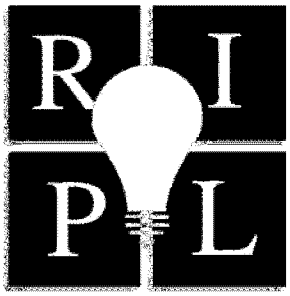
THE JOHN MARSHALL
REVIEW OF INTELLECTUAL PROPERTY LAW



TAMING THE CODE: EFFECTIVELY IMPLEMENTING SOFTWARE PATENTS

ANTHONY E. ANDERSON

ABSTRACT

Software patents are a sore subject for many programmers. Although still in their infancy, they have managed to anger many of those in the programming community. Software patents started to evolve in the early 80's through multiple court decisions that eventually defined software as statutory patentable material. Although patentable, software has proven to be a formidable match for the examination process. The examination process has proven ineffective in properly examining software patent applications and as result multiple lawsuits based on frivolous patents have emerged. Potential battles such as the one between Creative and Apple over Creative's patent for a hierarchal file system have become examples for which opponents of software patents can rely. This comment proposes the creation of a third party entity that would be made up of the programming community that would police software patent applications prior to issuance. This entity would alleviate the strain on the USPTO as well as examiners while rebuilding the reputation of software patents and the USPTO.

Cite as Anthony E. Anderson, Comment, *Taming the Code: Effectively Implementing Software Patents,* 5 J. MARSHALL REV. INTELL. PROP. L. 382 (2006).

TAMING THE CODE: EFFECTIVELY IMPLEMENTING SOFTWARE PATENTS

ANTHONY E. ANDERSON*

INTRODUCTION

Within the programming community, the distaste for software patents is a phenomenon that cannot be ignored.[1] This resentment regarding software patents such as Amazon's "one click ordering"[2] has spurred debate as to the possibility of punishing companies[3] like Amazon in the open source community.[4] Many programmers are against software patents and have no problem voicing their opinions in open forums.[5] Such disputes give the impression that software patents are under the highest degree of scrutiny within the intellectual property world.

---

* J.D. Candidate I.P., May 2007, The John Marshall Law School. B.S. Computer Engineering Technology, DeVry University, Chicago, 2004. The author would like to thank The John Marshall Law School, my family, and the entire REVIEW OF INTELLECTUAL PROPERTY LAW. Specifically I would like to thank Kyle and Randy for taking the time and giving me the guidance to produce a publish-worthy comment. Most importantly, I would like to thank my wife, Kori, who dealt with the craziness of being married to a law student. Any and all errors are my own.

[1] *See* Lucas Van Grinsven, *New Free Software License Takes Aim at Patents*, MSNBC.com, Sept. 6, 2005, http://www.msnbc.msn.com/id/9225821/ (discussing with Georg Greve, the Free Software Foundation Europe's president, his attitude towards software patents and the open source community); *see also* Federico Biancuzzi, *RMS: The GNU GPL Is Here to Stay*, O'REILLY ONLAMP.COM, Sept. 22, 2005, http://www.onlamp.com/pub/a/onlamp/2005/09/22/gpl3.html?page=1 (quoting Federico Biancuzzi, founder of the Free Software Movement, "'Useless' is an understatement–in the software field, the patent system is harmful and unjust."); Electronic Frontier Foundation, Our Mission, http://www.eff.org/mission.php (last visited Sept. 25, 2005); Amy Kucharik, *Linuxworld: Lingering Patent Threats Worry Open Source Experts*, LINUXWORLD, Feb. 16, 2005, http://searchenterpriselinux.techtarget.com/originalContent/0,289142,sid39_gci1059267,00.html (quoting Bruce Perens, open source advocate, "The way the law is written in the United States, we could be shut down by a sufficient number of software patent lawsuits."); Stefan Krempl ET. AL., *Critics of Software Patents Nominated "European of the Year"*, HEISE ONLINE, Sept. 23, 2005, http://www.heise.de/english/.newsticker/news/64219 (announcing the nomination of two lobbyists against software patents).

[2] U.S. Patent No. 5,960,411 (filed Sept. 12, 1997) (describing "[a] method and system for placing an order to purchase an item via the Internet").

[3] *See* Lucas Van Grinsven, *New Free Software License Takes Aim at Patents*, MSNBC.com, Sept. 6, 2005, http://www.msnbc.msn.com/id/9225821/ (discussing how the inclusion of a clause in the new version of the General Public License ("GPL") might be added to punish those seeking to enforce software patents).

[4] "Open Source" is defined as an ideology behind the distribution and creation of software. Principles such as free distribution, viewable source code, and the allowance of derived works are firmly rooted in the definition of "open source.". *See* Open Source Initiative, The Open Source Definition, http://www.opensource.org/docs/definition.php (last visited 3/31/06).

[5] *See e.g.*, Slashdot.org. The popular technology website, Slashdot (http://slashdot.org) posts articles concerning many different technological issues and allows professionals and enthusiasts to comment on the articles often eliciting intellectual debate over a topic. In the "Patents Pending" section of Slashdot, a bulk of the comments written are often expressing extremely hostile feelings towards software patents and how the entire patent system should either be removed or overhauled. *Id.*

The problem plaguing software patents is ineffective use and regulation. In addition, procedural issues[6] concerning the prosecution of software patents in the United States Patent & Trademark Office ("USPTO")[7] as well as a common misunderstanding[8] as to what they represent severely jeopardizes the future of software patents. The resentment towards a specific patent or the patenting process in general however, tends to be rooted more in common misconceptions than in logic or facts.[9] Much of the uproar concerning software patents has risen to a frenzy, often characterizing software patents as "evil."[10]

Software patents and the issuing thereof are on the verge of becoming the proverbial snowball gaining size and momentum until it becomes a force out of control. Although there are numerous groups, programmers, and lobbyists[11] that disapprove of software patents in general, their disapproval is, to a large extent, based on the USPTO's inadequate handling of software patent applications during prosecution. The newness and the depth of software patents has created many issues. These issues become more pressing where the issuance of a patent grants a

---

[6] The process of the examiner finding prior art in regards to software patents is not an effective means by which to obtain information. Unfortunately, the scope of software is so large that to assume an examiner can effectively research all the needed information to properly issue a rejection is a serious problem. The scope of prior art can come from so many sources, the only true way to ensure the proper information is received by the examiner is to change the procedure by which the examiner obtains this information.

[7] U.S. PAT. & TRADEMARK OFFICE, U.S. DEP'T OF COMMERCE, MANUAL OF PATENT EXAMINING PROCEDURE § 2106 (8th ed. 2001, rev. 2005) [hereinafter MPEP 2005] (outlining patentable subject matter for computer-related inventions).

[8] *See supra* note 5 and accompanying text. The programming community contains a broad range of people ranging from the professional programmer to the teenager developing the new big idea. Many of these programmers don't understand patents and the purpose they serve. As a result common misunderstandings such as patents do not add to the general knowledge and that they serve no true purpose is only a testament to this ignorance.

[9] *See supra* note 5 and accompanying text. Additionally, many times the poster who is expressing anger over the USPTO's granting of a patent has neither read the patent nor understands the patent system. More often, the anger that is expressed by a poster is often based on comments made by other posters with each new poster expanding on the previous post. However, when the initial post is incorrect or makes an inaccurate assumption or assertion, the information is generally not tested as the statement is considered to be true and adds to the confusion.

[10] *See generally* NoSoftwarePatents.com, http://www.nosoftwarepatents.com/en/m/basics/webshop.html (last visited Sept. 25, 2005) (showing an extreme example of how a typical website could potentially infringe 20 issued patents).

