Jones, Economics of Software Reuse, COMPUTER, July 1994, at 106, 106)); see also David Bender, Recent Developments in Software Patents, in COMPUTER SOFTWARE PROTECTION 139, 146 & 215 n.4 (PLI Patents, Copyrights, Trademarks, & Literacy Property Course Handbook Series No. G-479, 1997) (noting that many companies, ranging in size from behemoths like IBM to small start-ups, have heavily invested in intellectual property for software and have depended on patents to protect that property). Bender argues that securing capital financing depends on the company's ability to obtain an intellectual property right such as patent, which justifies the heavy spending on the research and development of the software. See id. at 151-52.

Judge Newman noted:

Patent law has nicely fostered technological advance in the United States, for its principles are particularly suited to a free market system: it requires neither governmental intrusion nor federal funds to provide the incentive for industrial innovation; the innovation incentive is the direct consequence of the patent grant. I know of no major technological advance, no new industry or evolving technology, that has not participated in the patent
facial differences between software-related inventions and their more tangible, traditional counterparts, the patent system has thus far failed to consistently protect software inventions.28

Because software-related intellectual property is so valuable, developers desire the consistent enforcement of patent protection for software. However, contradictory Supreme Court and Federal Circuit opinions have created confusion in the area of software patents.29 Furthermore, some commentators have questioned whether the Patent Act,30 which has remained substantially unchanged since 1952, can adequately address the new technological and policy issues presented by software-related inventions.31

---

28 See Samuelson et al., supra note 20, at 2361-64. Samuelson and her colleagues argue that the patent system has vacillated between periods of over- and underprotection:
Concerns about overprotection [of software] contributed to the initial policy denying patents for software innovations. . . .

. . . . As the software industry became more commercially significant, the early concerns about possible overprotection . . . subsided and were replaced by concerns that the initially constrictive patent policy toward software innovation might, in fact, underprotect valuable aspects of programs. . . .

. . . . Because innovation in the software industry is typically incremental, not inventive [in the sense that tangible inventions are inventive, the patent system] will inevitably withdraw most patent protection from software and result in underprotection.

29 See Stern, supra note 20, at 208 (“[Recent Federal Circuit decisions] have left algorithm-related and computer-related patent law in such disarray that . . . it may be years before any equilibrium is reached. The outcome of cases will now significantly depend on the happenstance of panel composition.”). Richard Stern argues that the current level of uncertainty in software-related patent law is “not acceptable.” Id. “It is unfair to the software industry . . . upon which patents impact. It is not in the interest of the public that predictability of commercial affairs, business expectations, and security of investment should be impaired by this much legal uncertainty.” Id. For a discussion of the decisions that have fragmented software patent jurisprudence into its current state, see infra Part I.B-C.


31 See McDonald, supra note 3, ¶ 2 (arguing that “[t]he fundamental cause of the unpredictability and inconsistency, which might best be described as ‘doctrinal chaos,’ of recent decisions by the Court of Appeals for the Federal Circuit (CAFC) regarding patent protection for software inventions is simply that the current patent law was not designed to provide such protection” (footnote omitted)). Shawn McDonald observed that “[t]he current patent statute is based upon the mechanical innovation paradigm of the Industrial Revolution rather than the algorithmic innovation paradigm of the current Information Revolution.” Id. (footnote omitted). For a discussion of proposed amendments to the Patent Act, see infra Part III.B. Until Congress updates the Patent Act, the courts’ decisions on software patentability remain the final authority. Indeed, some commentators have speculated that Congress’s creation of the Federal Circuit signaled an abrogation of its responsibility to update the Patent Act. See McDonald, supra note 3, ¶ 13 (“The message
Under the Judicial Code, the United States Court of Appeals for the Federal Circuit has exclusive appellate jurisdiction over patent cases and thus usually serves as the final arbiter with respect to patent issues. Because the Federal Circuit has exclusive appellate jurisdiction, there is no possibility of a circuit split over patent law issues. Although the Supreme Court has jurisdiction to review Federal Circuit decisions, the Court often abstains from reviewing the Federal Circuit’s patent decisions due to the technical nature of the issues involved.

On the rare occasions when the Supreme Court grants certiorari to hear a patent appeal, the Federal Circuit often distinguishes or completely ignores the resulting decision, even if the precedent appears directly on point. Thus, in practice, the Federal Circuit represents the final authority on most patent issues. As Justice Stevens lamented, if the judges on the Federal Circuit (or the CCPA before it) disagree with Supreme Court precedent, they will not hesitate to distinguish the decision to the point of emasculation, if not ignore it.

sent by both the legislative and executive branches in creating the [Federal Circuit] was a clear and adamant request for judicial activism to take the place of legislative debate and resolution of patent law doctrine.

34 See § 402, 96 Stat. at 57.
35 See, e.g., State St. Bank & Trust Co. v. Signature Fin. Group, Inc., 149 F.3d 1368, 1370 (Fed. Cir. 1998) (holding that patent claims covering a computerized accounting system “are directed to statutory [patentable] subject matter”), cert. denied, 525 U.S. 1093 (1999). In addition, many of the Federal Circuit’s decisions are reversals of patent application rejections by the PTO Board of Appeals (“Board”). If the Federal Circuit reverses the Board’s decision and validates the patent at issue, that reversal usually becomes the final disposition of the case. See, e.g., In re Alappat, 33 F.3d 1526, 1536-37 (Fed. Cir. 1994) (reversing the Board’s rejection of a patent for a “smooth waveform display in a digital oscilloscope”); In re Abele, 684 F.2d 902, 903 (C.C.P.A. 1982) (reversing in part the Board’s rejection of a patent covering an improvement in CAT scan imaging technique); In re Freeman, 573 F.2d 1237, 1238 (C.C.P.A. 1978) (reversing the Board’s rejection of a patent covering “a system for typesetting alphanumeric information, using a computer-based control system in conjunction with a phototypesetter of conventional design”).
36 See, e.g., Diamond v. Diehr, 450 U.S 175, 192-93 (1981) (affirming the CCPA’s decision and validating a patent that covered a process for curing rubber); Parker v. Flook, 437 U.S. 584, 593-96 (1978) (reversing the CCPA’s decision and invalidating a patent that claimed a process for updating alarm limits in catalytic converters).
37 For a discussion of the CCPA’s dismissal of the Supreme Court’s holding in Flook, see infra notes 125-26 and accompanying text.
38 The Supreme Court has even adopted the rationale of the lower court in later cases. For instance, after the CCPA heavily criticized the “point of novelty” approach employed by the Supreme Court in Flook, the Supreme Court rejected the approach in its subsequent Diehr decision. See infra notes 125-27 and accompanying text.
Current software patentability doctrine is thus the result of an amalgam of contradictory Supreme Court and Federal Circuit cases. Not surprisingly, the law governing software patents has developed unevenly, causing great confusion.

Before the Federal Circuit’s decision in *State Street Bank & Trust Co. v. Signature Financial Group, Inc.*, inventors faced two substantial hurdles in their attempts to obtain patent protection for software inventions. The first hurdle, the mathematical algorithm exception to patentability, reflects the Supreme Court’s concern for the progress of technology. The Court feared that allowing an inventor to patent an abstract mathematical principle, as opposed to a specific application of that principle, could preclude other inventors from using that principle in other ways. The exception poses a significant hurdle to software patents, because virtually all software utilizes mathematical algorithms to accomplish its intended function.

Similarly, the business method exception to patentability prohibits inventors from patenting methods of doing business. The term, “methods of doing business,” has been applied broadly by courts, who, under its aegis, have purportedly invalidated patents covering inventions as diverse as bookkeeping systems and drive-in movie theaters. This exception poses a problem for software developers, because many software programs primarily implement novel methods

---

39 See *Diehr*, 450 U.S. at 204-05 (Stevens, J., dissenting). Justice Stevens chastised the CCPA for its disregard of the Supreme Court’s rationale in *Parker v. Flook*, 437 U.S. 584 (1978):

[The CCPA] construed *Flook* as resting on nothing more than the way in which the patent claims had been drafted, and it expressly declined to use the method of claim analysis spelled out in that decision. . . . [The CCPA’s] reading of *Flook*—although entirely consistent with the lower court’s expansive approach to § 101 during the past 12 years—trivializes the holding in *Flook*, the principle that underlies [*Gottschalk v. Benson*, 409 U.S. 63 (1972)], and the settled line of authority reviewed in those opinions.

*Diehr*, 450 U.S. at 204-05 (Stevens, J., dissenting).

40 For a discussion of this development, see infra Part I.B-C.


42 See *Gottschalk v. Benson*, 409 U.S. 63, 67 (1972) (relying on prior decisions which held that scientific principles and laws of nature could not be patented to decide the patentability of mathematical algorithms and explaining that “[p]henomena of nature, though just discovered, mental processes, and abstract intellectual concepts are not patentable, as they are the basic tools of scientific and technological work”). For a discussion of the genesis and subsequent development of the mathematical algorithm exception, see infra Part I.B.


44 For a detailed discussion of the business method exception, see Part I.C.

45 See *Hotel Sec. Checking Co. v. Lorraine Co.*, 160 F. 467, 469 (2d Cir. 1908).

46 See *Loew’s Drive-In Theatres, Inc. v. Park-In Theatres, Inc.*, 174 F.2d 547, 551-53 (1st Cir. 1949).
of doing business. Aggravating these problems was the fact that, prior to State Street Bank, courts had applied a variety of tests to determine whether a patent claim fell within either or both of these exceptions. The inconsistent application of the two judicial exceptions has resulted in a patchwork of fact-specific decisions, creating confusion among courts and practitioners alike.

A. Statutory Subject Matter

The power of Congress to establish a patent system arises out of the Constitution. The original justification for the U.S. patent system was not the inherent entitlement of inventors to the exclusive use of their inventions, but the provision of economic incentive. The proponents of the patent system reasoned that if the government granted inventors limited monopolies on the use of their inventions, inventors would have greater incentive to invent than if others could legally copy and subsequently undersell their inventions. Thus, the primary goal of U.S. patent law is to stimulate the advancement of technology via economic incentive. Accordingly, the current patent


48 See infra Part I.B (discussing the various manifestations of the Freeman-Walter-Abele test and the physical transformation test, all formulated to determine whether the presence of a mathematical algorithm removed an invention from the realm of patentable subject matter).


50 See U.S. CONST. art. I, § 8, cl. 8.

51 Under traditional analysis, allowing inventors exclusive use of their inventions stimulates invention. See Bender, supra note 26, at 151 ("The underlying philosophy of United States IP [Intellectual Property] law is that providing a degree of exclusivity will stimulate the incentive to create IP.").

52 The inventor must generally charge a higher price for a new piece of software than would competitors who had simply copied the inventor's innovation. This price differential reflects the research and development costs of the inventor. The absence of market protection for the invention in the form of a limited monopoly on the invention would negate all incentive to innovate. See id. at 151-52 ("A useful software package usually has a development cost of hundreds of thousands of dollars, if not millions. Absent a chance to recoup that investment, developers will have no incentive to develop, and investors no incentive to fund.").

53 Professor Chiappetta describes the rationale behind the Patent Clause: An incentive in the form of a limited right to preclude competition could be offered to inventors to encourage the investment of their time and resources in inventive endeavors which would progress the useful arts. The hope was that this encouragement would result in the desired growth in the useful arts which, in turn, would result in overall economic growth and a general increase in prosperity.
statute requires that all patentable inventions be truly novel, nonobvious, and useful, and that the inventor’s patent application disclose enough information to enable another to improve upon the invention and practice the invention upon expiration of the patent monopoly.

Congress historically has taken an expansive view of the patentability of new inventions. The 1790 Act authorized a patent for “any useful art, manufacture, engine, machine, or device, or any improvement therein not before known or used,” so long as the invention was “sufficiently useful and important.” In 1793, Congress modified the categories of patentable subject matter to include “any new and useful art, machine, manufacture or composition of matter, or any new and useful improvement [thereof].” The broad categories established by the 1793 Act largely remain intact to this day. Thus, to reside within the domain of “statutory subject matter,” an invention must fall into one of the following four categories: process, machine, composition of matter, and article of manufacture.


See id. § 103.
See id. § 101.
See id. § 112.
Act of Apr. 10, 1790, ch. 7, 1 Stat. 109 (repealed 1793).
§ 1, 1 Stat. at 110.
Act of Feb. 21, 1793, ch.11, § 1, 1 Stat. 318, 319 (repealed 1836).

In 1952, following the Supreme Court’s equation of “art” with “process,” see, e.g., Cochrane v. Deener, 94 U.S. 780, 788 (1876) (“In the language of the patent law, [a process] is an art.”); Cornng v. Burden, 56 U.S. 252, 267 (1853) (“A process ... is included under the general term ‘useful art.’”), Congress modified the statutory language to define as patentable subject matter “any new and useful process, machine, manufacture, or composition of matter,” 35 U.S.C. § 101 (emphasis added), and defined the term “process” as any “process, art or method,” id. § 100(b). See Act of July 19, 1952, ch. 950, 66 Stat. 792, 797 (codified at 35 U.S.C. §§ 100-101 (1994)); see also 1 CHISUM, supra note 23, § 1.01, at 1-5 & n.8 (discussing the evolution from “useful art” to “process”).

See 35 U.S.C. § 101. “Statutory subject matter” is a term of art in patent law. In order to be patentable, an invention must fall within the domain of statutory subject matter, which is limited to those categories enumerated in § 101 of the Patent Act. See 1 Peter D. Rosenberg, Patent Law Fundamentals § 6.00, at 6-3 (2d ed. 1997) (explaining that “[o]ne manner in which Congress may restrict the granting of patents is to limit that protection only to certain enumerated categories of invention, these specified categories of patentable or statutory subject matter being less extensive, even collectively, than the realm of inventive subject matter”). The term describes any invention that falls within the enumerated categories of § 101, while avoiding any judicial exceptions to § 101’s categories. See Diamond v. Diehr, 450 U.S. 175, 182 (1981). In addition, in order to constitute statutory subject matter, the invention at issue must avoid any judicial exceptions to the categories prescribed by § 101. See infra notes 65-71 and accompanying text for descriptions of judicial exceptions to the categories embraced by § 101. Courts should pursue the inquiry into whether a patent claim falls within statutory subject matter wholly apart from the Patent Act’s other requirements for patentability. See Diehr, 450 U.S. at 188-91; see also

Over the last 200 years, Congress has broadened the scope of patent coverage. In fact, the expressed intent of Congress in passing the 1952 Patent Act was to "include [as patentable subject matter] anything under the sun that is made by man."63 However, the Constitution limits the authority Congress may delegate to the Patent and Trademark Office to provide patent protection for inventions.64 In light of this constitutional limitation, the Supreme Court has established categories that define unpatentable subject matter. Traditionally, the Court has excluded the following categories from patentable subject matter: abstract ideas,65 natural phenomena,66 and laws of nature.67 These exceptions to patentability reflect the Court's belief that

---

Rosenberg, supra, § 6.00, at 6-3 ("Statutory subject matter is a substantive criterion of patentability separate and distinct from novelty, utility, and nonobviousness.").

63 S. REP. No. 82-1979, at 5 (1952); H.R. REP. No. 82-1923, at 6 (1952).

64 As Professor Chisum has noted, the Constitution provides separately for the promotion of the progress of science and of the useful arts. See I Chisum, supra note 23, § 1.01, at 1-5 n.9. Today, "useful arts" can be defined as "applied technology" and inventions in the "useful arts" are eligible for patent protection. Id. at 1-5. Discoveries in pure science, however, can only be secured, if at all, by copyright protection, because the Constitution provides "Authors ... the exclusive right to their ... Writings" for the promotion of "the Progress of Science." U.S. Const. art. I, § 8, cl. 8. Thus, Congress only has authority to grant patent protection to useful applications of technology, not to the science behind the technology. See I Chisum, supra note 23, § 1.01, at 1-5 & n.9.

65 Rubber-Tip Pencil Co. v. Howard, 87 U.S. (20 Wall.) 498 (1874), introduced the concept that abstract ideas do not merit patent protection. In that case, the Supreme Court held that "[a]n idea of itself is not patentable, but a new device by which it may be made practically useful is." Id. at 507. The claimed invention in Rubber-Tip Pencil was the attachment of a small piece of rubber eraser to the blunt end of a pencil. See id. at 505. The Court said in regards to this invention, "The idea of this patentee was a good one, but his device to give it effect, though useful, was not new. Consequently he took nothing by his patent." Id. at 507.

66 The natural phenomena exception arose out of Funk Brothers Seed Co. v. Kalo Inoculant Co., 333 U.S. 127, 130 (1948) ("He who discovers a hitherto unknown phenomenon of nature has no claim to a monopoly of it which the law recognizes. If there is to be invention from such a discovery, it must come from the application of the law of nature to a new and useful end."). In Funk Bros., the respondent attempted to enforce a patent covering a mixture of different bacteria strains used to inoculate plants from the entire family of the bacteria. See id. The Supreme Court held that the patentee did not actually invent anything, but had simply discovered an existing phenomenon of nature. Thus, the patentee did not deserve a patent for the discovery. See id. at 130-32.

67 O'Reilly v. Morse, 56 U.S. (15 How.) 62 (1853), represents the genesis of the law of nature exception to patentable subject matter:

The mere discovery of a new element, or law, or principle of nature, without any valuable application of it to the arts, is not the subject of a patent. But he who takes this new element or power, as yet useless, from the laboratory of the philosopher, and makes it the servant of man; who applies it to the perfecting of a new and useful art, or to the improvement of one already known, is the benefactor to whom the patent law tenders its protection.

Id. at 132-33 (Grier, J., concurring). Morse involved Samuel Morse's invention of the telegraph. In the patent's eighth claim, Morse rejected all limitations to the specific machinery used to operate the telegraph, in effect claiming the entire concept of electronic communications. See id. at 112-13. The Supreme Court invalidated the claim, explaining:
the government should grant a patent monopoly only for inventions that produce a useful physical manifestation, and not for simply innovative ideas.\(^68\)

As discussed earlier, in addition to these traditional exceptions, the courts have crafted two other exceptions to statutory subject matter that affect software patents: the mathematical algorithm exception\(^69\) and the business method exception.\(^70\) In the past, the mathematical algorithm and business method exceptions provided the highest hurdles to software developers seeking patent protection.\(^71\)

When applying for patent protection, inventors have traditionally characterized their software-related claims as processes.\(^72\) Concep-

"[Morse] claims an exclusive right to use a manner and process which he has not described and indeed had not invented, and therefore could not describe when he obtained his patent. The court is of opinion that the claim is too broad, and not warranted by law." \textit{Id.} at 113.

