900 17th Street, N.W.
Suite 1100
Washington, DC 20006
Phone: 202.783.0070
Fax: 202.783.0534
Web: www.ccianet.org
Computer & Communications Industry Association

Enlarged Board of Appeal
European Patent Office
Erhardtstrasse 27
80331 MUNCHEN
Germany

Re: Matter G 3/08, reffered under Article 112(1)(b) to the Enlarged Board of Appeal by the President of the European Patent Office.

The EPO President's referral asks the Enlarged Board of Appeal to clarify the application of the Section 52 exclusion of computer programs as expressed in divergent TBA decisions formulating and applying the required technicity.[1] In this submission, CCIA asks the Enlarged Board of Appeal to address these questions with reference to the goals of the patent system and the concerns inherent in the computer program exclusion.

The Computer & Communications Industry Association (CCIA) is dedicated to open markets, open systems, and open networks. CCIA members participate in the information and communications technology industries, ranging from small entrepreneurial firms to the largest in the business. CCIA members employ nearly one million people and generate annual revenues exceeding $200 billion. CCIA and its members subscribe to principle of "full, fair, and open competition." We believe that the patent system is intended to serve this goal by promoting innovation, and we believe that this requires taking a long-term perspective that transcends particular market and professional interests.

We begin by reviewing the recommendations of the President's Commission on the Patent System as the most complete official analysis prior to the EPC. We note the recent return of U.S. jurisprudence to the principles of the Supreme Court decisions in *Benson*, *Flook*, and *Diehr*. We then examine how research on the practical effects of patents illuminates the early skepticism towards patents on computer programs and how this early intuition is playing out within a greatly expanded and diversified universe of software. We review the problem as a reflection of software's unique economic characteristics and its characteristics as an extreme instance of complexity in products and services. The misfit between software and a unitary system designed for efficient protection of discrete industrial products is mapped onto a framework that shows how the characteristics of software diminish the benefits and increase the costs of patents.

We conclude that focusing on physical transformation as the essence of technicity is the best way of limiting the problems posed by patents on abstract subject matter.

Analysis Prior to the EPC

As the debate over the proposed Directive on Computer-Implemented Inventions made graphically clear, no issue has challenged the patent system so broadly and deeply as the patentability of software. Unfortunately, the *travaux preparatoires* for the European Patent Convention (EPC) offer no analysis of the underlying concerns, but section 52 brackets the exclusion so as to defer precision line-drawing to the wisdom of the EPO and national courts.[2]

The legislators of the EPC were aware that the U.S Patent Office was reluctant to issue patents on computer programs and did not want to disadvantage European companies by subjecting them to the reported problems of software patents.[3]
The report of the President's Commission reflected the concerns of the Patent Office:

> 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.
>
> The Patent Office now cannot examine applications for programs because of the lack of a classification technique and the requisite search files. Even if these were available, reliable searches would not be feasible or economic because of the tremendous volume of prior art being generated. Without this search, the patenting of programs would be tantamount to mere registration and the presumption of validity would be all but nonexistent.
>
> It is noted that the creation of programs has undergone substantial and satisfactory growth in the absence of patent protection and that copyright protection for programs is presently available.[4]

Indeed, the creation of computer programs flourished with virtually no patent protection for another 20 years, the generation of prior art has grown exponentially over the last 45 years, and software remains notoriously difficult to evaluate under the traditional criteria of the patent system. As the Commission concluded in 1966:

> The Commission believes strongly that all inventions should meet the statutory provisions for novelty, utility and unobviousness and that that the above subject matter cannot readily be examined for adherence to these criteria.

And recommended against patents for computer programs regardless of how they were claimed:

> A series of instructions which control or condition the operation of a data processing machine, generally referred to as a "program," shall not be considered patentable regardless of whether the program is claimed as: (a) an article, (b) a process described in terms of the operations performed by a machine pursuant to a program, or (c) one or more machine configurations established by a program.

The Commission's recommendation was cited in *Gottschalk v. Benson*,[5] the seminal 1972 U.S. Supreme Court decision that determined that mathematical algorithms are not patentable. The Court's unanimous opinion in *Benson* viewed the algorithm as too basic and fundamental: "Phenomena of nature, though just discovered, mental processes, and abstract intellectual concepts are not patentable, as they are the *basic tools* of scientific and technological work." 409 U.S. 63, 67 (emphasis supplied). And it was deemed useful in too many contexts: "The end use may vary from the operation of a train to verification of drivers' licenses to researching the law books for precedents." *Ibid*. at 68.

These concerns remain valid today. Patents on abstract subject matter present exceedingly difficult line-drawing problems compounded by the multilevel functional complexity of marketed products and the sheer volume of innovative activity. We now understand that these problems do not just burden patent offices; they can also burden innovators. They can inhibit cumulative innovation, create "patent thickets," devalue individual patents in favor of portfolios, impose high transaction costs, encourage strategic behavior, and threaten large investments in standards and tangible products.

But these symptoms took time to emerge. In complex technologies, both industry and professionals became accustomed to, enamored of, and eventually addicted to volume. For many years, the subject matter question remained framed mainly as struggle between an applicant and a patent office that complained of overwork, and the Court of Appeals for the Federal Circuit found it easy to side with the applicant. Besieged by applicants, confronting an unsympathetic court, and allowed to keep its own fees, the United States Patent and Trademark Office (USPTO) warmed to what would become its new mission: "To help customers get patents."[6] For a time, it was easy to see patents, especially the burgeoning of software patents, as assets in an expanding knowledge-based economy – with patent institutions at its heart. As a 2003 treatise observed:

> [B]road notions of patent eligibility appear to be in the best interest of the patent bar, the PTO, and the Federal Circuit [CAFC]. Workloads increase and regulatory authority expands when new industries become subject to the appropriations authorized by the patent law. Noticeably absent from the private, administrative and judicial structure is a high regard for the public interest.[7]

But eventually, as patent holders found new means of leveraging a ballooning number of exclusionary rights, the liability side of the ledger emerged.