[11] *Id. See generally* Electronic Frontier Foundation, The Patent Busting Project, http://www.eff.org/patent/ (last visited Sept. 25, 2005); Foundation for a Free Information Infrastructure Members, Index Page, http://www.ffii.org/members/index.en.html (last visited Sept. 25, 2005) (detailing the number of members which support the organization); *see also* Petition Against Software Patents, http://www.petitiononline.com/pasp01/petition.html (last visited Sept. 25, 2005).

> We have seen that many software patents covering well-known algorithms and techniques hinder the software industry in the United States of America and around the world. The Patent Office has shown that it does not understand software and cannot follow developments in the field, and frequently issues patents on well-known techniques and on simple ideas that programmers consider obvious. The causes of this are inherent in the nature of the software field and cannot be corrected.

Id.

monopoly to a single person or company.[12] Software is merely a new area of intellectual property and should be treated the same as any other patentable subject matter.[13] Though the potential for harm in this situation is great,[14] that does not mean those ideas contained in software patents should not be given protection.[15] Rather, the patenting process, as it relates to software, needs to be amended and molded around what has become an essential part of our everyday lives.[16]

This comment will focus on numerous aspects of software patents. The background discussion will include a review of how the USPTO currently handles and historically has handled software patents. The discussion will also analyze those software patents that have created a stir within the programming community. In light of these patents, the analysis will focus on the pros and cons of software patents and why the process by which software patents are examined must be adjusted to fit the specialized needs in examining software for patentability. The analysis will also discuss different positions of the software patent "war" from multiple perspectives.

Finally, the comment will propose that the prosecution of software patents needs to incorporate a higher level of scrutiny in order to ensure frivolous and obvious patents are not issued. Preventing these types of patents from issuing will enable the patent system to stay true to its initial intentions of promoting innovation without stifling the innovation it seeks to foster.

---

[12] 1-1 DONALD S. CHISUM, CHISUM ON PATENTS § 16.02(1)(c) (2004); *see also* Crown Die & Tool Co. v. Nye Tool & Mach. Works, 261 U.S. 24, 36–37 (1923). "The Government is not granting the common law right to make, use and vend, but it is granting the incident of exclusive ownership of that common law right, which can not be enjoyed save with the common law right. A patent confers a monopoly." *Id.*

[13] Patents, Patentability of Inventions 35 U.S.C. § 101 (2004). "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 therefore, subject to the conditions and requirements of this title." *Id.*

[14] *See* Electronic Frontier Foundation, The Patent Busting Project, http://www.eff.org/patent/ (last visited Sept. 25, 2005) (discussing the harm to individuals and smaller companies who cannot afford to defend themselves against forceful licensing agreements as well as the possible harm for unknown infringement as a result of illegitimate patent issuing); Stephen Shankland, *Group: Linux Potentially Infringes 283 Patents*, Aug. 1, 2004, http://news.zdnet.com/2100-3513_22-5291403.html?tag=nl. "Linux potentially infringes 283 patents, including 27 held by Microsoft but none that have been validated by court judgments, according to a group that sells insurance to protect those using or selling Linux against intellectual-property litigation." *Id.*

[15] *See* Richard Stallman, *Patent Absurdity*, ZDNET, June 20, 2005, http://news.zdnet.com/2100-9593_22-5754104.html?tag=st.prev (discussing the differences between patent and copyright law and how patent law has no place in the software world).

[16] UNITED STATES CENSUS BUREAU, COMPUTER AND INTERNET USE IN THE UNITED STATES: 2003, October 2005, available at http://www.census.gov/prod/2005pubs/p23-208.pdf . Software is the core of every computer. In order for a computer to run correctly it needs software to do so. Beyond the basic need for software to make a computer run, software is the piece which allows a computer to be become a specific tool in order to accomplish a specific task. The U.S. Census Bureau found that from 1984 to 2003 there was a 53.6 percent jump (8.2 percent in 1984 compared to 61.8 percent in 2003) in households with a computer. *Id.* at 1. The 2003 survey which included 113,126 total households, found 69,912 households had a computer. *Id.* at 2. An even more striking number exists when looking at the use of computers among children enrolled in grades K-12 in 2003. *Id.* at 7. The usage of computers at school by children K-12 topped out at 92.3 percent. *Id.* Taking these figures into account, the prevalence of computers in our everyday lives is staggering. All of these computers need software, often multiple types, demonstrating just how great an impact software has on the United States' society.

I. BACKGROUND

A. The History of Software Patent Treatment

The confusion concerning how software patents are to be handled began in *Gottschalk v. Benson*.[17] In *Gottschalk*, the Court found that a program written to convert signals from binary-coded decimal form into pure binary form on a digital computer was essentially a mathematical algorithm and thus not patentable.[18] The court did however state that perhaps these types of programs should be patentable, but added further that they were unable to speak on these matters during this case.[19] Three years later in *Diamond v. Diehr*, the court revisited the issue of mathematical algorithms and their patentability.[20] In *Diamond*, the court noted "[i]t is now commonplace that an *application* of a law of nature or mathematical formula to a known structure or process may well be deserving of patent protection"[21] The court reasoned that incorporating mathematical formula or algorithms into an invention was not alone grounds for denial.[22]

The Court in *Diamond* took a substantial step towards the patenting of software by deciding to look at the invention as a whole, including mathematical formulas or algorithms.[23] The Court distinguished *Diamond* from *Gottschalk* by stating that, in *Gottschalk*, the claimed invention was a new mathematical formula or algorithm in the abstract whereas in *Diamond*, the claimed invention was a process including a mathematical formula or algorithm.[24] As a result of *Diamond*, two exceptions to patentability remained, the mathematical algorithm exception and the business method exception.[25]

During the years following *Gottschalk* and *Diamond*, a series of inconsistent decisions plagued the Court of Custom and Patent Appeals. In cases such as *In re Freeman* (1978) and *In re Meyer* (1982), the court struggled to accurately apply the holdings from cases such as *Gottschalk* and *Diamond*.[26] As a result of the

---

[17] *See* Gottschalk v. Benson, 409 U.S. 63, 72 (1972) (holding that a mathematical algorithm for converting signals from binary-coded decimal form into pure binary form was not patentable but perhaps patent law itself should in fact be extended to cover programs).

[18] *Id.* at 65; *see also* Parker v. Flook, 437 U.S. 584, 596 (1978) (refusing to overrule *Gottschalk* without a clear decision from Congress).

[19] Gottschalk, 409 U.S. at 72..

[20] Diamond v. Diehr, 450 U.S. 175, 177–78 (1981) (examining the patentability of a process molding uncured rubber using an equation on a digital computer).

[21] *Id.* at 187.

[22] *Id.*

[23] *Id.* at 192–93.

[24] *Id.* at 165.

[25] Michael Guntersdorfer, *Software Patent Law: United States and Europe Compared*, 6 DUKE L. & TECH. REV. 12 (2003).