\(^68\) See \textit{id.}. The Court expressed concern that allowing an inventor to claim an idea, without requiring application of that idea to a specific physical manifestation, would produce two undesired results: First, the inventor would obtain a monopoly without providing disclosure sufficient to allow others to improve upon the invention. \textit{See id.} Second, the inventor would be able to exercise a patent monopoly over any machine that used the claimed principle, and not simply the inventor's embodiment of the invention. \textit{See id.} The Supreme Court's concern emanates from the Constitution's limitation of patent protection to the "useful arts." \textit{See supra} note 64.

\(^69\) The mathematical algorithm exception arose out of the Supreme Court's holding in \textit{Gottschalk v. Benson}, 409 U.S. 63 (1972). For a discussion of the evolution of the mathematical algorithm exception, see Part I.B.

\(^70\) \textit{Hotel Security Checking Co. v. Lorraine Co.}, 160 F. 467 (2d. Cir. 1908), provides the genesis of the business method exception, although commentators have questioned the exception's legitimacy. For a discussion of the exception and its shortcomings, see Part I.C.


A patent contains two components important to the § 101 subject matter analysis: the claims and the specification. A patent's claims denote the invention over which the applicant seeks a patent monopoly. As Professor Goldstein notes, "[I]ike a metes and bounds description in a deed to real property, a patent's claim or claims establishes the boundary of the property owner's exclusive rights." \textbf{PAUL GOLDSTEIN, COPYRIGHT, PATENT, TRADEMARK AND RELATED STATE DOCTRINES: CASES AND MATERIALS ON THE LAW OF INTELLECTUAL PROPERTY} 384 (rev. 4th ed. 1999). The structure of a patent's claims reflects the particular statutory categories relevant to the invention. Thus, a "process" claim corresponds to a statutory process, while a "means-plus-function" claim is a way to claim a process as a machine; the applicant claims a "means" for achieving each step in the invention's overall function.

Conversely, in order to comply with the Patent Act's disclosure requirement, a patent's specification includes a detailed description of the invention. \textit{See} 35 U.S.C. § 112 (1994). The specification may limit the scope of the claims because, while patent appli-
tually, this characterization makes sense: software is simply a set of instructions that directs a machine (usually a computer) to perform a series of steps to accomplish a desired result. However, for more certain patent protection, recent patent applicants have often characterized their software-related inventions as machines by employing the "means-plus-function" style of claim drafting.\textsuperscript{73}

The means-plus-function format characterizes the invention as a machine, rather than a process. This characterization attempts to avoid the pitfalls of the mathematical algorithm exception. But as the Federal Circuit recently noted, "for the purposes of a § 101 [statutory subject matter] analysis, it is of little relevance whether [a software claim] is directed to a 'machine' or a 'process,' as long as it falls within at least one of the four enumerated categories of patentable subject matter."\textsuperscript{74}

Similarly, in the last few years, patent applicants have begun to claim their software-related inventions as articles of manufacture. For example, the disputed claim in \textit{In re Beauregard}\textsuperscript{75} describes the invention as "'a computer usable medium in which a program code is embodied.'"\textsuperscript{76} Like the means-plus-function format, the article of manufacture format seeks to avoid the mathematical algorithm exception.\textsuperscript{77} However, \textit{State Street Bank} suggests that such artful draftsmen-
ship is now unnecessary. Indeed, one of State Street Bank's progeny has rendered the issue moot by holding that "[w]hether stated implicitly or explicitly [in prior decisions], we consider the scope of § 101 to be the same regardless of the form—machine or process—in which a particular claim is drafted."

B. The Mathematical Algorithm Exception

1. Supreme Court Foundations

The mathematical algorithm exception, the newest of the Supreme Court's patentability exceptions, grew out of Gottschalk v. Benson. In Benson, the Supreme Court applied to the field of digital computers its earlier holdings that "[p]henomena of nature . . . mental processes, and abstract intellectual concepts are not patentable." The patent application in Benson involved a method of programming a general-purpose digital computer that would convert signals from binary-coded decimal format to pure binary form. Notably, the only physical limitation on the scope of the claim was the "shift register," a generic piece of computer equipment, which implemented the conversion process. As the first Supreme Court decision to examine software-related inventions, Benson provided much of the rationale behind the subsequent body of mathematical algorithm case law. The Benson holding indicated that without a physical limitation upon the patent claims, an

under § 101. There is, therefore, no need to apply the mathematical algorithm . . . test.

Id. (footnote omitted). For a discussion of the mathematical algorithm exception, see infra Part I.B.

For a discussion of the potential impact of the State Street Bank decision on software patents, see infra Part II.B.

AT&T Corp. v. Excel Communications, Inc., 172 F.3d 1352, 1357-58 (Fed. Cir.) (citing, inter alia, State Street Bank to support validation of claims in AT&T's patent that were directed to a method for using a telecommunications system), cert. denied, 120 S. Ct. 368 (1999).

The other exceptions all arose in the nineteenth and first half of the twentieth century. See supra notes 65-67 and accompanying text.

409 U.S. 63 (1972). The Supreme Court defined the term "algorithm" as "a procedure for solving a given type of mathematical problem." Id. at 65.

Id. at 67.

See id. at 65-67.

Id. at 73; McDonald, supra note 3, ¶ 21. A patent applicant can include information in the patent's specification to limit the scope of the claim. See supra note 72. By thus narrowing the scope of the claim, the applicant can avoid a rejection based on an otherwise impermissibly broad claim. The physical limitation would have narrowed the scope of the claim to a particular physical implementation of the algorithm. Without such limitation, a claim would conceivably preempt use of the algorithm itself, and not simply a particular application of the algorithm. Thus, by failing to narrow his claim beyond a component found in every computer, Benson precluded use of his algorithm on any computer. See McDonald, supra note 3, ¶ 21-24.

See McDonald, supra note 3, ¶ 21, 25.
invention containing a mathematical algorithm is unpatentable per se, as it otherwise would preempt all future use of the algorithm. In the Court’s view, because they did not effect any physical transformation, Benson’s claims focused solely on the mathematical conversion. Thus, validating this type of patent would foreclose all future use of the algorithm at issue. Like the other exceptions to statutory subject matter, the mathematical algorithm exception seeks to ensure that the federal government does not preclude future inventors from applying an abstract mathematical principle by granting a patent on the principle itself. The Court implements the policy behind these exceptions by restricting patentability to specific applications of the principle.

The Supreme Court revisited the mathematical algorithm exception in Parker v. Flook. Flook’s patent application involved a method of updating alarm limits in a catalytic converter. Flook, the inventor, attempted to distinguish Benson on the ground that the process in Benson only involved mathematical algorithms, whereas the process in Flook involved steps additional to the mere calculation of numerical values. The Court, however, disagreed and held that a process is nonstatutory if the mathematical algorithm is the only novel step.

---

86 See Benson, 409 U.S. at 71-72 ("The mathematical formula involved here has no substantial practical application except in connection with a digital computer, which means that if [the patent at issue] is affirmed, the patent would wholly pre-empt the mathematical formula and in practical effect would be a patent on the algorithm itself.").

87 See id. at 65.

88 See R. Lewis Gable, An Historical Perspective on Patent Protection for Software—Everything Old Is New Again, in COMPUTER SOFTWARE PROTECTION, supra note 26, at 9, 21. As McDonal notes, the Court’s rationale for this rejection is questionable, because “[t]he subject matter of a claim, rather than its scope, is the sole ambit of § 101 eligibility analysis.” McDonald, supra note 3, ¶ 24 (citation omitted).

89 See supra note 68 and accompanying text.

90 The Benson Court expanded the holdings of earlier “scientific principle” cases by applying their “application” requirement to a process claim, as opposed to a product claim: “[Earlier cases dealt with] ‘product’ claim[s], while the present case deals with a ‘process’ claim. But we think the same principle applies. Here the ‘process’ claim is so abstract and sweeping as to cover both known and unknown uses of the [underlying scientific principle].” Benson, 409 U.S. at 67-68.

91 437 U.S. 584 (1978).

92 See id. at 585-86.

93 See id. at 589-90.

94 That is, the process falls within the mathematical algorithm exception.

95 See Flook, 437 U.S. at 594-95. To properly analyze Flook, one must distinguish statutory subject matter from the other requirements of the Patent Act. In addition to being within one of the enumerated categories of § 101, an invention must also satisfy the novelty requirement of § 102 and the nonobviousness requirement of § 103. See 35 U.S.C. §§ 101-103 (1994). The Flook Court’s “point of novelty” analysis commingles the requirements, focusing not on the presence or absence of statutory subject matter, but on whether the process meets the novelty and nonobviousness requirements without the algorithm at issue. Thus, if the algorithm is the only step in the process distinguishable from the prior art (the “point of novelty” in the invention), the claimed process fails the novelty inquiry and
This "point of novelty" analysis has had a profound impact on software patentability because the only novel aspects of most software-related inventions are algorithms performing either new functions or existing functions more efficiently.\(^9\)

*Flook* also established the proposition that post-solution activity is insufficient to render an otherwise nonstatutory algorithm patentable: "The notion that post-solution activity, no matter how conventional or obvious in itself, can transform an unpatentable principle into a patentable process exalts form over substance."\(^9\) Although the Court invalidated Flook's process, the Court nevertheless expressly restricted its holding to the facts of the case and noted that "a process is not unpatentable simply because it contains a law of nature or a mathematical algorithm."\(^9\)

Although the *Flook* dictum preserved a foothold for mathematical algorithms, the Court did not explicitly validate a process containing an algorithm until *Diamond v. Diehr*.\(^9\) *Diehr* involved a process for curing rubber, which included an algorithm that continuously calculated the remaining cure time for the rubber article.\(^10\) Because the "respondents' claims involve[d] the transformation of an article,"\(^10\) the Supreme Court validated the patent, noting that the Court's "conclusion regarding respondents' claims is not altered by the fact that in several steps of the process a mathematical equation and a programmed digital computer are used."\(^10\)

*Diehr* thus represents a withdrawal from the point of novelty approach of *Flook*.\(^10\) With this retreat, the Court in *Diehr* returned to the traditional definition of a statutory process: the "[t]ransformation . . . of an article 'to a different state or thing.'"\(^10\) Furthermore, the Court recognized the mathematical algorithm exception as a natural

---

1 Chisum, *supra* note 25, § 1.03[2], at 1-69.

---

96 See *supra* note 43 and accompanying text.

97 *Flook*, 437 U.S. at 590.

98 Id.


100 See id. at 177-78.

101 Id. at 184.

102 Id. at 185.

103 See id. at 189 n.12 (explaining that courts should apply the § 101 statutory subject matter analysis to the entire claim, and not just the algorithm.) The Court elaborated that the argument that a process is nonstatutory if the algorithm constitutes the only novel element in the process "would, if carried to its extreme, make all inventions unpatentable because all inventions can be reduced to underlying principles of nature which, once known, make their implementation obvious." Id.

104 Id. at 184 (quoting Gottschalk v. Benson, 409 U.S. 63, 70 (1972) (internal quotation marks omitted)).
corollary to the traditional framework of exceptions established in *O'Reilly v. Morse*\textsuperscript{105} and *Rubber-Tip Pencil Co. v. Howard*,\textsuperscript{106} rather than as a new genre of exception.\textsuperscript{107}

The Court characterized the claims in *Diehr* much more favorably than those in *Flook*, and this characterization shaped the Court’s § 101 subject-matter inquiry.\textsuperscript{108} Once the Court determined that Diehr’s use of the algorithm was part of a larger process, it deemed the process patentable despite the centrality of the algorithm to the process.\textsuperscript{109} In contrast, the *Flook* Court characterized the remaining process as “post-solution” despite the fact that the algorithm’s solution served a specific purpose.\textsuperscript{110} This characterization led to the conclusion that the process was unpatentable.\textsuperscript{111} *Diehr* thus retreats from the *Flook* Court’s nearly absolute prohibition against mathematical algorithms. As long as the algorithms are part of a larger process, the process itself may still be patentable.

The *Diehr* holding comports with traditional statutory subject matter exceptions. As long as an inventor claims an application of a scientific principle or abstract idea, or, as in this case, an algorithm within a process that contains other patentable steps, the claim recites statutory subject matter. However, if the inventor attempts to claim an idea or principle itself, or, as in this case, the algorithm without the surrounding, limiting process, the claim is nonstatutory.\textsuperscript{112} *Diehr* was the Supreme Court’s last word on the mathematical algorithm exception, and Federal Circuit opinions have established the remainder of the mathematical algorithm doctrine.\textsuperscript{113}

\textsuperscript{105} 56 U.S. (15 How.) 62 (1858).

\textsuperscript{106} 87 U.S. (20 Wall.) 498 (1874).

\textsuperscript{107} See *Diehr*, 450 U.S. at 185.

\textsuperscript{108} Compare id. at 175 (“[The applicants’] process admittedly employs a well-known mathematical equation, but they do not seek to pre-empt the use of that equation. Rather, they seek only to foreclose from others the use of that equation in conjunction with all of the other steps in their claimed process.”), *with Parker v. Flook*, 437 U.S. 584, 595 (1978) (“[I]f 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.”) (quoting *In re Richman*, 563 F.2d 1026, 1030 (C.C.P.A. 1977)).

\textsuperscript{109} See *Diehr*, 450 U.S. at 187.

\textsuperscript{110} *Flook*, 437 U.S. at 590.

\textsuperscript{111} See id. at 594-95.

\textsuperscript{112} See *Diehr*, 450 U.S. at 187 (“Our earlier opinions lend support to our present conclusion that a claim drawn to subject matter otherwise statutory does not become nonstatutory simply because it uses a mathematical formula, computer program, or digital computer. . . . [A]n application of a law of nature or mathematical formula . . . may well be deserving of patent protection.”).

\textsuperscript{113} See McDonald, *supra* note 3, ¶ 32.
2. The Freeman-Walter-Abele Test

Pursuant to the Supreme Court's opinions in Gottschalk, Flook, and Diehr, the CCPA, through a series of three decisions, developed a two-part test to determine whether a claimed invention fell within the mathematical algorithm exception.114 The first of these decisions, In re Freeman,115 involved a process that used a computer to control a phototypesetter.116 In Freeman, the CCPA followed a two-step process to determine if the mathematical algorithm exception prevented patent protection for the claimed invention. In the first step, the court sought to determine "whether the claim directly or indirectly recites an 'algorithm' in the Benson sense of that term," because only such a claim could preempt an algorithm.117 If the claim did contain a Benson algorithm, the second step of the test required the court to determine "whether in its entirety [the claim] wholly preempts that algorithm."118 However, the Freeman court did not address the second stage of this inquiry because it held that the claim did not recite a Benson algorithm.119

In re Walter involved a method of seismic prospecting and surveying, which estimated subsurface formations by using a mathematical algorithm to "cross-correlate" emitted vibration waves that subsequently returned through the earth.120 The CCPA emphasized that a court should consider the relationship between the algorithm and the remaining substance of the claim when deciding whether a process containing a mathematical algorithm constitutes statutory subject matter.121 To that end, the CCPA revised the second step of its mathematical algorithm exception analysis:

Once a mathematical algorithm has been found, the claim as a whole must be further analyzed. If it appears that the mathematical

---

114 See generally Gable, supra note 88, at 30-44 (reviewing the development and decline of the Freeman-Walter-Abele two-step test).
116 See id. at 1238.
117 Id. at 1245. Note the specific reference to a "Benson" algorithm. The CCPA distinguished the Benson algorithm from the generic definition of algorithm. Compare id. ("[A]s in Benson, they recite a 'procedure for solving a given type of mathematical problem.'" (quoting Gottschalk v. Benson, 409 U.S. 63, 65 (1972))), with In re Chatfield, 545 F.2d 152, 156 n.5 (The broader definition algorithm is "a step-by-step procedure for solving a problem or accomplishing some end." (quoting WEBSTER'S NEW COLLEGIATE DICTIONARY (1976))). As the Freeman court noted, "[b]ecause every process may be characterized as 'a step-by-step procedure...for accomplishing some end,' a refusal to recognize that Benson was concerned only with mathematical algorithms leads to the absurd view that the Court was reading the word 'process' out of [35 U.S.C. § 101]." In re Freeman, 573 F.2d at 1246.
118 In re Freeman, 573 F.2d at 1245.
119 See id.
120 618 F.2d 758 (C.C.P.A. 1980).
121 Id. at 760-61.
122 See id. at 765.
algorithm is implemented in a specific manner to define structural relationships between the physical elements of the claim (in apparatus claims) or to refine or limit claim steps (in process claims), the claim being otherwise statutory, the claim passes muster under § 101. If, however, the mathematical algorithm is merely presented and solved by the claimed invention . . . and is not applied in any manner to physical elements or process steps, no amount of post-solution activity will render the claim statutory; nor is it saved by a preamble merely reciting the field of use of the mathematical algorithm.\(^\text{123}\)

Thus, \textit{Walter} narrowed the scope of the second step of the inquiry by examining not simply whether the claim completely preempted the algorithm, but rather whether the algorithm is actually interrelated with the physical elements of the claim.\(^\text{124}\)

In addition to establishing the two-stage mathematical algorithm exception inquiry, \textit{Freeman} and \textit{Walter} signaled a retreat from the point of novelty inquiry of \textit{Flook}.\(^\text{125}\) The CCPA emphatically condemned the point of novelty approach in \textit{Walter}.

If [the point of novelty] approach were to be adopted it would immeasurably debilitate the patent system. We do not believe the Supreme Court has acted in a manner so potentially destructive. As an illustration of the utter failure of such an approach to resolve these questions, we offer the example of certain improvement inventions, wherein the improvement resides in the application of scientific truth, e.g., mathematical formulae, to previously-known structure or process steps.

Improvement inventions are expressly included within § 101 . . . . Yet a strict "point of novelty" approach to improvement inventions involving the application of scientific truth as the improvement would effectively place them, as a class, outside the coverage of § 101—and to no purpose.\(^\text{126}\)

The Supreme Court embraced the CCPA's rejection of the point of novelty approach in \textit{Diehr}.\(^\text{127}\) The Court's rejection of this ap-
ABORT, RETRY, FAIL

proach reaffirmed the conceptual distinction between the subject matter requirement of § 101 and the novelty requirement of § 102. This distinction paved the way for later CCPA and Federal Circuit decisions that more closely analyzed the relationship between mathematical algorithms and the particularized claims at issue.

