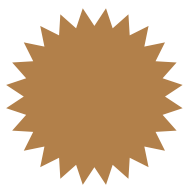
4iP Council

4iP Council
Research Award Winner 2020
Third Place

3

Patentability of Computer Programs in Europe

by Yulia Borisova
LL.M. Candidate in Law of Internet Technology at Bocconi University (Milan, Italy)



Rigorous empirical research on intellectual property

4iP Council is a European research council dedicated to developing high quality academic insight and empirical evidence on topics related to intellectual property and innovation. Our research is multi-industry, cross sector and technology focused. We work with academia, policy makers and regulators to facilitate a deeper understanding of the invention process and of technology investment decision-making.

www.4ipcouncil.com

Suggested citation

Borisova, Yulia. Patentability of Computer Programs in Europe. (February 2021). 4iP Council.

Patentability of Computer Programs in Europe

Yulia Borisova

Summary

In the modern world patent claims become more complex, which is caused by the development of technology and novelty of examined spheres. Insufficient understanding of certain concepts and gaps in regulation undermine legal certainty and decrease the level of intellectual property rights protection. A detailed analysis of existing deficiencies is needed for correct interpretation and further improvement of legal framework.

The subject matter of this paper is focused on software inventions, which emerged in the practice of European Patent Office under the umbrella of "computer program having a technical character". This concept is built without a proper definition of "computer program" and remains insufficiently researched. However, such patents are already granted. The main questions examined in the paper include the following:

- What does Article 52 of the European Patent Convention understand under unpatentable "programs for computers as such"?
- Are there any elements in a computer program that are not excluded by Article 52 and thus can qualify as inventions?
- What is the role and the meaning of "further technical effect" that must be produced by a computer program in order to be patentable?

The paper comprises an introduction, three parts and a conclusion, and considers both European and US experience. The introduction presents the paper's objective, the first part works with the exclusion of Article 52, the second part analyses patentable elements of a computer program, and the third part is devoted to "further technical effect" concept. The conclusion summarizes the key outcomes, which might help to resolve the existing uncertainties in relation to software patents.

Table of Contents

Introduction

During the recent decades there have been many debates about whether the "software patents" should be allowed or denied. In practice such patents have been already granted for some years[1], and the number of patent applications will likely only continue to increase with time. According to the European Patent Office's ("**EPO**") statistics for 2019, computer technology was the third field in terms of the number of patent applications (after digital communication and medical technology), and "the second fastest-growing field, fueled by the rise of artificial intelligence".[2]

The legal framework for patents (including software patents) in the European Union is based primarily on the European Patent Convention ("**EPC**") and Guidelines for Examination in the European Patent Office ("**EPO Guidelines**").

Article 52 of the EPC excludes "programs for computers as such" from patentability. However, the EPO has introduced two concepts in order to circumvent such exclusion: "computer-implemented invention" ("**CII**") and "computer program having a technical character".

These new concepts focus on a certain result produced by software, but disregard the fact that the basic term "computer program" remains unclear. There is no definition of this term in the EPC or EPO Guidelines. However, as it has been correctly noted in doctrine, "it is impossible to regulate this industry without perfect understanding of software".[3] It does not mean that EPC or EPO should attempt to define this complex software engineering term in detail, but as long as this term is used, it is necessary to agree what it means.

The paper's objective is to prove that computer programs are patentable, and that proper definition of this term can help to resolve many uncertainties concerning software patentability. It should be noted that the question is not about whether the actual words "computer program" should be used in the patent application or not, but about proper understanding of these words. The conclusions will be based on the analysis of fundamentals of patent law, the elements and structure of a "computer program", as well as the practice in Europe and the United States ("**US**").

1. Programs for Computers

1.1. Invention

Patent law protects inventions. Although the EPC does not contain a precise definition of "invention", Rule 43 of the Implementing Regulations to the EPC mentions the eligible categories: product, process, apparatus or use. Based on this wording, the EPO's Glossary defines invention as a product, process, apparatus or any new use thereof.

Article 52 of the EPC lists certain exceptions, which "as such" do not constitute a "product, process, apparatus or any new use thereof". Such exceptions include, among other things, (a)

[1] Eric Sutton, "Software Patents. A Practical Perspective", Version 4.1, May 2019, page 6.
[2] EPO. Patent Index 2019, page 4.
[3] Mihai Avram, "Software Legal Protection: Shaping the EU Software Patent", Amsterdam Law Forum, Vol. 6:2, page 21.

mathematical methods, (b) schemes, rules and methods for performing mental acts, playing games or doing business, and (c) programs for computers.

In the US the approach is similar. According to Section 101 of U.S. Code (Title 35), invention is a process, machine, manufacture, or composition of matter, or any new and useful improvement thereof. A patent claim must be directed to one of the four statutory categories above.[4] The main difference from the EU is that the law is silent on what is not patent eligible subject matter. However, courts have created exceptions such as laws of nature, abstract ideas, and natural phenomena. As a general rule, also excluded but subject to analysis are scientific and fundamental truths, mathematical expressions, formulas, algorithms and mental processes.[5]

That said, "software" or "computer program" can be eligible to patent protection only to the extent they qualify as a process or another protected category from the list, and do not fall under exclusions.

