Faculty of Law
Lund University

COPYRIGHT V. PATENT

The issue of overlapping protection of computer software in Intellectual Property Law

JAMR32
Intellectual Property Law
In A Global Perspective

Dena Dervanović

Table of Contents

1. Introduction

This essay will analyze the nature of computer software as a subject matter of intellectual property law. It will separately address the features, benefits, issues and shortcomings of both patent and copyright modules of protection currently being used, all the while weighing the pros and cons of each type of protection.

The primary issue at hand here is the fact that the rapid development of technology is not always duly followed by appropriate means of intellectual property protection, which can cause troubling loopholes or overlaps. This is the reason why the topic of the overlapping protection of computer software is so important to discuss. As software and microprocessors became more ubiquitous, it is a necessity to bring about proper intellectual property protection for the aim of giving certain rights to software developers, as well as to entice creativity and innovation.

A landmark case study is incorporated as one of the main points of the essay, for the purpose of argumentum a contrario software patentability and the perils overlaps in protection can bring about. The essay will aim for an international aspect of the issue, thus will start by taking into account relevant international documents all the while taking into consideration certain national systems, in particular the United States system, as well as the regional, European system. The reason behind taking the U.S. and Europe in consideration is their close connection in regard to intellectual property protection, trade aspects etc. It is a very important relationship whose policies largely shape both systems. Relevant case law will be addressed in the course of the analysis.

The essay will conclude with a summary of the deliberation as well as a critical answer to the essay topic question.

2. The features of computer software

Where does software belong?

The complexity of computer software can be somewhat easily summed up in the definition of MacQueen, Waelde and Laurie[1] that says computer programs (software)

[1] Hector MacQueen, Charlotte Waelde, Graeme Laurie, *Contemporary Intellectual Property Law and Policy*, Oxford University Press (2008), p. 61

are a set of instructions designed to perform certain tasks on a computer. The performance of those tasks is a process displaying the analysis of the functions that ought to be performed like algorithms that go on to being restated by a programmer in a computer language, i.e. source code that then goes onto being translated by a computer into machine-readable language that is object code. [2] Software is increasingly becoming more in-demand, and more important for personal as well as business use.[3]

2.1 Copyright protection of software

The nature of computers being a mix of specific characteristics – hardware (machine), software (writing), and artistry – makes computers an interesting area of protection in intellectual property law, especially concerning software.[4] The software industry had followed suit with a fast-forwarded evolution of technology, making this particular category of protection under intellectual property law somewhat complex.

This is depicted even better through the image of the interests of persons in touch with software; namely, its developers need protection from infringement but also need the possibility of working on and building up on the works of others without infringement. Further on, users of software want an affordable, contemporary piece of software suited for their needs whereas competitors seek to develop software as close to the existing ones, so they could compete with it on the market.[5]

According to Article 2(1) of the Berne Convention that reads:

> "(1) The expression 'literary and artistic works' shall include every production in the literary, scientific and artistic domain, whatever may be the mode or form of its expression, such as books, pamphlets and other writings; lectures, addresses, sermons and other works of the same nature; dramatic or dramatico-musical works; choreographic works and entertainments in dumb show; musical compositions with

[2] Ibid
[3] Arne Kolb, *Protection of Computer Software*; Lilian Edwards, Charlotte Waelde, *Law and the Internet*, 3rd edition Hart Publishing (2009), p. 335
[4] Robert P. Merges, Peter S. Menell, Mark A. Lemley, *Intellectual Property in the new technological age*, Revised fourth edition, Wolters Kluwer Law & Business, Aspen Publishers (2007), p. 979
[5] Arne Kolb, p. 335

> or without words; cinematographic works to which are assimilated works expressed by a process analogous to cinematography; works of drawing, painting, architecture, sculpture, engraving and lithography; photographic works to which are assimilated works expressed by a process analogous to photography; works of applied art; illustrations, maps, plans, sketches and three-dimensional works relative to geography, topography, architecture or science."[6]

Article 2(1) of Berne does not explicitly name software as protected work in the list of examples it provides. However, since the term 'production' is so broad, it is considered that Article 2(1) in fact, protects software as well.[7]

