No. 08-964

IN THE
Supreme Court of the United States

BERNARD L. BILSKI AND RAND A. WARSAW,
Petitioners,

v.

DAVID J. KAPPOS, UNDER SECRETARY OF COMMERCE FOR INTELLECTUAL PROPERTY AND DIRECTOR OF THE UNITED STATES PATENT AND TRADEMARK OFFICE,
Respondent.

ON WRIT OF CERTIORARI TO THE UNITED STATES COURT OF APPEALS FOR THE FEDERAL CIRCUIT

BRIEF FOR FREE SOFTWARE FOUNDATION AS *AMICUS CURIAE* IN SUPPORT OF RESPONDENT

Jerry Cohen, Esq.
Counsel of Record
Burns & Levinson LLP
125 Summer Street
Boston, Massachusetts 02110
(617) 345-3000

Attorney for Amicus Curiae Free Software Foundation

BATEMAN & SLADE, INC. BOSTON, MASSACHUSETTS

TABLE OF CONTENTS

TABLE OF AUTHORITIES

CASES:

STATUTES:

MISCELLANEOUS:

INTEREST OF *AMICUS CURIAE*[1]

The Free Software Foundation ("Foundation"), founded in October 1985, pioneered a movement of information exchange that initially had significant impact on software development practices, but later also inspired changes in broader fields of great public benefit. In addition to having been a major developer in the field of software, it provides via its legal, technical, and administrative infrastructure an umbrella for further software development by other programmers around the world. As an important part of this worldwide free software movement, the Foundation has been both an inspiration for, and a close observer of, related social movements and organizations promoting legal sharing of innovative knowledge and cultural works in a variety of fields. The Foundation's mission is focused on free software,[2] which gives all users the freedoms to study, copy, modify, and redistribute their own changed versions.

The Foundation supports free software development directly in three main ways. First, through its GNU Project, launched in 1983 by the Foundation's creator to provide a fully free and functional operating system, it has provided necessary services like

---

[1] Pursuant to Supreme Court Rule 37, no counsel for a party authored this brief in whole or in part, and no counsel or party made a monetary contribution intended to fund the preparation or submission of this brief. No person other than the *amicus curiae*, its members, or its counsel made a monetary contribution to its preparation or submission. Counsel of record for all parties have filed with the Court letters of consent to the filing of briefs by *amici curiae*.

[2] See the Foundation's Free Software Definition, *available at* http://www.gnu.org/philosophy/free-sw.html.

project hosting, development servers, network bandwidth, legal support, and funding for programmers. Second, it provides a set of copyright licenses which can be used by any author as the terms under which to distribute his or her own software and documentation. Rather than forbid users from modifying and redistributing the software, these public copyright licenses encourage sharing and modification, providing a way for authors to commit their work to a commons and users to share and improve those works without fear. The Foundation's primary license is the GNU General Public License (GNU GPL). In addition to its own licenses, the FSF keeps an up-to-date list of licenses published by other companies and organizations which meet the Free Software Definition.[3] Third, it engages in public education and advocacy work to raise awareness about free software to create more opportunities for its use and development, and to ensure that the freedoms outlined in the Free Software Definition are respected.

Over the last 26 years, the GNU Project has successfully developed the utilities and applications that, together with the kernel Linux (also distributed under the GNU GPL) written by Linus Torvalds and contributors around the globe, form a complete free software operating system used by a growing number of individuals, governments, and companies in place of Microsoft's Windows or Apple's OS X. This operating system, though often erroneously referred to simply by the name of its kernel Linux, is called GNU/Linux. Though the

---

[3] See the Foundation's Various Licenses And Comments About Them, *available at* http://www.gnu.org/philosophy/license-list.html.

GNU/Linux operating system is now fully functional, the GNU Project remains active, continuing to refine its software and develop new applications.

One of the GNU Project's most fundamental tools is the GNU Compiler Collection (GCC), which has become the standard for programmers working on Unix-like systems around the world.[4] GCC is used heavily in both the private and public sectors. In 1992 the US Air Force awarded New York University (NYU) a contract to build a compiler for the Ada programming language, which is used extensively in systems deployed by the Department of Defense. The contract specified that the compiler was to be released under the GNU GPL, and the copyrights assigned to the Free Software Foundation.[5] This compiler, GNAT, is an extension of the GNU Compiler Collection (GCC), and is commonly used to compile avionic systems, including the Lockheed Boeing 787 Dreamliner, C130 Hercules aircraft and Raytheon Ship Self-Defense System (SSDS).

Though itself a nonprofit, the Foundation's publicly available licenses are known worldwide and have been adopted (as proposed or with variations) and implemented by hundreds of thousands of people outside the GNU Project who are also creating,

---

[4] See Vinod Valloppillil, *Open Source Software, a (New?) Development Strategy*, Aug. 11, 1998 (hereinafter "VALLOPPILLIL"), *available at* http://iowa.gotthefacts.org/ 011607/6000/ PX06501.pdf ("GCC is the most widely used compiler in academia & the OSS world. In addition to the compiler a fairly standardized set of intermediate libraries are available as a superset to the ANSI C libraries.").