In re Abele established the Freeman-Walter-Abele test in its final form. Abele sought to patent an improvement in the field of computerized tomography, commonly known as a CAT scan. The improvement involved a new method of interpreting the data received from the scanning apparatus that required fewer scans to obtain a clear image. Claims 5 and 6 of the patent application read:

5. A method of displaying data in a field comprising the steps of

calculating the difference between the local value of the data at a data point in the field and the average value of the data in a region of the field which surrounds said point for each point in said field, and

displaying the value of said difference as a signed gray scale at a point in a picture which corresponds to said data point.

6. The method of claim 5 wherein said data is X-ray attenuation data produced in a two dimensional field by a computed tomography scanner.

The CCPA concluded that the claims constituted a mathematical algorithm under the first step of the test, relying on the “calculating the difference” language of claim 5.

process is unpatentable under § 101, not because it contains a mathematical algorithm as one component, but because once that algorithm is assumed to be within the prior art [and thus not novel], the application, considered as a whole, contains no patentable invention.

---

128 See Diehr, 450 U.S. at 189-91.
129 See discussion infra Part I.B.3.
130 684 F.2d 902 (C.C.P.A. 1982).
131 The court explained in detail the mechanism of computed tomography:

Basically, computed tomography provides an image representing a transverse slice of the body. This slicing is accomplished by rotating an X-ray source and a detection means around the perimeter of the section to be viewed. The source and detection means are placed 180° from each other to allow the detection means to measure the attenuation of the beam as it passes through the plane of interest. When enough measurements have been taken, a computer is implemented to mathematically interpret the data, which is then displayed as a reconstruction of the slice on, inter alia, a television screen for diagnostic purposes.

Id. at 903.
132 Id. at 908 (emphasis added).
133 See id. at 907.
Instead of requiring that the mathematical algorithm define the structural relationship between the physical elements of the claim, the CCPA panel expanded the scope of the second step by requiring only that the claims applied the algorithm in any manner to physical elements or process steps. Moreover, if the claims with the mathematical algorithms subtracted recite statutory subject matter, the claims fall within statutory subject matter when incorporating the algorithm. This remains true even if the claims are inoperative without the algorithms.

When applying the revised test to claim 6, the CCPA found that the mathematical algorithm pertained to the physical elements of the claim, namely the X-ray data. Conversely, the CCPA held that claim 5, which did not refer to any specific data to which the mathematical algorithm might apply, was "directed solely to the mathematical algorithm portion of appellants' invention and is, thus, not statutory subject matter under § 101."

Viewed cumulatively, Freeman, Walter, and Abele established a two-stage test to determine whether a claim containing an algorithm falls within statutory subject matter. The first step inquires whether the algorithm is in fact a mathematical algorithm as defined by Benson. If the claim includes a Benson algorithm, the second step of the test requires the court to determine whether the algorithm applies only to some physical elements of the claim or the patent instead claims solely the algorithm itself. Both stages of the Freeman-Walter-Abele test have proven problematic, and the Federal Circuit subsequently revamped its mathematical algorithm exception inquiry through a series of cases in 1994.

---

134 The inquiry in Walter focused on whether the mathematical algorithm actually interrelated with the structure of the physical elements of the claim. See supra text accompanying notes 123-24.

135 See In re Abele, 684 F.2d at 906; Gable, supra note 88, at 35-36.

136 See In re Abele, 684 F.2d at 907 ("[i]f the claim would be 'otherwise statutory,' albeit inoperative or less useful without the algorithm, the claim likewise presents statutory subject matter when the algorithm is included." (citation omitted)).

137 See id. at 908.

138 Id.

139 See supra notes 86-89 and accompanying text.

140 See Vietzke, supra note 127, at 7.

141 See Arrhythmia Research Tech., Inc. v. Corazonix Corp., 958 F.2d 1053, 1062-64 (Fed. Cir. 1992) (Rader, J., concurring) (arguing that the Freeman-Walter-Abele test ignores Supreme Court precedent established in Benson, Flook, and Diehr and criticizing the test's dependence on a definition of "'mathematical algorithm'" that "remain[s] vague" and is "[w]ithout a statutory anchor"); see also Gable, supra note 88, at 39-43 (analyzing In re Schrader, 22 F.3d 290 (Fed. Cir. 1994), and detailing the Federal Circuit's "struggle[ ] with each of [Freeman-Walter-Abele's] two steps" in that case); Vietzke, supra note 127, at 7-8 (describing the Freeman-Walter-Abele test as "cumbersome and sometimes unworkable").
3. The 1994 Cases

Through a series of decisions in 1994, the Federal Circuit rejected the Freeman-Walter-Abele test in favor of a more workable approach. Arrhythmia Research Technology, Inc. v. Corazonix Corp. foreshadowed the demise of the old two-stage test, when it recognized that the Freeman-Walter-Abele test was not the only method available to determine whether a mathematical algorithm renders a claim invalid. Moreover, a concurring opinion in Arrhythmia Research strongly criticized the two-step test in favor of a more certain approach.

In re Schrader represents the Federal Circuit's last foray into the Freeman-Walter-Abele test. Schrader illustrates many of the difficulties inherent to that test. In Schrader, the Federal Circuit invalidated the applicant's attempt to patent a method of calculating competitive bids on a series of related items. The claimed invention determined the winning bid by calculating whether a single bid on all of the items or the aggregate of multiple bids on the individual bids totaled a greater sum. Because the disputed claim recited no physical limitation, the court concluded that it failed the second step of the Freeman-Walter-Abele test. Although Schrader seemingly stretched the definition of mathematical algorithm beyond the bounds of logic, the case is

---

142 958 F.2d 1053 (Fed. Cir. 1992).
143 See id. at 1058 ("[T]he Freeman-Walter-Abele analysis is not the only test for statutory subject matter, and this court has stated that failure to meet that test may not always defeat the claim . . . ." (citations omitted)).
144 See id. at 1066 (Rader, J., concurring) ("[T]he Supreme Court has focused this court's inquiry on the statute, not on special rules for computer art or mathematical art or any other art . . . [T]he Supreme Court's most recent message is clear: when all else fails (and the [Freeman-Walter-Abele] algorithm rule clearly has), consult the statute.").
145 22 F.3d 290 (Fed. Cir. 1994).
147 See id.
148 See In re Schrader, 22 F.3d at 291.
149 See id. at 291-92.
150 See id. at 293-94 ("[T]here is nothing physical about bids per se. Thus, the grouping or regrouping of bids cannot constitute a physical change, effect, or result.")
151 In attempting to reconcile its overly broad conception of when a mathematical algorithm should preclude patentability with the Supreme Court's limited expression of the mathematical algorithm exception in Diehr, the Schrader majority argued:

There is no inconsistency between [the holding in Schrader] and the statement in Diehr that the mathematical algorithm exception is limited to those algorithms that express a law of nature, a natural phenomenon, or an abstract idea. Schrader's algorithm relates to two obvious and familiar modes of human behavior: that potential buyers naturally may submit bids on one, some, or all of the items available for sale, and that sellers may naturally choose that combination of bids that maximize their profits.
152 Id. at 293 n.8 (citation omitted). The Schrader court's conception of an auction as a mathematical algorithm strains the Supreme Court's construction of that term in Diehr. Although Schrader's invention may have lacked utility or novelty, it is difficult to argue that it recites an abstract mathematical algorithm. See Gable, supra note 88, at 39-40.
most notable for Judge Newman’s dissent. Her dissenting opinion laid groundwork for the Federal Circuit’s eventual abandonment of the Freeman-Walter-Abele test and set the stage for a new standard.

In the landmark en banc decision, In re Alappat, the Federal Court implicitly deserted the Freeman-Walter-Abele test in favor of a return to the statutory language of §101 and its Supreme Court interpretations. Alappat invented a “a rasterizer for creating a smooth waveform” on an oscilloscope. The PTO Board of Appeals rejected the patent application, “because it ‘[merely] reads on a general purpose digital computer “means” to perform the various steps under program control.’” The Federal Circuit, however, disagreed. The court held that the inclusion of a “programmed general purpose computer” as an element in the claims [did] not justify holding [the claims] unpatentable as directed to nonstatutory subject matter. . . . Such programming creates a new machine, because a general purpose computer in effect becomes a special purpose computer once it is programmed to perform particular functions pursuant to instructions from program software.

The Alappat court rejected the prior notion of the mathematical algorithm as “an overly broad, fourth category of subject matter excluded from §101.” The Federal Circuit instead read Benson, Flook, and Diehr as “an attempt by the Court to explain a rather straightforward concept, namely, that certain types of mathematical subject matter, standing alone, represent nothing more than abstract ideas until reduced to some type of practical application.” Thus, Alappat established a new standard for mathematical algorithms; Alappat’s subject

---

152 See, e.g., Gable, supra note 88, at 43 (discussing Judge Newman’s dissent in Schrader).
153 Compare In re Schrader, 22 F.3d at 297 (Newman, J., dissenting) (“The test [for statutory subject matter] is simply whether the mathematical formula or equation is all that is claimed, or whether the procedures involving the specified mathematics are part of a useful process. When the latter requirement is met the subject matter is statutory.”), with In re Alappat, 33 F.3d 1526, 1544 (Fed. Cir. 1994) (en banc) (stating that “the proper inquiry in dealing with the so called mathematical subject matter exception to §101 alleged herein is to see whether the claimed subject matter as a whole is a disembodied mathematical concept . . . [or] a specific machine to produce a useful, concrete, and tangible result” and that the specific machine would be patentable, but the disembodied mathematical concept would not).
154 33 F.3d 1526 (Fed. Cir. 1994) (en banc).
155 See id. at 1542-44; see also Gable, supra note 88, at 44 (“Without mentioning the Freeman-Walter-Abele test by name, the Federal Circuit stated that §101 determinations must be made in accordance with the primary authorities, that is, the statutory language of §101 and the trilogy of Supreme Court decisions Benson, Flook and Diehr.” (footnotes omitted)).
156 In re Alappat, 33 F.3d at 1544.
157 Id. at 1544-45 (quoting the PTO Board’s opinion).
158 Id. at 1545 (citations omitted).
159 Id. at 1543.
160 Id.
matter inquiry focused on whether the claim recites a practical application of the algorithm or only covers the algorithm itself. This new standard fit neatly within the framework of traditional judicial exceptions.\footnote{Cf. O'Reilly v. Morse, 56 U.S. (15 How.) 62, 132-33 (1853) (commenting that, although "[t]he mere discovery of a new element, or law, or principle of nature, without any valuable application of it to the arts, is not the subject of a patent," an application of such is patentable subject matter).} More importantly to software developers, it "placed a Federal Circuit imprimatur on the proposition that . . . claims embodying mathematical algorithms are statutory," where physical apparatus limits the scope of the claim to a practical application of the algorithm.\footnote{Bender, supra note 26, at 169.} The court also reemphasized that a proper § 101 inquiry should analyze the claim as a whole, not just pieces thereof:

It is thus not necessary to determine whether a claim contains, as merely a part of the whole, any mathematical subject matter which standing alone would not be entitled to patent protection. Indeed, because the dispositive inquiry is whether the claim as a whole is directed to statutory subject matter, it is irrelevant that a claim may contain, as part of the whole, subject matter which would not be patentable by itself.\footnote{In re Alappat, 33 F.3d at 1543. The Alappat court also noted: [A]n analysis wherein one attempts to identify whether any part of a claim recites mathematical subject matter which would not by itself be patentable is not an improper analysis. Such a dissection of a claim may be helpful under some circumstances to more fully understand the claimed subject matter. Nevertheless, even in those cases wherein courts have applied a variant of the two-part analysis of In re Freeman, as amended by In re Walter, the ultimate issue always has been whether the claim as a whole is drawn to statutory subject matter. Id. at 1543 n.21 (citations omitted).}

While resolving many issues in the software patent controversy, \textit{Alappat} perpetuated one unfortunate artifact of the mathematical algorithm exception: it continued to place dispositive weight on the patent's claim format. In \textit{Alappat}, the Federal Circuit relied heavily upon the fact that the applicant drafted his claim in means-plus-function format.\footnote{See Gable, supra note 88, at 52-53.} After \textit{Alappat}, a software developer could therefore reasonably ensure grant of a patent by limiting the scope of the patent claims to a practical application of a mathematical algorithm and by drafting the claims in means-plus-function format. The \textit{Alappat} majority did leave open the question of how drafting claims in other formats would affect patentability, thereby muddying the waters yet again for future software patent applicants.\footnote{\textit{Alappat} and its progeny ignited a new scholarly debate: a dispute over the best method of claiming software inventions. See, e.g., Chiappetta, supra note 53, at 114 (postulating that "[the \textit{Alappat} court's] willing abandonment of the [Freeman-Walter-Abele]/mathematical algorithm analysis in [means-plus-function claims] set the tone and direction for its}
several mathematical algorithm cases subsequent to *Alappat*, its mathematical algorithm jurisprudence remained virtually unchanged until *State Street Bank*.

C. The Business Method Exception

Like the mathematical algorithm exception, the business method exception has thwarted the efforts of many software patent applicants. However, the judicial foundations of the business method exception are far murkier than those of its mathematical algorithm counterpart. According to the district court judge in *State Street Bank*, “[a]s established by a series of older cases, business methods are unpatentable abstract ideas.” Some commentators, however, believed that the business method exception’s analytic value is “dubious.” Moreover, “no [appellate] court majority has ever held that [an otherwise patentable invention] was *per se* unpatentable simply because the method was directed to a way to conduct business.”

Although earlier cases alluded to a prohibition against patents claiming methods of doing business, most scholars consider the Second Circuit’s decision in *Hotel Security Checking Co. v. Lorraine Co.* as the “genesis of the business method exception.” The invention in that case was a “method of and means for cash-registering and account-checking,” intended to prevent hotel and restaurant personnel

---

166 See, e.g., *In re Trovato*, 42 F.3d 1376, 1381 (Fed. Cir. 1994), withdrawn, 60 F.3d 807 (Fed. Cir. 1995) (en banc) (per curiam); *In re Warmerdam*, 33 F.3d 1354, 1358-60 (Fed. Cir. 1994).

167 See Vietzke, *supra* note 127, at 8 (observing that the business method exception “has remained a significant and difficult obstacle within the PTO to some software-related inventions”).

168 See id. (“In comparison to the mathematical algorithm exception, the business method exception is much less well defined.”).


171 Id. at 403.

172 160 F. 467 (2d Cir. 1908).

from committing fraud. In an oft-cited passage, the Second Circuit found that "[a] system of transacting business disconnected from the means for carrying out the system is not, within the most liberal interpretation of the term, [patentable material]." Thus, the business method exception was born.

Scholars have criticized the business method exception for a variety of reasons. First, courts have never properly defined the term "business method." Furthermore, the analytic value of the exception is suspect. One scholar has commented: "Nearly every case that supposedly invoked this rule simply restated the longstanding proposition that naked ideas, bereft of anything physically inventive, are not patentable. At best, these allusions to business were unnecessary. At worst, they caused confusion."

Moreover, the business method exception was merely dictum in the decision that purportedly created the exception. As Judge Newman noted in her Schrader dissenting opinion, "the [Hotel Security Checking]..."
CORNELL LAW REVIEW

Checking] court discussed the 'obviousness' of the system of records kept to prevent embezzlement by waiters at considerably greater length than whether the subject matter was 'statutory.'

Finally, the exception is itself an exercise in meaningless complexity. Judge Newman commented: "[T]he jurisprudence does not require the creation of a distinct business class of unpatentable subject matter. [Hotel Security Checking and its progeny] simply reaffirm that the patent system is directed to tangible things and procedures, not mere ideas." In other words, courts could almost universally have invalidated any patent claiming an unpatentable business method on grounds other than the business method exception. Nonetheless, the PTO incorporated the business method exception into its lexicon and has, until recently, used it to challenge numerous patent applications.

181 In re Schrader, 22 F.3d 290, 298 (Fed. Cir. 1994) (Newman, J., dissenting).
182 Id. (citation omitted).
183 See State St. Bank & Trust Co. v. Signature Fin. Group, Inc., 149 F.3d 1368, 1375 (Fed. Cir. 1998) ("The business method exception has never been invoked by this court, or the CCPA, to deem an invention unpatentable. Application of this particular exception has always been preceded by a ruling based on some clearer concept of [the Patent Act] or, more commonly, application of the [mathematical algorithm] exception . . . ."). cert. denied, 525 U.S. 1093 (1999). A patent attorney described the confusion created by the Hotel Security Checking decision as follows:

While the seemingly clear import of the [Hotel Security Checking holding] was that an invention of a process had to be directed to a physical means, the [case] would be, for nearly a century, enshrined as holding that all business systems were per se unpatentable.

This pattern would persist. Courts would declare that there must be a physical nexus by the employment of an inventive physical means. These cases would then be fallaciously recited for the principle that business methods are not patentable. As time passed, these misinterpreted cases were queued up by authors to lend support to the myth that business systems or methods are per se improper subject matter for patents. A phantasmic body of law had been created.

184 Del Gallo, supra note 170, at 408. But see In re Patton, 127 F.2d 324, 327-28 (C.C.P.A. 1942) ("[A] system of transacting business, apart from the means for carrying out such system, is not within the purview of [the Patent Act] . . . ."). Although Patton seems to suggest that business methods are per se unpatentable, one can also read the passage as simply requiring that a patent claim an implementation of a method, and not simply the abstract method. Such a reading supports Del Gallo's hypothesis and comports with the Supreme Court's statutory subject matter framework, as discussed supra Part IA. Cf Del Gallo, supra note 170, at 415 ("Where the physical means for carrying out a business method have been novel and inventive, patents have been upheld as within the purview of subject matter eligibility.").

Until recently, the official patent examiners' manual contained a reference to the business method exception. See supra note 175; see also In re Schrader, 22 F.3d at 297-98 (Newman, J., dissenting) ("The [PTO] Board [of Appeals] . . . relied on the 'method of doing business' ground for finding Schrader's subject matter non-statutory under section 101."); In re Howard, 394 F.2d 869, 870 (C.C.P.A. 1968) ("The [PTO] Board of Appeals affirmed the [patent examiner's] rejection . . . 'since the claims are drawn to a method of doing business.'" (quoting the opinion of the PTO Board)); In re Patton, 127 F.2d at 327-28 (affirming under the business method exception the PTO Board's rejection of a patent covering a fire-fighting apparatus for use during air attacks).
The business method exception poses a high hurdle for software developers, because many software applications simply automate methods of doing business. For instance, the popular personal finance software package Quicken® basically automates the process of writing and recording checks. If any part of this application is novel and useful, it should normally qualify for patent protection. Since the application is primarily a method of doing business, however, one could argue that the entire invention is nonstatutory under the business method exception.

