THE SOFTWARE PATENT CONUNDRUM AND OPPORTUNITIES FOR LEGAL COMMUNITY

Saurabh Prabhakar*

Introduction

The rapid growth of Digital Computers has marked the advent of a new chapter in the development of human society. It has been instrumental in vaulting mankind from the Industrial Era into the Knowledge Era. The Digital Computer has produced a profound and largely benevolent impact on all aspects of human society. It has been a key driver of economic prosperity in all parts of the world. The success of IT revolutions in developing countries like India, China and Eastern Europe has enabled alleviation of poverty in societies which are home to almost 40% of the world's population. The development of the borderless World Wide Web (www) and new medium of communications like voice-chat, email, social-networking etc. have made physical separation moot and have led to increased global interaction. This has resulted in greater understanding of cultural uniqueness and global bonhomie between civilizations. Increased computerization has produced significant change in the political landscape as well. The barrier-less access to information and development of concepts like e-governance has led to the creation of a well-informed and therefore empowered citizenry. This has resulted in increased accountability in government and has strengthened the foundation of democratic institutions. Therefore, whether it is economic, cultural, political or social, it is impossible to exclude any realm of human society which has not been revolutionized since the advent of the Digital Computer.

If the Digital Computer can be proclaimed the physical carrier of the knowledge revolution, then it would not be an exaggeration to claim that the software that drives the Digital Computer is the proverbial soul of this revolution. Without the ability to conveniently program the computer to carry out tasks as mundane as adding two numbers or as sophisticated as launching a space shuttle to Mars, this revolutionary invention of man would have been nothing more than a fascinating box of transistors and wires. Thus the contribution of the Computer Software industry towards the development of society is by no means smaller than the Semiconductor industry which created the Digital Computer.

In order to expand and prosper, the Computer Software industry, like any other industry, requires an ecosystem which is conducive to innovation

* Member of Consulting Staff, SLEC R&D.

and protective of property rights. However the relationship between the Intellectual Property Rights (IPR) regimes and the Computer Software industry has been marked by inflexibilities and inconsistencies. On one had the IPR regimes in European Union (EU) and India have refused to grant patents for software programs. On the other hand the regime in the U.S. has vacillated between blanket refusals[1] to that of categorical acceptances[2] on matters of software patents. This wide variance between various IPR regimes across the globe has weakened the protection of inventions in the software industry. Additionally, the ambiguity in application (copyright vs. patent) and interpretation of statutes (patentable or not-patentable) within the same regime is embroiling software manufacturers in expensive and time-consuming litigation. As the pervasiveness of computer software in human lives increases, this environment of indifference and ambiguity towards the protection of the IP of software industry would eventually impact growth thereof.

The uniqueness of Computer Software industry has also played a part in confounding the problem of software patents. Firstly, the products of software industry have no physical form as they exist inside digital computers as series of 1s and 0s. Their function is to program the computer to execute a series of instructions i.e. an algorithm. This raises serious challenges for the patenting regimes which were developed to protect inventions in the form of machines or physical processes which have well-defined form and application. Secondly, unlike industries like Manufacturing, Pharmaceuticals, Bio-technology etc. which have high entry costs and require long time to develop inventions, the software industry has low cost of entry and rapid rate of development. Thus the volume of IP that needs protection is very high. Thirdly, the low cost of entry and high profitability has created an industry which is an eclectic collection of large corporations, small start-up companies and a vibrant open-source community. This has created conflicting demands on the IPR regimes from the opposite ends of the spectrum, both in matters of grant and enforcement of protection.

This paper intends to present that how different yardsticks have been used by the U.S. Courts in deciding certain cases pertaining to granting and infringement of patents. It discusses how these judgements and the role of USPTO (United States Patent & Trademark Office) have created serious challenges for the software industry. It also invites the Indian legal community to take up the opportunity to initiate meaningful patent reform

1. *Gottschalk* v. *Benson*, 409 U.S. 63, 175 USPQ 673 (1972).
2. *Diamond* v. *Diehr*, 450 U.S. 175, 209 USPQ 1 (1981).

based on judicial interpretations of the presented cases. This paper then seeks to raise questions for the legal community involved in the development of the jurisprudence of Intellectual Property Rights on how existing IPR regimes can be reformed to provide consistent patent protection for computer software inventions and at the same time appropriately limit its scope in order to meet the constitutional mandate of IPR i.e. to promote the progress of Science and Useful Arts.