Computer programs (software) are considered written works with utility purposes[8] (regardless of the fact that they are mostly composed of '1' and '0' in endless combinations), meaning that they are composed of a source code and an object code. The source code is written by a human person, and once finished the source code is then transferred into a machine code that produces the wanted effect of the source code. This machine code is the object code of software.[9]

Thus, one can say that the source code of computer software enjoys the protection given to literary works, regardless of its format of expression, as enunciated in the aforementioned Article 2(1) of the Berne Convention.[10] The historic overview of the development of software protection goes back to the WIPO Model Provisions on The Protection of Computer Programs from 1978,[11] a document relying mostly on copyright law in regards to software that shaped the general opinion on how software should be protected.[12]

Thus, what followed was a series of documents protecting software as literary works, such as the Computer Software Copyright Act in the US (1980), the Council Directive

---

[6] Berne Convention for the Protection of Literary and Artistic Works (1971)
[7] Silke von Lewinski, *International Copyright Law and Policy*, Oxford University Press (2008), p. 232
[8] Merges, Menell, Lemley, p.986
[9] Lewinski, p.233
[10] M. Ficsor, *The law of copyright and the internet*, Oxford University Press (2002), p.465
[11] WIPO Model Provisions on The Protection of Computer Programs, Publication N. 814 (1978)
[12] A.Kolb; L. Edwards, C.Waelde p. 337

from 1991 in the EU[13], WTO TRIPs Agreement (1994) and the WIPO Copyright Treaty of 1996.

The TRIPs Agreement incorporates this with its Berne Plus regime in Article 10(1) that reads:

> "Computer programs, whether in source or object code, shall be protected as literary works under the Berne Convention (1971)."

It is interesting to note that, under TRIPs, it is only the whole code, the whole software that is protected as such – protection does not extend to specific portions of that code.[14] This is viewed in conjunction with Article 9(2) TRIPs that prohibits copyright protection for mathematical concepts, ideas, procedures and methods of operation.

WIPO Copyright Treaty (WCT) goes towards an affirmative language on copyright protection of software with its Article 4:

> "Computer programs are protected as literary works within the meaning of Article 2 of the Berne Convention. Such protection applies to computer programs, whatever may be the mode or form of their expression."

The provisions of TRIPs and WCT are almost identical, but TRIPs makes it one of its 'shall' provisions, whereas WCT adopts a very *fait accompli* stance that computer programs 'are' protected.[15]

On the level of the European Union, the European Software Directive plays a key role in understanding the position of the Union. A short analysis of the content and scope of the Directive follows, with the aim of depicting a better picture of copyright protection of software for the purpose of this essay's attempt at proving that copyright is a better way of protecting software than patents.

[13] 91/250/EEC.
[14] Sam Ricketson, Jane C. Ginsburg, *International Copyright and Neighbouring Rights, The Berne Convention and Beyond* volume I, Oxford University Press (2006), p. 517
[15] Ibid, p. 518

Namely, the EU Software Directive[16] has identified the necessity for legislative action in the field of software protection. The Directive expressly recognizes software as literary work, with protection granted to the expression of the idea and not the idea itself.[17] One of the most interesting parts of the Directive refers to its descriptions of infringement and exclusions from actions that constitute it. Namely, with the Directive, users are allowed to make back-up copies of software, observe and study the software – as enunciated in Article 5.

The protection of copyright is in force as soon as the software is created and there is no high threshold for obtaining protection for one's work. Thus, one of the most obvious arguments pro copyright protection of software is that there are no applications for protection that need to be assessed; there is nothing that leads to unnecessarily high costs and take a lot of effort to maintain over the course of years, which is the case with patent protection. Furthermore, copyright, as such, provides for somewhat of a control over dissemination and copying (through fair use).[18]

However, this is not to say that copyright is the perfect, tailor-made modus of protection for software. It does come with its own predicaments. Those include software's defenseless nature against copying. It is very easy to copy software, mostly because software was made to be copied.[19]

One of the most counterintuitive instances of copyright protection of software is the nature of software usage in itself. Namely, in order to use software, one makes at least one copy of the software in the RAM memory of the computer, which makes it impossible not to make copies of software.[20] It should be noted that these 'copies' are never saved. They are just the modus operandi of software utilization.