Recognizing this dilemma, Judge Newman in her Schrader dissent sought to put the business method exception to rest:

Indeed [the business method exception] is fuzzy; and since it is also an unwarranted encumbrance to the definition of statutory subject matter in section 101, my guidance is that it be discarded as error-prone, redundant, and obsolete. It merits retirement from the glossary of section 101.

... I discern no purpose in perpetuating a poorly defined, redundant, and unnecessary "business methods" exception, indeed enlarging (and enhancing the fuzziness of) that exception by applying it in this case.185

Unfortunately, until its decision in State Street Bank, the Federal Circuit declined to follow Judge Newman's advice; instead, it allowed the uncertainty generated by the business method exception to continue to confound judges and scholars alike.186

II

STATE STREET BANK: A WATERSHED

In the spring of 1996, the U.S. District Court for the District of Massachusetts decided State Street Bank & Trust Co. v. Signature Financial Group, Inc.,187 a suit between two mutual fund management corpo-
rations. The suit was a declaratory action in which State Street Bank &
Trust ("State Street Bank") sought to invalidate U.S. Patent No.
5,193,056 (the "'056 Patent"),\textsuperscript{188} assigned to Signature Financial
Group ("Signature").\textsuperscript{189} The '056 Patent covered a "'data processing
system'" for administering a "'Hub and Spoke'" configuration of mu-
tual funds.\textsuperscript{190} The patent consisted of six claims, all using the means-
plus-function format, only the first of which was an independent
claim.\textsuperscript{191} Basically, the Hub and Spoke configuration involves an in-

\begin{footnotesize}
\singlespacing
\begin{itemize}
  \item[188] For a discussion of the '056 Patent's specification and claim, see \textit{infra} notes 190-91
  and accompanying text.
  \item[189] \textit{See State St. Bank}, 927 F. Supp. at 504.
  \item[190] \textit{Id.} at 504 (quoting '056 Patent). The patent's specification described the invention
  as following:

  The present invention provides a data processing system and method
  for monitoring and recording the information flow and data, and making
  all calculations, necessary for maintaining a partnership portfolio and part-
  ner fund (Hub and Spoke) financial services configuration. In particular,
  the data processing system provides means for a daily allocation of assets of
  two or more funds (Spokes) that are invested in a portfolio (Hub). The
  data processing system determines the percentage share (allocation ratio)
  that each fund has in the portfolio, while taking into consideration daily
  changes both in the value of the portfolio's investment securities (as deter-
  mined by market prices) and in the amount of each fund's assets (as deter-
  mined by daily shareholder purchases and redemptions). The system also
  allocates to each fund the portfolio's daily income, expenses, and net real-
  ized and unrealized gain or loss, calculating each fund's total investments
  based on the concept of a book capital account, thus enabling determina-
  tion of a true asset value of each fund and accurate calculation of allocation
  ratios between the funds. The data processing system also tracks all the
  relevant data, determined on a daily basis for the portfolio and each fund,
  so that aggregate year-end income, expenses, and capital gain or loss can be
  determined for accounting and for tax purposes for the portfolio and for
  each fund.

  U.S. Patent No. 5,193,056 to Boes, col. 4, issued Mar. 9, 1993 (Data Processing System for
  Hub and Spoke Financial Services Configuration) [hereinafter '056 Patent].
  \item[191] \textit{See State St. Bank}, 927 F. Supp. at 505. The lone independent claim of the '056
  Patent recited the following:

  A data processing system for managing a financial services configuration of
  a portfolio established as a partnership, each partner being one of a plural-
  ity of funds, comprising:

  \begin{enumerate}
    \item[(a)] computer processor means for processing data;
    \item[(b)] storage means for storing data on a storage medium;
    \item[(c)] first means for initializing the storage medium;
    \item[(d)] second means for processing data regarding assets in the portfolio and
      each of the funds from a previous day and data regarding increases or de-
      creases in each of the funds, assets and for allocating the percentage share that
      each fund holds in the portfolio;
    \item[(e)] third means for processing data regarding daily incremental income, ex-
      penses, and net realized gain or loss for the portfolio and for allocating such
      data among each fund;
    \item[(f)] fourth means for processing data regarding daily net unrealized gain or
      loss for the portfolio and for allocating such data among each fund; and
    \item[(g)] fifth means for processing data regarding aggregate year-end income, ex-
      penses, and capital gain or loss for the portfolio and each of the funds.
  \end{enumerate}

\end{itemize}
\end{footnotesize}
vestment structure in which mutual funds ("Spokes") pool their assets in an investment portfolio ("Hub"), organized as a partnership.\textsuperscript{192} Because of the unique administrative challenges inherent to the complex financial structure, Signature created a computer program to manage the financial calculations required for the daily maintenance of the fund and to store the resulting information for accounting purposes.\textsuperscript{193}

State Street Bank’s declaratory action arose out of failed licensing negotiations between the parties.\textsuperscript{194} State Street Bank served as custodian for several multi-tiered mutual funds of a form similar to those described in the ‘056 Patent\textsuperscript{195} and attempted to acquire a license for the software from Signature. When the negotiations broke down, presumably in an effort to preempt an infringement suit by Signature, State Street Bank sought declaratory judgment on four counts: noninfringement, patent misuse, patent invalidity, and unenforceability due to inequitable conduct.\textsuperscript{196} Signature counterclaimed, alleging unfair and deceptive trade practices under Massachusetts state law and seeking a declaratory judgment that State Street Bank had entered into a valid and binding licensing agreement for the ‘056 patent.\textsuperscript{197}

In ruling on State Street Bank’s motion for partial summary judgment, the district court held that the patent for the claimed invention was invalid because it fell within both the mathematical algorithm and business method exceptions.\textsuperscript{198} The court then dismissed both of Signature’s counterclaims.\textsuperscript{199} Signature subsequently appealed the district court’s decision to the Federal Circuit.\textsuperscript{200}

A. A New Look at Old Requirements

On appeal, the Federal Circuit determined that the substantive issue—whether the patent claimed statutory subject matter—was a

\textsuperscript{192} See State St. Bank, 927 F. Supp. at 504.
\textsuperscript{193} See id. at 505.
\textsuperscript{194} See id. at 506.
\textsuperscript{195} See id.
\textsuperscript{196} See id. at 504 n.1. Noninfringement, invalidity, and patent misuse are all affirmative defenses to a patentee’s allegations of infringement. See 35 U.S.C. § 282 (1994) ("The following shall be defenses in any action involving the . . . infringement of a patent and shall be pleaded: (1) Noninfringement . . . , (2) Invalidity of the patent or any claim in suit . . ."); 6 CHISUM, supra note 23, § 19.01, at 19-5 ("The third major defense [to an infringement action] is [patent] misuse . . . . If a patent owner exploits his patent in an improper manner by violating the antitrust laws or extending the patent beyond its lawful scope, the courts will withhold any remedy for infringement . . .").
\textsuperscript{197} See State St. Bank, 927 F. Supp. at 504 n.1, 516.
\textsuperscript{198} See id. at 514-16.
\textsuperscript{199} See id. at 517.
The court then held that de novo review was proper for both allegedly fact-based questions relating to claim construction and issues related to statutory construction. In reversing the district court, the Federal Circuit held that the '056 Patent's claims "are directed to statutory subject matter." The Federal Circuit began its discussion by criticizing the district court's characterization of the patent's machine (means-plus-function) claims as processes: "'[M]achine' claims having 'means' clauses may only be reasonably viewed as process claims if there is no supporting structure in the written [specification] that corresponds to the claimed 'means' elements." For instance, although claim 1 refers only to a "computer processor means," the specification disclosed "a personal computer," and the Federal Circuit viewed the claim as limited to the disclosed physical structure. More importantly, the court noted:

[F]or the purposes of a § 101 [subject matter] analysis, it is of little relevance whether [a patent claim] is directed to a "machine" or a "process," as long as it falls within at least one of the four enumerated categories of patentable subject matter, "machine" and "process" being such categories.

This statement contrasts sharply with the prior confusion regarding the importance of claim format in patentability determinations and may significantly simplify the § 101 analysis in future cases.

The Federal Circuit then addressed the district court's holding that Signature's patent fell within the mathematical algorithm and business method exceptions to statutory subject matter. The panel was clearly hostile to going beyond the three traditional exceptions.

---

201 See id.
202 See id.
203 Id.
204 Id. at 1371. For an explanation of the importance of including physical limitations in the patent's specification, see supra note 84.
205 '056 Patent, supra note 190, col. 13.
206 Id. col. 7.
207 See State St. Bank, 149 F.3d at 1371-72.
208 Id. at 1372.
209 See supra notes 72-79 and accompanying text for a discussion of the central role that claim format formerly had on patentability. The court went on to note that "although we do not make this determination here, the judicially created exceptions, i.e., abstract ideas, laws of nature, etc., should be applicable to all categories of statutory subject matter . . ." State St. Bank, 149 F.3d at 1372 n.1.
Before examining the disputed exceptions individually, the court noted:

The plain and unambiguous meaning of § 101 is that any invention falling within one of the four stated categories of statutory subject matter [i.e., process, machine, article of manufacture, or composition of matter] may be patented, provided it meets the other requirements for patentability set forth in [the Patent Act], i.e., those found in §§ 102, 103, and 112 [novelty, utility, nonobviousness, and adequacy of disclosure, respectively].

The repetitive use of the expansive term “any” in § 101 shows Congress’s intent not to place any restrictions on the subject matter for which a patent may be obtained beyond those specifically recited in § 101. . . . Congress intended § 101 to extend to “anything under the sun that is made by man.” Thus, it is improper to read limitations into § 101 on the subject matter that may be patented where the legislative history indicates that Congress clearly did not intend such limitations.211

Thus, the court reiterated the rationale behind its holding in Alappat that “it is improper to read into § 101 limitations as to the subject matter that may be patented where the legislative history does not indicate that Congress clearly intended such limitations.”212 Having established the background principles for § 101 inquiries, the court proceeded to deal individually with each exception.

The court first examined the mathematical algorithm exception. It rejected the district court’s use of the Freeman-Walter-Abele test and noted that, even if the two-step test adequately determines whether the claim contained a mathematical algorithm, “after Diehr and Alappat, the mere fact that a claimed invention involves inputting numbers, calculating numbers, outputting numbers, and storing numbers, in and of itself, would not render it nonstatutory subject matter, unless, of course, its operation does not produce a ‘useful, concrete and tangible result.’”213 Furthermore, the court noted:

The question of whether a claim encompasses statutory subject matter should not focus on which of the four categories of subject matter a claim is directed to—process, machine, manufacture, or composition of matter—but rather on the essential characteristics of the subject matter, in particular, its practical utility. Section 101 specifies that statutory subject matter must also satisfy the other “conditions and requirements” of [the Patent Act], including novelty, nonobviousness, and adequacy of disclosure and notice.214

---

211 State St. Bank, 149 F.3d at 1372-73 (quoting Diamond v. Chakrabarty, 447 U.S. 303, 309 (1980) (footnotes and citations omitted)).

212 In re Alappat, 33 F.3d 1526, 1542 (Fed. Cir. 1994).

213 State St. Bank, 149 F.3d at 1374 (quoting In re Alappat, 33 F.3d at 1544).

214 Id. at 1375 (footnote omitted).
Taken together, these two statements imply that the Federal Circuit has abandoned the old notion of the mathematical algorithm exception and replaced it with the statutory characteristics required of all patentable inventions. First, the invention must fall into one of the four enumerated categories of statutory patentable subject matter. Second, the invention must be novel, useful, and nonobvious.

In determining whether Signature’s invention deserved patent protection, the court did not even consider whether an algorithm was present in the claimed invention. The court replaced the former judicial requirement that a claim embodying an algorithm effect a physical transformation with the statutory requirements of novelty and utility.215 This new approach is logically appealing: If a software-related invention is new and useful, it is necessarily an application of an algorithm. An algorithm in the abstract can never have utility; only a specific application of an algorithm has sufficient utility to garner patent protection.216 The new inquiry follows the original rationale of the mathematical algorithm exception, recognizing that a patent should not preempt all future uses of the algorithm.217 The new inquiry, however, allows a developer to secure protection for a software invention that employs a mathematical algorithm to achieve a new and useful result.218

The parallel between mathematical algorithms and abstract ideas further supports the court’s holding in State Street Bank: While abstract ideas are themselves ineligible for patent protection, useful applications of those ideas are patentable.219 Similarly, while one may not patent an algorithm standing alone, a new and useful application of that algorithm is patentable.220 Accordingly, an invention may constitute statutory subject matter, “even if the useful result [of the invention] is expressed in numbers, such as price, profit, percentage, cost, or loss.”221

After disposing of the mathematical algorithm exception, the court targeted the business method exception. Noting that “[t]he

215 See id. at 1374-75.
216 Because an algorithm, like an abstract idea, does not actually accomplish a tangible or useful result, it cannot have utility. See Rubber-Tip Pencil Co. v. Howard, 87 U.S. (20 Wall.) 498, 507 (1874) (“An idea of itself is not patentable . . . .”).
217 See State St. Bank, 149 F.3d at 1374.
218 See id. at 1375.
220 See State St. Bank, 149 F.3d at 1373-75. Compare Rubber-Tip Pencil, 87 U.S. (20 Wall.) at 507 (“An idea of itself is not patentable, but a new device by which it may be made practically useful is.”), with State St. Bank, 149 F.3d at 1373 (“[T]he transformation of data . . . by a machine through a series of mathematical calculations . . . constitutes a practical application of a mathematical algorithm, formula, or calculation, because it produces a useful, concrete and tangible result . . . .” (internal quotation marks omitted)).
221 State St. Bank, 149 F.3d at 1375.
business method exception has never been invoked by [the Federal Circuit], or the CCPA, to deem an invention unpatentable,” the court decided to “take this opportunity to lay this ill-conceived exception to rest.” The court then observed that “[a]pplication of this particular exception has always been preceded by a ruling based on some clearer concept of [the Patent Act] or, more commonly, application of the abstract idea exception based on finding a mathematical algorithm.”

Like its rejection of the mathematical algorithm exception, the court’s dismemberment of the business method exception reflects the underlying premise that courts should not limit the patentable subject matter under §101 when congressional intent indicates an expansive view of patentability. In discussing several of the cases purportedly relying on the business method exception to invalidate a patent, the court noted that the basis of the rejections in all the cases was either that the patent merely claimed an abstract idea, or that the claimed invention failed one of the novelty, utility, and nonobviousness requirements of the Patent Act. The court further stated:

Even the case frequently cited as establishing the business method exception to statutory subject matter, Hotel Security Checking Co. v. Lorraine Co., did not rely on the exception to strike the patent. In that case, the patent was found invalid for lack of novelty and "invention," not because it was improper subject matter for a patent.

Rather than endorsing a special subject matter exception for business methods, the Federal Circuit agreed with the most recent PTO patent examination guidelines: “Claims should not be categorized as methods of doing business. Instead such claims should be treated like any other process claims.”

Thus, the Federal Circuit in State Street Bank expressed its disapproval of subject matter exceptions beyond the traditional prohibitions against patenting abstract ideas, laws of nature, and natural

222 Id.
223 Id.
224 See In re Alappat, 33 F.3d 1526, 1542 (Fed. Cir. 1994).
225 See State St. Bank, 149 F.3d at 1375-76; see also In re Schrader, 22 F.3d 290 (Fed. Cir. 1994) (rejecting the claimed invention as an abstract idea under the mathematical algorithm exception); In re Meyer, 688 F.2d 789 (C.C.P.A. 1982) (same); In re Maucorps, 609 F.2d 481 (C.C.P.A. 1979) (same).
226 See State St. Bank, 149 F.3d at 1375 & n.12; see also In re Howard, 394 F.2d 869 (C.C.P.A. 1968) (rejecting the patent for lack of novelty, without reaching the business method exception); Dann v. Johnston, 425 U.S. 219 (1976) (invalidating the patent under § 103's nonobviousness requirement, without reaching the business method exception).
227 State St. Bank, 149 F.3d at 1376 (citation and footnote omitted).
228 Id. at 1377 (quoting Examination Guidelines for Computer-Related Inventions, 61 Fed. Reg. 7478, 7479 (1996)).
phenomena. In so doing, the Federal Circuit brought software-related inventions back under the aegis of traditional patent subject matter analysis. Both business method and mathematical algorithm claims that fall within the four enumerated classes of statutory subject matter need only comply with the Patent Act's requirements of novelty, utility, and nonobviousness in order to garner patent protection. Such claims do not need to meet specialized requirements of physical transformation or limit themselves to technical subject matter. As long as they claim practical applications of an algorithm or a business method, rather than abstract ideas, they constitute statutory subject matter. Additionally, the claims need not fall into any particular classification of subject matter; they may claim processes, machines, or articles of manufacture.

B. Implications for Software Patents

To gauge the potential impact of State Street Bank, one need only look at the parties that filed amicus curiae briefs in the case. The Information Technology Industry Council (ITI), as well as both Visa and Mastercard, filed amicus briefs before the Federal Circuit. As of 1996, the PTO had issued 1331 patents covering banking and business transactions and undoubtedly issued countless more patents for other software programs. State Street Bank provides the holders of these patents with an unprecedented degree of certainty. Perhaps more significantly, it has opened the door to countless other inventors seeking patent protection for software-related inventions.

---

229 See id. at 1373 ("The repetitive use of the expansive term 'any' in § 101 shows Congress's intent not to place any restrictions on the subject matter for which a patent may be obtained beyond those specifically recited in § 101.").

230 See supra notes 210-12 and accompanying text.

231 See supra notes 210-12 and accompanying text.

232 See supra note 215 and accompanying text.

233 See supra notes 215-21 and accompanying text.

234 See State Street Bank, 149 F.3d at 1372 & n.2. Although this portion of the State Street Bank decision was dictum because the claims at issue did indeed read on machines, the Federal Circuit explicitly adopted that rationale in its later decision, AT&T Corp. v. Excel Communications, Inc., 172 F.3d 1352, 1357-58 (Fed. Cir.) (holding that "the scope of § 101 [is] the same regardless of the form—machine or process—in which a particular claim is drafted"), cert. denied, 120 S. Ct. 368 (1999).

235 The Information Technology Industry Council "represents the leading U.S. providers of information technology products and services ... [which] had worldwide revenues of $405 billion in 1996[,] ... employ more than 1.5 million in the United States ... [and] are responsible for ... over 50% of all information technology research and engineering." About ITI, supra note 1. ITI has 28 members, including Apple Computer, Cisco Systems, IBM, Sony Electronics, and Xerox. See id.