Patents: Definitions and Constitutional Objective (US)

A patent constitutes a limited time grant to the patentee, his heirs or assigns, a right to exclude others from making, using, offering for sale, selling or importing the invention for the term of the patent in the jurisdiction of the granting state. A patent is not the grant of right to make use or sell, only the right to exclude others.[3]

A patent can be obtained for the invention or discovery of a new and useful process (process, art or method), machine, manufacture, or composition of matter, or any new improvement thereof.[4]

The word "Patent" originates from the Latin word *patere*, which means "to lay-open" (i.e., to make available for public inspection). A patent is a set of exclusive rights granted by a state to an inventor or their assignee for a limited period of time in exchange for public disclosure of an invention. The motivation for patent grants is captured in the US constitution as follows:

"To promote the Progress of Science and useful Arts, by securing for Limited Times to Authors and Inventors the exclusive Rights to their respective Discoveries and Writings."[5]

The intention of Patent Law is to promote the progress of Science and Technology by encouraging inventors to publicly disclose their novel inventions to society. The public disclosure of an invention serves the following purposes:

1. It enables the use of the invention by society by building required machinery or processes
2. The disclosed invention can be used as a building block for future inventions by other inventors.

As an incentive for full disclosure the state grants the inventors a limited period right by which only they can commercially exploit their invention by selling, licensing, transferring or mortgaging.

---

3. *Herman* v. *Youngstown Car Mfg. Co.*, 191 F. 579, 584-85, 112 CCA 185 (6th Cir. 1911).
4. 35 United States Code S. 101.
5. U.S. Constitution Art. I, § 8.

Hence the yardstick to measure the success of any patenting regime is whether its implementation encourages innovation in society and whether the innovations are subsequently disclosed thereto. Justice Breyer observed in his dissenting opinion in *Laboratory Corporation of America Holdings* v. *Metabolite Laboratories Inc.*,[6] "… too much patent protection can impede rather than 'promote the Progress of Science and Useful Arts,' the constitutional objective of copyright and patent protection."

Software Program: Definition and Applications

A software program (also a computer program) is a sequence of instructions written to perform a specified task for a computer.[7] The Oxford Advanced Learner's Dictionary defines a program as a set of instructions in CODE that control the operation or functions of a computer.[8] An algorithm on the other hand is defined as a set of rules that must be followed when solving a particular problem.[9] Therefore, a software program is an algorithm which is used to solve a problem using a computer. The "set of rules" of an algorithm correspond to the "set of instructions in CODE" of a computer program. The distinction between the two terms is important as it would be used when determining the patentability of a software program.

Modern software programs can be quite sophisticated with code spanning over millions of lines. Therefore, a standalone software program which itself is an algorithm can consist of several other algorithms which work together to produce the desired result. For e.g. a word processor like Microsoft Word is an example of a computer program whose objective is to represent textual information in an organized manner. Within the word processor there can be sub-programs whose objective might be to carry out smaller tasks for e.g. spell-check or grammar check. Therefore, a single innovation in a software product can actually consist of several other innovations. Additionally, inventions in software may require combining several existing inventions in a bottom up fashion. These characteristics place demands on the scope of patents.

A software program can be used for a variety of purposes. The execution of a program can result in a tangible physical transformation for example Computed Numerically Controlled (CNC) machines[10] which are

6. *Labcorp* v. *Metabolite*, 548 U.S. 124, slip op. at 2 (2006), (Stephen Bryer, dissenting).
7. Stair, Ralph M. *et al.*, PRINCIPLES OF INFORMATION SYSTEMS, 6th ed. 2003, p. 132.
8. Oxford Advanced Learner 's Dictionary of Current English, 7th ed. 2005, p. 1206.
9. *Ibid*, p. 36.
10. Siegel, Arnold, "Automatic Programming of Numerically Controlled Machine Tools", Control Engineering, Volume 3 Issue 10 (1956), pp. 65-70.

run using a software program. Some other category of programs may not produce any tangible physical transformation or movement for example a word processing application which simply captures user provided data. In more sophisticated programs, an artificial intelligence program may itself create a software program without the intervention of a human operator. These different characteristics pose serious questions for patenting regimes when determining patentability of inventions.