[26] *See In re* Freeman, 573 F.2d 1237, 1247 (C.C.P.A. 1978) (reversing a board rejection concerning an invention which used a computer program in conjunction with an apparatus claimed; finding that the examiner read *Gottschalk* too broadly); *In re* Meyer, 688 F.2d 789 (C.C.P.A. 1982) (affirming an examiner's rejection of an invention claiming to carry out complex tests and analyze such data because the invention was found to be a mathematical formulae or algorithm which was not applied to physical elements and merely represented a mental process); *see also In re* Lowry, 32 F.3d 1579, 1583 (Fed. Cir. 1994) (reversing a PTO rejection of applicant's claim which presented a model of how to store data in memory based in part on the theory that even though the claim's

inconsistency, the court implemented a two-step analysis in determining the patentability of software patents.[27] The first question the court must ask is whether the invention directly claims mathematical formula or algorithms.[28] If so, the second question is whether the invention involves formula or algorithms in some physical process.[29] If the answer to the second question is yes, the claimed invention is for statutory subject matter and thus patentable.[30]

In 1992, *Arrhythmia Research Technology, Inc. v. Corazonix Corp.* represented a significant decision by the Federal Circuit.[31] The court held that the claimed invention[32] was statutory subject matter. [33] This is because even though the basis of the claimed invention focused around mathematical formula or algorithms, the process physically changed one signal into another.[34]

State Street Bank & Trust Company v. Signature Financial Group represented another great shift in how a mathematical formula, embodied within a software program, is to be handled.[35] In *State Street Bank*, the claimed invention was a financial system which would make all the necessary calculations for maintaining a partner fund.[36] Due to the complexity and the speed with which these calculations needed to be performed, it was necessary and proper to have this process performed by a computer.[37] The court examined the invention and found it to be statutory

---

stored data had no physical structure itself, the bits of information constituted the essence of an electronic structure); *In re* Schrader, 22 F.3d 290 (Fed. Cir. 1994) (rejecting a patent claim for a system which proposed a way of bidding on items such as contiguous pieces of land that upon successful completion of the bidding process, the software would interpret the bids in order to find which bid would maximize the sellers profits); 1-1 DONALD S. CHISUM, CHISUM ON PATENTS § 1.03 (6) (2004) (citing *In re* Alappat, 33 F.3d 1526, 1545 (Fed. Cir. 1994)). PTO erred in rejecting as nonstatutory subject matter applicant's claims stating:

> (1) properly interpreted, the claim's means limitations are restricted to the corresponding circuitry in the specification, and equivalents thereof, making the claim a true machine or apparatus claim; (2) assuming that the "mathematical algorithm" exception applies to true apparatus claims, the claimed apparatus does not fall within that exception; (3) Supreme Court decisions exclude laws of nature, natural phenomenon and abstract ideas; mathematical concepts are not a fourth excluded concept; (4) given the foregoing, the proper inquiry in dealing with the so called mathematical subject matter exception to § 101 ... is to see whether the claimed subject matter *as a whole* is a disembodied mathematical concept, whether categorized as a mathematical formula, mathematical equation, mathematical algorithm, or the like, which in essence represents nothing more than a "law of nature," "natural phenomenon," or "abstract idea." and (5) a programmed digital computer may represent patentable subject matter.

Id. (footnotes omitted, emphasis original).

[27] *In re Alappat*, 33 F.3d at 1544.

[28] *Id.*

[29] *Id.*

[30] *Id.* at 1545.

[31] Arrhythmia Research Tech., Inc. v. Corazonix Corp., 958 F.2d 1053 (Fed. Cir. 1992).

[32] *See id.* at 1054 (claiming an electrocardiograph which, in conjunction with a mathematical algorithm, was able to help predict the risk levels for ventricular tachycardia).

[33] *Id.* at 1059.

[34] *Id.*

[35] State St. Bank & Trust Co. v. Signature Fin. Group, 149 F.3d 1368 (Fed. Cir. 1998).

[36] *Id.* at 1371.

[37] *Id.*

subject matter.[38] Furthermore, the court in *State Street Bank* dismissed both the mathematical algorithm exception[39] and the business method exception.[40] Instead, the court viewed the invention's practical utility in addition to considering the requirements of non-obviousness and novelty.[41] *State Street Bank* created a broad standard stating that instead of looking at the formula and process, the practical utility of the claimed invention was the standard.[42] The court also reasoned that as long as a "tangible result" was reached, mathematical formula or algorithms should be patentable.[43]

In 1999, the court revisited the broad standard announced in *State Street Bank,* but gave it a more narrow construction.[44] In *WMS Gaming*, the court found that software, in essence, created a specialized purpose for what is considered to be a general-purpose machine.[45] The software code, when executed on the computer, converts the computer into that of only one specific use.[46] Although seemingly more expansive, the holding in *WMS Gaming* is still somewhat narrower than that in *State Street*. In *WMS Gaming*, the court effectively required a change of a

---

[38] *Id.* at 1375.

> 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 Title 35, including novelty, nonobviousness, and adequacy of disclosure and notice. For purpose of our analysis, as noted above, claim 1 is directed to a machine programmed with the Hub and Spoke software and admittedly produces a "useful, concrete, and tangible result." This renders it statutory subject matter, even if the useful result is expressed in numbers, such as price, profit, percentage, cost, or loss.

Id. (citations and footnote omitted).

[39] *Id.* at 1373 n.4. "This has come to be known as the mathematical algorithm exception. This designation has led to some confusion . . . . By keeping in mind that the mathematical algorithm is unpatentable only to the extent that it represents an abstract idea, this confusion may be ameliorated." *Id.*

[40] *Id.* at 1375. "We take this opportunity to lay this ill-conceived exception to rest. Since its inception, the "business method" exception has merely represented the application of some general, but no longer applicable legal principle . . . ." *Id.*

[41] State St. Bank & Trust Co. v. Signature Fin. Group, 149 F.3d 1386, 1375 (Fed. Cir. 1998); *see also* AT&T Corp. v. Excel Commn., Inc., 172 F.3d 1352, 1353 (Fed Cir. 1999) (reversing PTO rejection of a claim for ". . . a message record for long-distance telephone calls that is enhanced by adding a primary interexchange carrier ("PIC") indicator" by examining claims as a whole).

[42] *See State St.*, 149 F.3d at 1375.

[43] *Id.* (citing *In re* Alappat, 33 F.3d 1526, 1544 (Fed. Cir. 1994)).

[44] WMS Gaming, Inc. v. Int'l Game Tech., 184 F.3d 1339, 1343 (Fed. Cir. 1999) (deciding on ". . . patent claims [for] a slot machine that decreases the probability of winning while maintaining the external appearance of a standard mechanical slot machine. The decreased probability of winning permits higher payoffs, which attracts players.").

[45] *Id.* at 1348 (quoting *In re* Alappat, 33 F.3d 1526, 1545 (Fed. Cir. 1994) holding "[a] general purpose computer, or microprocessor, programmed to carry out an algorithm 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.'").

[46] *Id.*

generalized computer into one that was specialized, whereas *State Street* focused on the "tangible result" and practical utility.[47]

B. How the United States Patent Office Decided to Handle Software Patents

The 1981 edition of the *Manual of Patent Examining Procedure* ("MPEP") began to deal with the patenting of mathematical formula and algorithms in § 2110.[48] The MPEP followed *Gottschalk* in viewing an "algorithm, or mathematical formula like a law of nature, which cannot be the subject of a patent."[49] The MPEP also drew six components from *Diamond*.[50] Ultimately, the MPEP boiled the examination of software claims down to a rather simple definition as to when a software program is considered non-statutorily patentable material. The MPEP stated that if the claims state *only* a mathematical algorithm or calculation, the claims are not statutory

---

[47] *Compare* WMS Gaming, Inc. v. Int'l Game Tech., 184 F.3d 1339, 1348–1349 (Fed. Cir. 1999) (finding that a mathematical formula or algorithm is patentable in a means-plus-function claim because the computer is changed from a general purpose computer to a specialized one), *with* State St. Bank & Trust Co. v. Signature Fin. Group, 149 F.3d 1368, 1374–1375 (Fed. Cir. 1998) (stating the focus of whether a claim encompasses statutory subject matter should be based on its practical utility as well as it's tangible result).

[48] U.S. PAT. & TRADEMARK OFFICE, U.S. DEP'T OF COMMERCE, MANUAL OF PATENT EXAMINING PROCEDURE § 2110 (4th ed. 1979, rev. 1981).

[49] Gottschalk v. Benson, 409 U.S. 63, 72 (1972).

[50] 1-1 DONALD S. CHISUM, CHISUM ON PATENTS § 1.03 (6) (2004).