[5] For further information about this arrangement, see generally http://www.adacore.com/home/company/customers/ featured-projects/ & http://cs.nyu.edu/cs/review95/schonberg.ps.

adopting and modifying software for both commercial and noncommercial purposes. The Foundation's software licenses are used by 65% of the more than 200,000 coding projects listed by Black Duck.[6] The aggregate effect of their work, done with reliance on a model of freedom, has promoted science and the useful arts, the objects of copyright and patent systems set forth in U.S. CONST. art. 1, § 8, cl. 8; has limited monopoly and restraint of trade, such limitation being an important object of federal and state statutes and common law; and has promoted freedom of expression, an object of federal and state constitutions. The extent of Foundation President Richard Stallman's contribution to free software and its social movement has been recognized by many institutions, including a Fellowship awarded to him by the MacArthur Foundation in 1990.[7]

The impact of the free software movement extends far beyond people who directly identify themselves as its members; it has influenced many other software and non-software social movements. In 1998, a group of people calling themselves "open source" split off from the movement in disagreement, in order to avoid discussion of the ethical and the social ramifications of free software in favor of values that would appeal directly to businesses, i.e., efficiency, security and cost-savings. While the Foundation is firmly opposed to the active refusal of open source to focus on computer user freedom, the two groups often collaborate on the actual writing of software code, and because open source proponents still use

---

[6] See http://www.blackducksoftware.com/oss/licenses/#adoption.

[7] See http://www.macfound.org/site/c.lkLXJ8MQKrH/b.1142703/k.787E/Fellows_List__August_1990.htm).

free software copyright licenses, it is common to talk about them together as Free, Libre and Open Source Software, or FLOSS, for short.

Bruce Perens, co-founder of the Open Source Initiative, has explained the connection between Richard Stallman and the Foundation's notion of free software, and open source: "My intent has always been for Open Source to simply be another way of talking about Free Software, tailored to the ears of business people, and that it would eventually lead them to a greater appreciation of Richard Stallman's arguments."[8] FLOSS writings, distributed under the terms of the GNU GPL and inspired in substantial part by the Foundation's work, have become staples of data handling and other information systems, telecommunications, video and sound files creation and control, scientific research, bioinformatics, health care, enterprise management and – of equal importance – have been useful in the study of artificial intelligence as a science apart from industrial applications. Although it does not achieve the public visibility or recognition of proprietary software systems like those developed by Microsoft and Apple, a closer look shows that many, if not most, uses of computers today in both the public and private sectors involve FLOSS.

Free software, specifically software distributed under the GNU GPL, is also used heavily by the government of the United States. A 2003 report commissioned by the Department of Defense ("DOD"), entitled "Use of Free and Open Source Software (FOSS) in the U.S. Department of Defense" ("DOD Report") credits Richard Stallman and the

---

[8] See http://perens.com/works/articles/State8Feb2008/.

Free Software Foundation directly. The DOD Report shows that roughly 50% of all of the FLOSS used at the DOD is licensed under the GNU GPL, and evaluates what a possible world within the DOD without this software would look like, stating:

> The main conclusion of the analysis was that FOSS software plays a more critical role in the DoD than has generally been recognized. FOSS applications are most important in four broad areas: Infrastructure Support, Software Development, Security, and Research. One unexpected result was the degree to which Security depends on FOSS. Banning FOSS would remove certain types of infrastructure components (e.g., OpenBSD) that currently help support network security. It would also limit DoD access to--and overall expertise in--the use of powerful FOSS analysis and detection applications that hostile groups could use to help stage cyberattacks. Finally, it would remove the demonstrated ability of FOSS applications to be updated rapidly in response to new types of cyberattack. Taken together, these factors imply that banning FOSS would have immediate, broad, and strongly negative impacts on the ability of many sensitive and security-focused DoD groups to defend against cyberattacks.[9]

---

[9] For a publicly available copy, see http://www.terry bollinger.com/index.html#dodfoss.

In addition, NASA has also recognized the importance of free software by creating its own repository of freely licensed works.[10] While the military and other divisions of government may not have to worry directly about their own use of patents, the DOD Report shows that their operation depends on the continued development worldwide of the kind of free software they use today – the kind of free software facilitated by the Foundation.

Software licensed under the GNU GPL is also critical to the private sector. A world without the Foundation and FLOSS in the private sector would look vastly different. Free software has even been incorporated "under the hood" into a number of consumer products. While these products impose unethical restrictions on users and do not respect free software values, they are nonetheless built using its software. Free software also is crucial to the business of many modern technology companies, including IBM and Novell. These companies may claim patents are necessary for their software businesses, but in many ways they actually rely on and profit from software produced by those with no interest in patents.

Even Microsoft's Bill Gates has criticized the impact of patents on software development. In a memorandum sent to Microsoft's upper management on May 16, 1991, Mr. Gates cited an article written by Richard Stallman for the League of Programming Freedom:

---

[10] See Matt Asay, "*NASA takes open source into space*," CNET NEWS, July 22, 2009, *available at* http://news.cnet.com/8301-13505_3-10292950-16.html; see also Nasa Ames Open Source Software, http://opensource.arc.nasa. gov/.

> Patents: If people had understood how patents would be granted when most of today's ideas were invented, and had taken out patents, the industry would be at a complete standstill today. I feel certain that some large company will patent some obvious thing related to interface, object orientation, algorithm, application extension or other crucial technique. If we assume this company has no need of any of our patents then they have a 17-year right to take as much of our profits as they want. The solution to this is patent exchanges with large companies and patenting as much as we can. Amazingly we haven't done any patent exchanges that I am aware of. Amazingly we haven't found a way to use our licensing position to avoid having our own customers cause patent problems for us. I know these aren't simple problems but they deserve more effort by both Legal and other groups. For example we need to do a patent exchange with HP as part of our new relationship. In many application categories straightforward thinking ahead allows you to come up with patentable ideas. A recent paper from the League for Programming Freedom (available from the Legal department) explains some problems with the way patents are applied to software.[11]

---

[11] See http://antitrust.slated.org/www.iowaconsumercase.org/011607/0000/PX00738.pdf.