While EPO and European courts have labored to work within Article 52(2) and corresponding national legislation, the Court of Appeals for the Federal Circuit in the U.S. faced no statutory constraints. The Federal Circuit virtually eliminated constraints on subject matter by limiting Supreme Court precedent to mathematics. It finally set up its own test in the controversial *State Street* case, requiring only "a useful, concrete, and tangible result."[8] At the same time, the *State Street* decision swept aside the longstanding judicial exclusion of methods of doing business, in part by placing inordinate emphasis on the word "any" in a patent statute enacted 46 years earlier.[9] The court did not define "concrete" and "tangible," in effect collapsing subject matter limitations into the utility test (the U.S. equivalent of "industrial application") while focusing on the result rather than the invention itself. The court, which already had a strong reputation as a "booster of its specialty,"[10] made it clear that it would be hostile to subject matter challenges, and the "useful, concrete, and tangible result" test was never applied in court to exclude any subject matter.

The effect of *State Street* was to sweep the liberal professions and applied social sciences into the patent system with no inquiry, no analysis, and no input from the tens of millions of professionals affected by this seismic policy change.[11] This led to notorious excesses such as patents on tax strategies, donations, and even patent practice itself.[12] The U.S. Government for a time argued in WIPO patent harmonization discussions that it was "best practice" to open the patent system to all areas of human activity, a proposition that was clearly rejected by the rest of the world.[13]

The *State Street* decision stands as an object lesson in the dangers of semantic jurisprudence and the political economy of patents. Had the U.S. Government entertained a consultation prior to the Federal Circuit's *State Street* decision, it would probably have found a consensus against business method patents, just as the European Commission and the UK Patent Office found a clear consensus that non-technological business methods should not be patentable.[14] By abolishing the infrequently applied but commonly understood exclusion of "methods of doing business," the Federal Circuit created a land rush of applications and an instant constituency that would contest any congressional effort to roll back business method patents.

Return to Supreme Court Precedent in the U.S.

As the hidden liabilities of too many low-quality patents began to emerge, along with critical studies of the patent system by the FTC, the National Academies of Science, and economists Adam Jaffe and Josh Lerner,[15] the U.S. Supreme Court took a renewed interest patent law. There followed a series of high-profile reversals of the Federal Circuit's historically "pro-patent" jurisprudence.[16] Although the Supreme Court had not spoken on patent-eligible subject matter since *Diamond v. Diehr* (1981),[17] it made clear that it intended to do so by accepting certiorari in *Labcorp v. Metabolite*.[18]

Sitting en banc in *In re Bilski* (2008), the Federal Circuit reformulated limits to patentable subject matter for processes (the most abstract of the four subject matter categories in 35 U.S.C. §§100(a),101). Reflecting a renewed sensitivity to Supreme Court precedent but without directly overruling *State Street*, the *Bilski* court found the State Street test inadequate. The decision drew on Supreme Court language to require that either a transformation of matter or use incident to a particular machine is needed to establish that a process is patentable.[19]

Unlike the Federal Circuit, TBAs and European courts have had to draw lines within the framework of Article 52 and parallel national laws. The TBAs have done so through an intensified focus on how the technicity requirement should be formulated and applied. As the referral to the EBA observes, the line has been drawn by using different, and inconsistent, tests of technicity. The 2001 Fraunhofer/Max Planck Institute study commissioned by the German government found that software closer to "engineering science" was more likely to be held patentable, but that the line was not drawn clearly or consistently by either the German courts or the EPO.[20]

Since the European Patent Convention came into effect, the universe of computer programs has expanded dramatically in variety, pervasiveness, social impact, and economic significance. Software has come to permeate virtually every profession, discipline, and undertaking. While some would like to think that this means that all software must be patentable, the greatly enhanced importance of software argues for close and careful scrutiny of what kind of software, if any, fits within the patent system, bearing in mind that within the system, it will be treated no differently than a life-saving drug.

Aggregate Liability and Private Costs

The determination of patentability is critical not only as to whether it enables innovators to file for patents – but for whether they are obliged to search for, avoid, and negotiate patents held by others. Patent jurisprudence is often framed only in terms of benefits of owning patents – and not in terms of the constraints, costs, and risks imposed on innovators by the patents of others. This is not just a matter of not copying. Recent research shows that the risk of being sued for patent infringement is directly related to the size to the scale of R&D.[21] Research also shows that copying is alleged (<u>not</u> proven) in a minor fraction of U.S. infringement cases – and only 2.6 percent in the case of software![22] Patent law's intolerance of independent invention may be a poor fit for software, especially given the tension it creates with copyright-based expectation of originality (discussed below).

Until recently, there has been little to connect the patent regime directly to economic benefits and costs. As a result, courts were free to assume that the patent system was working as imagined and that more patents produced more innovation. However, survey evidence has made it clear that a variety of means other than patents are used to appropriate returns from innovation, and that patents are the preferred means only in a few industries.[23] Bessen and Meurer use event studies combined with well-established means of valuing patents to show that the regime can operate as a net tax on innovation, at least outside of pharmaceuticals and chemicals – and that innovation in software appears to suffer disproportionately.[24]

Characteristics of Software

Without a careful effort to think through the appropriateness and formulation of limits, as the framework of EPC Article 52 requires, the patent regime is bootstrapped into nearly every facet of human life – despite the fact that it is a monolithic regime crafted for an industrial economy. Yet software is written, not manufactured, and can be reproduced and distributed at virtually no cost thanks to a commodity infrastructure of globally networked computers. Paradoxically, the ease of creating software creates a superabundance of recorded functionality – part of the glut of digital information that pervades contemporary life, but without the order and accessibility provided by citations, hyperlinks, and search engines.