237 See ITIC Amicus Brief, supra note 24, at 13-14.
Admittedly, the patent system may be less than a perfect vehicle for protecting software-related inventions.\textsuperscript{238} However, no other existing regime can provide better protection for software-related inventions;\textsuperscript{239} existing proposals for sui generis protection schemes suffer both theoretical and practical deficiencies.\textsuperscript{240} Consequently, patent protection still presents the best scheme of protection for software-related inventions.

The Federal Circuit's \textit{State Street Bank} decision provides a new level of certainty for inventors seeking protection for their software-related inventions.\textsuperscript{241} \textit{State Street Bank} (and more explicitly, \textit{AT&T v. Excel}) disposed of the necessity of drafting claims in a particular manner to obtain patent protection for software-related inventions.\textsuperscript{242} Prior to the \textit{State Street Bank} decision, claim format was often dispositive in litigation involving a patent containing mathematical algorithms.\textsuperscript{243} Scholars debated the relative merits of identifying inventions by using process claims, machine claims, or article of manufacture claims.\textsuperscript{244} Some scholars advocated machine claims in the means-plus-function format for software-related inventions.\textsuperscript{245} Courts

\begin{itemize}
\item \textsuperscript{238} See Samuelson et al., \textit{ supra} note 20, at 2343-47 (describing features unique to software that render the present patent system an ineffective form of software protection). Moreover, since the Federal Circuit decided \textit{State Street Bank}, many commentators in the software and electronic commerce industries have declared the current patent system an unfit method of protecting computer software and methods of conducting business transactions online. \textit{See infra} note 260.
\item \textsuperscript{239} \textit{See infra} Part IIIA.1-3.
\item \textsuperscript{240} \textit{See infra} Part IIIA.4.
\item \textsuperscript{241} Amendments to the current Patent Act, however, could make the current patent system even more amenable to the protection of software-related inventions. \textit{See infra} Part III.B.
\item \textsuperscript{242} \textit{See State St. Bank}, 149 F.3d at 1372; \textit{see also} \textit{AT&T Corp. v. Excel Communications}, Inc., 172 F.3d 1352, 1357-58 (Fed. Cir.) (discussing \textit{State Street Bank} and concluding that "whether the invention is a process or machine is irrelevant" for the purposes of § 101 analysis), \textit{cert. denied}, 120 S. Ct. 368 (1999).
\item \textsuperscript{243} See Chiappetta, \textit{ supra} note 53, at 108-10 (noting that, under the \textit{Freeman-Walter-Abele} test, software claims were virtually unpatentable without a physical structure in the claims, forcing a claim format that presented the invention as a machine or article of manufacture); Kass, \textit{ supra} note 49, at 830 (arguing that the \textit{Alappat} court allowed the patentee's claims primarily because they utilized means-plus-function format); Yoshida, \textit{ supra} note 43, at 479 ("[C]ases decided subsequent to \textit{Diehr} show that the substance of a program can indeed be patented, as long as the claim description includes some reference to a means for executing the program's functions.").
\item \textsuperscript{244} See Kass, \textit{ supra} note 49, at 807 ("Practitioners and commentators have developed several approaches to satisfying the \textit{Benson}, \textit{Flook}, and \textit{Diehr} requirement of physical transformation for processes that involve mathematical algorithms. Such approaches usually entail linking the algorithm or program with otherwise patentable subject matter to pigeon-hole the invention into a statutory category . . . .")
\item \textsuperscript{245} \textit{See id.} at 850. Lawrence Kass explains:
\begin{quote}
If [a means-plus-function] claim format is accorded substantive effect, a software or mathematical algorithm "process" claim, phrased as a "computer means for" accomplishing the software or algorithm's function, becomes a "machine" claim, so long as the specification recites a computer.
\end{quote}
tended to view such claims as more "physical" than mere process claims, which might not recite any physical structure at all. Other scholars advocated describing software as an article of manufacture and claiming a floppy disk as the invention, instead of the true invention, the program itself. These scholars asserted that the physical changes in the disk caused by the software constituted the necessary physical transformation to render a patent claim valid.

*State Street Bank* and its progeny eliminated the need for such artful drafting. In the words of the court, "for the purposes of a § 101 analysis, it is of little relevance whether [a patent’s claim] is directed to a ‘machine’ or a ‘process,’ as long as it falls within at least one of the four enumerated categories of patentable subject matter,” which include processes, machines, and articles of manufacture. In the wake of *State Street Bank* and *AT&T*, inventors are now free to claim their inventions in whichever format provides the most effective protection.

*State Street Bank* also dismissed the mathematical algorithm exception as an independent rationale to reject patent protection for software-related inventions. Since its inception in *Benson*, the mathematical algorithm exception had proven a significant obstacle to inventors seeking protection for software-related inventions. Moreover,

As a result, the claim would be automatically deemed patentable subject matter under § 101.

*Id.* The patent at issue in *In re Freeman* successfully employed this strategy. *See In re Freeman*, 573 F.2d 1237, 1238 (C.C.P.A. 1978) (validating claims framed in means-plus-function format, which covered "a system for typesetting alphanumeric information, using a computer-based control system in conjunction with a phototypesetter of conventional design").

See Kass, supra note 49, at 866 (noting that the courts have perpetuated a "legal fiction of attributing physicality to software claims drafted in means-plus-function format" and arguing that "the patent practitioner should always claim software inventions as machines or apparatuses in means-plus-function format"); Yoshida, supra note 43, at 479 (arguing that the inclusion of physical means in claims has enabled inventors to claim software that would not have been patentable if claimed as a process).

See, e.g., Chiappetta, supra note 53, at 114. ("[The article of manufacture claim format] is preferable to providers of software products. It offers the same escape from the mathematical algorithm/[Freeman-Walter-Abele] difficulties and provides better enforcement options than the machine format."). *In re Beauregard* features a patent that utilizes the article of manufacture claim format. *See In re Beauregard*, 53 F.3d 1583, 1584 (Fed. Cir. 1995) (vacating the PTO Board’s rejection of a patent that claimed a floppy disk encoded with a software program as an article of manufacture).

See Chiappetta, supra note 53, at 118-19 (discussing *In re Lowry*, 32 F.3d 1579 (Fed. Cir. 1994), and noting the Federal Circuit’s reliance on "the physical changes in the medium caused by the data structure" in its article of manufacture analysis).


See *State St. Bank*, 149 F.3d at 1373-75.
when the inventors did manage to obtain patents for such inventions, they faced uncertainty regarding the viability of those patents in infringement actions.\textsuperscript{251} Now, however, the Federal Circuit, most often the final authority on patent matters,\textsuperscript{252} has removed the hurdle of the mathematical algorithm exception from the path of these inventors.

Finally, \textit{State Street Bank} also extinguished the controversial business method exception. Although, as the Federal Circuit itself noted, the PTO had already removed the business method exception from its lexicon by the time of the \textit{State Street Bank} decision,\textsuperscript{253} the exception continued to haunt patent holders seeking to enforce patents claiming business methods;\textsuperscript{254} indeed, respected authorities continued to list business methods among the exceptions to statutory subject matter.\textsuperscript{255} In \textit{State Street Bank}, the Federal Circuit finally decided to "lay

\textsuperscript{251} See Chiappetta, supra note 53, at 91 (describing mathematical algorithm jurisprudence as a "30-year quest for a solution"); Gable, supra note 88, at 22 ("Despite the [Supreme] Court's express limitation, \textit{Benson} had the consequence of suggesting that a computer programs [sic] are unpatentable."); Yoshida, supra note 43, at 457 ("Since the [\textit{Benson} era], the [PTO] and the courts have struggled to resolve which computer program-related inventions represent patentable subject matter and which fall under the exceptions to patentability relating to laws of nature and ideas.").

\textsuperscript{252} See supra notes 32-40 and accompanying text. Moreover, because the Supreme Court has denied \textit{State Street Bank}'s petition for certiorari, the Federal Circuit's \textit{State Street Bank} decision presently stands as a conclusive statement of the law. See \textit{State St. Bank & Trust Co. v. Signature Fin. Group, Inc.}, 525 U.S. 1093 (1999); see also \textit{AT&T Corp. v. Excel Communications, Inc.}, 172 F.3d 1352, 1357-58 (Fed. Cir.) (following and extending the rationale of \textit{State Street Bank}), cert. denied, 120 S. Ct. 368 (1999).

\textsuperscript{253} See \textit{State St. Bank}, 149 F.3d at 1377. On the PTO's recent treatment of the business method exception, the Federal Circuit noted:

\begin{quote}
[I]t comes as no surprise that in the most recent edition of the Manual of Patent Examining Procedures (MPEP) (1996), a paragraph [authorizing patent examiners to reject patents under the business method exception,] was deleted. . . . This acknowledgement is buttressed by the U.S. Patent and Trademark 1996 Examination Guidelines for Computer Related Inventions which now read:

Office personnel have had difficulty in properly treating claims directed to methods of doing business. Claims should not be categorized as methods of doing business. Instead such claims should be treated like any other process claims.

\end{quote}


\begin{quote}
Numerous patent treatises recite the long-established principle that business plans and systems are not patentable . . . . In effect, the '056 Patent grants \textit{Signature} a monopoly on its idea of a multi-tiered partnership portfolio investment structure . . . . Because such abstract ideas are not patentable . . . as methods of doing business . . . , the '056 Patent must fail.
\end{quote}

\textit{Id.} (internal quotation marks omitted).

\textsuperscript{255} See, e.g., 1 \textit{CHISUM}, supra note 23, § 1.03[5], at 1-75 ("The decisions hold that business 'plans' and 'systems' are not patentable, even though they may not be dependent upon the aesthetic, emotional, or judgmental reactions of a human.").
this ill-conceived exception to rest."\textsuperscript{256} As a result, the thousands of patents covering business methods have gained newfound security.\textsuperscript{257}

The \textit{State Street Bank} decision's greatest significance lies in the cumulative impact of the individual rules that it established. Taken together, the rejection of the business method exception, the limitation of the mathematical algorithm exception, and the holding that claim format is irrelevant with respect to the subject matter inquiry establish software as deserving of patent protection as any other, traditional technology. After \textit{State Street Bank}, the PTO and courts will examine a software-related invention under the traditional statutory requirements of novelty, utility, and nonobviousness; they will no longer scrutinize whether the invention includes a mathematical algorithm or business method, or whether the claims are formatted as processes, machines, or articles of manufacture. Thus, if a software-related invention "produces a 'useful, concrete, and tangible result' \ldots, even if the useful result is expressed in numbers," the invention exhibits the qualities that the Framers of the Constitution, as well as Congress, established as prerequisites for awarding a patent monopoly.\textsuperscript{258}

\section*{III}
\textbf{PATENT: THE OPTIMAL FORM OF PROTECTION FOR SOFTWARE?}

\textit{State Street Bank}'s elevation of software to the ranks of generally patentable technologies raises the issue of whether software should be eligible for such patent protection. The Federal Circuit certainly supports software patentability, as long as the claims meet the novelty, utility, and nonobviousness requirements of the Patent Act. Commentators, however, do not universally agree.

Scholars have raised various arguments against affording patent protection to software-related inventions. Some point to the success of the software industry to date, despite the absence of the certainty of patent protection provided by \textit{State Street Bank}.\textsuperscript{259} Others acknowled-

\textsuperscript{256} \textit{State St. Bank}, 149 F.3d at 1375.
\textsuperscript{257} Of course, the decision to finally provide certain protection to business method patents has engendered controversy of its own. See infra note 260.
\textsuperscript{258} \textit{State St. Bank}, 149 F.3d at 1375 (quoting \textit{In re Alappat}, 33 F.3d 1526, 1544 (Fed. Cir. 1994)).
\textsuperscript{259} See, e.g., Against Software Patents: The League for Programming Freedom, 14 Hastings Comm. & Ent. L.J. 297, 297 n.\^{} (1992) ("New monopolies, known as software patents \ldots, have taken away our freedom of expression and our ability to do a good job."); Lee A. Hollaar, \textit{Justice Douglas Was Right: The Need for Congressional Action on Software Patents}, 24 AIPLA Q.J. 283, 285 (1996) (noting that "[p]ast commentators have questioned the need for, or advisability of, patents for software-based inventions" and discussing the reasons for their skepticism). Similarly, Professor Goldstein has noted that anyone trying to promote enhanced intellectual protection for software "will need to explain why—if copyright, patent, and trade secret law are so lacking—there was such an extraordinary outpouring of
edge that market failures inherent to the software market may prevent the full growth of the software industry without some form of intellectual property protection, but argue that patent protection is not the best remedy for these failures.\footnote{\textsuperscript{260}} Assuming that software needs some software innovation during the 1980s, when no alternative forms of intellectual property protection were in prospect." Paul Goldstein, Comment, *Comments on A Manifesto Concerning the Legal Protection of Computer Programs*, 94 *COLUM. L. REV.* 2573, 2575 (1994).

\footnote{\textsuperscript{260} For example, Professor Samuelson and her colleagues argue that software inventions are categorically incompatible with the patent system and claim that suitable protection for software can only be found in a new, sui generis regime:}

> [T]he more profound problem with using patent law to protect functional program behavior, user interfaces, and the industrial design of programs that produce behavior is that these innovations are typically of an incremental sort... Patent law requires an inventive advance over the prior art before it grants protection. Protecting incremental innovations in program behavior through patent law would thwart the economic goals of the patent system: to grant exclusive rights only when an innovator has made a substantial contribution to the art and advanced competition to a new level.

Samuelson et al., *supra* note 20, at 2346 (footnotes omitted). Another school of thought disagrees about the necessity and appropriateness of sui generis protection for software, but agrees with Professor Samuelson that patents do not provide the proper protection for software inventions. See, e.g., Jane C. Ginsburg, Comment, *Four Reasons and a Paradox: The Manifest Superiority of Copyright over Sui Generis Protection of Computer Software*, 94 *COLUM. L. REV.* 2559, 2572 (1994) ("It is not at all clear that a noncopyright 'fix' to the system of software protection is available, or desirable, at least as a matter of U.S. law.").

More recently, several commentators have criticized the *Post-State Street Bank* patent system, arguing that it allows patents for developments in software and electronic commerce that should not be patentable. For example, U.S. Patent No. 5,960,411 ("411 Patent"), granted on September 28, 1999 and assigned to Amazon.com, Inc., covers Amazon.com's "one click" ordering system. See U.S. Patent No. 5,960,411 to Hartman et al., issued Sept. 28, 1999 (Method and System for Placing a Purchase Order via a Communications Network). Under this system, a customer seeking to purchase an item from Amazon.com's electronic commerce site need only click one button to effect the transaction, bypassing numerous data entry and confirmation steps. Within a month after receiving the patent, Amazon.com filed for and ultimately received a preliminary injunction against competitor Barnesandnoble.com, Inc., who used a similar system to facilitate transactions with its customers. See Amazon.com, Inc. v. Barnesandnoble.com, Inc., 79 F. Supp. 2d 1228, 1249 (W.D. Wash. 1999). Many insiders in the software and electronic commerce industries have criticized both the "411 Patent and the *Amazon.com* decision, arguing that allowing Amazon.com a patent on such a noninventive development awards Amazon.com an undeserved monopoly and retards the development of the electronic commerce industry as a whole. See, e.g., Lawrence Lessig, *Patent Problems*, STANDARD \textsuperscript{\#} 6 (Jan. 21, 2000) <http://www.thestandard.com/article/display/0,1151,8999,00.html>. Professor Lessig argues:

> [A patent on a method of online commerce, such as Amazon.com's "one click"] gives the holder a monopoly over a way of doing business that gets instantiated in technology. That's troubling enough in real space, where not all ways of doing business can be expressed in technology. But in cyberspace, there is no limit to the potential of this sort of patent. Every method of doing business in cyberspace by definition is instantiated in technology—code. So every method in principle becomes subject to a patent.  

*Id.* However, this Note argues that a distorted interpretation of subject matter under § 101 is not necessary to address the concerns with electronic commerce patents; the other requirements for patentability—novelty, utility, and nonobviousness—adequately address these concerns. See *infra* notes 331-33 and accompanying text. Many of the arguments...
form of intellectual property protection, commentators have posited several alternatives to patent law: trade secret law, copyright law, trademark law, and sui generis protection schemes. However, none of these existing alternatives to patent law can fully protect software, and the theoretical infirmities and administrative difficulties of a sui generis protection scheme will likely prove insurmountable. A closer analysis of the alternatives to patent protection reveals that

---

against software and electronic commerce patents focus on the fact that these patents often contain insufficient "inventive" material to deserve monopoly protection, see, e.g., Seth Shulman, Software Patents Tangle the Web, TECH. REV., ¶ 9 (Mar./Apr. 2000) <http://www.techreview.com/articles/ma00/shulman.htm> ("'The Patent Office is issuing patents for blindingly obvious things just because they are being done with software or on the Internet.' . . . [T]he patents are already causing 'a chilling effect on electronic commerce.'" (quoting Professor James Boyle of American University)), or the PTO's inability to effectively uncover prior art that might render the patented invention obvious or non-novel, see id. ¶ 22-29. Shulman notes:

A key problem is that software programming—especially in its early days—was famous for its lack of a published paper trail and for the informal exchange of code and techniques among programmers. These poor "non-patent" records, combined with the PTO's late arrival to the software game, mean the agency examiners who scrutinize applications often have tremendous difficulty establishing exactly when an invention was first made.

Id., ¶ 23. This Note argues that the past confusion over whether algorithms and business methods are statutory subject matter has improperly shifted the focus from the analysis of novelty, utility, and nonobviousness. The Federal Circuit's State Street Bank decision should refocus attention on these traditional requirements. More importantly, the inter partes examination procedure advocated in this Note, see infra notes 347-49 and accompanying text, would allow interested third parties to intervene in the examination process, drawing to the attention of the PTO any relevant prior art that might render the patent at issue obvious or non-novel. These two factors, working in conjunction, would ensure that every patent granted by the PTO is truly meritorious.

---

261 See, e.g., McDonald, supra note 3, ¶ 5-6 (explaining that, because of the patent system's past ineffectiveness at protecting software, many software "developers were accustomed to maintaining their algorithms and software code as closely guarded secrets").

262 For example, Professor Ginsburg argues:

[T]he . . . essential premise that copyright law ill befits computer programs because the law does not protect works that 'behave,' betrays too cramped an appreciation of the subject matter and scope of copyright protection. Copyright does, to some extent, protect 'behavior,' whether of computer programs or of other works of authorship.