Specific free software programs aimed at individual computer users' desktops have also seen wide success recently. The Mozilla Firefox browser alone, available as free software under a number of licenses including the GNU GPL, has reached a market share of over 20% in a field that was once so dominated by Microsoft's Internet Explorer that the company was prosecuted over antitrust concerns.[12] In addition to the millions of users using free software browsers, much of the Internet they are connecting to itself is built on free software. Though it is sometimes deployed with proprietary extensions, the majority of Web sites are based on the free software Apache Web server.[13]

The Foundation has been in a unique position to contribute to and witness the development that has happened as a result of free software over the course of the last 26 years, but also to witness the development that did not and does not happen whenever the specter of patent infringement hangs over the heads of people with insufficient resources to defend against such threats. It has seen the effort undertaken by free software developers to defend themselves against software patents -- both as a pre-

---

[12] See Emil Protalinski, "*Safari, Chrome, Firefox steal share from IE, Opera in January*," ARS TECHNICA, Feb. 6, 2009, *available at* http://arstechnica.com/microsoft/news/2009/02/january-2009.ars; Aza Dotzler "*Firefox: 270 million*,", May 4, 2009, *available at* http://weblogs.mozillazine.org/asa/archives/ 2009/05/firefox_at_270.html.

[13] See Erez Reuveni, "*Authorship in the Age of the Conducer*," 54 J.COPYRIGHT SOC'Y U.S.A. 285, 293 (Jan. 2007) ("Today, roughly 70% of Web server software [*sic*] runs on Apache Web server, free software produced under the open-source model.").

emptive measure to avoid legal disputes, and in response to specific threats from patent holders.

In some cases, the free software community has simply done without certain software or functionality because of threats from patents. Fedora, a community distribution of GNU/Linux sponsored by Red Hat, cannot include software to play MP3s or unencrypted DVDs because of patent concerns.[14] For similar reasons, free software support for the various video codecs used by multimedia on the Web today remains hit-or-miss.

As part of its publication of a new version of the GNU GPL, the Foundation ran an international community comment process, soliciting feedback from companies and individuals about how the license needed to be improved. This process, which ran for over a year and accumulated over 2,500 comments,[15] showed that many free software developers were deterred by the threat of patents and sought protection. For much of its history, the Foundation has grappled with the very real possibility of patent suits against users of free software. Users of free software have been trapped by unforeseen patent claims and forced into costly settlements and cross-licensing agreements.[16] While the respective parties involved in these suits may have reached settlements, the larger questions for all users and de-

---

[14] See http://fedoraproject.org/wiki/ForbiddenItems.

[15] See http://www.fsf.org/news/gplv3_launched.

[16] See, e.g., Ina Fried, "*Microsoft, TomTom settle patent dispute*," CNET NEWS, Mar. 30, 2009, *available at* http://news.cnet.com/8301-13860_3-10206988-56.html.

velopers of free software involving the threat of a patent suit are left unsettled.

The Foundation has been at the center of an emerging and now overwhelming new economic and cultural model of sharing creativity and a community adopting that model. The Foundation has distinct observations of its own, and of its larger community, to offer the Court as to the interaction of this activity with copyright and patent systems. As discussed below, the patent system has not met the constitutional purpose of advancing useful arts in the case of software creation, but rather has been an obstacle to advancement. The Foundation can attest to the fact that the size and shape of this obstacle cannot be understood merely by examining actual instances of patent litigation; the mere threat of such litigation against parties without the ability to cross-license or afford sufficient representation has a substantially documented[17] chilling effect on innovative free software development, and by extension, on other socially beneficial movements inspired by free software.

The Foundation also has published the GNU Free Documentation License (GNU FDL).[18] The Foundation originally created the GNU FDL to provide a basis for documentation for the GNU Project to be shared and adapted the same way that the GNU Project's software is. However, it can readily be applied to any written reference work, and other projects have used this license as the basis for sharing information and collaborating. Noteworthy among

---

[17] See http://www.gnu.org/patent-examp/patent-examples.html.

[18] See http://www.gnu.org/licenses/fdl.html.

these is Wikipedia, an online encyclopedia which generally can be edited by anyone who visits its web site. Creators of Wikipedia have said that they were inspired by the Foundation's ideas.[19] Wikipedia used the GNU Free Documentation License exclusively as the license for all of its articles until June 2009,[20] and at that time it covered more than 340,000 texts.[21] The vast majority of articles on Wikipedia today are now dual-licensed, with the GNU FDL still available as an option for its users.

Those same ideas have also given birth to other movements to create works which can be shared and changed, even when they did not use the licenses published by the Foundation. Creative Commons publishes its own licenses that allow people to distribute and modify the works they cover, and are meant to be applied to a wide variety of works; current estimates suggest that these licenses have been used by 250 million works at minimum worldwide.[22] Most of these licenses grant permissions and have associated conditions similar to those in the Foundation's licenses. Executives at Creative Commons have frequently cited the Foundation as providing the basis for their own efforts.[23]

---

[19] See http://icommons.org/articles/commoner-profile-10-questions-for-jimmy-wales; see also http://mixergy.com/wikipedias-founder-jimmy-wales/.

[20] See http://lists.wikimedia.org/pipermail/foundation-l/2009-June/052616.html.

[21] See http://en.wikipedia.org/wiki/User:R._fiend/How_many_articles_does_Wikipedia_really_have%3F.

[22] See http://wiki.creativecommons.org/Metrics.

[23] See http://creativecommons.org/weblog/entry/13568.