While unique in some respects, software is also an extreme case of product complexity that is characteristic of computers, semiconductors, and other tangible technologies. This complexity departs radically from the one-patent-per-product technology where the patent system seems to work best. In the case of software, hundreds or thousands of potentially patentable functions may be needed for a single product. If the market is product-driven (as most are), the value of individual patents is greatly diluted.[25] However, the complexity of the product also makes it possible to accumulate large portfolios of patents. Portfolio racing feeds on itself, adding to the overall volume of patenting. Hence the so-called “patent paradox” in which large numbers of patents are sought and received in sectors where they are less valuable.[26]

It may be easy to design around known individual patents in complex environments, but in large numbers patents can present formidable obstacles that are very costly to identify, delineate, evaluate, and navigate around. Portfolios have a different competitive dynamic than individual patents.[27] Large firms seek “freedom to operate” by cross-licensing portfolios, providing nonexclusive access to tens of thousands of patents owned by competitors. In other words, portfolios enable major stakeholders to buy their way out of the exclusivity that patents are designed to provide.

This works well enough in concentrated markets segments where there are relatively few competitors, as is typically the case when there are barriers to entry such as the need to construct sophisticated manufacturing facilities. But software development is so barrier-free, distributed, and heterogeneous that cross-licensing cannot practically be applied to the large number of differently situated participants. As noted by former Microsoft CTO Nathan Myhrvold: “On the hardware side it's easy for large organizations to cross-license with each other, but that doesn't work in software, because so many patents are held by individuals and you can't really cross-license on a small scale.”[28]

Although commonly presented as a defensive solution, portfolios serve to disadvantage new entrants who lack portfolios of their own. Without portfolios to cross-license, small and medium-sized enterprises (SMEs) cannot acquire the freedom of action that portfolio owners enjoy.[29] The European Commission omitted this problem in its presentation of the Directive on Computer-Implemented Inventions. Similarly, the parliamentary rapporteur's summary of the proposed directive touted the advantages of software patents to SMEs without acknowledging the barriers they face from patent portfolios – an omission that provoked strong criticism from economists.[30]

This imbalance is not a problem in copyright. Moreover, the imbalance in patent actually undermines the value of copyright to small developers. Lawyers typically view the applicability of both patent and copyright to software as simply options for protecting different aspects of the program, but from the programmer's perspective, adverse practical effects flow from patents. While copyright is acquired automatically and easily observed in practice, applying for, monitoring, and defending against patents is very costly. By precluding independent invention, patents defeat copyright-based investments in original work. Hence, the concern of independent developers that "patents are stealing our copyrights" fuels much emotion behind the software patent debate.

Low barriers to patenting, whether in terms of inventive step or subject matter, increase the absolute number of patents, increasing the power of portfolios as well as the number wielded by non-practicing entities. This fragmented ownership diminishes the prospects for freedom of action and increases risk and uncertainty for all producers, but especially for small producers who face both portfolio owning incumbents and opportunistic trolls.[31] As the costs of searching for and evaluating patents rise, effective notice and disclosure are lost, and opportunities for ambush increase. Moreover, the ease with which trivial technology can be inadvertently incorporated into components endows trivial patents with undeserved negotiating value. The more widely infringed, the less the incentive for any one company to invalidate the patent, since its competitors will free-ride on the costs of invalidation.[32] Better to settle and encourage the patent owner to pursue one's competitors.

The heterogeneity of software directly compounds the problem of functional complexity. This results in an overabundance of information of uncertain quality – and with it opportunities to exploit asymmetric information, differences in vulnerability, and differences in bargaining strength. While useful in dealing with competitors, present and potential, portfolios are not effective against non-practicing entities that have no legal exposure to the patents of others. Large deep-pocketed producers offer the greatest potential rewards, but the high costs of evaluating validity and infringement means that trivial patents can also be used to extract small settlements from SMEs who cannot justify the cost of evaluating the patent's validity.[33]

Why Software Is Unique

Most of the problems outlined apply to complex products as a whole – software simply offers the worst case. But pure software also has unique economic characteristics. As pure information, software is nonrival and nondepletable – characteristics that can move costs to zero as the costs of complementary processing, storage, and transmission technologies have dropped. Like other forms of pure information, software can be marketed through a wide range of complements, such as hardware, services, or advertising that eliminate the need for individual transactions. While software has always been nonrival and nondepletable in principle, these characteristics could not be exploited fully because software was bound to costly standalone computers. The microcomputer revolution, the Internet, and the Web (and the declining associated costs of processing, storage, and transmission) have provided a vast, deeply commoditized global infrastructure, within which both reproduction and distribution are virtually costless and instantly available. These factors together have led to radically different production models that enable widely distributed, differently motivated individuals to work cooperatively on large-scale software. With the help of production software their work can be modularized and exchanged over the Internet on a daily basis – and then reassembled and tested every night.

The combined effect of these factors has enabled a variety of open source production and licensing models to flourish without facing the costs associated with tangible products. Copyright is embraced by consensus because it is based on behavioral norms that do not preclude independent creation. It simply constrains controllable objectionable behavior and cannot be used as a weapon against inadvertent infringers.[34] It imposes no obligation to search for documents generated by lawyers. There is no stockpiling of arsenals, no need for defensive acquisitions or freedom of action. It works for large and small developers regardless of business model.

The special economic characteristics of software diversify business models rather than simply changing them. Industrial-style models remain an option. Software can still be developed in-house as an exclusively controlled product of large investments that is heavily promoted and distributed on tangible media through conventional retail in shrink-wrapped boxes. However, high-cost legacy models should not be favored at the expense of new models that make more efficient use of software's unique characteristics.

The importance, and industry embrace, of open source models can be seen in the policies of standards-setting organizations (SSOs) where the standard is intended to be implemented in software. Since open source models are based on free distribution, there is no opportunity to charge royalties. So SSOs, including W3C, OASIS, and OAGi, have adopted royalty-free licensing as their default policy. This allows software to have the widest possible distribution in keeping with its basic economic characteristics.