There are two distinct concepts involved in the development of a software program:

1. The algorithm designed for the implementation of the desired task, i.e., the idea.
2. The manner in which the designed algorithm is implemented, i.e., the code.

The cognizance of these two concepts of developing a software program are important as the former can be protected by Patent regime and the latter can be protected by the Copyright Regime. Therefore effective protection of a software program requires protection using both the regimes.

Understanding the Software Patent Conundrum: U.S. Patent Laws and Court Rulings

The criteria for patentability of software programs vary across jurisdictions. The EU does not grant patents for Computer Program or Computer Implemented Business Methods.[11] Even India which is the epicentre of the world's software development industry does not grant patents for software programs.[12] The stance of the U.S. on software patents has however been ambiguous. Their patent laws have a very broad definition of patentability which does not explicitly refer to software program. However it is the very ambiguity of U.S. patent regime which makes it an excellent candidate for gaining vital insights in the procedural and jurisprudence issues involved in patenting software programs. There are valuable lessons to be learnt from understanding the judgement of U.S. Courts on some of the landmark cases involving patents by both software and legal professionals.

---

11. EPC, Art. 52 para. 2, which describes "what is not regarded an invention," section (c) describes "schemes, rules and methods for performing metal acts, playing games or doing business, and program for computers," as ineligible for patent grants.
12. The Patents Act 1970 (as amended on 2002), Chapter II, §3, which refers to "what are not inventions", clause (k) describes "a mathematical or business method or a computer program *per se* or algorithms," as a non-patentable invention.

The U.S. Code Title 35 Article 101 (35 U.S.C. §101) defines patentable inventions as follows:

"Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefore, subject to the conditions and requirements of this title."

Software programs cannot be defined as machine, manufacture or composition of matter. Therefore, they fall under the category of new and useful processes or new or useful improvements over existing processes for the purposes of requesting patents.

The first challenge to the validity of software programs as patents was captured in *Gottschalk* v. *Benson*.[13] This 1972 case involved a claim for a patent on a method of programming a general purpose digital computer to convert signals from Binary Coded Decimal (BCD) form to pure binary form. The claims were not limited to any particular art or technology, to any particular apparatus or machinery or to any particular end use. In deciding the case, the Court introduced a benchmark test for evaluating the eligibility of a process patent claim, now known as the "Machine or Transformation Test". The Court ruled that:

"…a process patent must either be tied to a particular machine or apparatus or must operate to change articles or materials to a different state or thing."

The Court ruled that the mathematical formula involved in the patent claim had no practical application except in connection with a digital computer. This meant that if the patent is granted it would wholly pre-empt the mathematical formula and in practical effect would be a patent on the algorithm itself. The Court defined "algorithm" as a "procedure for solving a given type of mathematical problem," and such an algorithm is like a law of nature which cannot be the subject of a patent.

This decision generated serious implications for patenting compute software. The decision that algorithms cannot be patented struck at the very definition of computer software, thereby rendering any invention which involved only a computer program and no machinery or transformation, ineligible for patent under the statutes. The decision also gave rise to subjectivity in evaluation of software patents as terms like transformation, articles and state change were not explicitly defined. Since the decision of

13. *Supra n.* 1.

Benson case implied that articles had to be physical objects, it created severe restrictions in accepting transformation of electrical signals by software programs as eligible under the statutes for patents. Therefore, the Benson case loaded the patent regime of the U.S. against granting patents for software programs.