[16] 2009/24/EC of the European Parliament
[17] Article 1(1) and 1(2) Software Directive
[18] Diane Rowland, Uta Kohl, Andrew Charlesworth, *Information Technology Law* fourth edition, Routledge (2012), p. 355
[19] Ibid, p.357
[20] Merges, Menell, Lemley, p.1024

If the advantage of software is its interoperability[21], its user-friendliness, then that is at the same time its very downfall, since it is equally easy to pirate it. Thus, copyright has the problem of enforcement[22] and Internet does not come to rescue in this case.

However, copyright as is, protects the expression of an idea, but not the idea itself. This particularity has proven to be a source of debate when it comes to software protection – is it enough? Hence, a new direction of software protection emanated from this debate – the one of software protection through patents that will be discussed *infra.*

2.2 Patent protection of software

Patents, by their nature, provide for a larger (but shorter) scope of protection thus making them much more appealing to inventors, innovators and software developers alike. In the U.S. particularly, this has been a trend with computer software. However, in Europe, the European Patent Convention[23] contains a software exclusion from patentability.[24] The language of the exclusion is quite clear in Article 52(2)(c) and specifically addresses computer programs as a non-patentable subject matter.

However, there have been a number of cases where this exclusion was debated. A limited number of prominent cases will be examined in this regard. The said case law has built a few approaches to the question of patentability of software, including but not limited to: the 'any hardware' approach or the Aerotel[2526] 'technical effect' 4-step test[27]. In the European Patent Office, the 'any hardware' approach was developed in the Pension Benefits Systems Partnership case[28] and Hitachi[29] case.

The 'any hardware' approach consists of finding grounds for patentability of software if it requires any hardware in order to function. This means that if the software in

[21] See Microsoft Corp v. Commission of the European Communities T-201/04 [2007] 5 CMLR 11
[22] Ibid
[23] EPC (1973), Article 52(2)(c)
[24] A.Kolb; L. Edwards, C.Waelde p. 346
[25] Aerotel v. Telco Holdings [2007] Court of Appeal of England and Wales, The United Kingdom
[26] N.B. Aerotel is a UK case. Nevertheless it is a very important one to mention since it established a well-known test.
[27] Lionel Bently, Brad Sherman, *Intellectual Property Law* 3rd edition, Oxford University Press (2009), p. 405
[28] Benefits Systems Partnership case T931/95 [2001] OJ EPO 441
[29] Hitachi/Auction Method, T258/03 [2004] OJ European Patent Office 575

question requires a computer to function, it is patentable, regardless of novelty or the inventive step.[30]

The Aerotel test is there to find if the invention in question falls under the exclusion provision through a 4-step test.[31] What has been said in connection with the Aerotel test was a principle that originated in the case of Shopalotto.com's Application[32] that the invention ought to have a certain level of a technical effect that reaches beyond being capable of loading itself into a computer.

This means that the respective software is examined so as to see whether it largely contributes to the art, in a manner that the contribution does not solely consist of the explicitly excluded patent subject matter.[33]

When it comes to the stance of the European Patent Office, it is a bit more relaxed. Regardless of the somewhat straightforward language of Article 52(2)(c), the EPO Guidelines for Examination of Patentability[34] explain how one can still go about patenting software. It is deemed as a 'computer-implemented invention' and if the software itself has patentable technical features along with a corresponding apparatus that it is implemented in, it can be patented.

Furthermore, the Guidelines clarify that if the software itself has such technical character that its operation amounts to a further technical effect than software is expected to do it is patentable software.

The message one can take from the EPO Guidelines when it comes to patent protection of computer software is the following:

In general, software is not patentable subject matter. However, it is not entirely excluded from patentability. Thus, if one wants to patent software, they will have to go through a lengthy, time-consuming, expensive application process where they will have to prove that the software in question in fact fulfills the criteria for patenting

[30] Bently, Sherman, p. 414
[31] Ibid, p. 405
[32] Shopalotto.com Ltd's Application [2005] EWHC 2416 (Pat); [2006] R.P.C. 7; Intellectual Property Office, The United Kingdom, para. 9
[33] Macqueen, Waelde, Laurie, p. 421
[34] European Patent Guidelines, Part G Patentability, para. 3.6; EPO

enunciated in the EPO Guidelines. It is exactly that lengthy proving process that is the *trick of the trade.*

This was also the stance exerted in IBM/Computer Programs that says: [35]

> "[…] if a computer program is capable of bringing about, when running on a computer, a further technical effect going beyond… normal physical effects, it is not excluded from patentability..."[36]

This makes it much simpler to patent computer software, no doubt. The question remains, however, whether this is a proper way to tackle the issue of protection of software.