Drawing Lines

Patent law is a strict liability regime that demands meticulous and costly line drawing in multiple contexts: in examination, in oppositions, in clearance searches, in subsequent disputes, and ultimately in court. Lines are needed to determine novelty, inventive step (nonobviousness), industrial application (utility), enablement, scope of claims, and so on. These binary determinations are imposed on a complex, nuanced world. Investigation and interpretation are needed, and when conflicts arise, the stakes are high. In fact, the stakes may be high enough for one side to justify huge expenditures, which may oblige the other side to counter. U.S. figures show an average cost of over $15,000 for a legal opinion on a patent's validity and another $13,000 or more for an opinion on whether a particular function or component infringes the patent.[35] Even if the assertion is contestable, legal expenses can be expected to be average $767,000 when the amount at risk is less than $1 million -- and much more when the amount at stake is higher.[36] These figures do not even include the cost of distracting key personnel from the operation of the business, which is disproportionately high for SMEs. Facing these costs and risks, it is economically rational for SMEs to pay $20,000 for a license – rather than take the first step of evaluating a defense.

The cost of determining boundaries is higher in software and business methods because of the abstract nature of the language.[37] Abstract terms can lead multiple interpretations, and different terms can be used to mean the same thing, which makes searching difficult. This indeterminacy leads to fuzzy boundaries, indeterminacy, and uncertainty – a line-drawing burden that is multiplied by the immense volume of relatively unorganized prior art. Depending on how permissive the patent regime is, patents can exist at overlapping levels of abstraction, extending from code-level algorithms to internal data process to user-oriented features to system-level configuration to business methods.

The questions of technicity and technology differ from level to level, although fully developed software programs typically span multiple levels of abstraction. Under the EPC Article 52, it seems clear that the extreme ends of the spectrum are non-technical, but there is no guidance for the territory in between. Nor is there guidance in the EPC for how the different levels may interact with each other, i.e., whether technical levels can incorporate or bootstrap non-technical levels into patentable inventions.[38]

TRIPS is of less in terms of what falls within a field of technology, since it leaves the definition of technology to national law.[39] In stark contrast to the explicit and much-discussed application of copyright to computer programs in TRIPS, there was no discussion about whether computer programs should be deemed technology. The "all fields of technology" clause of Article 27(1) had nothing to do with the treatment of computer programs but was inserted to ensure that pharmaceutical products were protected globally.[40] At the time that TRIPS was negotiated, it was generally assumed that programs were not patentable. Statutory exclusions were commonplace, and even in the U.S., there was considerable uncertainty as to the scope of patent-eligible software. When the EPC was amended to incorporate the "all fields of technology" provision, the delegates explicitly and rightly declined to amend the computer exclusion in deference to the political process underway at the European Commission.

Heterogeneity and Asymmetry in Software

The Supreme Court's view of the algorithm in *Benson* illuminates another aspect of the notice problem: Abstract claims that are not tied to a particular field or physical implementation burden the many fields in which they may be used. The more basic the logical function, the more widespread the potential uses and the less likely that users will have specific knowledge of the patent.[41] A strict liability regime presents great dangers because of the immeasurable risks that can result from the ubiquitous use of pure software.

Where natural barriers to entry are significant (semiconductors, computers) cross-licensing effectively limits notice problems among major stakeholders. But, as noted above, cross-licensing is not feasible when the interests of millions of software developers of varying size must be assessed and negotiated. Asymmetries of knowledge and resources, and therefore opportunities for holdup, are far greater in software. There are even attacks on mere users, such as retail websites, although many of these presumably go unreported because they are settled before a lawsuit is filed and the companies involved are not newsworthy.[42]

Cost-Benefit Framework

Individual patents always have private value. But assessing and interpreting the EPC requires a systemic perspective, and the empirical understanding of innovation and the patent system that has emerged over the past 35 years shows the wisdom of identifying specific limitations to be filled in based on experience over time. Unfortunately, a number of the divergent TBA decisions suggest a weakening of standards over time that has gradually undermined confidence in the system as a whole.

The lack of an institutional focus on the functioning of the patent system as a whole, including business practice and effects on innovation, naturally limits the ability of the system to correct aberrations. There is no economic or policy framework that would allow TBAs (or the PTO and the Federal Circuit) to maintain a broad perspective. The inevitable result seems to be a jurisprudence mired in legal semantics that make little or no sense to the public.

Hence the appeal of the cost-benefit framework developed by James Bessen and Michael Meurer in *Patent Failure*. This work reveals chemicals and pharmaceuticals to be the one sector where the aggregate value created by the patent system clearly outweighs aggregate costs, confirming what survey evidence, anecdote, and intuition have long indicated. Patents are essential to protecting the large investments needed to test and bring drugs to market. Molecules are well-defined and unambiguous. A single patent can effectively protect the entire marketed product. There is little risk that unknown patent owners will be needed to keep the product on the market. Competitors are few, and they consult the patent database regularly to learn of each other's research.

Under those circumstances, the patent system works. However, the reasons that patents work well for pharmaceuticals reflect the very reasons that patent system works poorly for software. The difference in benefits and costs can be summarized in terms of the factors that derive from the extreme complexity of software, as well as from distinct character as digital information:

Diminished benefits

a. Dilution of individual patents due to complexity of products and the need for access to many patents.[43]

b. Less investment to be protected in absolute terms (no costly research, no clinical trials)

c. Less investment in patent-protectable value. Design, integration, and debugging are important elements of software value that cannot be effectively protected by patents.[44]

d. Alternative means of appropriating returns to innovation, especially the automatic, virtually costless, and uncontroversial use of copyright. Secrecy, contract, first-mover advantages, and a wide range of complements-based strategies are also available.

e. No demonstrated benefit. History shows that the advent of patent protection for software under the U.S. Court of Appeals for the Federal Circuit had no discernable impact on software development.

Costs: Increased risk of liability

a. Complex products are especially vulnerable to hold up and ambush. If injunctive relief is available to the patent holder, the entire product can be forced off the market.[45]

b. The sheer number of possibly patentable functions in software combined with the sheer number of programmers and software firms. Many patents will belong to others creating widely dispersed and unforeseeable sources of liability, and many of these will be non-practicing entities or firms in other industries, rather than well-known competitors.[46]

c. Common underlying functionality that may be incorporated in widely different contexts may be preempted by patents.[47]

d. The importance and ease of propagating software standards can lead to large-scale liability, thereby as a powerful inducement for strategic behavior by patent applicants and owners.[48]

e. The symmetry and scale needed for effective cross-licensing is lacking. “Mutually assured destruction” (MAD) is partially effective as a deterrent among producers but ineffective where there is little competitive overlap.[49]

f. Under a permissive regime, patentable functions can be found at many overlapping levels of abstraction. This layering adds to the problem of identifying, designing around, and/or negotiating patent rights.