Ginsburg, supra note 260, at 2559-60. Professor Miller also advocates copyright protection for software:

Computer programs and other more traditional literary works, and the processes by which they are created, are quite similar. Computer programs, like other literary works, are expressive. The imagination, originality, and creativity involved in writing a program is comparable to that involved in more time-honored literary works and far exceeds various mundane efforts that have long enjoyed protection under the copyright rubric.

Miller, supra note 19, at 983-84 (footnotes omitted).

263 See, e.g., Kellner, supra note 19, at 1013 ("[C]omputer and software manufacturers should look to trade dress law [a subset of trademark law] to protect the 'look and feel' of their user interfaces.").

264 See Samuelson et al., supra note 20, passim; see also id. at 2312 n.6 (listing various proposals and articles on sui generis protection of software).

265 See infra Part III.A.4.
while some of them complement patent law in protecting software-related inventions, none of them can effectively replace the vital role patent law plays in protecting software.

A. Alternative Forms of Protection

1. Trade Secret

One traditional method that software developers have frequently used to secure their intellectual property is trade secrecy.\(^{266}\) The Uniform Trade Secrets Act provides:

> "Trade secret" means information, including a . . . program . . . that:
> (i) derives independent economic value, actual or potential, from not being generally known to, and not being readily ascertainable by proper means by, other persons who can obtain economic value from its disclosure or use, and
> (ii) is the subject of efforts that are reasonable under the circumstances to maintain its secrecy.\(^{267}\)

A distinguishing feature of trade secrecy is the fact that developers need not, and in fact must not, disclose the protected information to secure protection.\(^{268}\) This feature represents both the advantage and the disadvantage of trade secret protection.

The prohibition against disclosure seems to provide a great benefit to developers. Not only do they get protection for their software, but in so doing they also need not even disclose how they produced it. In contrast, to obtain patent protection, developers must disclose their inventions, potentially allowing others the opportunity to improve upon them. In fact, perhaps because of the past uncertainty regarding software patents, much of the software development community accepted this rationale and kept many, if not most, software development efforts secret.\(^{269}\)

Unfortunately, the mandate of secrecy actually works against the developers and even society as a whole. Rather than promoting information exchange and technological innovation, trade secrecy encourages developers to hoard their inventions; this forces software developers to "spend much of their efforts reinventing the wheel because there is not an adequate collection of how problems have been

\(^{266}\) See McDonald, supra note 3, ¶ 6-7.


\(^{268}\) Compare id. § 1(4)(ii) (requiring "efforts that are reasonable under the circumstances to maintain [the] secrecy" of the invention or information to qualify for trade secret protection), with 35 U.S.C. § 112 (1994) (requiring "a written description of the invention, and of the manner and process of making and using it, in such full, clear, concise, and exact terms as to enable any person skilled in the art . . . to make and use the same" to qualify for patent protection).

\(^{269}\) See Hollaar, supra note 259, at 286 (noting that "[u]ntil recently, patents were discouraged and commercial developers kept their work secret").
solved in the past.”270 While the software industry has prospered in the absence of proper patent protection, even greater innovation might have occurred had patent protection been available as a viable source of protection.271 In fact, Lee Hollaar, a professor of computer science at the University of Utah,272 claims that despite the growth of the software industry, “much software development has not been particularly innovative”273 and attributes the massive growth in the industry to the availability and rapid development of more powerful, less expensive computer hardware.274 This is perhaps due in part to the ease with which hardware manufacturers can patent new computer hardware inventions. Thus, even though individual software developers might have greatly benefitted from the protection of trade secrecy, the basic policy goals behind intellectual property protection demand a form of protection that encourages more information exchange and synergy between competing developers.275 Although other intellectual property protection schemes may allow disclosure and the resultant sharing of information, only one existing scheme requires them: patent law.276

2. Trademark

Trade dress is a form of trademark protection designed to protect the “look and feel” of products and their packaging.277 As such, it has the potential to provide protection for the aesthetic aspects of...

270 Id.
271 See id.; see also McDonald, supra note 3, ¶ 8 (“[T]rade secrecy creates barriers to innovation and progress within the software industry.”).
272 See Hollaar, supra note 259, at 283 n.*.
273 Id. at 285.
274 See id. at 285-86.
275 The ultimate goal of the intellectual property protection system in the United States is not to reward inventors, but to encourage innovation. See supra notes 51-53 and accompanying text.
276 While the Copyright Act provides for deposit and registration of copyrighted works, see 17 U.S.C. §§ 407-412 (1994 & Supp. III 1997), “[s]uch registration is not a condition of copyright protection,” id. § 408(a). However, registration may be a prerequisite to some infringement claims. See id. § 412. In contrast, patent law requires disclosure of the invention as a prerequisite to receiving any type of protection. See 35 U.S.C. § 112 (1994).
277 Trade dress protection arises out of section 43(a) of the Lanham Act, the federal trademark statute codified at 15 U.S.C. § 1125(a). See 1 J. THOMAS McCARTHY, McCARTHY ON TRADEMARKS AND UNFAIR COMPETITION § 8.01[5] (3d ed. 1996). Section 43(a) is often referred to as a federal unfair competition law. See GOLDSTEIN, supra note 72, at 363. In ruling on a trade dress infringement claim, a judge should look at “the total image or overall impression of plaintiff’s product,” in comparison with that of the defendant’s product. 1 MCCARTHY, supra, § 8.01[1][a], at 8-2. If the defendant’s product “is likely to cause confusion, or to cause mistake, or to deceive as to the affiliation, connection, or association” between the plaintiff’s product and the defendant’s, then the plaintiff’s claim should succeed. 15 U.S.C. § 1125(a)(1)(A) (1994).
software interfaces. Although trade dress protection has not been extensively tested in the computer software context, at least one commentator feels that it holds great potential for protecting the aesthetic aspects of computer programs.

However, under trademark law, trade dress protection does not extend to the functional characteristics of a product. Thus, while trade dress may be able to protect the user interface of a particular software application, it necessarily falls short of protecting the vital functionality of the application. Protecting functionality is hence the exclusive domain of patent protection. Nevertheless, trade dress protection may serve as an ideal complement: By patenting the "guts" of the program and obtaining trade dress protection for the look and feel of the interface, a software developer may be able to secure comprehensive protection for the entire application.

3. Copyright

Some developers have attempted to rely on the copyright laws to protect the software code itself. However, copyright, like trademark, is designed to protect not the functional characteristics of inventions, but merely "the literal expression of the software code." In fact, the First Circuit in a recent case held that, while "'[i]t is now well settled that the literal elements of computer programs, i.e., their

---

278 Laura Kellner notes the advantage of trade dress protection over copyright protection:

Copyright law protects individual static elements, but not composite dynamic wholes. Conversely, trade dress law, a subset of trademark law, protects composite wholes because trade dress is the "look and feel" of a product or its packaging. Where copyright law has failed to protect user interfaces, trademark law might succeed.

Kellner, supra note 19, at 1013.

279 See id. at 1017 ("While trade dress infringement claims have become more common in all product areas over the past few years, only a few attorneys have aggressively sought trade dress protection for the 'look and feel' of user interfaces. Most plaintiffs in these cases simply do not pursue trademark protection at all." (footnote omitted)).

280 See id. at 1013 (describing the protection of "'look and feel' of a product" under trade dress law).

281 See id. at 1026 (admitting that, "[t]o prevail in a trademark case, the manufacturer [of the software] must also demonstrate that the trade dress it seeks to protect is 'nonfunctional'"; cf. Michael J. Schallop, Comment, Protecting User Interfaces: Not as Easy as 1-2-3, 45 EMORY L.J. 1533, 1536 (1996) (arguing that "a user interface's functional method of operating a computer should not be protected by copyright, trademark, or trade dress, because . . . such functionality constitutes patentable subject matter and should be protected, if at all, by patent law" (internal quotation marks omitted))).

282 See McDonald, supra note 3, ¶ 7 ("Absent patent protection for the software's underlying idea or concept, developers have relied on the copyright laws to protect the actual expression of the algorithm in the form of the software code itself.").

283 Id., ¶ 9. But see Miller, supra note 19, at 986 ("Works with utilitarian aspects, however, have been accorded protection since our first Copyright Act in 1790, which embraced maps and charts . . . Since [Baker v. Selden, 101 U.S. 99 (1880)], courts have reaffirmed repeatedly that functionality poses no per se bar to copyrightability." (footnotes omitted)).
source and object codes, are the subject of copyright protection," no functional elements of the program may be copyrightable because they fall within the "'method of operation'" exception to copyrightability.

Section 102(b) of the Copyright Act of 1976 excludes from copyright protection "any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied." Moreover, the legislative history of that section clearly indicates that only the actual expression of the software developer—the program's source and object codes and possibly aesthetic interface—and not the underlying functionality of the program deserves copyright protection. Nevertheless, some scholars still champion copyright as the preeminent form of protection for software. However, since the First Circuit's ruling in Lotus Development Corp. v. Borland International, Inc., an increasing number of commentators have advocated scaling back copyright protection so as to cover only the literal program code,


285 Lotus Dev., 49 F.3d at 816.


287 See H.R. Rep. No. 94-1476, at 10 (1976), reprinted in 1976 U.S.C.C.A.N. 5659, 5670 ("Section 102(b) is intended . . . to make clear that the expression adopted by the programmer is the copyrightable element in a computer program, and that the actual processes or methods embodied in the program are not within the scope of the copyright law.").

288 See, e.g., Robert A. Gorman, Commentary, Comments on A Manifesto Concerning the Legal Protection of Computer Programs, 5 Ala. L.J. Sci. & Tech. 277, 280 (1996) ("[The primary criticism of] the suitability of copyright protection for computer programs relates to the fact that programs have the utilitarian objective to achieve functional results in an efficient way. Yet copyright has never been regarded as an inappropriate protective regime for other works that have functional objectives." (footnote and internal quotation marks omitted)). Note, however, that Professor Gorman wrote this article before the Supreme Court's affirmance of Lotus Development, in which the First Circuit limited the scope of protection to nonfunctional elements of the software. See Lotus Dev., 49 F.3d at 815-19.

in either source or object form, and using patent protection to fill the resulting void.  

Conceptually, this approach makes the most sense. The goal of copyright law is to protect nonfunctional, expressive works, while patent law seeks to protect only functional technology. Thus, stretching copyright law to cover functional properties of software makes little sense. Additionally, from a policy standpoint, it would be unwise to grant a monopoly on functional behavior which does not meet the rigorous standards of novelty, utility, and nonobviousness under patent law.

Copyright serves a valuable function in the protection of software-related inventions by prohibiting literal copying of either the executable program code or the source code for the software. It also protects some elements of the user interface. But after *Lotus Development*, copyright can no longer viably protect the underlying functionality of the software itself; in many cases, the underlying functionality is the software’s primary source of value. Functionality is the domain of patent law; unless Congress legislates a sui generis protection scheme,

---

290 See, e.g., Dennis S. Karjala, *Copyright Protection of Computer Program Structure*, 64 BrooK. L. Rev. 519, 522 (1998) ("[C]opyright should protect computer program code from verbatim copying or slavish mechanical or electronic translations. [If o]ther program elements, such as structure, sequence, and organization (‘SSO’) and elements of software interfaces . . . are to receive intellectual property protection at all, that protection should be sought in patent . . . law . . . .") (emphasis omitted); McDonald, *supra* note 3, ¶ 8 ("Copyright law is designed to protect the individual expression of an idea rather than the idea itself.").

291 Congress’s power to grant both patents and copyrights arises out of Article I, section 8 of the Constitution. See U.S. Const. art. I, § 8, cl. 8 (authorizing Congress “[t]o 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”). Upon enacting the 1952 Patent Act, Congress noted:

The background, the balanced construction, and the usage current then and later, indicate that the constitutional provision is really two provisions merged into one. The purpose of the first provision is to promote the progress of science by securing for limited times to authors the exclusive right to their writings, the word “science” in this connection having the meaning of knowledge in general, which is one of its meanings today. The other provision is that Congress has the power to promote the progress of useful arts by securing for limited times to inventors the exclusive right to their discoveries. The first patent law and all patent laws up to a much later period were entitled “Acts to promote the progress of useful arts.”


292 In fact, this Note argues that providing protection for the functional aspects of software without meeting the Patent Act’s standard of inventiveness violates the Patent Clause of the Constitution. See infra notes 356-59 and accompanying text.

293 For example, the most valuable part of Signature’s invention in *State Street Bank* was the software’s functional ability to track the assets and losses of the various mutual funds. See *State St. Bank & Trust Co. v. Signature Fin. Group, Inc.*, 149 F.3d 1368, 1370 (Fed. Cir. 1998), cert. denied, 525 U.S. 1093 (1999). While the interface was certainly important, the underlying processes provided the software’s inherent value.
patent law provides the only comprehensive protection for the functional aspects of a software application.

4. Sui Generis Protection Schemes

Some commentators claim that the existing intellectual property protection regimes cannot effectively protect software-related inventions and argue that a new, sui generis scheme is necessary for software protection. Proponents of the sui generis protection scheme for software contend that "the incremental nature of innovation in software largely precludes patent protection." They liken software to semiconductors and point to the congressional adoption of the Semiconductor Chip Protection Act of 1984 to protect semiconductor-related inventions as evidence that existing regimes cannot adequately protect computer-related inventions.

The most recent and comprehensive proposal for sui generis protection of software inventions is A Manifesto Concerning the Legal Protection of Computer Programs. Samuelson et al., supra note 20. In this seminal article, the authors, comprised of lawyers and computer experts, surveyed the various existing modes of intellectual property protection and concluded that the existing intellectual property legal regimes failed to properly protect software. See id. at 2429-31. Moreover, they argue that patent law and copyright detract from one another by attempting to coexist in the software realm:

In addition, the application of both copyright and patent law to software innovations may impair the effectiveness of both forms of protection. It has also created considerable uncertainty about the scope of protection available from each. No one knows just where the boundary line between these domains does or should lie. The economic goals of both regimes can be thwarted when both are applied to a dual-character subject matter such as computer programs. This is especially true if copyright law, with its long duration of protection, its low creativity threshold, and its automatic protection, is construed so broadly that it encompasses technical innovations that should be regulated by patent law.

Id. at 2346-47 (footnotes omitted). The Manifesto represents the most persuasive and complete argument against software patent protection and in favor of a new system of protection. In addition, it represents one of the few recent proposals for sui generis protection of software. See id. at 2312-13 (conceding that "the idea of sui generis protection for software has generally fallen out of favor"). Thus, this Note will discuss sui generis protection primarily in light of the Manifesto's arguments.

The Manifesto makes the following analogy between software and semiconductor chips:

Congress recognized that the chip industry's products were . . . vulnerable to rapid imitative copying that undermined innovators' ability to recoup research and development costs . . . [S]oftware resembles "semiconductor chips whose industrial designs are rarely inventive. . . . To provide proper incentives for semiconductor designs, Congress passed the Semiconductor Chip Protection Act of 1984 (SCPA). It may eventually need to do the same for computer software, where the typically incremental nature of innovation also impedes the utility of patent protection.

Samuelson et al., supra note 20, at 2346 (footnote omitted).
A Manifesto Concerning the Legal Protection of Computer Programs (the "Manifesto") by Pamela Samuelson, Randall Davis, Mitchell Kapor, and J.H. Reichman advances three arguments against using patent law to protect software inventions.298 The first argument concerns the subject matter hurdle a software developer traditionally had to clear under § 101 before securing patent protection.299 Through a series of decisions culminating in State Street Bank, however, the Federal Circuit has conclusively established that software-related inventions are proper patentable subject matter, provided they meet the other requirements of the Patent Act.300 As a result, the Federal Circuit has rendered moot the Manifesto's subject matter argument.

The Manifesto's second argument contends that, because software inventions are most often innovative rather than inventive in nature, patent protection is inevitably underinclusive and overinclusive.301 In other words, some undoubtedly valuable software innovations may not qualify for patent protection because they fail to meet the Patent Act's requirements of novelty, utility, or nonobviousness,302 while others reserve too great a monopoly, stifling innovation.303

298 See id. at 2308.
299 See id. at 2344-45 ("[T]he Patent Office for nearly two decades, refused patent applications for software innovations on subject matter grounds. The U.S. Supreme Court seemed to concur in this legal position, and its decisions influenced the patent policies of other nations." (footnote omitted)).
300 See State St. Bank & Trust Co. v. Signature Fin. Group, Inc., 149 F.3d 1368, 1374-75 (Fed. Cir. 1998) (rejecting previous statutory subject matter barriers to a software-related patent, as long as it produces "a 'useful, concrete, and tangible result'" and meets the utility, novelty, nonobviousness, and disclosure requirements of the Patent Act), cert. denied, 525 U.S. 1093 (1999).
301 See Samuelson et al., supra note 20, at 2331 ("While innovation in program design occasionally rises to the level of invention, most often it does not."). The authors of the Manifesto use the word "innovative" to describe products of skilled effort that advance the state of the art but do not meet the patent standard of nonobviousness." Id. at 2330 n.67.
302 See id. at 2346 ("[T]he more profound problem with using patent law to protect functional program behavior, user interfaces, and the industrial design of programs . . . is that these innovations are typically of an incremental sort . . . Patent law requires an inventive advance over the prior art before it grants protection." (footnote omitted)).
303 The Manifesto reasons that, because "[i]t is quite possible to produce functionally indistinguishable program behaviors through use of more than one method[,] . . . holding a patent on one method of generating certain results could not prevent the use of another method, even if those results were the program's principal source of value." Id. at 2345 (footnote omitted). In order to prevent other developers from developing around a software patent, the PTO could allow developers to patent the results of the process, instead of the process itself. However, the Manifesto notes that course of action has the potential to stifle competition:

Yet if the Patent and Trademark Office were willing to issue a patent with claims for any means of achieving a particular set of results, such a patent would issue at a high level of generality and would inhibit competition in development of useful program behaviors out of proportion to the innovation actually contributed by the claimant.

Id.
In opposition to this view, Professor Hollaar asserts that the lack of adequate patent protection has created the current development environment in which software advances are incremental and evolutionary, as opposed to inventive and revolutionary. Under his theory, if software developers could rely on patent protection, they would be more willing to disclose their inventions and help produce more rapid, inventive developments in the software field. Thus, the problem of incremental innovation in the software field is not necessarily an argument against patent protection for software, but is more likely a symptom of the historical absence of predictable patent protection for software-related inventions.

Further, Professor Hollaar argues that if more developers secured patent protection for their software-related inventions, other developers’ “concern over patent infringement could . . . accelerate[] the trend in the computer system development industry towards using software modules purchased from [the patent-holding] developers.” The modular software development techniques, which have recently gained momentum with the adoption of C++ and other object-oriented languages, encourage code reuse to promote faster

---

304 See Hollaar, supra note 259, at 286-88 (arguing that, because of the lack of active patent protection of software, software developers must practice their trade in secrecy and “spend much of their efforts reinventing the wheel because there is not an adequate collection of how problems have been solved in the past”).

