Intellectual Property Protection for Computer Programs: Developments, Challenges and Pressure for Change

by

Rosa Maria Ballardini *

* Assistant Professor of IP Law Hanken School of Economics

Nordic Journal of Commercial Law
Issue 2013#2

1 Introduction

This book is a study of how computer programs have challenged the thinking about and the actual use of intellectual property rights (IPRs) around the world. In general, the intellectual property (IP) system is governed by the same rules and applies equally to all fields of developments. However, the particular nature of computer software has challenged these fundamentals.

Software is a pluralistic product that contains several elements, each of which could fall into different categories of IP laws. Computer programs can be defined as "a combination of computer instructions and data definitions that enable computer hardware to perform computational or control functions"[1]. IP protection applies differently depending on the manner in which those instructions and definitions are expressed. Currently, several protection mechanisms are available for software, including copyright, patents, trademarks, contracts, licensing agreements, and technical measures of protection. It has been suggested that none of these mechanisms, if used individually, successfully provide an adequate level of protection to computer programs.

Software technology started to appear during the 1950s, when the first computer programs were developed. As soon as it became evident that computer software was a highly complex technology that required large monetary investments, both a wide market for software and significant potential for monetary rewards were envisaged. Debates regarding the adequacy of the existing protection mechanisms for software were then initiated. It became immediately clear that each of the existing IP protection mechanisms possessed certain limitations when applied to software. These problems have generated a global debate regarding the extent to which software should be afforded IP protection. Several perspectives have been presented, and a large variety of solutions have been proposed. Some of the proposals embrace rather extremist views. For example, some suggest the complete abolition of IP protection for software in favour of an IP-free regime or the development of a completely new protection mechanism tailored to the special needs of software. In contrast, others see the IP system as playing an extremely important role in securing investments and ensuring progress in the software field and thus advocate a strong level of protection. In addition, several more balanced proposals have been launched, many of which have attempted to shape the existing IP rules such that they meet the needs of computer software. This thesis effectively contributes to this debate by providing a number of balanced, reasonable, and feasible answers to some of the major issues in the software intellectual property ecosystem. The proposed solutions were generated within the framework of the currently existing body of IP laws.

[1] IEEE Std 610.12-1990, Standard Glossary of Software Engineering Terminology (1990), New York, *The Institute of Electrical and Electronics Engineers*, at 66.

2 Structure of the book

This dissertation comprises an introductory chapter, discussing the background, the research aims, the methods implied, and the research results achieved, and four original essays. The four essays cover a relatively wide range of themes, the common denominator being the process of adapting intellectual property rights to computer software.

Regarding copyright law, this thesis provides new perspectives on the protection of the functional elements of computer programs in the European context in *Essay I*.

Copyright is a globally accepted mechanism for protecting computer programs.[2] Copyright can adequately protect certain aspects of software (namely, the code itself). However, the principles upon which the copyright system is based create several issues with software copyright. Among these issues, the most discussed (but unsolved) problem is most likely that copyright generally does not extend to utilitarian products and thus cannot provide protection to the software's functions (the "idea-expression" dichotomy). Other shortcomings include the contrast between the long duration of copyright protection and the short lifespan of software as well as the challenges inherent in determining the "originality" of the code. New technological developments have generated other problems, such as the difficulties created by the increased use of modularisation and object-oriented designs in computer programming[3].

With regard to copyright, this thesis focuses on the issues related to the "idea-expression" dichotomy by proposing a workable model that could be used in European jurisdictions. This book considers the related developments that have occurred in the United States. However, the decision to propose solutions for Europe was driven by the absence of a consistent interpretative framework for addressing these issues in Europe (as the relevant American jurisprudence is relatively extensive).

The primary problem is the pluralistic nature of software, which is simultaneously literary and functional. Computer programs are unique in that the software functions are fully independent from the literary lines of code. In other words, even though the source codes of two programs might look completely different, the codes can perform the exact same function and produce the same (or similar) instructions. This means that with computer programs the expression is often an integral part of the idea itself such that the two are inseparable and, therefore, drawing the line between the literary parts of programs and the mechanical ones has been extremely challenging. On the other hand, however, such a distinction is important for the purposes of IP protection because of the different scope of copyright and patents, namely the fact that

[2] See, for examples, the WIPO Treaty on Trade Related Aspects of Intellectual Property Rights, Marocco, 15 April 1994 (TRIPs).

