ASSESSING THE APPLICATION OF INTELLECTUAL PROPERTY LAW FOR SOFTWARE DEVELOPMENT IN RELATION TO INTERNATIONAL LAW AND THE INDUSTRY RESPONSE

Rattasapa Chureemas[1,*]

Abstract

Software has been continually developed to comply with new technologies around the world by programmers or software engineers. In doing so, they have intended to create tools facilitating people in many activities, such as working, communication, and learning through either online or offline channels. For industry, software is an important system applied in the processing of hardware. As can be seen, software has been developed every day, such as for knowledge, processing, prototypes, and alogism. Nonetheless, there is an absence of legal mechanisms used to protect programmers' or software engineers' rights. Hence, Intellectual Property Rights (IPRs) are the crucial mechanism in protecting such rights in the software industry. In some aspects, software products are able to be processed through registrable and non-registrable schemes. However, the legal mechanisms for IPRs protection are still problematic. Consequently, this article will explore such concepts under the laws of the European Union, the United States, and the United Kingdom.

Keywords: Intellectual Property Rights, Software Engineering, Software Industry, Patent, Copyright

1. INTRODUCTION

Since the 1950s (Wirth, 2008) most industry sectors have employed computer software to operate their businesses. Therefore, many software engineering companies have been established during the modern economic era. One of the most well-known companies in this sector is IBM, which was established in 1957 (Amy, n.d.) with the 'Fortran' language. This led to success of the company in introducing its first Algol

[1,*]Dr. Rattasapa Chureemas obtains a Doctor of Laws from Rangsit University, Thailand. Currently he is working as a Lecturer in the Bachelor of Laws Program, department of Sociology, Faculty of Social Science, Srinakharinwirot University. Email: m.rattasapa@gmail.com

version to the market and improving computer performance. Since the 1960s, software engineering has improved in line with new technologies for suitable use in society (Wirth, 2008). Programmers and software engineers have attempted to develop software for facilitating human activities (Amy, 2018) through the use of computers, such as for work, communication, and learning. Software engineering concepts have been used to improve software programs for the increased performance of hardware mechanisms, supporting the efficiency of production processes (Amy, 2018). Under the response of the software industry, acceleration of software development has been limited, possibly concerned with the inability of legal mechanisms to respond promptly. However, while software programs are rapidly developed every day, it is unclear what legal mechanisms of IPRs can protect software products of diverse types, including knowledge, processes, prototypes, and algorithms, especially in entirety, from the development of the initial concept of a new idea, until the creation of usable, saleable software.

Intellectual Property Rights (IPRs), defined as "the rights given to persons over the creations of their minds [which] usually give the creator an exclusive right over the use of his/her creation for a certain period of time" (World Trade Organization, n.d.) are the best answer as a legal mechanism capable of protecting software products. IPRs are identified based on the concept of accessing the property rights of people regarding their intellectual creations. IPRs are similar to the rights relating to real property, such as land, buildings, or estate. Additionally, IPRs include the rights to buy, purchase, or license the property, and the ability to protect property from use by others without the creator's permission (U.S. Congress, 1990). IPRs related to software products include copyrights, patents, and may also include trade secrets (some papers reveal how a trade secret can be implemented to prevent use of a software product). Therefore, management of IPRs for software products involves registrable and non-registrable processes, which work differently regarding the expression of copyrights and patents, but which are overlapping, depending on the process of the software product and what is necessary to be protected under the IPRs law.

However, the application of IPRs for software products has some difficulties, including the definition of the product, and terms of the software and computer programs. Therefore, the development of legal protection for a software product, until now, has been recognized under the doctrine of '*sui generis*' (U.S. Congress, 1990), meaning of its own kind or class (Black, 1979), which allows for the uniqueness of protection for a software product.

This paper will discuss the theory of IPRs and the theory of IPRs under *sui generis*, a new principle that the world has tended to recognize in applications for IPR protection of

software. Analysis of how to apply for an IPR for protecting a software product in partial response to the software industry is also included.

