

# IP AND SOFTWARE

The WIPO Latin America and Caribbean (LAC) Regional Seminar on Intellectual Property and Software in the 21<sup>st</sup> Century: Trends, Issues, Prospects took place in San José, Costa Rica, on August 19 and 20 with the participation of ten governments in the region and over 200 representatives of the software industry. The Seminar provided crucial information to demonstrate the role of software in economic development for countries in Latin America. It covered four themes: IP rights protection of software and its relation to economic development; business models and licensing in the software industry; development of standards in the software field; and the role of public authorities and private companies in software development.

The Seminar highlighted the relevance of, and relationship between, IP rights and software to the governments and software industry of the LAC region. Software procurement and development decisions, standards-development policies, tele-communications and information and communication technology policies are all affected by how IP rights in software are provided, licensed and enforced. This relationship is horizontal from an IP perspective, cutting across both patent (patenting of software) and copyright questions (licensing, particularly open-source). This sector-specific approach to the issues stimulated intense interaction among public authorities and the software industry representatives. The dynamism of the local software industry also served as a catalyst in showcasing rich practical experiences in the issues under debate.

WIPO Magazine invited Mr. **ANDRÉS GUADAMUZ GONZÁLEZ** to develop the topics under discussion at the Seminar. Mr. Guadamuz, a lecturer at the SCRIPT Law and Technology Centre, University of Edinburgh, is an expert on IP and software, and has developed interesting analyses on issues including software patents, open source software licenses and the interplay between proprietary and open source software. In this article Mr. Guadamuz focuses on the issue of software patents, the topic of his presentation in the Costa Rica Seminar.



## Software Patentability: Emerging Legal Issues

Andrés Guadamuz González

There is little doubt that the software industry is still one of the powerhouses of the global economy. Despite the recent financial downturn, global worldwide spending on software amounted to some US\$257 billion in 2007. Given its economic importance, it is clear that any discussion about the legal protection awarded to software is of the utmost interest to producers, consumers and all economies that share, or want to share, in the growing demand for computer programs.

Software has been remarkably difficult to classify as a specific form of IP subject matter because its dual nature presents particular difficulties for those trying to draw analogies with existing legal categories. This is why there have been attempts to classify it as subject to copyright, patents or trade secrets and even to a *sui generis* software right. It is indicative of the complexity of the debate that it has gone on for more than 20 years, and, if the recent interest in the topic is anything to go by, will continue for years to come.

But what makes the legal classification of software so difficult? The problem may lie in the fact that software is not a monolithic work but that it has several elements that could fall into different types of IP protection. If we define software as a set of instructions to a computer that bring about a certain result, then the manner in which those



instructions are expressed should give us an idea about the type of IP protection that applies. These instructions are initially expressed as source code – lines of instructions in a computer language. As source code is expressed in written form, it is therefore logical to define software as being subject to copyright protection as a literary work. This is indeed the existing approach towards software protection in several international treaties. For example, Article 4 of the WIPO Copyright Treaty (WCT), Article 10 of the World Trade Organization's TRIPS Agreement<sup>1</sup> and section 3 of the European Council Directive 91/250/EEC on the Legal Protection of Computer Programs all define software as literary works subject to copyright protection.

However, software is not only the source code that operates in a computer; software has to be compiled into object code – machine-readable instructions that can be directly executed by the computer. This translation usually has no bearing on the type of protection awarded to software because the object code is a direct result of the source code, and one can argue that its legal status should be indistinguishable from that of source code.

The problem with strict classification of software as a literary work arises when one considers that computer programs have other elements that are not usually protected by copyright. Software is not only a literary expression; the lines of code have a function that is not dependent on their grammatical construction. The source code from a computer program can be completely different from that of another program, and yet have the same functionality to produce a similar set of instructions that achieve a similar result. This is at the heart of the idea/expression dichotomy that is often at the forefront of the software protection debate.

It is clear that copying large parts of source code and inserting them into another program constitutes copyright infringement. However, this type of infringement is relatively rare, and the real problem has become the protection of non-literary elements contained in software.

Is there copyright infringement for the copying of the functional aspects of a computer program? The answer has been a very complex and lengthy 'yes'. This is evidenced by the initial application of the idea/expression dichotomy to software and then by the inception and reliance on the (rather clunky) so-called *Abstraction-Filtration-Comparison*<sup>2</sup> doctrine

in the U.S., which has been both applied and criticized by the U.K. courts. Most recently, the case of the protection of the functional elements of computer software has been thrown open again in the U.K. with the case of *Navitaire v Easyjet*, where the High Court significantly diminished copyright protection of non-literal elements by finding that copyright protection should not be extended to the functional aspects contained in software.