Two things that are often overlooked by those in favour of patent protection of software are: [37]

> 1) The inventive step requirement – most software is a compilation of existing techniques and knowledge that aim to solve a well-known problem.[38]
>
> 2) In Europe, the EPC clearly excludes software from patentable subject matter.

Taking into consideration the nature of patent protection and the fact that it is there to encourage innovation, at first sight the software patent protection does not sound worrying. However, if applied to software, patent protection risks gaining a completely different character indeed – because software is supposed to be interoperable, it would bar competition anti-competitiveness and obtain somewhat of a monopoly feature, one can see that patenting software would hinder further development of the software industry.

---

[35] Macqueen, Waelde, Laurie, p. 421

[36] T935/97 IBM/Computer Programs [1999] EPOR 301 and T1173/97 IBM/Computer Programs [2000] EPOR 219

[37] Diane Rowland, Uta Kohl, Andrew Charlesworth, p. 356

[38] Dennis S Karlaja, *Copyright protection of computer software in the United States and Japan: Part 1*, EIPR 195 (1991)

3. *Alice in Software Patent Land*
The Alice Corporation v. CLS Bank case

3.1 Factual circumstances

Alice Corporation v. CLS Bank[39] case is undoubtedly one that caused a big stir in the IP world. The background of the case is that Alice, an Australian company in possession of '479, '510, '720 and '375 patents that all deal with computerized trade platforms, i.e. electronic methods and programs. These platforms engage into financial transactions where third parties settle duties between two other parties with the aim of eliminating risk.

The patents owned by Alice are concerned the concept of third parties acting as guarantors for transactions, i.e. escrow. In 2007, CLS Bank filed a lawsuit against Alice seeking a declaratory judgment of non-infringement and invalidity of the following Alice's patents: '479, '510 and '720. Alice filed a countersuit which claimed infringement, which eventually led to CLS moving for summary judgment stating that the infringement could not have happened in the United States.

Both parties issued several cross-motions, incorporating all four patents. This eventually led to the District Court deciding that Alice's patents were invalid due to the fact that the idea behind them was too abstract which could, in turn, preempt the use of that idea in general.

This case split the opinions of the judges of the U.S. Court of Appeals for the Federal Circuit (CAFC). After the first panel's decision, a hearing *en banc* was held for reaching a *per curiam* decision. This did not help unify the opinions on one or the other side of the issue of patentable subject matter, however. The result was five separate opinions, none supporting the other (by a 10-judge panel). Eventually, seven judges ruled for the non-patentability.[40]

[39] Alice Corporation Pty. Ltd. v. CLS Bank International, et al., 134 S. Ct. 2347 (2014)
[40] Jesse Adland, *Alice Corp. v. CLS Bank International*: Challenges in Identifying Patentable Subject Matter, Intellectual Property and Technology Law Journal volume 26, Aspen Publishers (2014), p.20

The case went up to the U.S. Supreme Court that ruled unanimously that the claims were not eligible for patent protection. The case will be analyzed *infra*, starting with the relevant law, and proceeding with a detailed display of the relevant case law that impacted the decision.

3.2 The law

The U.S. Supreme Court has taken into consideration The U.S. Patent Act (35 USC) with special focus on the utility requirement in §101, as well as the conditions enunciated in §102 and §103. §101 serves as the first-instance checkpoint for determining patentability. Once an invention is deemed patentable under §101, the invention undergoes a test of whether it fulfills the criteria for patentability in §§102, 103.[41]

It is important to note that software is protected by the U.S. Copyright Act of 1976 (with its 1980 amendment that includes quite an explicit definition).[42] It has also been affirmed by the U.S. courts in cases like Apple Computer Inc. v. Franklin Computer Corp.[43] This goes to showcase the existing overlap.