The categorical acceptance of patent claims involving software programs was provided by the 1981 case *Diamond* v. *Diehr*.[14] The case involved determination of whether a process for curing synthetic rubber which includes in several of its steps the use of mathematical formula and a programmed digital computer is patentable subject matter under 35 U.S.C. § 101. The patent examiner invoked the Benson judgement stating that involvement of a computer program in a patent claim constituted non-statutory subject matter. The Court ruled that a physical or chemical process for moulding with precision synthetic rubber falls within the 35 U.S.C § 101 categories of patentable subject matter. Thus a subject matter otherwise statutory does not become non-statutory simply because it uses a mathematical formula, computer program or digital computer. The Court noted that the Benson decision does not preclude a patent for any program servicing a computer. This decision relaxed the constraints on filing patent claims which involved a combination of computer software and physical phenomena. This opened the doors for filing patents in the embedded systems domain but still precluded a large segment of software industry inventions from being eligible for patents. In the absence of clear definition of articles and transformation, computer software which did not involve any physical transformation was still considered as non-statutory subject matter under 35 U.S.C. § 101.

The expansion of statutory subject matter to include computer software which did not involve any physical transformation of an article was achieved in 1998 vide the United States Court of Appeals, Federal Circuit decision in the case of *State Street Bank* v. *Signature Financial Corp*.[15] The case involved the eligibility of a patent entitled "Data Processing System for Hub and Spoke Financial Services Configuration" under 35 U.S.C. § 101. The Court ruled that a process patent which involved no physical transformation of an article was a statutory subject matter under 35 U.S.C § 101 if it produces a "useful, concrete and tangible result." The impact of this decision was to bring under the gambit of statutory subject matter a broad spectrum of software and business method inventions which

14. *Supra n.* 2.
15. *State Street Bank* v. *Signature Financial Corp*, 149 F. 3d 1368 (1998).

were previously considered as non-statutory. Expectedly this decision was a major impetus behind the boom in software and business method patents in the late 90s and early 2000. Coincidentally this time period also overlapped with the rapid growth in the computer software and internet industry.

The honeymoon period for software patents however was short-lived. In a 2008 case of *In re Bernard L. Bilski and Rand A. Warsaw*,[16] the United States Court of Appeals, Federal Circuit reiterated that the "machine-or-transformation test" introduced in the Benson verdict as the applicable test for statutory matter under 35 U.S.C § 101 and stated that the "useful, concrete and tangible" test in State Street Bank verdict[17] should no longer be relied upon. Though the Court conceded that "…the more challenging process claims of the twenty-first century are seldom so clearly limited in scope as the highly specific, plainly corporeal industrial manufacturing process of Diehr; nor are they typically as broadly claimed or purely abstract and mathematical as the algorithm of Benson.," it also established the primacy of the "machine-or-transformation test" to determine whether a process claim is tailored narrowly enough to encompass only a particular application of a fundamental principle rather than to pre-empt the principle itself. The Court also commented that "…future developments in technology and the sciences may present difficult challenges to the machine-or-transformation test, just as the widespread use of computers and the advent of the Internet has begun to challenge it in the past decade. Thus, we recognize that the Supreme Court may ultimately decide to alter or perhaps even set aside this test to accommodate emerging technologies. And we certainly do not rule out the possibility that this Court may in the future refine or augment the test or how it is applied." While this comment kept the possibility of refinement or rejection of the "machine-or-transformation test" in the longer term, in the short term it firmly shut the door on the wider interpretation of statutory matter to include software patents of all kind.

Thus, over a period of more than four decades the attitude of the U.S. Patent Regime has varied from conservative (Benson), to categorical acceptance (Diehr), to extremely liberal (State Street Bank) and finally back to the conservative stance but with the possibility of a liberal stand in future in acknowledgement of developments in the field of computer and internet technologies. However this vacillation in attitude and jurisprudence of the Patent Regime towards patents involving digital computer processes

16. *In re Bernard L. Bilksi and Rand A. Warsaw*, 545 F. 3d 943, 88 USPQ 2d. 1385 (2008) (en banc).
17. *Supra* n. 15.

has raised serious questions and challenges for the Computer Software Industry.