305 See id. at 286 (“An active patent system for software-related technology would make [existing software-development] techniques known. Then others could improve upon a patent or ‘invent around’ a patent by using new solutions.”).

306 See Lemley & O'Brien, supra note 24, at 256-59. Mark Lemley and David O'Brien argue that the “public goods problem” has stifled the creation of reusable, patentable software components. Id. at 268-69. Under the public goods problem, “the costs of exclusion are high,” so that inventors cannot easily exclude free riders from benefiting from the good; “consumption of the good is ‘nonrivalrous,’” so that the good is nondepletable and thus never gains value through scarcity. Id. at 268. The reason for this problem, they contend, is that “[u]ntil now, legal doctrine has not optimally encouraged software reuse.” Id. at 304. They conclude that the shift toward patent protection for software is encouraging “because it promises to couple strong protection for novel [software] components with a free market in interoperability.” Id.

307 Hollaar, supra note 259, at 287. Professor Hollaar speculates that, if developers feared liability for patent infringement in actions filed by other developers, they might be more willing to negotiate for a license on existing software inventions, rather than building them from scratch. See id. at 286-87. However, the danger of becoming the subject of an infringement action has been historically low, due to both the relative paucity of software-related patents and the uncertainty of their validity. Cf. McDonald, supra note 3, ¶¶ 6-9 (describing software developers’ traditional aversion to relying on the patent system for software protection and analyzing the detrimental results of the patent system’s historical failure to protect software). State Street Bank has changed this calculus, and the fear of infringement will become a strong motive once software developers begin to utilize the newfound security of patent protection.

308 C++ is the object-oriented extension to the C programming language. See Hollaar, supra note 259, at 287 n.12. The majority of modern consumer software applications are either in C or C++. Java and Visual Basic, two other popular software development languages, both use the object-oriented paradigm. See Gary Cornell, Visual Basic 5 from
and more efficient programming practices and development cycles. In turn, more efficient development enables more rapid innovation and an overall improvement in the rate of growth in the computer software sector. Moreover, the new object-oriented programming paradigm of software development "seems to be the dominant programming paradigm these days, having replaced the structured programming techniques that were developed in the early 1970s," and it is better suited to patent protection than earlier, structured programming techniques. Because a software module accomplishes a discrete task, a novel, useful, and nonobvious module is more readily identifiable and patentable than a new innovation encompassed within a larger, older program.

Thus, while the Manifesto argues that the incremental progress of the state of the art in software precludes effective patent protection, a better view might be that affording patent protection to software would allow the software industry to break away from the incremental

the Ground Up 374-75 (1997); 1 Cay S. Horstmann & Gary Cornell, Core Java: Fundamentals 104-05 (1997).

Object-oriented programming has largely replaced the old, structured programming paradigm. See Cornell, supra, at 374; 1 Horstmann & Cornell, supra, at 104. Structured programming develops specialized, highly interrelated data structures and manipulates the application-specific data with algorithms. See 1 Horstmann & Cornell, supra, at 105. This structured approach allows for a high degree of specialization, but requires developers to write substantially from scratch the entire code for a particular application. In contrast, under the object-oriented programming paradigm, software developers design programs as collections of independent modules, each of which performs a discrete task. These independent modules communicate with each other through standard interfaces, so that they can easily be reused without modification. See id. Thus, if a developer creates a novel and useful module, the developer can obtain patent protection for the module and subsequently license its use to other developers.

See Hollaar, supra note 259, at 287 ("The use of purchased modules or libraries not only minimizes concerns about patent infringement, but can result in faster software development. Further, because the module developer can spend more effort producing a comprehensive program, greater capabilities or more efficient operations can result.").

Object-oriented programming has largely replaced the old, structured programming paradigm. See Cornell, supra, at 374; 1 Horstmann & Cornell, supra, at 104. Structured programming develops specialized, highly interrelated data structures and manipulates the application-specific data with algorithms. See 1 Horstmann & Cornell, supra, at 105. This structured approach allows for a high degree of specialization, but requires developers to write substantially from scratch the entire code for a particular application. In contrast, under the object-oriented programming paradigm, software developers design programs as collections of independent modules, each of which performs a discrete task. These independent modules communicate with each other through standard interfaces, so that they can easily be reused without modification. See id. Thus, if a developer creates a novel and useful module, the developer can obtain patent protection for the module and subsequently license its use to other developers.

See Hollaar, supra note 259, at 287 ("The use of purchased modules or libraries not only minimizes concerns about patent infringement, but can result in faster software development. Further, because the module developer can spend more effort producing a comprehensive program, greater capabilities or more efficient operations can result.").

Object-oriented programming has largely replaced the old, structured programming paradigm. See Cornell, supra, at 374; 1 Horstmann & Cornell, supra, at 104. Structured programming develops specialized, highly interrelated data structures and manipulates the application-specific data with algorithms. See 1 Horstmann & Cornell, supra, at 105. This structured approach allows for a high degree of specialization, but requires developers to write substantially from scratch the entire code for a particular application. In contrast, under the object-oriented programming paradigm, software developers design programs as collections of independent modules, each of which performs a discrete task. These independent modules communicate with each other through standard interfaces, so that they can easily be reused without modification. See id. Thus, if a developer creates a novel and useful module, the developer can obtain patent protection for the module and subsequently license its use to other developers.

See Hollaar, supra note 259, at 287 ("The use of purchased modules or libraries not only minimizes concerns about patent infringement, but can result in faster software development. Further, because the module developer can spend more effort producing a comprehensive program, greater capabilities or more efficient operations can result.").

See Hollaar, supra note 259, at 287 ("The use of purchased modules or libraries not only minimizes concerns about patent infringement, but can result in faster software development. Further, because the module developer can spend more effort producing a comprehensive program, greater capabilities or more efficient operations can result.").
improvement model and experience truly inventive progress. Additionally, modern, object-oriented programming techniques more easily lend themselves to inventive advances and therefore to patent protection.\textsuperscript{313} Granting patent protection to software would thus encourage this type of development and would introduce a self-reinforcing cycle of enhanced inventiveness in the software industry. Furthermore, "dividing property entitlements by patenting software components may leave the developers of integrated software programs no choice but to negotiate licensing agreements with others in the industry,"\textsuperscript{314} stimulating an increase in software technology sharing and resulting in further progress in the software industry.\textsuperscript{315}

Finally, the \textit{Manifesto} argues that unique characteristics of software render the patent system administratively unmanageable as a vehicle for its protection.\textsuperscript{316} However, these arguments are true for patents in general, not just for software inventions.\textsuperscript{317} Admittedly, the patent system is not perfect.\textsuperscript{318} However, this Note argues that the present patent system even with its imperfections is capable of effectively dealing with software-related inventions and is therefore preferable to a sui generis protection scheme, which is untested and for which there is no persuasive justification. Moreover, if the present system proves administratively unworkable for software, it will surely be an easier task for Congress to simply update the Patent Act than to promulgate a wholly new sui generis protection scheme for software.\textsuperscript{319} Instead of underscoring the need to create a new system

\begin{footnotes}
\item[313] See supra notes 311-12 and accompanying text.
\item[314] Lemley & O’Brien, supra note 24, at 295.
\item[315] See id. at 295-96.
\item[316] See Samuelson et al., supra note 20, at 2345-46 n.134 (claiming that “the high costs and long delays required to obtain [patent] protection for program innovations” are often prohibitive, especially for small developers with limited resources). In addition, the \textit{Manifesto} argues, “If and when a patent finally issues, it may also come after the useful commercial life of the product embodying the innovation, because of the fast pace of innovation in the industry.” Id. at 2346 n.134. Consequently, the \textit{Manifesto} proposes “[a] form of legal protection for the first years of a technical innovation in software [that] would, accordingly, be better tailored to the needs of software entrepreneurs.” Id.
\item[317] See Hollaar, supra note 259, at 285 (“If those complaints have merit, Congress should address them by reforming the patent system as a whole, not by excluding one area of technology.”). On the other hand, the Federal Circuit has a history of vacillating on the issue of software patents. See supra Part I.B-C.
\item[318] For suggestions on possible improvements to the Patent Act, see infra Part III.B.
\item[319] See Goldstein, supra note 259, at 2575 (suggesting that the sui generis scheme of Samuelson and her colleagues will have to overcome substantial congressional inertia against change). Professor Gorman’s doubts about the efficacy of the \textit{Manifesto}’s proposal are more fundamental:

I am doubtful, however, that the sui generis legal regime the [\textit{Manifesto}’s] authors propose is warranted or practicable. If the \textit{Manifesto}’s basic premise—that copyright does not provide meaningful protection, and thus economic incentives, for the computer program designer—is unsound, then the argument for supplemental \textit{sui generis} protection is fundamentally
\end{footnotes}
of protection, administrative concerns thus justify maintaining and updating the current patent system.

B. Proposed Amendments to the Patent Act

This Note argues that, after State Street Bank and subsequently AT&T, the patent system represents an adequate protection scheme for software-related inventions. However, it is by no means perfect. As Justice Douglas noted in Benson, software patentability raises issues that can best be resolved by congressional action. While recent Federal Circuit decisions, most importantly State Street Bank, have clarified many of the conundrums in past software patentability doctrine, judicial action cannot so easily solve other problems endemic to the patent system. Amending the Patent Act might be a better solution to these deficiencies; the State Street Bank decision provides valuable guideposts for Congress to follow.

This Note suggests two possible amendments to the Patent Act. The first proposal would simply codify the holding of State Street Bank. The second, more controversial proposal would remedy problems that plague patent applicants generally. Several commentators have proposed changes to the Patent Act, and this Note adopts elements of their suggestions with some modifications.

weakened. I believe that to be the case. I also believe that, if such a new legal regime were to be created, it would add little to—and would in important respects seriously conflict with—the intellectual property regimes already long in place. Finally, I believe that the Manifesto proposal presents serious practical difficulties of implementation.

Gorman, supra note 288, at 278-79.

320 This Section argues that since these decisions, the patent system has emerged as the best available protection for software-related inventions. For other options that have been suggested, see supra Part III.A.

321 See Gottschalk v. Benson, 409 U.S. 63, 73 (1972) (“If [computer] programs are to be patentable, considerable problems are raised which only committees of Congress can manage .... The technological problems [inherent to software] indicate to us that considered action by the Congress is needed.” (footnotes omitted)).

322 For instance, many scholars had viewed both the business method and the mathematical algorithm exceptions as per se bars to software patentability. See supra Part I.B-C. However, the Federal Circuit in State Street Bank conclusively held that these doctrines should not prevent the patenting of otherwise patentable software inventions. See supra Part II. On the other hand, the Federal Circuit has a history of vacillating on the issue of software patents. See discussion supra Part I.B-C.

323 For instance, judicial action cannot remedy the Manifesto’s contention that the time delay between patent application and patent issuance acts as a de facto bar to software patents for many small developers. See supra note 316. This problem stems from PTO procedures and is a deficiency that redress by litigation will never be able to address. However, congressional restructuring of the patent examination process could alleviate the problem. See infra Part III.B.2 (proposing procedures that would streamline the statutory scheme for protection of software patents).

324 See, e.g., Hollaar, supra note 259, at 297 (“By making three simple changes to the patent statute, Congress can clarify the patentability of software-based inventions and enhance their protection.”); Stern, supra note 20, at 213 (proposing a “petty-patent system
1. Explicitly Define Software as Patentable Subject Matter

The historical animosity and subsequent uncertainty exhibited by both the PTO and courts toward software-related inventions have been problematic to the patent system. The State Street Bank decision settled the confusion by returning to the traditional patentable subject matter analysis for software-related inventions and dispensing with both the mathematical algorithm and business method exceptions. While this Note applauds the Federal Circuit's decisive action in State Street Bank, its decision, while persuasive, does not carry the same weight as an affirmative statement by Congress.

To ensure the continued viability of the State Street Bank holding, Congress should amend the Patent Act to explicitly embrace software as patentable subject matter. Congress has at least three options for promoting software patentability. First, it could define mathematical algorithms and business methods as separate categories of patentable subject matter under §102. Second, it could clearly indicate in its legislative history that while software is not patentable per se, it is patentable when included in part of a machine or article-of-manufacture [that] would effectively supersede patent protection for all computer-related questionable statutory subject matter, whether called an algorithm, method of doing business, printed matter, or an abstract idea); McDonald, supra note 3, ¶ 123 (calling for "guidance from the legislature" and arguing that "policy decisions inherent in protecting software are simply too complicated to allow adequate solution through the [Federal Circuit's] legislating from the bench"). Although Professor Stern's proposal is actually more of a usurpation of the Patent Act than an amendment to the Act itself, it contains suggestions that are also helpful in amending the Act itself.

For example, the business method and mathematical algorithm exceptions obfuscated the field for decades. See supra Part I.B-C. See supra Part II. On the other hand, the Federal Circuit's exclusive appellate jurisdiction over patent matters gives every Federal Circuit patent decision considerable authority. See supra notes 32-39 and accompanying text. Moreover, the Federal Circuit's recent decision in AT&T Corp. v. Excel Communications, Inc., 172 F.3d 1352, 1357-58, 1360 (Fed. Cir.), cert. denied, 120 S. Ct. 368 (1999), reaffirms its commitment to uphold software patents that meet the statutory requirements for patentability. However, the fundamental policy questions surrounding software patents demand a legislative solution, rather than reactive judicial lawmaking.

Cf. Stern, supra note 20, at 208 ("[State Street Bank's predecessors] in the Federal Circuit have left algorithm-related and computer-related patent law in such disarray that, without legislative intervention, it may be years before any equilibrium is reached. The outcome of cases will now significantly depend on the happenstance of panel composition."
(emphasis added)). While State Street Bank is certainly a strong move toward certainty in software patents, another Federal Circuit panel could just as easily reinstate the prior confusion with a decision based on the now-defunct exceptions. However, in light of AT&T's extention of the State Street Bank rationale in upholding a process claim that implemented a mathematical algorithm, see AT&T, 172 F.3d at 1357-58, 1360, Stern's concerns may no longer hold true. Arguably, AT&T indicates a shift in the Federal Circuit toward consistent, favorable treatment of software patents. In any event, congressional ratification of the State Street Bank and AT&T holdings would provide more reliable protection for software.
claim. Finally, and most desirably, Congress could indicate that software itself constitutes patentable subject matter within the existing framework of § 101.

Congress should not explicitly define algorithms and business methods as statutory subject matter, because such categorization would fail to acknowledge their nature as abstract ideas. As State Street Bank notes, mathematical algorithms and business methods are not patentable per se, but their useful applications are patentable. Similarly, Congress should not legitimize software merely as an attachment to machines or articles of manufacture. In addition to needlessly limiting the scope of protection for software, such a formulation would perpetuate the legal fictions and contortionist claim drafting practices that have permeated mathematical algorithm jurisprudence for the last three decades.

Congress should instead revise the existing statutory framework to ensure that otherwise patentable subject matter is not rendered unpatentable solely because it contains a business method or mathematical algorithm. By doing so, Congress would not only establish a consistent mode of subject matter analysis, it would also provide the proper scope of protection to software-related inventions. These inventions would be patentable, but only if they satisfied the requirements of novelty, utility, and nonobviousness. Therefore, this Note proposes the following amendment to § 101 of the Patent Act:

Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title. So long as such an invention or discovery conforms to the conditions and requirements of this title, the fact that the invention or discovery contains or implements a business method or mathematical algorithm shall not affect the eligibility of such invention or discovery for patent protection under this title.

Following the Federal Circuit's rationale, this amendment does not distinguish between processes, machines, and articles of manufacture. Moreover, the proposed amendment comports with the constitutional requirement that all patented inventions be new and useful by explicitly mandating compliance with other patentability requirements of the Patent Act, such as § 102 (utility) and § 103 (nonobviousness).

329 See State St. Bank & Trust Co. v. Signature Fin. Group, Inc., 149 F.3d 1368, 1375, 1377 (Fed. Cir. 1998) (requiring a claim embodying a mathematical algorithm or business method to produce a "useful, concrete, and tangible result" in order to constitute statutory patentable subject matter), cert. denied, 525 U.S. 1093 (1999).
330 See supra notes 72-79 and accompanying text.
332 The emphasized portion is the author's proposed amendment to § 101.
333 See AT&T, 172 F.3d at 1357-58; State St. Bank, 149 F.3d at 1375.
ness). By codifying the results of State Street Bank and AT&T, the proposed amendment provides a dependable standard that courts, the PTO, and inventors can rely on.

2. Streamline the Patent Application Process

Arguably, by simply codifying the results of State Street Bank and AT&T, Congress will fully enable the patent system to effectively protect software. However, if it decides to amend the Patent Act, Congress may also wish to take more substantial—and admittedly controversial—action to fix the patent system's much criticized application process, especially for software patent applications. As the Manifesto points out, the protracted length of time between patent application and issuance adversely affects the desirability of patent as the primary mode of protection for software. In many cases, the extended examination period renders a software patent virtually useless because the patented invention has become obsolete by the time the patent issues. Typically, patent prosecution takes a long time because the PTO has to search for the relevant prior art to make novelty and obviousness determinations. Ironically, however, prior art in the software field is largely undocumented and highly disorganized.

---

334 See Samuelson et al., supra note 20, at 2345 n.134. The patent application and examination process works as follows: First, the applicant submits a patent application to the PTO. See Patent and Trademark Office, U.S. Dep't of Commerce, Manual of Patent Examining Procedure § 601 (7th ed. 1998) [hereinafter MPEP]. The application contains both a specification (describing the invention and any prior art), see id. § 608.01, and claims (describing, in explicit detail, the invention), see id. § 608.01(i). Next, the PTO searches its databases for prior art that would render the claimed invention in the application unpatentable. See id. § 704. If the PTO finds such prior art, it rejects some or all of the application's claims. See id. § 706.02. The applicant may then amend the claims in response to the examiner's rejections. See id. This process continues until the patent conforms to the examiner's requirements or the examiner issues a Final Rejection. See id. § 706.07. Following the Final Rejection, the applicant may either appeal to the PTO Board of Appeals or abandon the application. See id. This iterative process is referred to as examination and typically lasts from several months to several years. Only after the completion of the examination process will the patent issue, thereby conferring protection on the invention. See id. § 701. For more details on the patent examination process, see id. §§ 701-24.