> 1. The claims must be considered as a whole. It is inappropriate to dissect the claims into old and new elements and then to ignore the presence of the old elements in the analysis. The "novelty" of any element or steps in a process, or even of the process itself, is of *no relevance* in determining whether the subject matter of a claim falls within the 101 categories of possibly patentable subject matter.
> 2. When a claim containing a mathematical formula implements or applies that formula in a structure of process which, when considered as a whole, is performing a function which the patent laws were designed to protect (e.g., transforming or reducing an article to a different state or thing), then the claim satisfies the requirements of § 101.
> 3. When a claim recites a mathematical formula (or scientific principle or phenomenon of nature), an inquiry must be made into whether the claim is seeking patent protection for that formula in the abstract. (If the claim does seek protection for such a mathematical formula, it would be non-statutory under 35 U.S.C. 101).
> 4. A mathematical formula as such is not accorded the protection of our patent laws, ... and this principle cannot be circumvented by attempting to limit the use of the formula to a particular technological environment. Similarly, insignificant post solution activity will not transform an unpatentable principle into a patentable process.
> 5. When a claim as in *Parker v. Flook,* 198 USPQ 193 (1978), is drawn to a method for computing an "alarm limit" (which) is simply a number, the claim is non-statutory under 35 U.S.C. 101 because *Flook* "sought to protect a formula for computing this number.
> 6. It is now commonplace that an *application* of a law of nature or mathematical formula to a known structure or process may well be deserving of patent protection.).

Id. (citations omitted)(emphasis added).

subject matter.[51] The MPEP demonstrated this by providing an example of a software program and stating that absent some type of associated machine to accomplish a specific purpose, the code would merely represent an idea or abstract concept of the programmer and would thus not constitute statutory patentable material.[52]

In 1996, the USPTO published an updated version of the guidelines to be used when examining computer-related inventions.[53] Within this version of the guidelines, the USPTO explained how examiners should determine if software can be considered statutorily patentable. The 1996 guidelines stated that a statutory invention is one in which the claims define a useful machine in terms of either its hardware or its combination of hardware and software.[54] The general thrust of the 1996 guidelines stated that if the software is defined alongside hardware and performs a valid non-obvious function, it is patentable.[55] The 1996 guidelines are consistent with the previous two-step examination process as well as subsequent decisions such as *State Street Bank* and *WMS Gaming*.[56]

The current version of the MPEP construes software in yet an even simpler way by refining the 1996 guidelines.[57] The MPEP states that if the idea is solely a mathematical process, without a link to an actual process, it is a nonstatutory process and thus not patentable.[58] Examining the previous case law, legislation, and manuals concerning mathematical formula and algorithms, the USPTO determined that if code, when claimed with a hardware counter-part, provides a practical result and is a part of an entire process as a whole, it is patentable.

Essentially, case law since *Diamond* proves only that a mathematical formula or algorithm, represented as a piece of software code, is indeed patentable if claimed

---

[51] *Id.*

[52] *Id.*

[53] Examination Guidelines for Computer-Related Inventions, 61 Fed. Reg. 7478 (Mar. 29, 1996).

[54] *Id.* § (IV)(b)(2)(a) stating:

> *(a) Statutory Product Claims.*-If a claim defines a useful machine or manufacture by identifying the physical structure of the machine or manufacture in terms of its hardware or hardware and software combination, it defines a statutory product.
>
> A machine or manufacture claim may be one or two types: (1) A claim that encompasses any and every machine for performing the underlying process or any and every manufacture that can cause a computer to perform the underlying process, or (2) a claim that defines a specific machine or manufacture. When a claim is of the first type, Office personnel are to evaluate the underlying process the computer will perform in order to determine the patentability of the product. (footnotes omitted).

Id.

[55] *Id.*

[56] State St. Bank & Trust Co. v. Signature Fin. Group, 149 F.3d 1368, 1371 (Fed. Cir. 1998); WMS Gaming, Inc. v. Int'l Game Tech., 184 F.3d 1339, 1343 (Fed. Cir. 1999).

[57] MPEP § 2106 (IV) (B) (1) (8th ed. 2001, rev. 2005).

[58] *Id.* ("In practical terms, claims define nonstatutory processes if they: - consist solely of mathematical operations without some claimed practical application (i.e., executing a 'mathematical algorithm'); or - simply manipulate abstract ideas without some claimed practical application.")(citations omitted).

and structured correctly.[59] The main concern during the examination process lies in the determination of "prior-art" upon which to make a rejection.[60] Now that the barriers to filing a patent for software have been broken down and justification has been given, how does the USPTO make sure claims are non-obvious? This is where the examination process breaks down. Software is akin to a living organism. It is always evolving. How do you determine what is prior art when common practice is to build upon and use existing code in the creation of a new invention?

II. ANALYSIS

A. The Software Patent, a Necessary Evil

The patent system was created in order to encourage and foster innovation through the disclosure and expression of ideas.[61] Patents have long spanned many different areas of industry from biology to mechanical to engineering to the simple entrepreneurial progression.[62] There are opinions that software patents should not be entitled to the same type of protection as ideas in other industries.[63] Although the examination process that software patents undergo may be flawed, that does not mean software is not entitled to the same patent protection as other patentable subject matter. Software is a relatively new technology and like any new technology when it begins to be patented, there are problems that an examiner must take into consideration. With software patents these problems are: (1) the lack of prior art; (2) the way examiners determine obviousness; and (3) determining novelty. These

---

[59] *See generally* Diamond v. Diehr, 450 U.S. 175, (1981); Arrhythmia Research Tech., Inc. v. Corazonix Corp., 958 F.2d 1053 (Fed. Cir. 1992); *In re* Alappat, 33 F.3d 1526 (Fed. Cir. 1994); *In re* Lowry, 32 F.3d 1579 (Fed. Cir. 1994); State St. Bank & Trust Co. v. Signature Fin. Group, 149 F.3d 1368 (Fed. Cir. 1998); WMS Gaming, Inc. v. Int'l Game Tech., 184 F.3d 1339 (Fed. Cir. 1999); AT&T Corp. v. Excel Commn., Inc., 172 F.3d 1352 (Fed Cir. 1999).

[60] Patents, Patentability of Inventions, 35 U.S.C. § 102 (2004).

[61] PAT CHOATE, HOT PROPERTY: THE STEALING OF IDEAS IN AN AGE OF GLOBALIZATION 27 (Alfred A. Knopf ed., Borzoi Book 2005) (". . . [T]he basic concept was simple: patents and copyrights encouraged inventors and authors to produce more new and useful creations. These innovations could help the U.S. progress. And as the details of these creations became public, the general knowledge of the nation would be expanded.").

[62] *See* Bob Mims, *Success Or Failure: Patent May Be The Difference*, DELEWAREONLINE: THE NEWS JOURNAL, Sept. 04, 2005, http://www.delawareonline.com/apps/pbcs.dll/article?AID=/20050904/BUSINESS/509040360/1003 (discussing how entrepreneurs use patents to capitalize on an invention that was years in the making).

[63] *See* Mark H. Webbink, *A New Paradigm for Intellectual Property Rights in Software*, 12 DUKE L. & TECH. REV. 4 (2005). Webbink states that had software patents been available in 1975, we would be locked into early version of programs such as VisiCalc and Wordstar. *Id.* This approach is flawed. When these programs were first introduced, they were truly unique and revolutionary. Inventions such as VisiCalc and Wordstar are those at which patents are directed. The logic that had these pieces of software been patented, people would be "locked" into using them is a fallacy. Historically this is not the case for patented inventions. Licensing is how industries have handled new inventions such as those for years. Rather than being locked in, companies would pay licensing fees to use the patented technology. Or rather, the common practicing of reverse engineering and designing around patents would have occurred. The comment somehow implies that software should not be entitled to the same protection as other inventions in every other industry, merely because it is software.

problems are not unique to software applications, rather they have been confronted each time the patent system is forced to adjust to a previously unknown area of technology.