3.3 The relevant case law and its impact on software patents

The U.S. Supreme Court has used several cases as basis for decision in Alice Corp that will be mentioned in the coming breakdown of the issues presented in the case. Namely, the Court used the two-step test on the patent ineligibility of abstract ideas set out in the Mayo case.[44] The two-step test is consisted of 1) determining whether the patent claim is directed at an abstract idea, law of nature etc. and 2) if the claimed idea contains enough inventiveness to be patentable.

This opinion of the Court stems from one of the first cases on the matter of patentability of abstract ideas – Gottschalk v. Benson,[45] where the Court deliberated

[41]Nikola L. Datzov, *Machine-or-Transformation Patentability Test: The Reinvention of Innovation*, Hamline L. Rev. 281 (2010), p. 311
[42] Diane Rowland, Uta Kohl, Andrew Charlesworth, p. 359
[43] Apple Computer Inc. v. Franklin Computer Corp. 714 F 2d 1240 (3rd Circ. 1983)
[44] Mayo Collaborative Services v. Prometheus Laboratories, Inc., 132 S. Ct. 1289 (2012).
[45] Gottschalk v. Benson, 409 U.S. 63,67 [1972]

whether binary code/algorithms of software can or cannot represent an expression of an abstract idea.[46] Here, the Court set the standard by saying that patent protection of an algorithm would pre-empt the public from using it, and thus goes against the core purpose of intellectual property protection.

That is the test of determining whether the claim is in fact an abstract idea. The abstract ideas ineligible for software patent protection found in Alice, were, inter alia, mathematical relationships/formulas, economic practices etc.

Furthermore, if the claim does indeed belong to the category of an abstract idea, a test of determining whether there is a segment of the claim or the claim as a whole that produces an effect of being something *significantly more* than just an abstract idea.
In this test, the delineation of the *significantly more* amounted to: improvements to another technology or field; improvements to the functioning of the computer itself (same principle as EPO exerted in IBM/Computer Programs, *vide supra*); meaningful limitations beyond generally linking the use of an abstract idea to a particular technological environment.

On the other hand, features that will not be eligible to amount to 'significantly more' would then be: adding the words "apply it" or their synonym to the abstract idea; instructions to implement the abstract idea on a computer; requiring a generic computer performing generic and well-known tasks and computer functions.
In further analysis of the case, the U.S. Supreme Court refers to the precedent case that is Bilski[47] where it invokes pre-emption as a source of issues that would arise from upholding a patent of such software: "would pre-empt use of this approach in all fields, and would effectively grant a monopoly over an abstract idea."[48]

Such monopoly would put innovation to a halt, instead of encouraging it, as was also stated in the Mayo case. The Court supplements this by citing from Myriad that "[…]

[46] Andrei Iancu, Jeremiah Helm, *Code on Disks and Hat Tricks – Is Computer Software on a Medium Really Patentable?* 90. J. Pat. & Trademark Off. Soc'y 97 (2008), p. 103
[47] Bilski v. Kappos, 130 S. Ct. 3218 (2010).
[48] Bilski v. Kappos, at 3231

abstract ideas are the basic tools for scientific and technological work."[49] Thus, they are ineligible for patent protection.

One of the most important tests confirmed in Alice came from Bilski: the machine-or-transform test that would not pre-empt the use of the fundamental principle (i.e. an operation that the invention would perform) by just attaching it to a machine/apparatus. [50]

What this test entails is that, under §101, a patentable subject matter will be a process that includes as specific machine or if the process ultimately changes, transforms into something different. [51] If an invention passes the machine-or-transform test, it is patentable, and if it does not, it is still not automatically un-patentable. Then the invention is subject to other tests that could determine patentability (however, this has yet to happen).

The machine-or-transformation test is ultimately essential for the clarification of patentable subject matter, as well as for establishing a fair balance between pre-emption and encouragement for innovation since it prevents patent owners from monopolizing and abusing their rights.[52] This test will make it easier to know what exactly is patentable, which is useful and practical for any innovation stakeholder and courts alike. However, this is not to say that this test is the supreme test for software patentability determination. [53]The Alice case ended with a unanimous decision against patentability of all claims in the case.