* * *

While supporting the result sought by Respondent (i.e., affirming the judgment of the Federal Circuit Court of Appeals of unpatentability of the claimed subject matter of the Bilski et al. patent application), the Foundation hopes this brief provides the Court a broader context and approach to software patenting limitation, which goes beyond the two part test derived by the Court of Appeals for the Federal Circuit from this Court's decision in *Diamond v. Diehr*, 450 U.S. 175 (1981). The Foundation is aligned in part, but only in part, with the separate opinion of Judge Mayer.[24]

Notwithstanding the contrary assertion at sec. I.C.3 (pages 36-44) of Respondent's brief, the Foundation submits respectfully that this case is an appropriate one to address patenting eligibility of computer software. Several amici aligned with both parties of this case so regard it and indeed the court of appeals eschewed a categorical exclusion of business methods and, instead, applied the machine-transformation test to a claimed process which was implicitly software-based though not reciting software directly in its broadest independent claim.

---

[24] See *In re Bilski*, 545 F.3d 943, 998-1011 (Fed. Cir. 2008).

SUMMARY OF ARGUMENT

The Foundation asserts that software patents hinder the progress of software development and distribution, are unjust and cause deleterious socioeconomic effects upon the advancement of technology in the United States for society's benefit. The Foundation further takes the position that computer software patents do not "promote the Progress of Science and useful Arts," and therefore are not constitutionally justified. Clarification or expansion of the "particular machine or transformation" test is appropriate to prevent the USPTO from granting software patents and to reduce uncertainty for those developing software.

An information processing algorithm with no physical manifestation of any sort is beyond the bounds of patentability. This Court has repeatedly ruled that an information processing algorithm with "insignificant post-solution activity" appended should still not be patentable. Notably, claims for an information processing algorithm loaded in a standard way onto a standard computer were repeatedly ruled to be invalid. The Court in *Diamond v. Diehr*, 450 U.S. 175, 182 (1981), set forth a clear criterion that information processing algorithms dressed in physical terminology are not eligible to be patents, although other, more involved types of software-on-a-machine could be patentable when "considered as a whole."

Unfortunately, subsequent rulings from the Federal Circuit have applied only the second half of this criterion, thereby elevating the exception to the rule and inventing a doctrine whereby *any* claim which includes software must be "taken as a whole" when

examined for patent-eligibility. If this line of Federal Circuit cases continues, and the holistic second half of the *Diehr* holding is treated as the entire ruling, patent examiners would be barred from inquiring whether claims such as those present in the Bilski patent are merely information processing claims recited in a manner that circumvents limitations on patent eligibility. Instead, examiners would be obligated to accept any artfully styled claim as patent-eligible.

The Foundation suggests that the Court could conform to its earlier precedent whereby claims for information processing with trivial physical dressing are excluded from patent-eligibility, either via a standard Section 103 analysis or via a comparable dissection under Section 101. However, either method requires striking down the artificial doctrine that inventions with software – and only inventions with software – must be "considered as a whole" in relation to Section 101.

There is wisdom in this Court's repeated attempts to ensure that information processing algorithms remain outside the scope of patent law, even in manifestations where "insignificant post-solution activity" is appended. Such a limitation on patent-eligible subject matter should be respected and enforced, and for good reason: allowing such patents has had perverse economic effects. First, litigation regarding software is increasingly targeted not at producers in the "information processing sector," but rather at parties in the general economy who are independently reinventing software in the course of business. Second, the increased risk of liability brought about by the expansion of patent law to include software and business methods has sparked

debate about the validity of the patent system at large.

ARGUMENT[25]

I. THIS COURT'S JURISPRUDENCE CONCERNING SOFTWARE PATENTS HAS APPROPRIATELY EXPRESSED A TEMPERED APPROACH, BUT SUBSEQUENT RULINGS BY LOWER COURTS THREATEN TO UPSET THE BALANCE STRUCK BY THIS COURT OVER THIRTY YEARS AGO.

A. THIS COURT HAS RULED THAT INFORMATION PROCESSING ALGORITHMS WITH "INSIGNIFICANT POST-SOLUTION ACTIVITY" ARE BARRED FROM PATENT-ELIGIBILITY.

There is little controversy that information processing algorithms in their pure, ethereal forms, with no physical component or manifestation of any sort, are excluded from patentability.[26] However, what is

---

[25] The Foundation acknowledges, with gratitude, the substantial work of research, compilation, analysis and drafting contributed to this brief by Ciarán O'Riordan and End Software Patents (ESP), a not-for-profit organization. See http://endsoftpatents.org/. Mr. O'Riordan is the Executive Director of ESP.

[26] Novel mathematics, for example, is outside the scope of patent-eligibility. "Whether the [mathematical] algorithm was in fact known or unknown at the time of the claimed invention ... it is treated as though it were a familiar part of the prior art." *Parker v Flook*, 437 U.S. 584, 591-92 (1978) (internal citation omitted).

under debate is how much of a physical manifestation an information processing algorithm must have before it is patentable.

In a trio of opinions issued over the span of nine years, this Court clearly rejected the patentability of an information processing algorithm with "insignificant post-solution activity" appended. First, in *Gottschalk v. Benson*, 409 U.S. 63 (1972), the Court quoted approvingly the 1966 President's Commission on the Patent System:

> Direct attempts to patent programs have been rejected on the ground of nonstatutory subject matter. Indirect attempts to obtain patents and avoid the rejection, by drafting claims as a process, or a machine or components thereof programmed in a given manner, rather than as a program itself, have confused the issue further and should not be permitted.[27]

Second, in *Parker v. Flook*, 437 U.S. 584 (1978), this Court made a more general statement, reiterating the position that loading an algorithm onto a standard computer is merely an attempt to circumvent recognized limitations: "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. A competent draftsman could attach some

---

[27] *Id.* at 72 n.5 (*quoting* "To Promote the Progress of ... Useful Arts," Report of the President's Commission on the Patent System, at 13 (1966)).

form of post-solution activity to almost any mathematical formula." *Id.* at 590.