Purpose is ultimately the most important factor to strive for. The question, what is meant by purpose, is better stated: what was the patent system created for? Patents were created to encourage the sharing of information, and through this sharing foster innovation.[64] Can this purpose be realized with software or is it a hopeless situation? It is simply wrong to believe that the purpose of patents cannot be embodied in software patents. Software patents, just as other utility patents, serve to disclose information in exchange for a limited monopoly of that information. It is this disclosure that adds to the common pool of information and as a result fosters innovation, not stifles it. People must stop condemning software patents to an early doom and realize that a new approach must be taken. No person can deny that the software patented in *Arrhythmia Research Technology, Inc. v. Corazonix Corp.* for the monitoring of signs for tachycardia was not only useful, but lifesaving.[65] Should this invention not have been patented because it is software?

B. The Suffocation of Innovation: The Negative Effect of Software Patents

The arguments against patents tend to surface when the patents themselves seem to frustrate the original purpose of the patent system. This occurs when frivolous patent applications are filed and the system is abused.[66] Although our patent system is designed to protect new ideas, exploitation of the system can occur.[67] This abuse seems most prevalent in the software world.[68] The adverse effects on the software world are the primary focus of those opposed to software patents.[69] Effects such as the substitution of patents for traditional research and development are the basis of arguments against software patents.[70] Those opposed also focus on the negativity of cross-licensing.[71] Cross licensing, although a standard practice, is probably the most devastating of all of the adverse effects on the software industry.

---

[64] *See* CHOATE, *supra* note 61 at 27, and accompanying text.

[65] *See generally* Arrhythmia Research Tech., Inc. v. Corazonix Corp., 958 F.2d 1053 (Fed. Cir. 1992).

[66] *See generally* Electronic Frontier Foundation, The Patent Busting Project, http://www.eff.org/patent/wp.php (last visited Sept. 25, 2005). The EFF lists numerous patents which were filed and issued that claimed simple technologies. *Id.*

[67] PAT CHOATE, HOT PROPERTY: THE STEALING OF IDEAS IN AN AGE OF GLOBALIZATION 27 (Alfred A. Knopf ed., Borzoi Book 2005) (". . . the basic concept was simple: patents and copyrights encouraged inventors and authors to produce more new and useful creations. These innovations could help the U.S. progress. And as the details of these creations became public, the general knowledge of the nation would be expanded.").

[68] Mark H. Webbink, *A New Paradigm for Intellectual Property Rights in Software*, 12 DUKE L. & TECH. REV. 5 (2005). There have been over 150,000 software patents issued in the last 22 years with many of the applications being dated after 1994. *Id.*

[69] *Id.* at 6. (stating findings that there is a negative correlation between a firm's patent focus and its research and development intensity).

[70] *Id.*

[71] *Id.* at 14–16. (discussing how cross licensing benefits the large companies by suppressing litigation costs while at same time effectively shutting out the smaller start-up software companies which aid in software innovation); *see also* Jim Dalyrymple, Macworld, *Is The iPod In Trouble?*

1. The Spiteful Opponent

Creative's[72] current fight against Apple and their Apple iPod[73] is a perfect example of how the problems in the software patent world are manifesting themselves. Creative filed a patent application on January 5, 2001 claiming a hierarchical structure for files on a portable music device.[74] Creative's patent was issued on August 9, 2005,[75] and the release of this information was documented in numerous news sources later that month.[76] September 1, 2005, articles were published stating that Apple's incredibly popular iPod possibly infringed on Creative's patent.[77] Creative stated in a press release that Apple had also filed for a patent in late 2002 for a "user interface in a multimedia player."[78] According to Creative, Apple's patent was later rejected.[79] In considering the issuance of Creative's patent, Craig McHugh, president of Creative's United States operation, stated only that they had identified Apple as being in violation of their patent and that historically, Creative has always been vigorous in defending their patent portfolio.[80]

How does one interpret the Creative versus Apple situation? No clear method exists for determining who is in the wrong and who is in the right. Creative followed the proper procedures for filing and obtaining a patent.[81] Is Apple an innocent victim? Many of those that oppose software patents do feel as if Apple is a bystander

---

Apple's Patent Woes, Macworld No. 11 Vol. 22, Nov. 1, 2005 (describing Creative Technologies patent infringement suit against the Apple iPod and speculating as to whether licensing issues may arise).

[72] *See* Creative.com, About Us, http://www.creative.com/corporate/about (last visited Oct. 31, 2005) ("Creative (NASDAQ: CREAF) is the worldwide leader in digital entertainment products for the personal computer and the Internet.").

[73] *See generally* Apple.com, ipod, http://www.apple.com/ipod/ipod.html (last visited Oct. 31, 2005).

[74] U.S. Patent No. 6,928,433 (filed Jan. 5, 2001) (describing in the abstract "[a] method, performed by software executing on the processor of a portable music playback device, that automatically files tracks according to hierarchical structure of categories to organize tracks in a logical order. A user interface is utilized to change the hierarchy, view track names, and select tracks for playback or other operations.").

[75] *Id.*

[76] *See* Anne Broache, *Creative Wins Patent For MP3 Player Interface,* CNET NEWS.COM, Aug. 30, 2005, http://news.com.com/Creative+wins+patent+for+MP3+player+interface/2100-1041_3-5844472.html?tag=st_lh; *see* also Chris Noon, *Apple Faces iPod Patent Threat,* FORBES.COM, Aug. 31, 2005, http://www.vnunet.com/forbes/news/2141698/job-apple-ipod-patent-threat (describing Apple possibly infringing on Creative's patent); Stuart Miles, *Creative Wins Patent Race For MP3 Player Interface,* POCKET-LINT.CO.UK, Aug. 31, 2005, http://www.pocket-lint.co.uk/news.php?newsId=1587.

[77] Laurie J. Flynn, *Apple Is Accused of Violating Software Patent,* N.Y. TIMES, Sept. 1, 2005, at C4; *see also* Brad Cook, *Creative Accuses Apple of Violating Its Patent,* THE MACOBSERVER, Sept. 1, 2005, http://www.macobserver.com/article/2005/09/01.7.shtml; David Richards, *Creative To Go After Apple,* SMARTOFFICE//NEWS, Sept. 2, 2005, http://www.smartofficenews.com.au/Computing/Industry?article=/Computing/Industry/News/J3U8F7F9.

[78] Press Release, Creative, Creative Awarded U.S. Patent On Its Invention Of User Interface For Portable Media Players, August 30, 2005 (http://www.creative.com/press/releases/welcome.asp?pid=12175).

[79] *Id.*

[80] *See generally* Laurie J. Flynn, *Apple Is Accused of Violating Software Patent,* N.Y. TIMES, Sept. 1, 2005, at C4.

[81] U.S. Patent No. 6,928,433 (filed Jan. 5, 2001).

in this situation and that Creative is taking advantage of them.[82] There are underlying facts that may support this opinion. Creative was one of the first companies to release a successful digital music player.[83] However, Apple now dominates the digital music player market with its now famous iPod.[84] It seems convenient that once Creative's patent issued they immediately named Apple as an infringer. As a remedy Creative could force licensing fees or attempt to file an injunction to halt Apple's alleged infringement. Either way, Creative stands to gain something in the digital player market whether through a market share, by attempting to limit Apple's popularity, or through licensing fees in which Creative can sit back and collect money from Apple while doing nothing. Although this situation may offend one's personal morals, is it really wrong?