335 See Samuelson et al., supra note 20, at 2346 n.134. Professor Samuelson and her coauthors note:

If and when a patent finally issues, it may... come after the useful commercial life of the product embodying the innovation, because of the fast pace of innovation in the industry. A form of legal protection for the first years of a technical innovation in software would, accordingly, be better tailored to the needs of software entrepreneurs.

Id.

336 Quoting from a 1966 report of the President's Commission on the Patent System, Justice Douglas lamented on the inadequacy of the patent system in Benson: "The Patent Office now [in 1966] cannot examine applications for programs because of a lack of a classification technique and the requisite search files. Even if these were available, reliable searches would not be feasible or economic because of the tremendous volume of prior art
Thus, the present examination procedure for software patent applications is both overly protracted and largely ineffective.

As a remedy to this gaping hole in the patent system, Professor Richard Stern proposes a "petty-patent system."\(^3\)\(^3\)\(^7\) Stern envisions the petty patent as a system that would "direct the Commissioner [of the PTO] to issue petty patents with only negligible prior examination, which would be limited to facial compliance with statutory requirements and implementing regulations.\(^3\)\(^3\)\(^8\) Professor Stern patterned his proposal on the "utility-model" system recently implemented by several countries—most notably Japan and Germany.\(^3\)\(^4\)\(^0\) Although it would not be part of the existing patent system,\(^3\)\(^4\)\(^1\) Professor Stern's system would still reside with the Patent Act in Title 35 of the United States Code.\(^3\)\(^4\)\(^2\) The proposed system is notable for the lack of an involved pre-issuance examination process and the consequent negation of the presumption of validity normally accorded to patents.\(^3\)\(^4\)\(^2\) To

---

\(^{3\)\(^7\)}\) Gottschalk v. Benson, 409 U.S. 63, 72 (1972) (quoting President's Comm'n on the Patent Sys., "To Promote the Progress of . . . Useful Arts": In An Age of Exploding Technology 19 (1966). The situation has not improved since the 1960s. With the explosive growth of the software industry and its use of trade secrecy to protect much of its intellectual property, see supra Part III.A.1, much of the prior art in software remains shrouded in obscurity, see Stern, supra note 20, at 216 n.144 (noting "the past difficulty experienced in the [Patent and Trademark] Office in finding relevant prior art in software cases"). For a discussion of more recent critiques of the PTO's ability to deal with prior art in the software field, see supra note 260.

\(^{3\)\(^8\)}\) Id. at 213; see also Richard H. Stern, A Sui Generis Utility Model Law as an Alternative Legal Model for Protecting Software, 1 U. Balt. Intell. Prop. L.J. 108, 112-13 (1993) (postulating that a petty-patent examination would not include an exhaustive iterative prosecution process and thus the petty patent would issue more quickly and without "the high front-end costs of a patent-type examination procedure").

\(^{3\)\(^9\)}\) See Stern, supra note 338, at 112 & n.6; see also Mark D. Janis, Second Tier Patent Protection, 40 Harv. Int'l L.J. 151, 159 (1999) ("More than sixty countries currently offer second tier [utility model] patent protection, including key patenting jurisdictions such as Germany and Japan. . . Most importantly, the European Commission is moving to expand the role of second tier regimes at the pan-European level.").

\(^{3\)\(^4\)\(^0\)}\) See Stern, supra note 20, at 221 (calling for "a sharp, bright-line divide between the respective domains of petty and regular patents").


\(^{3\)\(^4\)\(^2\)}\) See Stern, supra note 20, at 213-14 ("Petty patents [would issue] with only negligible prior examination, which would be limited to facial compliance with statutory requirements and implementing regulations. . . . That, in turn, implies a very slight presumption of validity."). Section 282 of the Patent Act establishes a presumption of validity for issued patents in infringement actions. See 35 U.S.C. § 282 (1994) ("A patent shall be presumed valid."). Only clear and convincing evidence of invalidity will rebut this presumption. See Texas Instruments Inc. v. United States Int'l Trade Comm'n, 988 F.2d 1165, 1177 (Fed. Cir. 1993) ("A patent is presumed valid and the party asserting invalidity must overcome this presumption by clear and convincing evidence establishing facts which support the conclusion of invalidity."); Chrysler Motors Corp. v. Auto Body Panels, Inc., 908 F.2d 951, 953 (Fed. Cir. 1990) ("Under § 282, a patent is presumed valid. This presumption of validity places the burden of persuasion as well as the burden of going forward on the party asserting invalidity."). The presumption of validity is one of patent's primary sources of
counteract this deficiency in the examination process, Professor Stern proposes a more robust reexamination proceeding, in which interested parties can challenge the validity of the issued petty patent without resort to expensive trials. Stern reasons:

The [exhaustive examination] procedure should be post-issuance, rather than pre-issuance, in order not to delay prompt registration of the subject matter and attachment of rights to the owner, and opposers should be allowed to submit art to the Office, explain its relevance, and controvert the owner’s arguments as to the significance of the art.

Successful navigation of the post-issuance examination process would presumably confer the traditional presumption of validity to petty patents.

Multiple benefits would arise from conducting the post-issuance examination proceeding. Examiners currently conduct the ex parte pre-issuance examination proceedings in order to ensure the confidentiality of the invention until the inventor secures patent protection. However, if the patent has already issued at the time of examination, the PTO can utilize an inter partes proceeding, which mirrors the current reexamination proceeding. This inter partes proceeding would allow interested parties to introduce relevant prior

value, because it shifts the burden in an infringement action onto the alleged infringer to prove invalidity as a defense.

Stern, supra note 20, at 214-16 ("Post-registration administrative opposition (revalidation) should also be available to permit those in the software industry to bring to the Patent and Trademark Office's attention prior commercial software products that anticipate [or render invalid] a registered algorithm, rather than engage in infringement litigation." (footnotes omitted)). Under the current patent system, an interested party, including the patentee, may file for a reexamination proceeding in the PTO in order to introduce previously undisclosed relevant prior art and reexamine a patent in light thereof. See MPEP, supra note 354, § 2212. During the reexamination proceeding, a patent examiner will issue a new determination of patent validity. See id. § 2258.

Stern, supra note 20, at 215 n.143.

Cf. Patent Applications Preserved in Confidence, 37 C.F.R. § 1.14(a) (1999) ("No information will be given concerning the filing, pendency, or subject matter of any application for patent, and no access will be given to, or copies furnished of, any application or papers relating thereto [with limited exceptions, such as if the application has been abandoned].").

Congress recently enacted an optional inter partes reexamination procedure, effective November 29, 1999. See Optional Inter Partes Reexamination Procedure Act of 1999, Pub. L. No. 106-113, §§ 4601-08, 113 Stat. 1501A-567, 1501A-567 to -572 (codified as amended in scattered sections of 35 U.S.C.). The inter partes reexamination will likely allow the PTO to more easily find and evaluate the impact of relevant prior art on the patent at issue, ensuring that fewer nonmeritorious patents are upheld. However, because the patent system continues to implement the involved preissuance examination procedure, the new reexamination does nothing to alleviate the major problem this Part seeks to address: the unduly burdensome time and expense involved in obtaining a patent. In contrast, Stern’s proposal would vastly reduce the scope of the preissuance examination, allowing patents to issue more quickly and with less expense. See supra notes 342-44 and accompanying text.
art to challenge the novelty, utility, or nonobviousness of the patented
invention. The inter partes nature of the proceeding would promote
more comprehensive searches of the prior art, because prospective
searchers would have a vested interest in finding prior art that
would invalidate the patent at issue. The inter partes proceeding
would also remove from the PTO the burden of searching the prior
art.

The reduced time and expense of the pre-issuance prosecution
makes Professor Stern's petty-patent a facially appealing form of
software protection. The petty-patent proposal also addresses con-
cerns about the issuance of patents for meritless inventions. Unfor-
nately, however, theoretical and administrative deficiencies could
render Professor Stern's proposal as impractical as any other sui
generis protection scheme. Further, the most desirable feature of
Professor Stern's proposal, the post-issuance examination, could be
incorporated into the existing patent system. Therefore, if Con-
gress were to adopt such a procedure, it should do so within the cur-
rent patent system for all types of inventions.

Two primary premises motivate Professor Stern's advocacy
of a petty-patent system outside of the existing patent system.
First, he contends that mathematical algorithms do not constitute
patentable subject matter, and thus the current patent system
should not accommodate them. In light of State Street

---

347 Most European countries provide for this type of inter partes, post-issuance ex-
amination. See Stern, supra note 20, at 215 n.143.

348 The parties searching for prior art are usually competitors of the patentee, seeking
to have the patent invalidated. Not only will they have a strong motivation to find relevant
prior art, they will also have a better access to that prior art (because they likely have
produced some of it) and be much more well versed in the relevant art than the patent
examiner.

349 The PTO has traditionally had a difficult time searching prior art in the software
field. See supra note 336 and accompanying text.

350 See Stern, supra note 338, at 113 (describing "an intermediate level of required
technical merit" to which inventions must rise in order to qualify for petty-patent
protection).

351 See supra Part IIIA.4 for a discussion of the practical impediments to sui generis
protection.

352 See supra notes 337-44 and accompanying text.

353 Professor Stern's proposal shares a similar rationale with Professor Samuelson's
Manifesto. See supra Part IIIA.4.

354 See Stern, supra note 20, at 224-25. Professor Stern advocates Flook's "point-of-nov-
elty" approach, which Freeman, Walter, and ultimately Diehr rejected:

[My proposal] adopts the so-called point-of-novelt y . . . approach in distin-
guishing petty patent subject matter from regular patent subject matter. Under this approach, analysis focuses on how the claimed innovation is dif-
ferent from the prior art. If everything described in the claim is old and
conventional, except for a new algorithm (or other nonstatutory subject
matter), the claimed innovation is really a new algorithm.

Id. at 224, cf. supra notes 127-28 and accompanying text (describing the Supreme Court's
abandonment of the point of novelty approach in Diehr).
Bank, however, Professor Stern’s dogmatic reliance on the mathematical algorithm exception to distinguish between software and other inventions seems outmoded. Thus, the fact that mathematical algorithms comprise an integral part of software inventions does not justify a distinction between software patents and other types of patents.

Second, Professor Stern proposes lower standards of utility, novelty, and nonobviousness for petty patents than those required by the current patent system. He presumably justifies the lower standards by reasoning that most software advances, while meriting protection, are not as inventive as other types of inventions. However, both legal and economic arguments counsel against lowering the novelty, utility, and nonobviousness requirements of the current system. Legally, there is no distinguishing feature in software-related inventions that merits departure from the traditional standards of inventiveness required by the Constitution. As previously discussed, the innovation-invention dichotomy suffers from analytical deficiencies.

---


356 See Stern, supra note 20, at 218-21. Professor Stern replaces the traditional trio of patentability requirements with “originality, novelty, and technical merit advance.” Id. at 218. His originality requirement, which replaces the nonobviousness requirement, “is essentially the same as that for copyright law,” id. at 220, and is thus de minimis, see CRAIG JOYCE ET AL., COPYRIGHT LAW 81 (4th ed. 1998) (noting that copyright’s originality standard requires only “independent creation by the author, and a modest quantum of creativity”). Professor Stern’s “novelty requirement . . . is generally similar to that of patent law, adapted to a system in which rights depend on filing” as opposed to the current patent system’s first-to-conceive standard. Stem, supra note 20, at 220 (footnote omitted). His technical advance requirement “is not as high as that of patent law, and simply filters out routine or commonplace contributions.” Id. at 220. He further notes that “the term ‘innovation’ used throughout [his discussion of technical advance requirement] simply means the subject matter on which legal protection is sought and does not imply actual novelty and technical merit; the term is analogous to ‘alleged invention.’” Id. at 220-21.


358 See supra notes 301-15 and accompanying text.

359 Like Professor Stern, the Manifesto argues that software advances are more often innovative than inventive and asserts that the advances are so minor as not to qualify for patent protection under the Patent Act’s novelty, utility, and nonobviousness requirements. See Samuelson et al., supra note 20, at 2331 & n.69 (“While innovation in program-design occasionally rises to the level of invention, most often it does not.”).
One can also question the economic underpinnings of the utility model of software protection. Initially, a patent system that provides certainty and security to software inventors will stimulate inventive, and not just innovative, advances. Additionally, modern programming practices not only encourage inventive advances, but also promote easier identification and delimitation of patentable advances encompassed within larger, older programs by enabling module segregation. Finally, software advances that do not meet the patent system’s requirements do not deserve monopoly protection. Such advances already receive adequate market protection via lead time advantages. However, truly inventive software advances should receive the more substantial protection of a government-sanctioned monopoly. In sum, as Professor Janis notes, “second tier patent regimes ‘are hard to justify in terms of classical intellectual property theory . . . ,’ and . . . the economic justification for awarding protection to subpatentable innovation ha[s] never been articulated satisfactorily.

Thus, there is no reason to segregate Professor Stern’s post-issuance examination proposal from the present patent system. After all, even if the rapid pace of the software industry exacerbates the arduous examination process, inventors of nonsoftware inventions

---

360 See Janis, supra note 339, at 209 (arguing that the issue of whether lowering the inventiveness threshold “makes sense economically is . . . a highly contentious matter”). According to Professor Janis, the utility model depends on “the so-called ‘prospect’ theory of the patent system,” which argues, contrary to classical economic models of the patent system, that granting patent protection at the early stage of invention will encourage innovation by allowing the patentee to more readily finance the further development of the invention. Id. at 209-10. However, the utility model “seem[s] to assume that the subject matter of second tier applications will inevitably constitute modest variations on existing technologies . . . [and] is fundamentally not a technological prospect arising early in the course of development efforts.” Id. at 211-12. Thus, the economic justification for the petty-patent system is inconsistent with its goal of encouraging and rewarding merely innovative, as opposed to inventive, developments.

361 See supra notes 304-06 and accompanying text.

362 See supra notes 307-12 and accompanying text.

363 A software developer will receive some market exclusivity for its innovation simply as a result of the amount of time required for other developers to imitate the innovation. See Goldstein, supra note 259, at 2575 (speculating that one possible reason for the past growth in the software industry despite the lack of comprehensive protection for software-related advances might be the existence of “market niches [that] provided sufficient lead time to support software innovation”). This Note argues that only advances rising to the level of invention, as required by the Patent Act, merit the legal protection of monopoly status. For mere innovations, lead-time advantages provide sufficient reward, and corresponding incentive, to developers.


365 As Professor Hollaar notes, if software-specific complaints about inefficiencies of the PTO “have merit, Congress should address them by reforming the patent system as a whole, not by excluding one area of technology [from the patent system].” Hollaar, supra note 259, at 285.
likewise have to deal with the same high cost and delay of the current system.

If Congress ultimately decides that the protracted length of time between patent application and issuance sufficiently hinders the effectiveness of the patent system's protection of software to justify legislative action, this Note recommends modification of the existing patent system, rather than implementation of an entirely new, sui generis protection scheme for software. One way to limit the duration of patent application pendency would be to replace the current examination system with a summary application procedure. Such a procedure would diminish the time and expense currently required to obtain a patent. To prevent frivolous patents, the amendments should require comprehensive, inter partes, post-issuance examination proceedings, notice of which would be published to the public. Because these proceedings would take place after patent issuance, they would provide two key benefits: First, they would preclude delays associated with the present patent application procedure. Second, they could proceed inter partes, allowing interested parties to direct the Examiner to relevant prior art and relieving the PTO of the difficult burden of performing software prior art searches. The traditional presumption of validity accorded to an issued patent would not inhere to a software patent until it had undergone the post-issuance examination process.

Moving to a new system of patent examination will necessarily create substantial administrative costs. Additionally, the inter partes examination would probably be more expensive for the inventor than the ex parte examination under the current system because of the more adversarial nature of the proceeding. However, this increase in expense would be an investment for the patent holder, as it would presumably result in increased patent security; any opposing party would have the opportunity to challenge the patent during the examination proceeding. In addition, deferring the examination costs until after the patent issuance affords the inventor additional flexibility. The inventor can gauge the value of the invention before deciding whether to proceed with the examination; the inventor can also possibly finance the examination with proceeds from marketing the patented article or negotiating technology licenses.

Whether the benefits of the post-issuance examination outweigh the additional costs is an issue beyond the scope of this Note. However, in the event that Congress does decide to adopt a post-issuance, inter partes examination system for software, it would be both feasible and desirable for Congress to do so within the current patent system, rather than as a sui generis scheme. In addition, constitutional considerations militate against the lower inventiveness threshold common to sui generis protection schemes.
Conclusion

For purposes of intellectual property protection, software-related inventions do not possess any special characteristics that distinguish them from other forms of inventions. Thus, to qualify for protection they should meet the constitutional threshold of inventiveness. The patent system therefore represents the best mode of protection for software-related inventions. Since the early 1970s, however, the tension between the patent system and software-related inventions, and the consequent uncertainty regarding the validity of software patents, have prompted inventors and scholars to seek an alternative form of protection for software.

Commentators have advocated several alternative protection schemes for software-related inventions, including copyright, trade secrecy, trademark, and even sui generis protection schemes. Witnessing the explosive growth in the software industry over the last two decades, some critics have even disavowed any form of protection for software-related inventions. Like any other invention, however, software merits protection, as long as it meets the requisite standards of novelty, utility, and nonobviousness.

The federal government need not look any further than the current patent system to provide the proper level of protection for software-related inventions. In fact, the Federal Circuit’s recent State Street Bank decision represents a signpost on the path towards appropriate protection of software-related inventions: The presence of amorphous concepts such as mathematical algorithms and business methods in the patent’s claims should not be determinative of patentability. Instead, the patentability inquiry should focus on whether the claimed invention produces a tangible result and meets the other requirements of the Patent Act.

While State Street Bank provides insight, congressional amendment of the Patent Act will make the patent system even more amenable to software and provide the necessary guidance for the courts and certainty for developers.

This Note proposes two independent amendments to the Patent Act that will allow the patent system to better accommodate software-related inventions. First, the Patent Act should allow for the patentability of software qua software, and not merely as an attachment to a machine or an article of manufacture. Second, in response to concerns about the length of time the PTO takes to issue patents, Congress should consider restructuring the application procedure to defer the protracted examination period until after patent issuance. The benefits of this change would be two-fold: It would promote more expedient patent issuance and also allow inter partes examination, letting interested parties introduce evidence relevant to the patentability
determination. If Congress decides to implement such a procedure, it should do so within the current patent system. There is simply no sufficient justification for creating a special sui generis protection scheme for software. *State Street Bank* has reemphasized the patent system’s ample ability to provide protection to software developers and to enable them to share information and innovation in unprecedented fashion.