2. A BRIEF HISTORY OF SOFTWARE DEVELOPMENT

Following the development of computers in the 1950s, a computer is essentially a machine which has the capability for conducting calculations, with a system allowing for the combination of multiple calculations. Computer functions normally entail 2 components, namely the hardware and software. Successive improvements in software have occurred since the 1970s in line with the IBM project (Mege, 2014). During the following period, after the 1970s, programming companies attempted to develop and improve software systems to be more sufficient. These included the Microsoft Word program in the mid-1980s, Linux kernel and the internet innovation in the 1990s, the Netscape Navigator browser by use of the C and C+ languages after 1998, and many others (Yost, 2018). Furthermore, software has been developed for use in mobile applications and new software is likely to be developed for new technology in the future.

The definition of software has various meanings depending on specific philosophies. These definitions of software demonstrate that *'Computer programs and associated documentation such as requirements, design models and user manuals, may be Generic - developed to be sold to a range of different customers e.g. PC software such as Excel or Word, or Custom-made - developed for a single customer according to their specifications'* (Long, n.d.). Most definitions are presented in specific laws, such as in the Indian Copyright Act where it is defined as '*a set machine-readable medium, capable of causing a computer to perform a particular task or achieve a particular result, includes any electronic or similar device having information processing capabilities*' (Gupta, 1996).

The definition of software in the European Commission, states that *'a computer program shall include programs in any form, including those which are incorporated into hardware. This term also includes preparatory design work leading to the development of a computer program provided that the nature of the preparatory work is such that a computer program can result from it at a later stage*' (Directive 2009/24/EC on the legal protection of computer programs, 2009). Therefore, software has been classified into two categories; first of all, system software – to coordinate activities and functions for hardware (i.e. as an intermediary between hardware and the interface application); second, application software – to interface with the user for coordinating a computer's activity (i.e. to provide a specific function to the user and show a specific ability from the computer) (Rob & Etnyre, 2015).

The computer hardware has no ability to work without functional

software, as a computer performs by using software for processing computer circuit systems. Software development is faster than hardware development, thought to be due to the lower requirements for initial capital expenses and because programmers can achieve high wages in software development (Mege, 2014).

The history of software development presents software as an interface between computers and humans, which has seen incredibly rapid development and change, even to become part of the human body. Software is used in everyday life and is enveloped all the time. Thus, the programing and software industry has attempted to find the best solution in using IPRs for protecting their software products, either by copyright or patent. However, the function of software is a key point as some programmers think that software is not the expression of ideas, but is developed from processes and mechanisms. Notwithstanding, this point should be analyzed based on responses from the industrial sector.

3. ANALYSIS

Intellectual Property rights (IPRs) have a purpose to protect the rights of creators in reference to their creations of the mind (World Intellectual Property Organization, 2004) and are similar to the property laws which protect physical property. IPRs include intangible creations of human intellect (Craig, 2012). These are different to the rights of property, particularly tangible goods. For example, when a seller sells his or her own car to a buyer, the seller's full rights of property will be transferred to the buyer without owning a license. On the other hand, if the seller holds an IPR, the seller will receive the exclusive owner right to sell or transfer the rights to a buyer via the owning of a license.

Between the 1970s and 1980s, WIPO members generated and discussed the problem of what IPRs (Copyrights or Patents) should be implemented to protect software products. In 1985, WIPO and UNESCO agreed that copyright would be implemented to protect software products, as software products are a type of expression work, similar to other works under copyright. Later, copyright was widely accepted for implementation in protecting software products under a *sui generis* approach. As software's basic use is irrelevant in its qualification as a literary work, it can be protected under an IRP, even though this extends only to works of expression. On the other hand, if the software was created using a technical solution, it can be protected under patent (World Intellectual Property Organization, 2004).

Under the principle of copyright, in which works of literature should be present and touchable in order to be shown the same rights as tangible property, some software products are not suitable and therefore cannot apply for copyright. Furthermore, this paper suggests the extent to which IPRs should be applied for computer software in response to the industry

sector.