Third, in *Diamond v. Diehr*, 450 U.S. 175 (1981), the Court directly reiterated its two previous holdings, while also acknowledging that bona fide, patent-eligible inventions may *include* a software component:

> A mathematical formula as such is not accorded the protection of our patent laws, *Gottschalk v. Benson*, 409 U.S. 63 (1972), and this principle cannot be circumvented by attempting to limit the use of the formula to a particular technological environment. *Parker v. Flook*, 437 U.S. 584 (1978). Similarly, insignificant post-solution activity will not transform an unpatentable principle into a patentable process. *Ibid*. To hold otherwise would allow a competent draftsman to evade the recognized limitations on the type of subject matter eligible for patent protection. On the other hand, when a claim containing a mathematical formula implements or applies that formula in a structure or 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 [Section] 101.

Id. at 191-92.

The phrase "considered as a whole," in relation to subject matter eligibility, does not appear in Sections 100 or 101 of Title 35 of the United States Code (although it does appear in section 103 concerning obviousness determinations); rather, it originates in the above-quoted statement from *Diehr*, 450 U.S. at 192, and therefore its meaning should be evaluated in the context of the balance in which it was presented. In particular, the *Diehr* ruling demarked a dichotomy between designs that have only "insignificant post-solution activity," on the one hand, and processes which, "considered as a whole," fall under the scope of patent-eligible subject matter, on the other. The full statement directs that an inquiry first be made into whether a claimed invention is information processing with "insignificant post-solution activity." Once undertaken, that inquiry may lead to the conclusion that the invention involves significantly more and should be patent-eligible.[28]

[28] This inquiry is carried out in the text of the *Diehr* opinion. For example, the Court determined that the equation-plus-alarm claim that was at issue in *Flook* was merely an attempt to patent an equation, whereas Diehr's machine went well beyond the underlying software. See *Diehr*, 450 U.S. at 186-87.

B. THE FEDERAL CIRCUIT'S BROAD APPLICATION OF THE "CONSIDERED AS A WHOLE" DOCTRINE RELIES UPON A MISLEADING INTERPRETATION OF SUPREME COURT PRECEDENT, AND THIS HOLISTIC APPROACH HAS RESULTED IN THE EXCEPTION SWALLOWING THE RULE.

The *Diehr* analysis is correct, but as applied by the Federal Circuit, has improperly evolved into a one-sided doctrine whereby all claims, including software claims, are preliminarily "taken as a whole" when assessing their Section 101 eligibility for patenting. Rulings by the Court of Appeals for the Federal Circuit in *In re Alappat*, 33 F.3d 1526, 1543 (Fed. Cir. 1994), *State Street Bank & Trust Co. v. Signature Financial Group, Inc.*, 149 F.3d 1368, 1374 n.6 (Fed. Cir. 1999), and *AT&T Corp. v. Excel Communications Inc.*, 172 F.3d 1352, 1356-59 (Fed. Cir. 1999), are based only on the second half of the *Diehr* standard. That is, these rulings state that a claim must be considered "as a whole," while wholly disregarding the counterbalancing statement that some inventions are merely unpatentable formulae with "insignificant post-solution activity."

The result is an extreme doctrine that has damaged patent law, and allowed the patentability of elements that would not be patentable under any of the above Supreme Court rulings. Under this distorted automatic approach to the "considered as a whole" doctrine, the Bilski claim (intended to be read as information processing with a not-novel, obvious physical step appended) and the typical Beauregard claim (for a standard computer memory device upon

which is loaded a new work of software) are nevertheless deemed patentable, even though they could (and should) easily be dissected into an information processing step and "insignificant post-solution activity." When viewed through the Federal Circuit's clouded lens, even the claim in *Flook* would be read as a patent-eligible alarm, instead of the dressed-up, unpatentable equation that the *Diehr* ruling took it to be.

Such a result is wrong, and the Court in this case has the opportunity to prevent the "considered as a whole" language of *Diehr* from being considered as *the* whole of that ruling, which it is not. If left unchecked, an incorrect legal standard would prohibit examiners from inquiring whether claims such as those present in the Bilski patent are merely information processing claims recited in a manner that circumvents limitations on patent-eligibility, as opposed to bona fide patent-eligible inventions. Instead, examiners would have to accept any correctly worded claim as patent-eligible, thereby elevating form over substance.

C. THE SECTION 103 INQUIRY COULD ACHIEVE THE SAME EFFECT, BUT IS ALSO BLOCKED BY THE "AS A WHOLE" DOCTRINE.

The preceding discussion of the Section 101 inquiry advises a dissection of a patent application that is in many ways comparable to the inquiry of non-obviousness under 35 U.S.C. § 103. When evaluating the non-obviousness of a patent application, the following process is prescribed: "Under 103, the scope and content of the prior art are to be de-

termined; differences between the prior art and the claims at issue are to be ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background, the obviousness or nonobviousness of the subject matter [as a whole] is determined." *Graham v. John Deere Co. of Kansas City*, 383 U. S. 1, 17 (1965).