Costs to competition

g. Patent portfolios create barriers to entry in product markets since newcomers lack the patents needed to secure freedom of action through cross-licensing.[50]
h. Strong network effects in software can create and sustain dominant providers who can use patents to exclude rivals – simply by using patents to limit access to an interface, even if the patented technology is trivial.[51]
i. There are scale and scope economies in managing knowledge and patents limit competition from individuals and SMEs that are normally an important source of innovation in software. Large companies benefit from the lower costs of in-house legal services, and only the largest companies can afford to evaluate and navigate patent thickets in areas of commercial interest. The high costs of professional help in assessing validity, infringement, and other legal issues (see transaction costs below) may force individuals and small businesses to license questionable patents. The smaller the enterprise, the greater the opportunity costs of patent disputes.
j. The opacity of patent portfolios encourages nonspecific threats against competitors and their users.[52] It shifts competition from product value to legal indemnification, which favors the most deep-pocketed companies.

Transaction costs

k. Exceptionally high search costs in software due to the sheer volume of functionality within overlapping layers of abstraction. Prior art is often available only in the form of software itself, where it is not organized or readily searchable.
l. Interpretation of claims is unpredictable and indeterminate because of the use of multivalent and abstract language, making evaluation of validity and infringement costly and uncertain.[53]

Drawing the Line Under the EPC

The President’s referral reflects a renewed commitment to drawing meaningful, consistent, and effective limits in light of the changing nature of innovation and the economic and social goals of the patent system. As the referral suggests, the line should be clear not just for patent professionals but for the intended beneficiaries of the system – innovators and the public.[54]

We now have a far richer understanding of what computer programming encompasses and what it can do. “Programming” can be editorial in nature (HTML coding) and simplified authoring tools can enable unsophisticated people to program.[55] Software may represent the ultimate democratization of technology – or merely the digitized enhancement of most human activity.

At the same time, software has challenged the patent system greatly, vividly exposing some of the inherent tensions.[56] It has illuminated the great difficulty of drawing bright lines in an expanding and volatile landscape of innovation. It has led the way in exposing fissures within a unitary patent system, showing that there are in reality two different systems: one where disclosure still works (centered on chemicals and pharmaceuticals) and one where patents are only read by lawyers. The vast scope of software and the diversity of interests in software have deeply politicized the patent system.[57] It's not just complex product producers against discrete product producers. Upstream interests are pitted against downstream interests. SMEs are pitted against MNEs. Software developers are pitted against patent attorneys.[58]

Alone among the world's patent offices, the European Patent Office has recognized the immensity of the political, economic, and legal challenges facing the patent system in a globalized economy.[59] We commend the president's referral for addressing the fundamental challenge of how to draw limits on the patent system from within in terms that have lacked, and perhaps defy, definition. We urge the Enlarged Board of Appeal to meet this challenge with the intellectual force that it demands.

Recommendations

Containing the problem of inadvertent infringement means charging a manageable community of technology professionals with practical working knowledge of the patent database. This cannot be done if volumes of patents spill promiscuously into every area of human endeavor, including the routine uses of general purpose computers. Defining technicity in terms of physical transformation of matter can prevent this problem in a way that can be clearly understood by the public.

Drawing a line in terms of physical transformation would still leave questions surrounding the distinction between physical and informational. For example, *Bilski* allows intangible representations of physical transformation.[60] Although this distinction finds no support from Supreme Court precedent, it still helps confine the benefits and burdens to a relatively discrete professional community. Another permissive but still meaningful way to draw the line would be to treat analog (but not digital) processes as physical.

The alternative prong of the *Bilski* test, which would allow a process tied to particular machine, was not explored because no machine was involved in that case. However, the USPTO Board of Patent Appeals and Interferences has found that processes implemented on general-purpose computers do not qualify under this standard.[61] This is consistent with the Supreme Court's decision in *Parker v. Flook*, which would view the use of a general purpose computer as a routine, insignificant add-on[62] -- unlike the rubber curing process which was the subject of the ruling in *Diehr*.

The *Bilski* test parallels the principle of "further technical effect" – i.e., something outside of the logical box of general purpose computing that necessarily links the software to tangible subject matter. Otherwise it is too easy to confuse the software with the box itself. [63] When the concepts of "technical" and "technology" have become democratized and diluted by the hundreds of millions who have some level of technical skill, it is essential that "technical" end up meaning more than whatever the judge does not know how to do. The idea of partitioning "technical" and "non-technical" inside a continually expanding universe of general-purpose computing has become an increasingly futile and untenable exercise in semantics.

We recognize that the EBA will hear many different perspectives on where and how the line should be drawn. It is often difficult to separate short-term interests from long-term goals, and some of our members are internally conflicted in this regard. But it is now clear that permissive standards for patentable subject matter inevitably create a constituency deeply invested in accumulated portfolios and institutionalized portfolio strategies, while future innovators that would benefit more from less cluttered, more open prospects are not heard from. Because of the delays built into the patent system, systemic problems lag behind the immediacy inevitably surface, whether in the form of administrative overload or an addiction to volume in professionals and industry, or to the derivative phenomena of thickets, inadvertent infringement, trolls, and disclosure failure. While these problems may be endemic to information technology, in software they have reached extremes of symptom and public controversy, thanks in large part to the unique economic characteristics that have made software so rich, diverse, and ubiquitous.

The Referral Questions

With this in mind, we offer the following answers to the questions posed in the EPO President's Referral to the Enlarged Board of Appeal. The questions fall into two categories. The first concerns whether the exclusion can depend on the form of the claim rather than the essence of the invention. There the answer is clear: The computer program exclusion must be respected as a matter of substance, not form. Otherwise, the system will belong to the high priests who know the magic words.