3.1 Software under Copyright Protection Law

The principal of copyright protection for literature works, assigns rights to the person who first created the artistic works. Copyright can cover all creative works and expressions of ideas but is not concerned with the underlying ideas or processes of thoughts (U.S. Congress, 1990). Therefore, copyright protection might be created for implementation, recording, and implementation of a software's source code, but not program logic, algorithms, systems, or layout (Ambrogi, Katz & Hammer, 2012). Computer software has been accepted under copyright in the international sector according to Article 2 of the Berne Convention 1896/1979 and the TRIPs Agreement article 10 which refers even to source code and object code which shall be protected under copyright on behalf of the Berne Convention 1971 (Mege, 2014). Meanwhile, Article 4 of the WIPO Copyright Treaty of 1996 also guarantees software protection under the Bern Convention (Software Industry Promotion Agency, 2015).

The protection of software under copyright has been applied as substantive law in various countries such as through the European Directive 91/250 EEC of 1991 which clarifies the types of software protected under law (Directive 2009/24/EC on the legal protection of computer programs, 2009). The 1998 Copyright Act of the United States mentions computer software as a literary work which is protected under law (The United States Copyright Office, 2016). The 1985 Copyright Act of Japan includes the wording '*Program*', '*Program work'* and *'Program Language'* (Durney, 1991).

Computer software has even been accepted and guaranteed under copyright law under both international and domestic laws. However, there are some problems of concern to programmers. Firstly, some parts of a software product are not covered by copyright protection. Secondly, there is an issue regarding fair use and the associated exceptions of exclusive rights (Software Industry Promotion Agency, 2015).

As mentioned above, creation of software products is a varied process but only the expression of creativity in the software product is protected under copyright protection. On the other hand, the underlying ideas, procedures, methods, mathematical concepts, or algorithms are not eligible for copyright protection, as they are not an expression of creativity (Software Industry Promotion Agency, 2015).

One problem regarding the copyright of software products is the principle of fair use, especially with consideration to reverse engineering. By this reason, a user can copy copyrighted work without permission under the limits of exemption, such that a competing company can use this to renew an algorithm which will then be the initial implementation of a software product and as such new

software (Software Industry Promotion Agency, 2015).

3.2 Software under Patent Protection Law

According to some, the lack of copyright protection for software products, has led to attempts by the world convention to find other IPRs to protect software products. One of best IPRs is patent protection, for the reason that software development makes use of creativity under processes, methods, systems, calculation, frameworks, and algorithms, which is relevant to the patent protection principle.

Since the 1990s, the patent idea has become apparent in considerations of legal protection of software and has been considered in response to questions from programmers as a program or source code under title can be protected by copyright. Under patenting and copyright laws, codes and computer programs cannot be described as touchable inventions that can be protected under such laws. Later, during the software patent crisis, attempts were made to protect software products under patent law. The World Convention and some individual countries have begun accepting software products to register under patent law (Place, 2005).

Article 27 of the TRIPS Agreement, under China Patent Law, was revised on April 1, 2017 under the State Intellectual Property Office (SIPO) of China, which revised its examination guidelines for patents to extend the process to software products and computer programs under China patent law (European Patent Office, 2017). These included "*Computer program per se*", which belonging to the rules and methods for mental activities, is ineligible for patent protection, but also *"inventions relating to computer programs*" which is patentable. A software claim may be drafted in the form of "*medium plus computer program process*" or as an apparatus claim including a component implemented by a computer program (Huang, 2017).