As the *Diehr* ruling acknowledges, the combination of information processing and machinery is sometimes greater than the sum of its parts; the same could be true of combinations in a Section 103 context. But in a parallel manner, this Court's ruling in *KSR v. Teleflex*, 550 U.S. 398 (2007), points out that if a combination of elements in the prior art is "obvious to try," the combination does not pass the conditions of Section 103. Given a set of rules for information processing – such as Bilski's hedging scheme or any computationally-intensive number-crunching scheme – it is blatantly obvious to try loading the algorithm onto a standard computer. In fact, in *Northern Telecom v. Datapoint*, 908 F.2d 931, 940-41 (1990), the Federal Circuit ruled that loading an algorithm onto a computer is a "mere clerical function." That is, a software-plus-computer claim consists of one piece of prior art (an information processing algorithm), a second piece of prior art (a standard, unmodified computer), and the obvious-to-try combination of one with the other.

II. FOR MANY SOFTWARE DEVELOPERS, THE PATENT SYSTEM IS UNJUST.

Development in a domain where ideas are patentable carries risks. In certain domains, such as the manufacturing of vehicles and pharmaceuticals,

product developers are almost always large companies. Because they have the resources for manufacturing, and thereafter attaining the necessary government approval for their product, it is reasonable to assume these companies also have the resources to perform patent searches, get legal opinions and, if necessary, mount a defense when accused of patent infringement. Manufacturing vehicles and pharmaceuticals is inherently expensive, with or without the patent system, so manufacturers can be assumed to be prepared to expend money. The expense of the patent system does not change who can participate, but rather, simply increases the cost.

None of these factors apply for software. Software is developed not just by large companies, but also by small companies, project communities, students, and individuals. Participation in software development has no fundamental need to consume any resource (other than, perhaps, one's time spent at a computer). Thus, it cannot be assumed that developers active in the field of software have the resources to perform patent searches, nor that they expect to incur expenses for their activity. This leaves many developers in a situation where they cannot afford the legal resources necessary to minimize their risk of infringement, and if accused of violating a patent, they cannot afford to defend themselves. The system is disproportionately expensive, by orders of magnitude.

This inability to participate on an even basis amplifies the problem, but there is also a deeper problem: losing control of one's computing in his or her daily life. Because individuals can write software, they can help themselves and solve their own problems. Given that software development includes

common activities such as making a webpage, the freedom to use a computer as one sees fit for his or her daily life is a fundamental form of expression, just as using a pen and paper is.

III. COMPUTER SOFTWARE PATENTS CAN COMPROMISE COMPATIBILITY AND DATA FORMAT SPECIFICATIONS.

In software, access to data format specifications holds a role of importance unparalleled in other fields. Examples of data that come in specific formats include email, images, and word processing documents. In the context of writing an email reader, a word processor, or an image viewer, being blocked from reading, modifying, or writing in the required data format is equivalent to being banned from writing a functional program for that task.

This gives too much power to the holder of a patent on a widely used format. In terms of the proper functioning of the software industry, this power is harmful to competition. In terms of individuals, it translates to banning them from writing useful software for themselves and for others. In some fields of development, the barriers made by patents may spur useful innovation when developers search for a different way to accomplish a patented task. With data formats, this type of innovation is impossible to encourage because by reading or writing the data differently, the software would be failing its compatibility objective.

Software patents can also hold back freely implemented standards such as those that structure the World Wide Web. HTML version 4 has been the

standard for web pages since 1997. Efforts are ongoing to produce its successor, HTML5. This effort is making progress, except concerning which video format to recommend. Due to patents, the popular "MPEG H.264" video format cannot be recommended because it cannot be implemented without permission. The other option was the patent-free Ogg Theora format, but when improving the quality of this format was discussed, Chris DiBona of Google said, "Here's the challenge: Can [T]heora move forward without infringing on the other video compression patents?"[29]

Yet another way in which compatibility is essential is in the look and control of an application. If one develops an innovative word processor which doesn't resemble any existing word processor, it will be difficult for people to use. Software users have certain expectations. Using new design paradigms can be useful, but it adds initial inconvenience for the users, so this change should only be made if the benefits clearly outweigh the annoyance to users. Having a strange interface because of patent problems is not beneficial for computer users.

---

[29] See http://www.mail-archive.com/whatwg@lists.whatwg.org/msg15476.html.

IV. THE CONTINUED BROAD ALLOWANCE OF SOFTWARE CLAIMS PLACES UNJUST RESTRICTIONS ON EVERYONE AND HAS RESULTED IN PERVERSE ECONOMIC EFFECTS AND AN EPIDEMIC OF LITIGATION.

Litigation regarding software is increasingly targeted not at producers in the information processing sector, but at persons in the general economy who are independently reinventing course-of-business software.

One large software project has been the subject of two analyses. First, the Linux kernel – which is the kernel of the widely used GNU/Linux operating system – was examined by patent attorney Dan Ravicher, who announced on August 2, 2004, that he had found no court-validated patents to be infringed by the Linux kernel, although 283 *issued* patents existed which could potentially be used to support patent claims against it.[30] Thereafter, Microsoft in the year 2007 began claiming that the Linux kernel violates 235 of its patents - although the patents have never been specified.[31]

Yet the kernel Linux is just one component of the GNU/Linux operating system. The kernel Linux's human-written source code is publicly available, and contains approximately 4,000,000 (four million) lines

---

[30] See http://www.osriskmanagement.com/press_releases/press_release_080204.pdf.

