The Software Patent Thicket: A Question of Disclosure[1]

Rosa Maria Ballardini[2]

Abstract

The purpose of this article is to propose that more extensive disclosure in patenting of software inhibits the growth of patent thickets. The abstractness of software patents claims being one of the main reasons behind the proliferation of overly broad, obvious patents in the field, a more detailed description of the program is a necessary requirement to be added to the description of the invention.

In incremental component industries like computer software, where new developments necessarily build upon existing technologies and innovation occurs through small improvements rather than real breakthroughs, inventions are usually highly complex and often protected by multiple intellectual property rights. As a consequence numerous licenses are required to produce even a single commercial product. Furthermore, in the software field, the vast proliferation of patents in a relatively short period of time, together with the lack of relevant prior art and, in particular, the special nature of software as an abstract technology, has led to non-novel, obvious, and too broad patents being issued. The Blackboard patent no. 6,988,138 recently litigated in the US (Blackboard v. Desire2Learn) represents but one of the many examples available. This horizontal dense overlapping of multiple patent claims is termed "patent thicket"[3].

Specifically, patent thickets refer to two different types of concerns: "overlapping" problems, related to the "quality" of the patents, on the one hand, and problems related to the vast amount of patents issued, on the other. The former set of problems is the direct cause of the latter ones. Both concerns might appear during the life-span of software patents, although at different stages.

This article pursues the subject, first by investigating reasons and consequences of the software patent thicket. Both the American and European perspectives are considered. In particular, the US software patent landscape serves as a background for the discussion on the European situation. Surprisingly, in fact, while awareness over the negative effects of the software patent thicket is currently arising in the US, with the PTO recently embracing a more restrictive approach, the EPO seems, instead, to be following an opposite track, by increasingly relax the requirements for the granting of such patents.

This analysis then leads to a "software patents paradox": while the amount of software patents applications keeps on growing exponentially, the strong thicket affecting the field has radically reduced their private and social value. Various business strategic

---

[2] Doctoral candidate, HANKEN- Swedish School of Econ. & Buss. Adm., Helsinki, Finland. Email: rosamaria.ballardini@hanken.fi.

[3] C. Shapiro, 2000, "Navigating the Patent Thicket: Cross Licenses, Patent Pools, and Standard–Setting" (March 2001). Available at SSRN: http://ssrn.com/abstract:=273550 or DOI: 10.2139/ssrn.273550.

reasons are said to lie behind this trend. However, whether such a behavior is justified from a patent law point of view is highly questionable. The fact that patents are of limited value to their holders casts serious doubts over their importance to the society. In this way, software patents might collude with the essence of patent law as a tool to afford temporary private monopolies so as to provide innovation incentives for the overall benefit of society.

Building on these bases, the article argues that the time has come to take a step back and look at the problem again from its roots as to avoid proposing of wrong policy recommendations and inefficient solutions. In particular, the article suggests that patent thickets concerns in the computer programming field all converge around the ´abstract´ nature of the software patents claims. Any proposal for reform should then stem from this point.

Addressing the problem from a "patentable subject matters" perspective seems highly undesirable. Patent officers and legislators have been trying unsuccessfully to deal with this question for the past thirty years. This is particularly evident in Europe, where the exclusion of computer programs "as such" from the patentability field has taken the EPO Boards of Appeal on a crusade of trying to draw up the boundary between patentable and non-patentable objects in software. In the US some recent cases such as the currently pending *en banc* Bilski appeal and, in particular, the *ex parte* Langemyr and *ex parte* Wasynczuk PTO Board of Patent Appeals and Interferences' decisions represent but another demonstration of the difficulty to define patentable objects in software.

Efforts should, instead, focus on increasing disclosure in the applications. In this respect, would-be patentees should be required to provide not only the description of the invention, but also the source code (or part of it), flowcharts and, generally, a more detailed description of the program. This would succeed in making the claims "less abstract". When applying for a patent, in fact, one does not necessarily need to have produced any line of source code. A source-code-disclosure policy would, on the one hand, constitute a "prove" that the applicant have an invention and thus slow down the filing of very uncertain applications and, on the other hand, would make the claims less abstract by tightening patent protection to the function produced by the mentioned source code. Source codes can be considered as the "genes" of software and a disclosure policy be built on the line of the escrow of biological samples in biotechnology patents. Overall this would help in delimiting the software patents fuzzy ´boundaries´ and would, consequently, reduce the thicket.

A source-code-disclosure policy might be essential not only for patent officers to evaluate the sufficiency of disclosure of the applications, but also to support judges in dealing with questions of infringement. Software inventions, in fact, do not often transpire "from the face" of the patents.[4] Reverse engineering, a tool traditionally used for accessing technical information not apparent in patents, might fail in the computer programming field, due to both legal (at least under the American jurisdiction)[5] and

---

[4] Pamela Samuelson, Randall Davis, Mitchell D. Kapor, & J.H. Reichman, *A Manifesto Concerning the Legal Protection of Computer Programs*, 94 Colum. L. Rev. 2308, 2314 (1994).

[5] Lemley, Mark A., Cohen, Julie E., "Patent Scope and Innovation in the Software Industry", California Law Review, Vol. 89, p. 1, 2001. Available at SSRN: http//ssrn.com/abstract=282790 or DOI: 10.2139/ssrn.282790.

technical reasons. Also competitors and new inventors, who necessarily need full access to the patented program in order to improve upon it, would clearly benefit from such a practice.

Pretending to solve the overall highly complex problem would be very presumptuous. The many issues involved, in fact, undoubtedly call for different solutions that need to be taken on various fronts. However, by looking at this subject from a perspective focused on the disclosure requirement, the author hopes to make a positive contribution to the discussion in the area.

Key Words: Software patents; Patent thickets; Comparative analysis- Europe and the United States