In the EU, under the European Patent Convention, article 52, software products (computer programs) are not eligible for patenting. However, EU patenting may depend on an individual court decision (Knauer, 2015) (Free Software Foundation Europe Organization, 2010). Software protection under patent law in the UK has legal similarly to the European Patent Convention of article 52. Nevertheless, inventions of computer programs cannot be determined as touchable items, but various court cases in the UK have been decided in favour of protecting a software product under patent law. The UK court involved in one such decision provided the following reason "*the practical benefit of the invention is that it presents a new and improved interface to application programmers and makes it easier for them to write application software for the multi touch device. The device is in a real*

practical sense, an improved device" (HTC v Apple, EWCA Civ 451, 2013), with the approach of using the technical contribution of the software, which is patentable. In addition, to be eligible under patent protection the UK High Court has set up five signposts to test and determine if a software product should not be excluded from patent protection, following modification of the HTC v Apple case:

'1. Whether the claimed technical effect has a technical effect on a process which is carried on outside the computer.

2. Whether the claimed technical effect operates at the level of the architecture of the computer; whether the effect is produced irrespective of the data being processed or the applications being run.

3. Whether the claimed technical effect results in the computer being made to operate in a new way.

4. Whether there is an increase in the speed or reliability of the computer; and

5. Whether the perceived problem is overcome by the claimed invention as opposed to merely being circumvented' (Intellectual Property Office, BL O/184/15, 2015).

It is important to consider the benefits of protecting software under patent. Initially, in the 20th century, most software innovation consisted of development for increasing technology making hardware work sufficiently. The creation of software is a highly complex process, involving the use of semiconductor technology, and using technological tools for development of virtual reality, interactive systems, and better user interfaces, especially regarding artificial intelligence (Sucontphunt & Siriborvornratanakul, 2017). Therefore, sole copyright protection cannot cover some of the processes involved in the implementation of software, those which are not present in the expression of ideas but are present in the underlying ideas, and as such are not eligible for copyright protection. As the function of copyright can only protect expressions of creativity which have been written by the programmer, and does not cover the ideas behind this expression, such as the procedures, methods of operation, mathematical concepts, or the algorithms used (Yang, 2012). However, patent protection can solve the problem of the underlying ideas itself, which have not been protected under IPRs. Patent protection of software applies benefit in four basic areas (Stallman, 2002). Firstly, patenting offers protection of ideas and the use of those ideas. Secondly, registration under patent law confirms the exclusive right to the software. Thirdly, patenting offers a long-term protection period of 20 years. Lastly, patenting allows for monopolization of the exclusive right to the patented material, and protection is offered even for non-intentional copying of patented works in the situation of resistance or when another party cannot prove that their work is their own idea and not taken from the patented work.

4. CONCLUSION

Therefore, following the above-mentioned explanation and analysis of the IPR copyright and patent protection of software products, both copyrights and patents have different functions in regard to works of expression, as well as different advantages and disadvantages regarding the protection of software and software development. Due to their different characteristics and objectives in the protection of works, and also due to the fact that it is difficult to identify the individual components of the software product itself, including difficultly in deciding how to separate between the unpatentable underlying ideas and the patentable creative expression of ideas in the software product, it is difficult to decide which parts should be protected under IPRs.

However, considering the historical development of software, and in accordance with intellectual property laws, developers and engineers aim to create new software, and therefore should be protected under IPRs. However, under IPRs only the source code and object code are covered by copyright law, while the system processes and algorithms used in its development cannot be covered by copyright law. Therefore, not all components of the software program are protected. If the effort of driving the composition of ideas in the software is to be protected under IPRs, then patent law is the answer to be used to close the loophole of copyright law. This is likely to solve the problem that is of concern to programmers and software engineers.

Nevertheless, this does not correspond to benefit in the software industry, as the restrictions in the process of patent law are complex, and costly, such that this mode of protection would not be valuable in a commercial comparison with copyright. It might cause difficulty in developing software and disruption for the industry. Owing to the main problem of software protection under patent law, it is necessary to determine which patent relates most closely to the software that a programmer is attempting to write. This is not possible, as some patents that are pending consideration are confidential. Patents are published after the period of consideration, which may be as long as 18 months, which is enough time to finished implementing a program, or even transferring it to the user, such that a developer may acknowledge only after finishing or sending their software to the user that some of the content of the program is already protected by patent. It is known that software development relies on independent software developers to maintain a competitive edge in the software market. As the cost of patent application is high, protection of software by this method may stop independent software developers from continuing to develop software. Moreover, the patent application uses a long period of time, sometimes over one year to complete the process, such that it is not appropriate to the rapidly developing software industry.