[31] See Roger Parloff, "*Microsoft Takes On The Free World*," FORTUNE (May 14, 2007), *available at* http://money.cnn.com/ magazines/fortune/fortune_archive/2007/05/28/100033867/.

of source code. Accounting for the two suggestions discussed in the preceding paragraph about the number of patents possibly violated, one ends up with one instance of patent infringement for every 14,275 to 17,191 lines of code. Taking the one calculation one step further, given the fact that complete GNU/Linux operating systems, often distributed with sets of applications, can contain software with more than 225 million lines of source code, we arrive at the possibility of 13,160 or 15,848 patent infringements per complete distribution.[32] The resulting legal entanglements would be simply absurd. This massive uncertainty is what this Court can end by clearly excluding software ideas on computers from patentability.

A second example is the previously mentioned MPEG H.264 video format. This one format is covered by patents from numerous companies and institutions, including Columbia University, Electronics and Telecommunications Research Institute of Korea (ETRI), France Télécom, Fujitsu, LG Electronics, Matsushita, Mitsubishi, Microsoft, Motorola, Nokia, Philips, Robert Bosch GmbH, Samsung, Sharp, Sony, Toshiba, and Victor Company of Japan (JVC).[33]

Not only do these two examples demonstrate the absurd situation created by granting patents on something as complex and abstract as software,[34] but

---

[32] See http://www.upgrade-cepis.org/issues/2005/3/up6-3Amor.pdf.

[33] See http://www.mpegla.com/news/n_03-11-17_avc.html.

[34] Because software is a set of instructions to achieve a goal, there is often very little difference between the description of the goal itself and a description of one developer's method of

the latter instance is also an example of an important standard the implementation of which depends on permission from patent holders. In other industries, practitioners may know which companies in their field are applying for patents which might be pertinent. In software, the number of practitioners is too great for it to be possible to know which companies or groups one should monitor for applications or pertinent patents. As researcher Ben Klemens, who estimates the cost of software patents to the United States economy to be $11.26 billion per year, points out, "Any company with a web site could be liable for software patent infringement."[35] In fact, the risks of patents on software are not limited to companies that consider themselves software companies, and Klemens notes that among the companies facing litigation for violating one particular software patent are CDW Corp., Motorola, the Green Bay Packers, OfficeMax, Caterpillar, Kraft Foods, ADT Security Services, AutoNation, Florida Crystals Corp., HearUSA, Tire Kingdom, and Boca Raton Resort and Club.[36]

The increased risk of liability brought about by the expansion of patent law to include software and business methods has sparked debate about the validity of the patent system at large. Thus, there is wisdom in the Supreme Court's repeated attempts to ensure that information processing algorithms re-

---

achieving the goal. This abstract nature of software leads to overly-broad patents which cover many – or all – ways of solving a problem, rather than just covering the way the patent applicant chose to solve the problem.

[35] See http://endsoftwarepatents.org/28-february-2008:esp-releases-report-on-the-state-of-softpatents.

[36] Ibid.

main outside the scope of patent law, even in manifestations where "insignificant post-solution activity" is appended. The limitations on patent-eligible subject matter should be respected and enforced.

V. INNOVATION CAN BE, AND HAS BEEN, ACHIEVED IN THE ABSENCE OF SOFTWARE PATENTS.

It may be surprising to many people that Microsoft Corporation is an example of a company built without patents. In 2005, Grace Murray Hopper Award-winning software developer Dan Bricklin and New York Times journalist Randall Stross researched the patent ownership history of Microsoft. The pair found that in 1987, when Microsoft had only had one patent granted to them, their revenue was $350 million. By 1990, the corporation had five patents, and a revenue of $1.18 billion. By 1995, they still had only 77 patents, their revenue was $5.94 billion.[37] In other words, at the time Microsoft completed its Windows 95 operating system, Microsoft still had only 77 patents. The development of its flagship product was fostered by copyright, not patents. Though copyrights possess their own problems, those problems are more solvable under the current law, and copyrights do not block independent development as patents do.

On February 10, 2009, Microsoft announced that it had been granted its 10,000th patent. Since 1995,

---

[37] See Randall Stross, "*Why Bill Gates Wants 3,000 New Patents*," New York Times (July 31, 2005), available at http://www.nytimes.com/imagepages/2005/07/30/business/yourmoney/20050731_DIGI_GRAPHIC.html.

the company's premier position in the operating systems and word processor markets have not changed. More recently, since at least 2006, Microsoft has been approaching distributors of other operating systems and demanding payments of patent royalties. One could deduce that patents didn't help them compete with others, but did help them entrench the position gained. Also, Microsoft's current long-standing dominance and its recent use of its patents are strikingly similar to the problems that Bill Gates predicted in 1991 about industry standstill and patents being used to take profits of other software companies.[38]

The GNU/Linux operating system is another example of a top class operating system built on copyrights, not patents. GNU/Linux uses copyright much differently than Microsoft. Before 1995, GNU/Linux was developed by individuals, communities, universities, and those who developed software as a side activity to use for other work. The license used by the majority of free software, the GNU General Public License, does not allow distributors to use patents to require royalties[39] – thus excluding, for example, licensing the MPEG H.264 under its current terms.

---

[38] See *supra*, pp. 10-11, n.11.

[39] Section 7 of Version 2 of the GNU GPL states: "If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all."