The Software Patent Conundrum: Challenges for the Software Industry

The challenges that the computer software industry has faced in filing of patents for its inventions has been on two axes:

1. Establishment of the invention as statutory subject matter;[18] and

2. Demonstration of novelty[19] and non-obviousness.[20]

The origin of the first challenge has been on account of changing definition of statutory subject matter based on the various judgements made by the U.S. Court s from time to time. This has impacted the manner in which software patents have been drafted, inconsistency in the type of patents granted and lack of clarity when challenging the eligibility of patents.[21]

The origin of the second challenge has been the inability of the US Patent and Trademark Office (USPTO) in consistently applying the novelty and non-obviousness tests on patent applications for computer software. This has resulted in the grant of some poor quality patents which have generated barriers to innovation in the industry[22] and consequently development of strong resistance within the industry towards the granting of patents for software patents. The impact of both challenges is discussed in this section.

Establishment of the invention as statutory subject matter (35 U.S.C § 101)

The Benson judgement by U.S. Supreme Court established the "Machine-or-Transformation" test as the benchmark for evaluating the statutory eligibility of process patents. However the Benson Court's inability to provide a clear definition of terms like transformation, article and state change has introduced significant amount of subjectivity in interpreting process patent applications. The cognizance and understanding of the test in the software community is quite low. In fact, the first time software

18. 35 United States Code S. 101.
19. 35 United States Code S. 102.
20. 35 United States Code S. 103.
21. See Federal Trade Commission, "To Promote Innovation: The Proper Balance of Competition and Patent Law and Policy" (2003), p. 163, available at www.ftc.gov/os/2003/10/innovationrpt.pdf , last visited Apr 1, 2010.
22. *Supra n.* 18.

engineers encounter this test is when they discuss their patent application with their attorney. At a very high level all software implements an algorithm but the Benson judgement does not allow patenting of algorithms. But the overwhelming requirement for protecting their invention forces inventors to obfuscate the language of the patent which as such is not apparent from the application that the patent is sought on an algorithm. Richard Stallman describes such a patent: A software patent for application of "topological-sort" algorithm for carrying out recalculation in spreadsheets was filed as "compiling formulas into object code,"[23] thereby making it difficult for a developer of spreadsheet software to be familiar with inventions in his related field. Another example of invention obfuscation is the patent for reverse auction where buyers announce their price and sellers are allowed to bid in response. This patent held by Walker Asset Management is described as "methods and apparatus for a cryptographically assisted commercial network system designed to facilitate buyer-driven conditional purchase offers.[24]" This practice of obfuscation is extremely worrisome for the industry as it potentially broadens the scope of protection than what was actually sought and at the same time it makes detection of infringement (wilful or innocent) very difficult. Additionally, it also raises the cost of drafting as well as protecting the patents as special legal skills are required to obtain software patents. The cost factor may exclude smaller corporations or start-ups from appropriately protecting their inventions.

Another key aspect of the Benson judgement was that the Court determined that the patent claim was abstract and sweeping as to cover both known and unknown uses of the invention. This was found to be tantamount to protection of basic tools of scientific and technological work. Therefore, a vital guideline to applicants of software patents must be that their claims must not be overtly broad in the protection that they seek. Many a times this criterion creates a paradox for software inventors. Sometimes in software industry a lot of R&D effort and investment is directed towards the development of an idea. The application of that idea to various domains might be obvious but the novelty is in the idea itself. For example Monte Carlo Simulation[25] is applicable in numerous fields like

23. Richard Stallman, "The Dangers of Software Patents", Speech by Richard Stallman at Cambridge University, March 25, 2002, p. 3, available at http://www.ftc.gov/os/comments/intelpropertycomments/stallmanrichard.pdf, last visited Apr 1, 2010.
24. Edward J. Black,"Competition and Intellectual Property Law and Policy in the Knowledge-Based Economy", March 20, 2002, p. 5, available at http://www.ftc.gov/opp/intellect/020320black.pdf , last visited Apr 1, 2010.
25. N. Metropolis and S Ulam, "The Monte Carlo method", Journal of the American Statistical Association, vol. 44, 1949, pp. 335-341.

physics, mathematics, biology, semiconductors, finance etc. The applications of Monte Carlo Simulations are obvious but the real intelligence is involved in developing the method. Hence it may be desirable for an inventor to protect the idea rather than its application. In such cases filing a patent for all possible applications may not be feasible and protecting a specific application may lead to disclosure of the algorithm which may be counterproductive to protection. Hence the paradox is that if a patent is not applied for the invention is unprotected but at the same time the act of patenting the invention would lead to loss of protection. This paradox is responsible for various inventions in the software domain remaining in private domain and society is unable to benefit from the use of such inventions.