It should be noted that computer program is a part of software. In addition to computer program, software also includes all other elements which cannot be classified as hardware, primarily ancillary documentation and data files.[6] However, computer programs and software are sometimes also used as synonyms.[7]

1.2 Exclusion of Article 52

Programs for computers "as such" are excluded from patentability by Article 52 of the EPC. There is no definition of "program for computers" (whether "as such" or "not as such") in the EPC or EPO Guidelines. Thus, we can try to understand the scope of exclusion using general interpretation techniques.

Literal, systematic and purposive interpretation are the three primary methods in the European legal tradition.[8] According to the **literal rule**, in the absence of explicit definitions all words should be understood in their usual meaning.

The Collins English Dictionary[9] contains definitions of such terms as "program", "computer" and "computer program". A "program" is a set of instructions that a computer follows in order to perform a particular task. A "computer" is an electronic machine that can store and deal with large amounts of information. A "computer program" is a set of instructions for a computer to perform some task.

The Macmillan Dictionary[10] defines a "program" as a set of instructions that makes a computer perform an action or a particular type of work. A "computer" is defined as a machine that stores programs and information in electronic form and can be used for a

[4] Eric Sutton, page 57.
[5] Eric Sutton, page 60.
[6] Rosa Maria Ballardini, "Intellectual Property Protection for Computer Programs. Developments, Challenges, and Pressures for Change", Helsinki, 2012, page 11; Susan A. Dunn, "Defining the Scope of Copyright Protection for Computer Software", Stanford Law Review, January 1986, Vol. 38, No. 2, page 500.
[7] Susan A. Dunn, page 500; Eric Sutton, page 5.
[8] Koen Lenaerts, "Interpretation and the Court of Justice: A Basis for Comparative Reflection", at: https://core.ac.uk/download/pdf/216908204.pdf.
[9] Online edition at: https://www.collinsdictionary.com/dictionary/english.
[10] Online edition at: https://www.macmillandictionary.com.

variety of processes. A "computer program" is defined as a set of instructions stored inside a computer that allows the user to do a particular thing.

That said, in its ordinary meaning "program for computers" is understood as a set of instructions.

The **systematic method** determines the meaning of law by considering general legal context, including use of the same term in other legal acts. In many instances the term "computer program" is understood as a set of instructions, namely as a source code and object code. For example, Article 10 of the Agreement on Trade-Related Aspects of Intellectual Property Rights (TRIPS) states that computer programs, whether in source or object code, shall be protected as literary works under the Berne Convention. Source code is a set of instructions for computer written in human-readable programming language. Object (machine) code is a source code translated into machine-readable form. It represents a series of bits (zeroes and ones) that cannot be read by humans.[11]

WIPO Model Provisions on the Protection of Computer Programs also define computer program as a set of instructions capable, when incorporated in a machine-readable medium, of causing a machine having information-processing capabilities to indicate, perform or achieve a particular function, task or result.

U.S. Code (Title 17, Section 101) states that a "computer program" is a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result.

Thus, systematic interpretation also shows that "program for computer" is generally understood as a set of instructions.

The third method is the **purposive approach**. This technique interprets the enactment in the context of the law's purpose. There are at least two possible options here.[12] The first one is that the EPC's purpose is to distinguish the subject matter of patent law from that of copyright law. Thus, it is reasonable to assume that "program for computer" as such refers to the set of instructions.

The second one is that the purpose it to exclude all elements of a computer program. Expressive elements because they are protected by copyright. Functional elements because they are primarily mathematical methods and rules (methods) for performing mental acts (both excluded by Article 52).[13] Algorithms and interfaces are not directly excluded by Article 52, but as it is correctly noted in doctrine, "distinctions here are artificial, as any computer processes can be modelled mathematically or claimed to be a sequence of rules for performing acts".[14]

However, there are no sufficient grounds for such broad interpretation of this exclusion. Firstly, mathematical methods and rules (methods) for performing mental acts are

[11] Pamela Samuelson, Thomas Vinje, & William Cornish, "Does Copyright Protection Under the EU Software Directive Extend to Computer Program Behaviour, Languages and Interfaces?", page 3, at: http://ssrn.com/abstract=1974890.
[12] Brad Sherman, "Computer Programs as Excluded Patentable Subject Matter", page 4, at: https://www.wipo.int/edocs/mdocs/scp/en/scp_15/scp_15_3-annex2.pdf.
[13] Brad Sherman, page 4.
[14] Philip Leith, "Software and Patents in Europe", Cambridge University Press, 2011, page 141.

mentioned by Article 52 separately, so it is unreasonable to include them into "program for computer" definition. In particular, the EPO's practice confirms that programming is not equaled to mental acts. A process "which at least initially can take place in the designer's mind" can be a mental act and only "to the extent that it is a mental act it will be excluded from patentability".[15] Secondly, all other interpretation techniques show that "computer program" usually refers to a set of instructions. That said, it is fair enough to conclude that Article 52 of the EPC uses "programs for computers as such" in the same meaning.