The second category is the more difficult question of how to draw the line between technical and non-technical. We believe that the *Bilski* decision's return to early Supreme Court precedent represents a healthy return to a workable and meaningful line between patentable and unpatentable subject matter. The Enlarged Board of Appeal can similarly redraw this line within the framework of EPC 52 and European jurisprudence. The President's referral clarifies the need to do so and shows that despite the divergence, the right precedent can be found.

The Questions

Question 1: Can a computer program only be excluded as a computer program as such if it is explicitly claimed as a computer program?

The answer must be no. Otherwise, the exclusion could be routinely evaded.

Question 2: (A) Can a claim in the area of computer programs avoid exclusion under Article 52(2) (C) and (3) merely by explicitly mentioning the use of a computer or a computer-readable data storage medium?

No. A computer program is useless without a computer. Here, too, the exclusion would be rendered meaningless if it could be avoided by merely claiming the computer (or a storage medium) as well.

Question 2 (B): If Question 2 (A) is answered in the negative, is a further technical effect necessary to avoid exclusion, said effect going beyond those effects inherent in the use of a computer or data storage medium to respectively execute or store a computer program?

Yes, with "technical" understood as requiring a physical transformation – so as to confine the operation of the patent to a meaningful and manageable epistemic community.

Question 3: (A) Must a claimed feature cause a technical effect on a physical entity in the real world in order to contribute to the technical character of the claim?

Yes. However, to get beyond the circularity and uncertainty surrounding "technical," it should be clear that "technical" requires a material transformation of a physical entity.

Question 3 (B): If Question 3 (A) is answered in the positive, is it sufficient that the physical entity be an unspecified computer?

It is sufficient only if the effect changes physical aspects. This question goes to the heart of defining the physical aspect of "technical." A permissive but still reasonable definition would include transformation of analog signals, but not of logical or codified content.

(C) If Question 3 (A) is answered in the negative, can features contribute to the technical character of the claim if the only effects to which they contribute are independent of any particular hardware that may be used?

Question 4: (A) Does the activity of programming a computer necessarily involve technical considerations?

No, this begs the ultimate question of what programming should be considered technical. For example, use of HTML coding clearly should not be considered technical. Programming that directs the operation of a nuclear reactor can readily be considered technical because it will physical effects. The question is where to draw the line.

(B) If Question 4 (A) is answered in the positive, do all features resulting from programming thus contribute to the technical character of a claim?]

(C) If Question 4 (A) is answered in the negative, can features resulting from programming contribute to the technical character of a claim only when they contribute to a further technical effect when the program is executed?

Yes. While defining what kind of programming is or is not subject to the patent regime would help to provide practical guidance to the field, the "further technical effect" standard offers a clearer, more comprehensible line assuming that "technical" implies physical transformation. While expressed in European terms, this understanding of "technical" aligns with *Bilski* and U.S. Supreme Court decisions and would exert a useful, convergent influence on the evolution of subject matter jurisprudence in the U.S. and elsewhere.

Conclusion

After *State Street* and the European Commission's too-eager attribution of U.S. innovation to U.S. patent practice,[64] it may not seem advisable to hold up U.S. patent law as an example. But the focused holding of the U.S. Supreme Court in *Diamond v. Diehr* is often forgotten or mischaracterized. The Court merely held that an otherwise patentable physical process is not made unpatentable by the inclusion of a computer program as one of its elements. By accepting certiorari in *Labcorp v. Metabolite*, the Supreme Court showed its intent to revisit subject matter issues, even though it eventually chose not to do so in that particular case.[65] The Federal Circuit's long excursion from Supreme Court precedent has since been curtailed in *Bilski*. In rejecting the virtually boundless "useful, concrete, and tangible result" test of *State Street*, the Federal Circuit recognized that the Supreme Court would be inclined to jettison the test – just as it had recently reversed the Federal Circuit in a series of high-profile cases cutting back on the power of patent applicants and holders.[66]

Despite the divergence and opacity of the technicity tests, the Enlarged Board of Appeal has not previously had the opportunity to interpret the computer program exclusion. The referral gives the EBA a first opportunity to distill the collective wisdom of the President's Commission, the U.S. Supreme Court trilogy, and the legislated exclusion in the European Patent Convention – and to determine where and how the perimeters of a unitary patent system can be set to achieve the goal of promoting innovation. It should ensure that software as such is not straight-jacketed by inappropriate rules that serve particular private interests rather than the interests of the citizens of Europe and the world.

[1] We use the term "technicity" to refer to the various formulations of "technical contribution," "technical character," "technical effect," and so on regardless of operative structure.

[2] The EPC picked up the language of 52(2) from the recently negotiated Patent Cooperation Treaty (1970). *See* Justine Pila, *Dispute over the Meaning of 'Invention' in Article 52(2) EPC - The Patentability of computer-implemented Inventions in Europe.* International Review of Industrial Property & Copyright Law, Vol. 36, 2005. *Available at* SSRN: http://ssrn.com/abstract=593881.

[3] R. M. Hilty & C. Geiger, *Patenting Software? A Judicial and Socio-Economic Analysis,* International Review of Intellectual Property and Competition Law 3**6**(6), (2005). notes [14], [15]

[4] President's Commission on the Patent System, *To Promote the Progress of Useful Arts*, Report to the Senate Judiciary Committee, S. Doc. No. 5, 90th Cong., 1st sess. (1967)

[5] *Gottschalk v. Besen*, 409 U.S. 63, 72 (1972)

[6] cite to 2000 annual report

[7] Roger E. Schechter and John R. Thomas, *Intellectual Property*, West Hornbook Series, 2003, p. 314.

[8] This language was first used in *In re Allapat,* 33 F.3d 1526, 1544 (Fed. Cir. 1994), but only enshrined as a test in *State Street Bank v. Signature Financial Group,* 149 F.3d 1368, 1373 (Fed. Cir. 1998)

[9] State Street,149 F.3d 1368 at 1373. The distorted use of legislative history in State Street is documented in Brief for Computer & Communications Industry Association as Amicus Curiae Supporting Petitioner, *Lab. Corp. of Am. v. Metabolite Labs., Inc*., 543 U.S. 1185 (2005) (No. 04-607), 4–10 (Dec 23, 2005), available at http://www.ccianet.org/modules/patentPDFs/CCIALabCorpMeritsAmicus%5B04-607%5D.pdf.