Demonstration of novelty (35 U.S.C § 102) and non-obviousness (35 U.S.C § 103)

The software industry is a rapidly changing industry. The rate of innovation in the industry is high and is one of the most important factors in ensuring commercial success. The State Street Bank decision resulted in increasing the number of software inventions which were eligible for patents. The impact of this judgement was the deluge of patent applications related to software and business method at the USPTO. As a result the quality of patents granted has been impacted. The increase in number of questionable patent grants combined with the legal and financial difficulties in challenging of patents has impacted innovation in the software industry and has created a strong lobby of vocal critics of software patents within the industry.

In an October 2003 Federal Trade Commission (FTC) report[26] on patent regime's impact on innovation, several participants from the Software Industry acknowledged the existence of questionable patents in the industry. They also mentioned in the report that subjective and ambiguous process of construing claims makes avoiding infringement uncertain and deters innovation. They further identified the failure of the USPTO to examine all prior art (35 U.S.C § 102) and as a consequence grant of patents that are overtly broad or obvious (35 U.S.C. §103). A few key factors determined as the cause of the USPTO to inaccurately examine all prior art were: (1) the rapidly changing and complex nature of software and internet industries; (2) the absence of legal requirement for patent applicants to disclose source code; and (3) the relatively recent recognition of the validity of business method patents by the Courts.

26. *Supra n.* 21.

The record of USPTO in determining obviousness of the patent claim has also been questionable. Open Source Software pioneer Richard Stallman disparagingly stated that 90% of the software patents issued in the U.S. would not have passed the "Crystal City test", which meant that if the USPTO went outside to the news stand and got some computer magazines, they would see that these ideas are already known.[27] The sarcasm of Richard Stallman is not without merit. In September 1999, USPTO issued a patent to Amazon.com for "one click purchasing,"[28] which allowed customers to purchase items by just clicking once on an icon, permitting the website to access pertinent personal information previously stored in their database. It is hard to believe that such a concept was not obvious to a person having ordinary skill in the field of e-commerce. In fact barnesandnobles.com already had a similar feature on their website and was sued by Amazon.com for infringement within 3 weeks of obtaining the patent.

It would be appropriate not to pass judgement on the competence without acknowledging the challenges they face in carrying out due diligence. The technique of obfuscation used by inventors when drafting patent applications makes it difficult for the patent examiner to determine the right area for initiating prior art search. Additionally, various federal Court decisions like *Northern Telecom* v. *Datapoint Corp.*[29] and *Fonar* v. *General Electric*[30] have allowed filing of software patents without any obligation to disclose the source-code. This further complicates the determination of prior art. The software industry evolves rapidly and combined with an instant delivery via the World Wide Web (www) new technologies are adopted almost right after development. Thus the boundary of obviousness is being tested everyday and hence to make an objective call on the obviousness of a patent application is more difficult that it seems to seasoned and up-to-date software professionals.

Regardless of the root cause, the inability of USPTO to rigorously apply the criteria of novelty (35 U.S.C § 102) and obviousness (35 U.S.C. § 103) has resulted in patent grants which are having a negative impact on innovation in the software industry. In the rapidly changing software industry the 20 year validity of a questionable patent could severely impact innovation

27. *Supra* n. 20.
28. United States Patent 5960411,"Method and system for placing a purchase order via a communications network", available at http://patft.uspto.gov/netacgi/nph-Parser?patent number=5960411, visited April 2, 2010.
29. *Northern Telecom Inc.* v. *Datapoint Corp.*, 908 F. 2d 931, 15 USPQ 2d. 1321 (1990).
30. *Fonar Corporation* v. *General Electric Company*, 107 F. 3d 1543 (1997).

in the industry. The lack of effective mechanisms for third party challenges to patents compounds the harm to innovation caused by questionable patents.