VI. NOTABLE ANALYSES HAVE NOTED THE RISKS POSED BY SOFTWARE PATENTS.

Experts from around the globe repeatedly have concluded that software patents are not good for progress, innovation, or the economy:

- Bessen & Hunt – MIT: "'Thus the extension of patent protection to software did not generate a relative increase in R&D spending as predicted by the static model; instead, consistent with the dynamic model, R&D spending seems to have remained roughly steady or to have declined."[40]

- Bessen & Maskins: "The very large increase in software patent propensity over time is not adequately explained by changes in R&D investments, employment of computer programmers, or productivity growth. [...] We find evidence that software patents substitute for R&D at the firm level; they are associated with lower R&D intensity."[41]

- U.S. Federal Trade Commission: "The software and Internet industries generally are

---

[40] James Bessen & Eric Maskin, Sequential Innovation, Patents, And Imitation (Jan. 2000). Massachusetts Institute of Technology, Department of Economics Working Paper, *available at* http://www.researchoninnovation.org/patent.pdf.

[41] Bessen, James E. and Hunt, Robert M., An Empirical Look at Software Patents (March 2004). FRB of Philadelphia Working Paper No. 03-17, *available at* http://ssrn.com/abstract=461701.

characterized by five factors: (1) innovation occurs on a cumulative basis; (2) capital costs are low, particularly relative to the pharmaceutical, biotechnology and hardware industries; (3) the rate of technological change is rapid, and product life cycles are short; (4) alternative means of fostering innovation exist, including copyright protection and open source software; and (5) the industries have experienced a regime change in terms of the availability of patent protection. Panelists consistently stated that competition drives innovation in these industries. Innovation is also fostered by some industry participants' use of copyright protection or open source software. Several panelists discounted the value of patent disclosures, because the disclosure of a software product's underlying source code is not required. Many panelists and participants expressed the view that software and Internet patents are impeding innovation. They stated that such patents are impairing follow-on incentives, increasing entry barriers, creating uncertainty that harms incentives to invest in innovation, and producing patent thickets. Panelists discussed how defensive patenting increases the complexity of patent thickets and forces companies to divert resources from R&D into obtaining patents. Commentators noted that patent thickets make it more difficult to commercialize new products and raise uncertainty and investment risks. Some panelists also noted that hold-up has become a problem that can result

[42]

- Lord Justice Jacob, Supreme Court of Judicature: "The patent system is there to provide a research and investment incentive but it has a price. That price (what economists call 'transaction costs') is paid in a host of ways: the costs of patenting, the impediment to competition, the compliance cost of ensuring non-infringement, the cost of uncertainty, litigation costs and so on. There is, so far as we know, no really hard empirical data showing that the liberalisation of what is patentable in the USA has resulted in a greater rate of innovation or investment in the excluded categories. Innovation in computer programs, for instance, proceeded at an immense speed for years before anyone thought of granting patents for them as such. There is evidence, in the shape of the mass of US litigation about the excluded categories, that they have produced much uncertainty. If the encouragement of patenting and of patent litigation as industries in themselves were a purpose of the patent system, then the case for construing the categories narrowly (and indeed for re-

[42] US Federal Trade Commission, "To Promote Innovation: The Proper Balance of Competition and Patent Law and Policy" (October 2003), *available at*- http://www.ftc.gov/os/2003/10/innovationrpt.pdf . This 315-page review of the United States patent system dedicated thirteen pages to "The Software and Internet Industries." The Foundation quotes the conclusion to this section in full.

moving them) is made out. But not otherwise."[43]

- Report for the United Kingdom's Chancellor of the Exchequer: "The software industry in the USA grew exponentially without pure software patents, suggesting they are not necessary to promote innovation. The evidence suggests software patents are used strategically; that is, to prevent competitors from developing in a similar field, rather than to incentivise innovation."[44]

- Deutsche Bank Research: "Chances are that patents on software, common practice in the US and on the brink of being legalised in Europe, in fact stifle innovation. Europe could still alter course."[45]

- Report for The Netherlands Ministry of Economic Affairs: "There are particular threats to the European ICT industry such as the cur-

---

[43] *Aerotel Ltd. v. Telco Holdings Ltd.*, 2006 EWCA Civ 1371 (C.A. 2006, Supreme Court of Judicature, Court of Appeals (Civil Division), on appeal from the High Court of Justice, Chancery Division (Patents Court)), *available at* http://www.patent.gov.uk/2006ewcaciv1371.pdf.

[44] Gowers Review of Intellectual Property (Nov. 2006), *available at* http://www.hm-treasury.gov.uk/d/pbr06_gowers_report_755.pdf. This review, conducted at the request of the United Kingdom's Chancellor of the Exchequer, was published after a year-long "independent review into the UK Intellectual Property Framework."

[45] Deutsche Bank Research, "Current Issues, More Growth In Germany" (June 22, 2004), *available at* http://www.dbresearch.com/PROD/DBR_INTERNET_EN-PROD/PROD0000000000175949.pdf.

> rent discussion on the patent on software. The mild regime of IP protection in the past has led to a very innovative and competitive software industry with low entry barriers. A software patent, which serves to protect inventions of a non-technical nature, could kill the high innovation rate."[46]

CONCLUSION

The decision below should be affirmed, and further clarified or expanded to ensure that no patents are granted for software running on a computer. Beyond that, this Court should recognize how its decisions in *Diehr*, *Parker* and *Gottschalk* have been misapplied in a manner that is no longer acceptable, and acknowledge that patenting software is inconsistent with the Constitution's mandate.

Respectfully submitted,

/s/ JERRY COHEN
JERRY COHEN
Counsel of Record
BURNS & LEVINSON LLP
125 Summer Street
Boston, Massachusetts 02110
(617) 345-3000

Attorney for Amicus Curiae
Free Software Foundation

October 2, 2009

---

[46] PriceWaterhouseCoopers, Rethinking the European ICT Agenda, Report for The Netherlands Ministry of Economic Affairs (Aug. 2004), *available at* http://www.ez.nl/dsresource?objectid=62435&type=PDF.