[3] Lipton J, "IP's Problem Child: Shifting the Paradigms for Software Protection" (2006), 58 *Hasting Law Journal* 2, at 205-251.

copyright protects the *expression* of ideas, while patents the *implementation* of such ideas – so called "idea-expression dichotomy" doctrine in copyright law.

On these bases, the paper identifies two specific sets of problems inherent to software copyright:

- Risks of *under-protection*, because copyright does not protect the way that the program works.
- Risks of *over-protection*, due to the difficulty of drawing boundaries between the literary and functional aspects of software, which has lead copyright to be occasionally extended to functions, protecting software in a "patent-like" manner.

Essay I argues that these risks are particularly problematic in Europe, especially due to the traditional reluctance to afford patent protection to computer programs under the dictate of Article 52(2)(3) of the European Patent Convention and, accordingly, the solution is sought not only within the framework of the existing copyright laws, but also based on patent rules. The paper identifies the more pressing questions on the copyright and patent scope of protection for software and suggests that two specific rules should be considered for the European framework: first, the article advocates the adoption of explicit statutory limitations against non-literal infringement to keep the scope of software copyright within the sphere of literary works of art; and, second, the paper suggests that keeping software within the umbrella of patent law is essential in order to impede copyright from expanding to protect the functional elements of software in a "patent-like" manner. Indeed, some recent developments occurred in the legislative side, like the recently passed *SAS Institute v World Programming Ltd.* decision by the European Court of Justice, have confirmed the importance and validity of the issues discussed and the solutions proposed in *Essay I* of this dissertation.

The patent discourse is conducted in *Essay II* and *Essay III*, which discuss different problems related with the application of patent law to software. Overall, the discussion of the software patent framework in this thesis was triggered by the "crisis of the patent system"[4] (especially in the United States), which many claim was caused by the extension of patent protection to computer software.[5] Indeed, sings exist both to support and to defeat this claim. On the one hand, the question of patentable subject matter in computer programs continues to appear in legal disputes and has not been resolved. Some highly controversial software patents that have been granted have indicated the lack of novelty and inventiveness of these types of patents in general. The relatively low threshold for disclosure required for patents in the software field (e.g., there is no need to disclose the source code) is often controversial. Additionally, "patent thickets" are said to dominate the software patents sector. The software industry sees the patent

[4] Burk D, and Lemley M, *The Patent Crisis and How The Courts Can Solve It* (2009), The University of Chicago Press.

[5] Bessen J, and Meurer M, Patent Failure. How Judges, Bureaucrats, and Lawyers Put Innovators at Risk (2008), Princeton University Press.

system mostly as a cost rather than as an incentive for innovation, and companies operating in the field often claim to use their patents mainly for defensive purposes rather than to secure protection for their inventions. This information seems to suggest that the system is fundamentally flawed. However, given that the system has not yet kept progress from occurring in the field, it seems that methods of addressing these issues (regardless of their legitimacy) have emerged.

This thesis provides a richer understanding of the reasons for and consequences of extending patent protection to software. Overall, the analysis sheds light on the most challenging legal questions associated with the software patent framework and provides a solid foundation for proposing suitable solutions. With regard to patent protection, this thesis mainly proposes solutions tailored to Europe because of the background and personal interests of the researcher and because relatively few systematic studies on software patent protection exist within the European legal regime. Indeed, U.S. approaches, case law and literature have been widely analysed.

Essay II attempts to answer the question of whether the traditional European approach to computer-implemented inventions (CII) patents, which states that computer programs "as such" do not deserve patent protection because they do not have "technical" character and do not make a "technical" contribution, should be considered obsolete and thus be overruled. The study investigates and provides evidence of the discrepancies existing in the interpretation of the "technical" requirement for CII patents both at the EPO and at the national level, and it shows that a general lack of legal consensus exists regarding CII patentability. Furthermore, it argues that this disharmony might overall distort the European market and impede economic growth in the software field. *Essay II* concludes that the traditional European system, where only "technical" inventions receive patent status, has become obsolete under current rules and, accordingly, it formulates a more workable approach to enhance the clarity of Art. 52's dictate in the field of computer programs. Namely, the paper suggests that the traditional concept of the "inventive step" should be reformulated and that no elimination of non-technical claim features should be attempted.