The Alice case is essential for this matter since it has installed a major, radical change towards limiting the amount of software patents in the United States. However, Alice is also an interesting case from the perspective of the cases cited in it. Alice rounds up the criteria under which software can be considered patentable subject matter, and some of them were mentioned *supra*.

[49] AMP v. Myriad, US Supreme Court No. 12-398 [2013]
[50] Nikola L. Datzov, p. 296
[51] Ibid
[52] Ibid, p. 310
[53] Andrew Nieh, *Software Wars: The Patent Menace*, New York Law School Law Review volume 55 (2010/2011), p. 298

4. Conclusion

The issue of overlapping protection in intellectual property law is essentially a problematic one. Considering this patent/copyright overlap, it is not a stretch to imagine the difficulties a third party could have with understanding what kind of actions exactly would or would not constitute infringement. This could, in turn, have a chilling effect on the actual purpose and utility of the protected software.

If protection is given in order to encourage creativity and innovation, then it can easily be argued that having copyright protection of computer software is more than enough of an incentive.

Namely, copyright protection, with its properties, is quite sufficient a protection, and for a long period of time. To add to that another layer of patent protection would have the potential of hindering progress and the purpose of the subject matter of intellectual property protection – the society would not be able to fully use the protected software, due to the subject matter being overprotected.[54] Such overprotection would bring along economic implications – too high patent maintenance costs, it would discourage further development due to high production costs, etc.[55] And why would anyone be given a chance to abuse the system through this apparent loophole in intellectual property law? Using both types of protection can certainly amount to abuse, if one considers all the ways a copyright/patent holder can be 'safe' from infringement.

One can simply weigh the effect of both copyright and patent protection. If one opts for copyright, it is merely the expression of the idea that is protected. This leaves us with the possibility to use the idea and develop it in a different way without a fear of infringing.[56] Ultimately, this goes to show that copyright protection of software incites innovation. If one opts for patent protection of software, what happens is that the scope of the patent becomes too large and covers the idea itself under its umbrella, ultimately hindering innovation.

---

[54] Viva R. Moffat, *Mutant Copyrights and Backdoor Patents: The problem of overlapping Intellectual Property Protection,* Berkeley Technology Law Journal volume 19 (2004), p. 1515
[55] Ibid
[56] Diane Rowland, Uta Kohl, Andrew Charlesworth, p. 355

Albeit, neither is perfect. Copyright can be identified as the 'lesser evil' compared to patents. The problem with both systems is not only their modus of protection but also their different durations.[57] It is, in both cases, too long and ultimately looks out of context (mostly because it is completely out of context).

Essentially, if those in favour of copyright and those in favour of patent protection of software cannot find a middle ground, the only thing imaginable left is to create a *sui generis*[58] system of protection designed specifically for computer software. This brings along problems of its own, seeing that a compromise between the successes and shortcomings of both patents and copyright should be taken into consideration.

If this was to be the next step in the IP world, one must proceed with caution: the tailor-made software protection system needs to be, inter alia, not overprotective, but just enough, *lagom* [sic!], encompassing and allowing for technological innovation to fall under its scope over the course of years to come, but at the same time not too broad, otherwise it would not fulfill its purpose. It needs to be enforceable and not costly, to begin with. It would also needs to bridge two systems, the U.S. and the European one, since they have close ties.

For now, however, it would be interesting to follow the development of Google's patent application on a software that would provide a spoiler warning system on the content of books, movies, series etc. that the user is following (but has not had the chance to see/experience yet). Google filed the application on April 7th 2015 with the United States Patent and Trademark Office and due to the lengthy, costly process of patent applications, it will take a long while until we know the decision.[59]

---

[57] Laurence Diver, *Would The Current Ambiguities Within The Legal Protection Of Software Be Solved By The Creation Of A Sui Generis Property Right For Computer Programs?* 3 Journal Of Intellectual Property Law & Practice (2008), p. 138