It is necessary to consider why patenting of software has several more problems than copyright regarding protection of the rights to software in the industrial context. According to the analytical statement of Professor Eric Goldman at Santa Clara University School of Law, who has expertise in Internet Law, Intellectual Property, and Advertising Law (Goldman, 2013), there are 3 main problems of patenting software with respect to development in the software industry. Firstly, software has a short innovation life, with some software having a full product lifecycle in just a few years. Compared to the patent application period, some software may complete their commercial life before the patent application has been processed. Secondly, software will be produced without any patent incentive. This idea comes from looking at patent law on utilitarian grounds along with the idea that social welfare enhances software development by giving trailblazers a monetary reward in a restricted commercial term. Because the importance of software developers lies in the need to provide innovation to society, providing innovation under rights to exclude competition is also important. Thirdly, further problems for software patents including that programming protection requires too high a level of reflection, programming is difficult to depict, and that patent exploration by resulting trend-setters is too exorbitant. It is expected that the software industry will prefer the idea of using the legal machine of patent law to protect software product rather than copyright, as patent law requires that each new software meets the qualification of being a new and higher innovation, which is stricter than the requirements for the protection of such software under copyright.

Although, some authorities have tried to develop a process or legal requirement regarding patent law suitable for the protection of software products, there is hesitation in the software industry regarding future development for the reasons mentioned above. This is due to the complicated requirements in granting patents, the cost to obtain them, and also the fact that patentability legal requirements are not the same in each country. If the industry needs to cover their software internationally, they would be required to obtain a patent in each country, which is not good value for a commercial industry. In comparison, all members under the Berne Convention for the Protection of Literary and Artistic Works are automatically copyright protected. Consequently, copyright may be the smart response when looking for a candidate to protect programming items. Later we may apply change in the system, developing a new direction of copyrights more effectively able to secure programming items, and furthermore execute court measures relating to the escape clause of control.

REFERENCES

Place, A.G. (2005). The Evolution of Patenting Software. James Cook University law review, 12, 11. Retrieved March 22, 2018, from URL:http://www.austlii.edu.au/au/journals/ JCULawRw/2005/2.html

Amy J. K. (2018). *Cooperative Software Design: History of Software Engineering*. Washington: University of Washington, Retrieved March 18, 2018, from https:// faculty.washington.edu/ajko/books/cooperative-software-development/index.html.

Brian Craig. (2012). *Cyberlaw the Law of the Internet and Information Technology*. Pearson Education Inc.: Prentice hall Publishing, United State of America.

Deli Yang. (2012). Software Protection: Copyrightability vs Patentability. *Journal of Intellectual Property Rights*, 17, 160-164.

Durney, Edward G. (1991). Protection of Computer Programs under Japanese Copyright Law. *Pacific Basin Law Journal*, 9(1-2), 18–19. Retrieved March 22, 2018, from https://escholarship.org/content/qt8j05w13p/qt 8j05w13p.pdf.

Eric Goldman. (2013). Fixing Software Patents. *Forbes Tertium Quid Blog, Santa Clara University*. Retrieved April 26, 2018, from https://ssrn.com/abstract=2199180

European Patent Office. (2017). *China: Revision of SIPO's Examination Guidelines*. Retrieved March 26, 2018, from https://www.epo.org/searching-for-patents/helpful-resources/asian/asia-updates/2017/20170331.html.

Fredrick Otieno Mege. (2014) *Protection and Regulation of Intellectual Property Rights in Computer Software and Programs in Kenya* (Master Dissertation, The University of Nairobi, Kenya). Retrieved March 20, 2018, from http://hdl.handle.net/11295/77302.

Free Software Foundation Europe Organization. (2010). *Software Patents in Europe*. Retrieved March 23, 2018, from https://fsfe.org/campaigns/swpat/swpat.en.html.