Looking at the situation from an objective point of view we may come to a different conclusion. Creative's patent was filed first[85] while Apple filed for the same type of patent a year later.[86] Had Apple filed first and obtained the patent before Creative, the situation would have changed and the fighting would be reversed or non-existent. Apple would have a patent for its digital music player and would have effectively been able to prevent others from developing other digital audio players to compete. It would force other manufacturers to come up with a new type of digital music player that would not infringe upon Apple's patent. This role reversal seems to be acceptable over the current actual situation. The only reason this scenario is easier to accept is because the supposed spiteful company, Creative, would not be able to harm the successful company, Apple, by merely parading a piece of paper in their face. Look at the situation again, though. This has *nothing* to do with software patents, but rather who patented a patentable invention first. This is the world of patents. Using the Creative versus Apple situation to specifically condemn software patents provides no basis to justify the extinguishing of software patents.

---

[82] *See generally*, *Apple Is Accused of Violating Software Patent*, SLASHDOT.ORG, http://apple.slashdot.org/article.pl?sid=05/09/02/1615237&tid=123&tid=155&tid=3 (last visited Oct. 31, 2005). The comments on Slashdot tend to take the form of ranting rather than coherent comments. Slashdot, however, represents a very large section of the programming community spanning multiple countries and multiple industries. The general outcry when discussing patents is for an overall dismissal of the patent system as a whole. The percentage of those commenting who truly understanding the legal principles involved with intellectual property is low. As a result, a Slashdot post such as this is representative of the anger towards the patent system by people who don't necessarily truly understand it. Many of those whom comment are active in lobbying against software patents and patents as a whole. In order to reform the patent system, the USPTO's need to address the general perception that the community has so that it may solicit input rather than hate mail.

[83] *See* Phil O'Shaughnessy, *Creative Expands Nomad Family With New Portable Digital Audio Players,* CREATIVE LABS INC., Jan. 5, 2000, http://us.creative.com/corporate/pressroom/releases/welcome.asp?pid=6193 (describing their line of digital audio players known as the Nomad and the Jukebox).

[84] *See* Connie Guglielmo, *Apple's Jobs Taps Teen iPod Demand to Fuel Sales, Stock Surge*, BLOOMBERG.COM, Oct. 11, 2005, http://quote.bloomberg.com/apps/news?pid=10000103&sid=a58iozj_2jXM ("The iPod had an 82 percent share of the market in U.S. retail stores in the 12 months ended in August, up from 64 percent in the same period a year earlier, and 33 percent two years ago, according to Port Washington, New York-based NPD Group, Inc.").

[85] U.S. Patent No. 6,928,433 (filed Jan. 5, 2001).

[86] U.S. Patent Pub. No. 2004-0055446 (filed Oct. 28, 2002).

2. The Problem With Content

The issue of whether or not Creative's patent should have been issued is another story completely. Whether or not a patent should issue is the focus of many in their ongoing protest against software patents, and rightly so.[87] One author describes today's patent practices as functioning as nothing more than a defensive portfolio to prevent companies from threats of others even though they have successfully walled themselves off through cross-licensing.[88]

This might be true. The patent office is being flooded with software patent applications.[89] Some companies are filing multiple applications for the same idea, not in an attempt to double patent, but rather with minor changes in each application.[90] This causes a problem in that an examiner must research multiple patents on one idea where, in reality, the examiner should only need to review a few patents to obtain the same information.

The practice of slicing up patents into small divisions derives from an even bigger problem. Many of the software patents currently being issued should have never made it through the Patent Office. The patent examiner is held to strict guidelines when examining a patent.[91] If an application survives the patenting process, the application may issue.[92] This problem arose with respect to Creative's patent for a digital music player.[93] In Creative's patent application, the claims set out a device which will play digital music arranged in a hierarchal structure.[94] Although the portable music device may be patentable, the software providing the structure for the filing system is merely an element of the device as a whole and does not deserve protection on its own. This application was submitted on January 5, 2001.[95] The hierarchal structure laid out in the claims should have been rejected

---

[87] Mark H. Webbink, *A New Paradigm for Intellectual Property Rights in Software*, 12 DUKE L. & TECH. REV. 7 (2005) (comparing and contrasting the patent practices of the software industry to the patent practices of the pharmaceutical industry where software patents are being filed frivolously in mass numbers and the pharmaceutical industry is taking time to patent only that which they hope to protect).

[88] *Id.* at 18.

[89] *Id.* at 6. (stating how software companies such as Microsoft are spending more money on patents than on research and development).

[90] *Id.* Webblink describes a phenomenon that seems to be occurring in the technology world, specifically the computer industry, concerning patent filings and research and development. *Id.* Webblink uses Microsoft as an example stating that they would be increasing their patent filings from 2,000 to 3,000 in 2004. *Id.* However, there was no corresponding increase in research and development. *Id.* Interpreting these numbers, Webblink makes the logical assertion that the same research and development is being sliced smaller and smaller into individual patent applications rather than attempting to file broader patents encompassing the essence of the inventions. *Id.*

[91] *See generally* MPEP 2005.

[92] *Id.*

[93] U.S. Patent No. 6,928,433 (filed Jan. 5, 2001) (describing in the abstract "[a] method, performed by software executing on the processor of a portable music playback device, that automatically files tracks according to hierarchical structure of categories to organize tracks in a logical order. A user interface is utilized to change the hierarchy, view track names, and select tracks for playback or other operations.").

[94] *Id.*

[95] *Id.*

under § 103 obviousness.[96] An examiner would not have to look far to find prior art for such claims because any operating system works on the idea of hierarchal folder structuring.[97] It is logical to assume that when a certain function of a large computer will be shrunken down to a smaller size for portable use, the portable version will retain the same core elements for that function as the large computer. In other words, it makes sense that the smaller portable music player would arrange files much the same as its larger counter-part. If the digital music player *did not* keep the same process, it would likely have one very similar. In the Creative patent, Creative did nothing more than recite a modified hierarchal tree structure focused on tracks which are the same as files on computers.[98] For this reason, Creative's patent should not have been issued.

The more important question is, why did this patent issue? If the patent examiner cannot find a way to deny the application within the confines of the MPEP, the examiner has no choice but to issue the patent.[99] This is the largest problem facing software patents. Numerous patents have been issued where the subject matter was questionable in much the same way as Creative's patent.[100] Until the USPTO adjusts its methods of examining software patents, situations such as the Creative versus Apple scenario will continue to surface. If the UPSTO allows basic software ideas to be patented, innovation will be stifled.

C. The Blow to the Open Source Community

Those with the most to lose in the software patent war are within the open source community.[101] Basic software practice is based on reuse and modular

---

[96] Patents, Patentability of Inventions, 35 U.S.C. § 103(a) (2004). "A patent may not be obtained . . . if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art . . . ." *Id.*

[97] *See generally* AuditMyPC.com, *HFS is an acronym for . . .* , http://www.auditmypc.com/acronym/HFS.asp (last visited Oct. 31, 2005) ("Hierarchical File Systems. The inverted tree structure that most computer systems use for organizing directories and files."); *see also* PCMAG.COM, *hierarchical file system*, http://www.pcmag.com/encyclopedia_term/0,2542,t=hierarchical+file+system&i=44240,00.asp (last visited Oct. 31, 2005) (defining hierarchical file system).

[98] *See generally* U.S. Patent No. 6,928,433 (filed Jan. 5, 2001).

[99] *See generally* MPEP 2005.