[10]Declan McCullagh, *Left Gets Nod from Right on Copyright Law*, CNETNEWS.COM, Nov. 20, 2002, *available at* http://news.com.com/2100-1023-966595.html ("'A specialized court tends to see itself as a booster of its specialty. . . . Patents are clogging retail commerce on the Internet, and software (patents) in particular are apparently impediments to software development.'" (quoting Judge Richard Posner)).

[11] John R. Thomas, *The Patenting of Liberal Professions*, 40 B.C. L. REV. 1139 (1999).

[12] citations to patents

[13]The record suggests that this was unanimously opposed by European and other delegations, even though the U.S. delegation threatened to withdraw from the harmonization talks. *See* WIPO Standing Committee on Patent Law, *Seventh Session, Report,* SCP/7/8 (Geneva, May 2002) ¶159-173

[14] UK and EC consultations in 2000.

[15] FTC supra; http://www.nap.edu/catalog.php?record_id=10976; http://press.princeton.edu/titles/7810.html

[16] KSR v. Teleflex, 550 U.S. 398 (2007); eBay Inc v. MercExchange, L.L.C., 547 U.S. 388 (2006); Microsoft v. AT&T, 550 U.S. 437 (2007); Quanta v LG Electronics, 128 S.Ct. 2109 (2008); Medimmune v. Genetech, 549 U.S. 118 (2007)

[17] Diamond v. Diehr*,* 450 U.S. 175 (1981)

[18] After accepting certiorari in *Labcorp v. Metabolite,* the Supreme Court decided to dismiss the case without deciding it on the grounds that review was improvidently granted. *Labcorp* involved diagnostic information, and the subject matter issue was raised belatedly. Nonetheless, three justices issued a strongly worded dissent to the order of dismissal.

[19] *In re* Bilski*,* 545 F.3d 943, (Fed. Cir. 2008)

[20] Knut Blind and Jakob Edler (Fraunhofer Institut), Ralph Nack and Joseph Straus, (Max Planck Institut), Micro- and Macroeconomic Implications of the Patentability of Software Innovations. Intellectual Property Rights in Information Technologies between Competition and Innovation.

[21] James Bessen & Michael J. Meurer, Patent Failure, (1st ed. 2008)

[22] Christopher Anthony Cotropia & Mark A. Lemley, *Copying in Patent Law,* (Stanford Public Law Working Paper No. 1270160). *Available at* SSRN: http://ssrn.com/abstract=1270160

[23] W.M. Cohen, R.R. Nelson & J.P. Walsh, *Protecting their Intellectual Assets: Appropriability Conditions and Why U.S. Manufacturing Firms Patent (or Not),* (National Bureau of Economic Research, Working Paper No. 7552, 2000) (revised 2004).

[24] Bessen & Muerer, *supra* note [21], analyzing public U.S. companies over several decades and finding a shift attributable to Federal Circuit rulings leading to notice failure. While the data does not allow a discrete sector analysis for software, indicators such as the frequency of litigation and uncertainty of claim construction suggest that the costs are exceptionally severe for software and business methods.

[25] Don E. Kash & William Kingston, *Patents In a World of Complex Technologies,* 28 Science and Public Policy 1, 11 (2001).

[26] B.H. Hall & R.H. Ziedonis, *The Patent Paradox Revisited: An Empirical Study of Patenting in the US Semiconductor Industry, 1979-95,* 32 Rand Journal of Economics 1, 102-128 (2001). The Eli Lilly brief in *AT&T v. Microsoft* argues for exclusion of software from patentability on this principle. Brief of Amicus Curiae Eli Lilly and Company in Support of Petitioner, Microsoft v. AT&T, 550 U.S. 437 (2007) (No. 05-1056)

[27] R. Polk Wagner & Gideon Parchomovsky,*Patent Portfolios* . U of Penn. Law School, Public Law Working Paper 56; U of Penn, Inst for Law & Econ Research Paper 04-16. (March 1, 2005) *available at* SSRN: http://ssrn.com/abstract=582201 or DOI: 10.2139/ssrn.582201

[28] Interview with Nathan Myhrvold*,* PC Magazine, Sept. 9, 1997, at 30.

[29] While the same portfolio effects apply to other complex-product technologies, it is of less significance relative to other costs of doing business. It figures prominently in software because software development is uniquely open to small companies and individuals.

[30] An Open Letter to the European Parliament Concerning the Proposed Directive on the Patentability of Computer-Implemented Inventions of 8/25/03, *available at* http://www.sussex.ac.uk/Units/spru/nprnet/policy/patentdirltr.htm

[31] We use the term "troll" rather than "non-practicing entity" to denote a business based on being infringed. *See* Markus G. Reitzig et al., *On Sharks, Trolls, and Other Patent Animals—'Being Infringed' as a Normatively Induced Innovation Exploitation Strategy,* (Working Paper) *available at* http://ssrn.com/abstract=885914.

[32] Mark A. Lemley & Carl Shapiro, *Probabilistic Patents*. Journal of Economic Perspectives, Vol. 19, p. 75 (2005) (Stanford Law and Economics Olin Working Paper No. 288). Available at SSRN: http://ssrn.com/abstract=567883 or DOI: 10.2139/ssrn.567883

[33] Most actions small entities are not newsworthy. But s*ee* http://www.nytimes.com/2003/01/28/business/technology-users-uneasy-on-sbc-claim-to-patent-on-web-tool.html; *Retail e-Commerce Lawsuits Are Settled, But More May Be in The Works*, Internet Retailer, May 1, 2004, http://www.internetretailer.com/internet/marketing-conference/81025-retail-e-commerce-lawsuits-are-settled-but-may-be-works.html http://www.techcrunch.com/2008/07/17/channel-intelligence-sues-just-about-everyone-who-offers-wishlists/

[34] Of course, developers must be careful when they acquire code from outsiders since they will be held responsible for code that has been copied without permission.