Unique Opportunity for the Indian Legal System

The Indian legal system currently does not allow patents on computer software. The Indian Patents Act 1970 (as amended in 2002) explicitly defines computer software as non-statutory subject matter for obtaining patents. In chapter II of the act which defines "what are not inventions," section (k) states "a mathematical or business method or computer program per se or algorithms." However upon careful inspection of the act it can be determined that the Indian interpretation of non-statutory subject matter is similar to categorical acceptance of the *Diamond* v. *Diehr*[31] decision rather than the strict exclusion as determined by the *Gottschalk* v. *Benson*.[32] However the interpretation of Indian patent law is moot as Indian software companies have traditionally sought protection under the U.S. Patent system due to the demands of their business and interest in obtaining Indian patents has been negligible. This trend could however change on account of rapid growth in the Indian market and recession in the US market. Therefore, there is a unique opportunity for the Indian Legal System to steal a march over other international legal systems in developing the jurisprudence for dealing with the conundrum of patenting a broader spectrum of software patents as per the relaxed application of the State Street Bank decision.[33] Such reform of our patents law is in our national interests for two reasons. Firstly, it protects and promotes the progress of our burgeoning software industry which is not only a top foreign exchange earner but also a significant employer for the skilled population of India. Secondly, such a reform would generate unique growth opportunities for the nation by forming a symbiotic relationship between our highly skilled and experienced legal community and our globally focussed Computer Software industry.

The absence of interest of Indian software companies in India should not be seen as a deterrent to reform. In fact, the rigid stance of our patent laws in matters of Computer Software and the lack of interest of software industry in seeking Indian patents is actually a blessing in disguise for initiating meaningful reform. The reform of the U.S. patent regime is complicated by the fact that there are already several existing software patents granted at various times under different definitions of statutory eligibility. Hence reform in either direction (exclusion or inclusion) cannot be carried out as

31. *Supra n.* 2.
32. *Supra n.* 1.
33. *Supra n.* 14.

it impacts several existing patents in favourable or unfavourable ways. Thus any meaningful reform would be tied down by significant litigation activity by either existing patent holders or unsuccessful patent applicants. Indian legal regime can however initiate reform conveniently as it has a clean slate due to blanket refusal of software patents. We can benefit from the existing legal jurisprudence already available on this matter in various other jurisdictions for example U.S. and EU. Additionally we have a globally focussed Computer Software industry which can provide intelligent inputs on this issue based on their technical and business expertise. This unique amalgamation of factors could enable our legal scholars and practitioners to develop sophisticated and innovative jurisprudence which can address the requirements of protecting the inventions of 21st century. This would allow us to evolve from a leader in IT technology to a pioneer in offering a complete IT ecosystem.

Unravelling the Conundrum: Questions for Legal Community

The investigation of the judgements in some landmark cases in the U.S. Court's enunciate clearly the need for reform in the Patent Regime in order to address the challenges faced by Computer Software industry in obtaining patents for their inventions. The unique legal and technological environment in India puts us in an excellent position for analysing these challenges and initiating suitable reform.

The most pronounced requirement for reform is in the area of defining the statutory eligibility of software patents. This issue does not have a binary yes or no answer as an overly broad or restrictive definition of eligibility would have consequences at both ends of the spectrum. A highly restrictive criterion for eligibility as recommended by the Benson judgement would render a large cross-section of the industry incapable of effectively protecting its inventions. A liberal criterion for eligibility in context of the rapid growth and unique nature of software inventions can very easily overwhelm the system resulting in insufficient scrutiny before granting of patents. This can have an exactly opposite impact on innovation than intended by the law. A fine balance between the two approaches is required and it would require excellent synergy between legal and software communities to come up with right answer.

It is important for the legal community to address the paradox that software inventors face when deliberating the disclosure of their inventions to society. Though the patent regime does not allow protection of abstract ideas, sometimes it is the abstract idea which is the actual intellectual property. This is a profound question where there are strong arguments

which can be put forth by both sides. Is there an innovative legal solution to this conundrum?

Finally, there is requirement for reform in the area of balancing the rights of the IP owners and society. Just as the lack of proper incentive from society can deter the inventor from disclosing his work to society, the absence of appropriate rights to society against inventors who stand in the way of innovation can deter society from granting such exclusive rights thereto. The instances of questionable patent grants and their hindrance to innovation make a strong case for visiting this issue.