[100] *See* U.S. Patent No. 5,960,411 (filed Sept. 12, 1997) (describing "[a] method and system for placing an order to purchase an item via the Internet"); U.S. Patent No. 5,715,314 (filed Oct. 24, 1994) (describing "[a] network-based sales system includes at least one buyer computer for operation by a user desiring to buy a product, at least one merchant computer, and at least one payment computer"); U.S. Patent No. 4,873,662 (filed Aug. 15, 1980) (describing the method of creating hyperlinks on computers for use of navigation); U.S. Patent No. 6,389,458 (filed Oct. 30, 1998) (claiming "[m]ethod, apparatus and system for directing access to content on a computer network" utilizing popup windows); U.S. Patent No. 6,289,319 (filed Nov. 30, 1994) (encompassing the general ideas of online transactions); U.S. Patent No. 6,029,141 (filed Jun. 27, 1997) (describing an "internet-based customer referral system"). These patents should have been found to be obvious and rejected. The simple technologies being claimed would have been obvious to anyone skilled in the art of computers and programming.

[101] *See generally* Open Source Initiative, http://www.opensource.org/ (last visited Oct. 31, 2005) ("The basic idea behind open source is very simple: When programmers can read, redistribute, and

programming to allow for one piece of code to be integrated into another. This fosters better programs by allowing multiple programmers to edit and refine a piece of code with the hopes of creating the best code possible for that function.[102] The best example of this is Linux.[103] Linux is the ultimate representation of the open source community and is facing its own patent war as well.[104] The battle over Linux is just the beginning if things are not adjusted. Linux is a collaboration of many of the best and brightest programmers around the world.[105] Linux represents the technology surge over the last decade and how the open source community can achieve amazing results in the programming world. These accomplishments have led many to believe that open source should be the only type of software, free and distributed under the GNU General Public License.[106] It is the open source community that feverishly opposes software patents because they feel it is a threat to their livelihood.[107] However, these fears could be avoided if the patent system is adjusted to ensure quality among software patent applications.

III. PROPOSAL: ADJUSTING THE PROCESS TO ENSURE QUALITY

Realizing the flaws in the process of patenting software programs is only the first step toward fixing the problem. The next step is identifying a way to correct it. The current examination procedures for new patent applications need to be revised. A readily apparent flaw is the inability for an examiner to conduct a proper prior art search. This is not to say the examiner is not qualified to do so, but rather software

---

modify the source code for a piece of software, the software evolves . . . . People improve it, people adapt it, people fix bugs . . . . Open Source Initiative exists to make this case to the commercial world.").

[102] *Id.*

[103] *See generally* LINUX ONLINE, *What is Linux*, http://www.linux.org/ (last visited Oct. 31, 2005) "Linux is a free Unix-type operating system originally created by Linus Torvalds with the assistance of developers around the world. Developed under the GNU General Public License, the source code for Linux is freely available to everyone." *Id.*

[104] *See generally* Groklaw, http://www.Groklaw.net (last visited Oct. 31, 2005). The SCO versus IBM case began with an initial complaint filed on January 4, 2004. *Id.* SCO alleged that IBM used pieces of UNIX code, which it had obtained through a series of corporate acquisitions, in the development of Linux. *Id.* SCO filed a complaint which has now been amended alleging multiple contract breaches, copyright infringement, and unfair competition to name just a few causes of actions. *Id.* This lawsuit is being closely watched by the open source community as it directly affects the Linux world. SCO has gone as far as to warn users of Linux of their infringement. *Id.* This very well be one of the most pivotal intellectual property cases to be decided in the last decade.

[105] *Id.*

[106] *See generally* GNU General Public License, http://www.gnu.org/licenses/gpl.html (last visited Oct. 31, 2005).

[107] S*ee* Electronic Frontier Foundation: The Patent Busting Project, http://www.eff.org/patent/ (last visited Sept. 25, 2005); Petition Against Software Patents, http://www.petitiononline.com/pasp01/petition.html (last visited Sept. 25, 2005); NoSoftwarePatents.com, http://www.nosoftwarepatents.com/en/m/basics/webshop.html (last visited Sept. 25, 2005); Foundation for a Free Information Infrastructure, http://www.ffii.org/members/index.en.html (last visited Sept. 25, 2005) (detailing the number of members which support the organization).

is unique in the amount of prior art available. Software is a relatively new technology and has recently become paramount to our everyday lives.[108]

Software is somewhat different than other patentable technologies, the scope of which is hard to imagine. Programming is not something that is practiced by only professionals. Programming is being done by young children and teenagers as well as billion dollar companies. Some of these "enthusiasts" are even creating new technologies that are becoming staples of everyday life.[109] Beyond this problem, there is another whole sub-culture of the programming world that wish to dedicate their work to the public use for free.[110] These programmers work amidst themselves in their own community until their work is adopted and/or obtains some sort of success. It is unrealistic to assume than an examiner is in the best position to search for prior art, taking into account all that this information and innovation is happening on a daily basis and sometimes in somewhat secluded circles. Rather, the examiner needs some sort of help through a refinement of the examination process.

A first step would be the USPTO looking at a host of software patents as examples of where the examination process took a wrong turn.[111] Although, these applications may have seemed valid at first blush, further investigation could have discovered prior art. The USPTO needs to adopt special processes when it comes to examining software patents. In a re-examination of Creative's patent for its hierarchal file system, it will be apparent to an examiner that it did not meet the § 103 non-obviousness requirement.[112] In order to protect basic level software patents from issuing and stifling innovation, the USPTO needs to take steps to increase the scrutiny by which software patents are examined.

The first step in attempting to address these issues is to determine how to better aid the examiner during the examination process. Considering the most crucial and most difficult part of the examination is the location of prior art to serve as a basis for rejection or allowance, this should be the area of focus for change. The creation of the "Software Evaluation Panel" ("SEP") would address this issue. The SEP's main purpose would be to alleviate the examiner's reliance on his limited knowledge of software and history and add the cumulative knowledge of the programming community to be at his disposal. Generally, this board would provide prior art to the examiner who would then determine whether the information supplied to him by the SEP is sufficient to warrant a rejection of some kind.

The SEP would be comprised of a cross-section of the programming community. This cross-section would include those from the open-source community,[113] private business entities, small business entities, and any other members who would better serve to represent the programming community as a whole. The selection to the board would be through nomination by the respective community. Once on the SEP, the members would serve a limited term so as to promote ethical behavior in order to

---

[108] *Supra* note 16 and accompanying text.

[109] *See* Peter Norton, The Future of Peer-to-Peer File Sharing, http://www.glencoe.com/norton/online/ezine/display_article.phtml?id=226 (last visited Apr. 10, 2006) (explaining how Sean Fanning at 19 years old created Napster, which was the first commercially successful peer-to-peer computing model, while in college).

[110] *See supra* note 4.

[111] *See supra* note 100 and accompanying text.

[112] *See* discussion *supra* Part B.2.

[113] *See supra note* 4.

support re-election. The members of the SEP would be held to high level of confidentiality due to the necessary disclosures needed to be given to the SEP by companies. These disclosures would be integral to the effectiveness of the SEP. Although abuse is a concern, the diverse makeup of the SEP from the programming community should act as an additional deterrent to those hoping to abuse it. These members, cooperatively, would in theory represent the cumulative knowledge of the programming community.

Once elected to the SEP, the member's main purpose would be to serve as a fact finding entity to aide the examiner in the patenting process. Although perhaps well versed in patent prosecution, the members of the SEP will not act in any capacity as an examiner, they will only act to aide him in obtaining information. The SEP would receive software applications after submitted and assigned to an examiner and would be charged with searching for any information that might show that the submitted application is not worthy of patent protection. The information gathered by the SEP would then be given to the examiner who would make the decision if the prior art found by the SEP serves as grounds for a patent rejection.[114] At no point in time will the SEP make a determination on the patent application, only submit the information it has found in the course of its search.