## Patentability

It is precisely the difficulty with protecting non-literal elements of computer programs that has created the perceived need for the patentability of software. This is because patents are used to protect functional aspects of works. There is no idea/expression dichotomy in patent law. If an idea fulfils the requirements for patentability – it is a patentable subject matter, it is novel and involves an inventive step – then it will be awarded patent protection. American courts opened the door to the patentability of computer programs by allowing a patent for software that controlled manufacturing processes as early as 1981.

Similarly, Art 27.1 of the TRIPS Agreement states that patents should be awarded to all inventions irrespective of their field of technology. Subsequent cases have expanded patentability of software in the U.S. to what it is today. With the patent door open, and the seeming chaos in the copyright protection camp, it is not surprising that software companies rushed to get patents, resulting in an explosion of successful applications in the U.S. In 1986, the number of patents issued under classes usually deemed to be related to software amounted to 3,078. In 2006 alone, the United States Patent and Trademark Office (USPTO) issued 41,144 software patents, and the total issued by that year was 336,643.

The European Patent Office (EPO) has followed suit, and in a series of decisions of the Technical Board of Appeals, it has allowed limited patentability of computer-implemented inventions that involve a technical effect (or contribution, or process). These rulings have allowed for the limited patentability threshold to exist as long as the invention that is going to be implemented through a computer fulfils this requirement of technicality. It is well understood that the source code, or the literary and textual element of software, cannot be patented, but that if the software produces

<sup>1</sup> The Agreement on Trade-Related Aspects of Intellectual Property Rights

<sup>2</sup> Found in *Computer Associates International, Inc. v. Altai, Inc.*, (2nd Cir. 1992) 61 USLW 2434. In short, this test abstracts all the elements found in the computer program, filters out the not protectable ones and then compares what is left to search for similarities.



some sort of effect in the same way that an invention does, it will be awarded protection. The problem is that the precise definition of this technical effect or process has been very difficult to pinpoint in the period of over 20 years since the first ruling by the EPO came about.

Further cases have been toying with this distinction, but have often been muddled and/or contradictory. The bottom line is that the EPO has granted a large number of patents for computer-implemented inventions, some estimates claim that 30,000 patents had been issued by 2003. The EPO issued 8,981 patents classed under computing in 2007 alone.

The trend around the world has been increasingly to allow for the granting of software patents. Australia, Brazil, India and Japan allow for the granting of computer-implemented inventions in one way or another, which seems to indicate that most of the major patent-owning countries consent to the patentability of inventions through computer programs. It is expected that the number of nations granting software protection will grow in the future, and more patent examiners around the world will be presented with applications that describe some form of algorithm.

## Challenges for WIPO

It seems clear that as more and more software patents are issued by the largest patent-granting offices, there will be increasing calls for international harmonization on the topic. As mentioned above, international treaties do not really cover the issue of patentability of computer programs. The closest to a unified international approach is the inclusion of specific rules regarding computer programs in the Patent Cooperation Treaty (PCT).

Rule 39.1 of the PCT states that:

*“No International Searching Authority shall be required to search an international application if, and to the extent to which, its subject matter is any of the following: [...] (vi) computer programs to the extent that the International Searching Authority is not equipped to search prior art concerning such programs.”*

Similarly, Rule 67.1 reads:

*“No International Preliminary Examining Authority shall be required to carry out an international preliminary examination on an international application if, and to the extent to which, its subject matter is any of the following: [...] (vi) computer programs to the extent that the International Preliminary Examining Authority is not equipped to carry out an international preliminary examination concerning such programs.”*

While these are useful guidelines in the increasingly relevant PCT, they are procedural provisions that help searching and examining authorities in navigating their way around potentially conflicting subject-matter provisions. This author believes that there is need for a more substantive approach.

Some of these issues were amongst those discussed by the 200 participants at the *WIPO Regional Seminar on Intellectual Property and Software in the 21<sup>st</sup> Century*. Emotionally-charged topics were handled in a constructive manner which permitted most speakers to find room for agreement. Information and communication technologies (ICTs) are of paramount importance to developing economies; therefore, the legal aspects of their protection are of relevance to policymakers in developing regions. In this author's opinion, WIPO has risen to the challenge of informing Member States about the latest legal developments in an ever-evolving area.