Henry Campbell Black. (1979). *Black's Law Dictionary* (5) St. PaulMinn: West Publishing Co.

Intellectual Property Office. (2015). *The Patents Act 1977: whether patent application GB1222096.8 complies with section 1(2) of the Act*. Retrieved April 12, 2018, from https://www.ipo.gov.uk/p-challenge-decision-results/o18415.pdf.

John L. Ambrogi, Michele S. Katz, and Amy L. Hammer. (2012). *Intellectual Property Protection for Computer Software in the United States.* Retrieved March 22, 2018, from http://www. advitamip.com/wp-content/uploads/2012/02/ProtectingComputerSoftware022006.pdf

Lyle N. Long. (n.d.). *Introduction to Software Engineering.* Retrieved March 21, 2018, from https://www.academia.edu/33839846/Introduction_to_Software_Engineering.

Reinhard Knauer. (2015). Recent Case Law of the EPO Regarding Software/Business Method related Inventions. *Grünecker, Kinkeldey, Stockmair & Schwanhäusser.* Retrieved March 23, 2018, from https://www.yumpu.com/en/document/read/35180826/recent-case-law-of-the-epo-regarding-software-gra-1-4-necker.

Richard Huang. (2017). *China will lift curbs on software patents as of April 1*, *2017 -- SIPO revised the Patent Examination Guidelines*, Retrieved March 23, 2018, from https://www.linkedin.com/pulse/china-lift-curbs-software-patents-april-1-2017-sipo-richard-huang

Rob, Mohammad & Etnyre, Vance. (2015). Student Perceptions in Teaching Principles of Management Information Systems. *Journal of Education for Business*, 90, 1-6. Retrieved March 21, 2018, from https://doi.org/10.1080/08832323.2015.1074151.

Software Industry Promotion Agency. (2015). *Legal Framework of Intellectual Property for Computer Software in Kingdom of Thailand*. Retrieved March 19, 2018, from http://www.sipa.or.th/sites/default/files/publication/files/ip-book-web.pdf.

Tanasai Sucontphunt, and Thitirat Siriborvornratanakul. (2016). *I-AM programmer for Thailand 4.0.* Retrieved April 23, 2018, from http://as.nida.ac.th/gsas/article/i-am-programmer.

U.S. Congress. (1990). *Office of Technology Assessment, Computer Software, and intellectual Property-Background Paper*. Washington DC: U.S. Government Printing Office. 19–24. Retrieved March 18, 2018, from https://www.princeton.edu/~ota/disk2/1990/9009/ 900908.PDF.

Gupta, V K. (1996). Protection of Computer Software/Algorithm. *Journal of Intellectual Property Rights,* 1, 76 – 86.

Wirth, N. (2008). A Brief History of Software Engineering. *IEEE Annals of the History of Computing*, 30(3), 32-39.

World Intellectual Property Organization. (2004). *What is Intellectual Property?*. Retrieved May 25, 2018, from https://www.wipo.int/edocs/pubdocs/en/intproperty/450/wipopub450.pdf

World Intellectual Property Organization. (2004). *WIPO Intellectual Property Handbook: Policy, Law and Use*. Retrieved March 22, 2018, from http://www.wipo.int/edocs/pubdocs/en/intproperty/489/wipo_pub_ 489.pdf.

WTO. (n.d.). *What are intellectual property rights?*. Retrieved March 26, 2018, from https://www.wto.org/english/tratop_e/trips_e/intel1_e.htm

Laws and Suitcases

Berne Convention, 1896/1979
United States Copyright Act, 1998
Directive 2009/24/EC on the legal protection of computer programs [2009] L111/16 (7)
European Directive 91/250 EEC, 1991

European Patent Convention, article 52.
Indian Copyright Act, 1994
WIPO Copyright treaty, 1996s
HTC Europe Co Ltd v Apple Inc (Rev 1) [2013] EWCA Civ 451 (03 May 2013).