[58] Ibid, p. 125

[59] Google Inc, Appl. No. 13/647,244; United States Patent and Trademark Office

Bibliography

Books and Articles

- Andrei Iancu, Jeremiah Helm, *Code on Disks and Hat Tricks – Is Computer Software on a Medium Really Patentable?* 90. J. Pat. & Trademark Off. Soc'y 97 (2008)
- Andrew Nieh, *Software Wars: The Patent Menace*, New York Law School Law Review volume 55 (2010/2011)
- Arne Kolb, *Protection of Computer Software*; Lilian Edwards, Charlotte Waelde, *Law and the Internet*, 3rd edition Hart Publishing (2009)
- Dennis S Karlaja, *Copyright protection of computer software in the United States and Japan: Part 1*, EIPR 195 (1991)
- Diane Rowland, Uta Kohl, Andrew Charlesworth, *Information Technology Law*, fourth edition, Routledge (2012)
- Hector MacQueen, Charlotte Waelde, Graeme Laurie, *Contemporary Intellectual Property Law and Policy*, Oxford University Press (2008)
- Jesse Adland, *Alice Corp. v. CLS Bank International*: *Challenges in Identifying Patentable Subject Matter*, Intellectual Property and Technology Law Journal volume 26, Aspen Publishers (2014)
- Laurence Diver, *Would The Current Ambiguities Within The Legal Protection Of Software Be Solved By The Creation Of A Sui Generis Property Right For Computer Programs?* 3 Journal Of Intellectual Property Law & Practice (2008)
- Lionel Bently, Brad Sherman, *Intellectual Property Law* 3rd edition, Oxford University Press (2009)
- M. Ficsor, *The law of copyright and the internet*, Oxford University Press (2002)
- Nikola L. Datzov, *Machine-or-Transformation Patentability Test: The Reinvention of Innovation*, Hamline L. Rev. 281 (2010)
- Robert P. Merges, Peter S. Menell, Mark A. Lemley, *Intellectual Property in the new technological age*, Revised fourth edition, Wolters Kluwer Law & Business, Aspen Publishers (2007)
- Sam Ricketson, Jane C. Ginsburg, *International Copyright and Neighbouring Rights, The Berne Convention and Beyond* volume I, Oxford University Press (2006)
- Silke von Lewinski, *International Copyright Law and Policy*, Oxford University Press (2008)
- Viva R. Moffat, *Mutant Copyrights and Backdoor Patents: The problem of overlapping Intellectual Property Protection*, Berkeley Technology Law Journal volume 19 (2004)

Documents

Berne Convention for the Protection of Literary and Artistic Works (1971)
Computer Software Copyright Act in the US (1980)
European Union Council Directive 91/250/EEC
European Software Directive 2009/24/EC
European Patent Convention (1973)
European Patent Guidelines
WIPO Copyright Treaty (1996)

WIPO Model Provisions on The Protection of Computer Programs (1978)
WTO TRIPs Agreement (1994)
The U.S. Patent Act (35 USC)
The U.S. Copyright Act (1976)

Cases

- Aerotel v. Telco Holdings [2007] Court of Appeal of England and Wales, The United Kingdom
- Alice Corporation Pty. Ltd. v. CLS Bank International, et al., 134 S. Ct. 2347 (2014)
- AMP v. Myriad, US S. Ct No. 12-398 [2013]
- Apple Computer Inc. v. Franklin Computer Corp. 714 F 2d 1240 (3rd Circ. 1983)
- Benefits Systems Partnership case T931/95 [2001] OJ EPO 441
- Bilski v. Kappos, 130 S. Ct. 3218 (2010)
- Gottschalk v. Benson, 409 U.S. 63,67 [1972]
- Hitachi/Auction Method, T258/03 [2004] OJ European Patent Office 575
- IBM/Computer Programs [1999] EPOR 301, T935/97
- IBM/Computer Programs [2000] EPOR 219, T1173/97
- Mayo Collaborative Services v. Prometheus Laboratories, Inc., 132 S. Ct. 1289 (2012).
- Microsoft Corp v. Commission of the European Communities T-201/04 [2007] 5 CMLR 11
- Shopalotto.com Ltd's Application [2005] EWHC 2416 (Pat); [2006] R.P.C. 7; Intellectual Property Office, The United Kingdom

Patent pending

Google Inc, Appl. No. 13/647,244; United States Patent and Trademark Office