[35] American Intellectual Property Law Association (AIPLA), Report of the Economic Survey 2007, p I-83.

[36] AIPLA, p. I-90. Litigation costs are still comparatively less in Europe. See CJA Associates study of patent litigation insurance prepared for the Europe Commission: http://ec.europa.eu/internal_market/indprop/docs/patent/studies/pli_report_en.pdf, especially Appendix 3 at page 46: http://ec.europa.eu/internal_market/indprop/docs/patent/studies/pli_appendices_en.pdf.

[37] Bessen & Meurer, *supra* note [21], ch.9.

[38] *Diamond v. Diehr* addressed one aspect of this problem in allowing an unpatentable algorithm to be incorporated into patentable physical process.

[39] Reinier Bakels & P. Bernt Hugenholtz, *The Patentability of Computer Programmes: Discussion of European-level Legislation in the Field of Patent Software* 17, (commissioned for the European Parliament by the Directorate-General for Research, Apr. 2002), *available at* http://www.europarl.europa.eu/meetdocs/committees/juri/20020619/SoftwarePatent.pub.pdf*.*

[40] Graeme B. Dinwoodie and Rochelle C. Dreyfuss, *Diversifying Without Discriminating: Complying with the Mandates of the TRIPS Agreement,* 13 Mich. Telecomm. Tech. L. Rev. 445 (2007), *available at* http://www.mttlr.org/volthirteen/dinwoodie&dreyfus.pdf

[41] Ben Klemens, *The Rise of the Information Processing Patent*, Boston University Journal of Science and Technology Law, 14:1 (2008)

[42] *See supra* note [33]. The Foundation for Free Information Infrastructure was able to score political points with its "Patented European Webshop," which graphically shows twenty European software patents implemented in a website context despite the EPC prohibition of patents on software as such. http://webshop.ffii.org/

[43] *Supra* notes [25], [26]

[44] Elements of program may be patented, and design patent are available for visual designs, but design is generally too complex to be captured efficiently in patent claims.

[45] Although automatic injunctive relief in the U.S. ended with *eBay v. MercExchange*, unpredictably high damage awards (in which reasonable royalties are inappropriately keyed to the value of the entire product) can have similar effects. The formulation of damages has been the single most contentious issue in patent reform in the U.S.

[46] Klemens, *supra* note [41]

[47] This is the problem identified in *Benson*. A more recent example, which could have been enormously disruptive was BT's claim on hyperlinks. *See* Matt Loney, *BT Loses Hyperlink Patent Case,* Zdnet.co.uk, Aug. 23, 2002. A*vailable at* http://news.zdnet.co.uk/itmanagement/0,1000000308,2121257,00.htm

[48] BT Loses, *supra* note [47]. While strategic behavior by can be addressed within standards-setting organizations to some degree, the more difficult challenge is presented by nonparticipants. “Nonparticipant” ambush is especially problematic where drafts and discussions are open to the public. *See* http://www.ccianet.org/docs/papers/Kahin%20on%20Standards&Patents.pdf

[49] Mutually assured destruction depends on the likelihood that each side will have at least one patent that reads on the product of the other. http://www.eweek.com/c/a/Linux-and-Open-Source/Software-Patents-and-Mutually-Assured-Destruction/

[50] Economists letter to the European Parliament, *supra* note [30]

[51] Joseph Farrell, Intellectual Property in Network Industries

[52] For example: Microsoft’s nonspecific allegations of patent infringement against Linux. [cite to Fortune interview of Brad Smith, May 2007] Microsoft has yet to specify the patents that it claims are in infringed.

[53] Bessen & Meurer, *supra* note [21], at ch.9.

[54] Referral, p. 2 (Summary)

[55] As the referral notes (p. 12), “modern (high-level) programming languages do their utmost to render technical considerations unnecessary.”

[56] Brian Kahin, "Through the Lens of Intangibles,” in *Patents, Innovation and Economic Performance*, OECD 2004, *available at* http://www.si.umich.edu/~kahin/Lens.doc.

[57] In Europe, this was manifest most clearly in the debate and ultimate debacle of the Computer-Implemented Inventions Directive. In the United States, software interests have played an early, critical role in the push for strong patent reform and the inter-industry split over the scope of reform.

[58] P. Samuelson, M. Denber, R.J. Glushko, Developments on the Intellectual Property Front, 35 Communications of the ACM, No. 6, June 1992, pp. 33±39; see also Oz, Effy. 1998. "Acceptable Protection of Software Intellectual Property: A Survey of Software Developers and Lawyers." Information and Management 34(3): 161–73. The Oz survey is directed at developers at enterprises while the earlier work surveys developers as professionals.

[59] EPO, Scenarios for the Future, http://www.epo.org/topics/patent-system/scenarios-for-the-future.html; especially the “Blue Skies” scenario.

[60] 545 F.3d 943, at 985

[61] See summary of recent cases at http://www.patentlyo.com/patent/2009/01/the-bpai-and-the-machine-or-transformation-test-of-bilski.html

[62] pin cite to Flook

[63] Thus, the *Symbian* decision asked “what is a computer without an operating system?” – but one might as well ask, “what is a computer without a computer program?”

[64] Cite to 1997 Green Paper and 1999 Communication on the Community Patent. Both of these documents were written before the major problems in the U.S. patent system surfaced, beginning with the FTC and Department of Justice hearings in 2002.

[65] The Court's dismissal of the appeal on the grounds that review was improvidently granted should be viewed as an acknowledgment that the question of abstract patents was inadequately presented. *Labcorp* concerned only diagnostic information – not software or business methods, and the subject matter issue was raised belatedly. [Kellen, check this. My recollection is that the subject matter was raised only on appeal to the Supreme Court – or that the Supreme Court raised it itself.] Three justices felt strongly enough to dissent from the dismissal, making it clear that they would have voted to restrict the scope of patentable subject matter. cite to dismissal

[66] *Supra* note [16]