An important concern is maintaining the integrity of the SEP and who would provide funding. Although funding by the USPTO would probably be the most proper as the SEP would be considered part of the UPSTO, it is unlikely that the UPSTO would be able or want to do so. Rather, funding could be provided by those the SEP would be created to protect. All aspects of the programming community are hurt by frivolous software patents. It would serve the entire community better to avoid the frivolous patents and the litigation costs associated with them. The group in the best position to take advantage of that is the programming community themselves. As a result, it is likely the programming community would be willing to help and fund the SEP.

Ethical concerns will always surround the SEP. Members of the SEP could conspire to not "search" as hard for some patents as well as others. This ethical concern is present in any entity but more so on the SEP especially if funded by the community itself. The best defense against abuse is the use of elections with a short term limit as well as the structure itself. The structure would be composed of opposing entities such as a large corporate programming companies and the open-source community. Neither would like to allow the other to gain the upper hand and thus a checks and balances system would naturally occur.

This system would prevent items like Creative's patent from issuing.[115] It would also stop many of the frivolous patents at the examination level.[116] The SEP would function in parallel to the ABA's proposed patent reforms.[117] Although the ABA's proposed reforms address the issues noted here, the reforms depend on a third party policing the examination process.[118] Unlike the proposed reforms, the SEP would

---

[114] *See* Patents, Patentability of Inventions, 35 U.S.C. § 102, 103 (2004).

[115] *See generally* U.S. Patent No. 6,928,433 (filed Jan. 5, 2001).

[116] *See generally supra* note 100 and accompanying text.

[117] ABA, *A Section White Paper: Agenda for 21st Century Patent Reform* (2005).

[118] *Id.* The ABA's patent reform includes three specific provisions for dealing with the issuance and challenging of a patent. *Id* at i. The proposed reforms are "Create a Post-Grant Opposition

require no intervention by third parties and would further reduce the need for the proposed reforms in regards to software patents by eliminating the need for many software patents to be challenged because the SEP would have provided the information normally provided by the third party challenge. The earliest the patent reforms provide for the challenge of a pending application is at publication.[119] The best situation is having the SEP work in tandem with the proposed patent reforms. In addition to providing another method by which to challenge applications, the patent reform would serve as a final check on the SEP as it would allow a third party to challenge a software patent application that the SEP may have unethically failed to search properly.

The SEP could begin as a stand-alone entity functioning independent of the USPTO and working within the confines of the ABA's proposed reforms. The SEP would *be* the third party policing the publications and issuing of software patents. They would then relay this information to the appropriate entity in order to aid them in challenging the application. Although this system works well where there is a long delay between filing and issuing, as that time decreases, the SEP's ability to effectively research suspect applications will be severely limited. At that point, the SEP will have proven its effectiveness and will be able to state a case as to becoming a part of the USPTO. In becoming a part of the USPTO, the SEP would function even quicker by having access to applications not at publication, but at submission. This adoption by the USPTO would be based on the success of the SEP and thus would need to prove itself as a stand-alone entity first and foremost.

Although not a perfect system, it would silence many critics since it would be putting the system within their control. The main argument referred to numerous times by those opposing software patents is that the patents should never have issued in the first place. The SEP in conjunction with the proposed patent reforms would allow groups to represent themselves and aid in the examination process as well as having a final gateway provision via the proposed ABA patent reforms to police the SEP. If the software claims don't deserve protection, they will be able to prove it either through examination or through pre-grant challenges. Most importantly, software will receive protection without the animosity they are shown today.

IV. CONCLUSION

Software has become an integral part of our lives.[120] With such importance come inherent problems. One such problem is capitalism and the desire to turn a profit.

Proceeding", "Expand Pre-Grant Third Party Submissions", and "Expand Use of Inter Partes Re-Examination." *Id.* In all three proposed reforms, a third party is required to respond to the patent application. *Id* at 21–27. If the third party therefore is not paying attention to published and issued software patent applications, their window to challenge the validity of the patent via a cost effective manner may be severely limited. *Id.*

[119] *Id.* at 24. Of the three available proposals made by the ABA, expanding pre-grant third party submissions is the only pre-grant method to challenge patent applications. *Id.* This option only becomes available after the publication of the application where a third party may then submit art to aide the examiner. *Id.*

[120] *See supra* note 16 and accompanying text.

Patents were established to address both of these issues.[121] Patents offer a monopoly in exchange for disclosure in order to foster innovation.[122] However, software patents do not seem to hold true to this intention.

Software patents are relatively new to the patent world.[123] They have been looked at in many different ways and have evolved through case law.[124] Their most recent definition has been set within the MPEP.[125] Though the courts and the USPTO have set out to fit software within the traditional framework of the patent system, it is obvious that problems still remain. Enforcement of a frivolous software patents can manifest themselves in different ways and are most readily apparent where they affect the consumer through increased fees due to cross-licensing or the removal of a beneficial product.

Creative's patent is an embodiment of the problems accompanying software patents.[126] Make no mistake though, although Creative's legal strategy against Apple's iPod may not seem morally comforting, it has nothing to do with software patents. Patent owners enforcing their rights are not the problem, though some tend to focus their argument in that direction. The problem, however, lies in whether the patent should have existed in the first place. Creative's patent could have been rejected based on prior art or non-obviousness.[127] The fact that it wasn't is the problem.

There needs to be a more exacting standard for scrutinizing software patent applications. This can be obtained through a review board of those skilled in the art of programming. One step further, the review board could contain those in the programming community with a vested interest in protecting their trade from useless and baseless obstacles impeding innovation.

In 1899, U.S. Patent Office Commissioner Charles Duell proposed closing down the office because "everything that can be invented has been invented."[128] Charles Duell's statement will never come to fruition because technology is constantly changing and new areas of technology will continue to develop. Like software, all these new technologies will face hurdles in the patent world. The answer is not to

---

[121] PAT CHOATE, HOT PROPERTY: THE STEALING OF IDEAS IN AN AGE OF GLOBALIZATION 27 (Alfred A. Knopf ed., Borzoi Book 2005) (". . . the basic concept was simple: patents and copyrights encouraged inventors and authors to produce more new and useful creations. These innovations could help the U.S. progress. And as the details of these creations became public, the general knowledge of the nation would be expanded.").

[122] *Id.*

[123] The beginning of serious analysis of software patents by the courts began in 1972 with Gottschalk v. Benson, 409 U.S. 63, 72 (1972). From there, important decisions were decided in 1978, 1981, 1992, 1998, and 1999. Looking at software patent case law, it can be said that software patents are still relatively new in terms of interpretation and understanding by the courts.

[124] *See* Gottschalk v. Benson, 409 U.S. 63 (1972); Parker v. Flook, 437 U.S. 584 (1978); *In re* Freeman, 573 F.2d 1237 (CCPA 1978); Diamond v. Diehr, 450 U.S. 175 (1981); *In re* Meyer, 688 F.2d 789 (CCPA 1982); Arrhythmia Research Technology, Inc. v. Corazonix Corp., 958 F.2d 1053 (Fed. Cir. 1992); *In re* Lowry, 32 F.3d 1579 (Fed. Cir. 1994); *In re* Schrader, 22 F.3d 290 (Fed. Cir. 1994); State St. Bank & Trust Company v. Signature Financial Group, 149 F.3d 1368 (Fed. Cir. 1998); WMS Gaming, Inc. v. Int'l Game Tech., 184 F.3d 1339 (Fed. Cir. 1999).

[125] *See generally* MPEP 2005.

[126] *See generally* U.S. Patent No. 6,928,433 (filed Jan. 5, 2001).

[127] *See* discussion *supra* Part B.2.

[128] Adam Piore, *So Predictably Unpredictable*, NEWSWEEK, Sept. 16, 2002.

dismantle the patent system, but to adjust it so that it may be used for its true purpose of fostering innovation whether it be now or the future.