Essay III provides suggestions for how to interpret the patent law requirements in such a way to eliminate efficiently or reduce substantially the problems associated with "patent thickets" in the field of software within the European system. Patent thickets are caused by overlapping patent rights, on the one hand, and the large volume of patents being issued, on the other. Focusing on the reasons for overlapping patent rights, *Essay III* finds that overlapping software patents are caused by *the lack of relevant prior art in the field* and *the abstractness of software patent claims*. However, although several studies have been conducted and initiatives have been put into place to improve the prior art repositories, the abstractness of software patent claims has not been sufficiently considered. However, the study demonstrates that abstraction is actually one of the major causes of the growing software patent thicket and, as such, it represents a fundamental issue to be addressed. In particular, the paper suggests that the more abstract the claims, the more difficult to evaluate the concrete applicability of the inventions, and the more interpretation is required to assess the question of patentability. Therefore, extensive disclosure

is highly necessary in software patent applications in order to facilitate the process. Accordingly, the article argues in favour of enhancing disclosure for computer program patents in order to reduce the abstract nature of the software patents claims, better define the boundaries between patentable inventions and prior art, help narrowing the scope of the patents granted, and overall reduce the number of applications filed, this way smoothing the software patent thicket.

Finally, the discourse on software copyright and patents is contextualized within the framework of the free *libre* and open source software in *Essay IV*. Open source has become an integral part of the legal environment for software. Therefore, to provide a complete overview of the challenges associated with IP and software, it is necessary to properly consider this phenomenon. Essay IV aims to deepen understandings of the dynamics governing the simultaneous use of copyright, patents, and open source in a commercial context in order to expose the legal risks associated with this configuration and highlight the coping mechanisms that can be implemented in order to navigate these risks. The study specifically focuses on the problems that proprietary companies may encounter if they incorporate both open source and IP protected code in the final proprietary software that they release to the market. Thus far, the research has independently focused on either open source as a phenomenon or proprietary software, while few studies have investigated the relationship and interactions between the two. This luck of readily available information and literature material, as well as the very little Court cases handed down on the matter justified the use of an empirical study under the form of a case study research (CSR) method. Indeed, the case study was conducted with the intention of generating new knowledge rather than answering one, or few specific questions and, therefore, the most valuable contribution of the study lies in the new theoretical and practical knowledge that it presents, rather than in having provided specific solutions to the problems investigated.

3 Conclusions

This doctoral dissertation was written with four objectives in mind:

1. to improve understanding of the problems associated with applying intellectual property laws (copyright and patents) to computer programs;
2. to study how copyright and patents interact within each other's framework and within the overall IP framework when applied to computer programs (to introduce a new perspective to the overall discussion);
3. to propose practical, balanced and more workable solutions to the problems within the copyright and patent software frameworks in the European jurisdiction;
4. to study whether the legal risks associated with the *hybrid* protection models are problematic in practice, to illuminate the causes of the issues at stake, and to assess which coping mechanisms could be used to alleviate these problems.

This thesis has various theoretical and practical implications. The importance of computer programs around the world is undisputable. Currently, software can be found everywhere. However, software is still being developed. Thus, clear legal rules regarding software protection

are more necessary than ever. The boundaries of intellectual property protection for computer programs remain murky in several respects. As explained earlier, in fact, software has been remarkably difficult to classify within a specific category of intellectual property. The issue has been discussed for more than thirty years, and the fact that recent developments have not yielded a solution shows the complexity of the debate and the problems involved.

This thesis has engaged some of the most controversial aspects of the debate surrounding the application of the most widely used IP protection mechanisms for software: copyright and patents. The end result is a holistic study that actively and substantially contributes to the academic discussion of software IP protection. The findings are practical, readily available tools that, if implemented, might enhance clarity and legal security in this important field of technology. The results of the study could generally serve scholars, lawyers, software companies, and policymakers.