This conclusion is supported by the EPO's approach in the cases of *MICROSOFT/Clipboard Formats I and II (2006).* The EPO noted that a computer program is a sequence of instructions, which just has a potential of being performed and achieving an effect when loaded into, and run on, a computer. An invention implemented in a computer system represents, instead, a sequence of steps actually performed and achieving an effect.[16] These decisions distinguish CII from computer program "as such".

The opinion that computer program per se is a set or combination of instructions, which enable the computer hardware to perform certain function, can also be found in doctrine.[17]

The same approach is presented in the US case *Allvoice Developments US, LLC v. Microsoft Corp. (2015)*, where it was recognized that a claim was directed to patent ineligible software per se. The claimed interfaces were "software instructions", and thus did not fall within any of the categories of eligible subject matter.[18] That said, software per se, or program for computer as such, is a set of instructions, a literary work protected by copyright.

1.3. Consistent Terminology

The English version of the EPC uses the term "programs for computers" (in Article 52), while the English version of the EU Directive 2009/24/EC[19] ("**Software Directive**") uses the term "computer program". Despite the fact that literal interpretation and a balanced approach suggest to treat these two terms as synonyms, use of different terminology does not add legal certainty. Moreover, it can be argued that these terms are not fully identical. As noted in doctrine, "software is now run on all kinds of different devices in addition to computers, from mobile telephones to medical instruments, machine tools, and cars, to give a few examples".[20] In this context, "computer program" can be viewed as a general term referring to a program run on any device, while "program for computer" refers only to one particular device (computer).

The same discrepancy exists in the German version of the documents. The EPC refers to "Programme für Datenverarbeitungsanlagen", while the Software Directive to "Computerprogrammen". Only the French version is consistent and refers to "programmes

[15] EPO Case G 0003/08, page 52, at: https://www.epo.org/law-practice/case-law-appeals/recent/g080003ex1.html.
[16] Andrew Murray, "Information Technology Law, Oxford University Press, 4th edition, 2019, page 250; EPO Case T 0424/03 at: https://www.epo.org/law-practice/case-law-appeals/recent/t030424eu1.html; EPO Case T 0411/03 at: https://www.epo.org/law-practice/case-law-appeals/recent/t030411eu1.html.
[17] Rosa Maria Ballardini, page 11.
[18] Eric Sutton, pages 59-60.
[19] Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the legal protection of computer programs.
[20] Catalina Martínez, "Expanding Patents in the Digital World: The Example of Patents in Software", page 57, at: www.i3pm.org/files/misc/CEIPI-ICTSD_Issue_5.pdf.

d'ordinateur" both in the EPC and the Software Directive, which serves as additional proof that "programs for computers" and "computer programs" are meant to be the same thing.

2. Patentable Elements

2.1. "Not as Such"

Computer program is a complex and pluralistic concept consisting of various elements, which can be protected by different intellectual property rights.[21] The primary debate is between patent and copyright laws.

The term "computer program" has two dimensions. In the narrow sense, it is a set of instructions, a subject matter of copyright protection (literary work). In the broad sense, it also includes other elements (functionalities). As it can be concluded from the above analysis, "program for computer as such" referred to in the EPC's Article 52 is a set of instructions protected by copyright.

There can be at least **two approaches** to patent protection of computer program "not as such". The **first one** is to look for an invention "**outside**" of a computer program, in a certain outcome implemented or produced by it. This approach gave rise to CII and "computer program having a technical character".

The concept of CII is based on the assumption that Article 52 contains a deliberate ambiguity "designed to allow a patent to be awarded to an invention which contains a software element but is not solely software based".[22]

According to the EPO Guidelines, CII is an invention which involves the use of a computer, computer network or other programmable apparatus, where one or more features are realized wholly or partly by means of a computer program.

Thus, although CII involves the use of computer program, is not about software patentability. For the purposes of CII "computer program" is understood in the narrow sense as a set of instructions. CII is just about inventions related to software. The European Commission Report of 2008 states that the notion of a CII has been introduced in order to distinguish such inventions from software inventions.[23] A good example of CII is *VICOM case (1986)*, where invention was examined as if software was not present, and patent claim was accepted because computer program "does not form part of the image processing methods claimed".[24]

As for "computer program having a technical character', it is patentable if it produces "further technical effect". However, this is not the only justification for protection, for the reasons described below, as this concept (as opposed to CII) is based on a broad understanding of "computer program".

---

[21] Rosa Maria Ballardini, page 1.
[22] Andrew Murray, page 245.
[23] European Commission, "Study of the effects of allowing patent claims for computer-implemented inventions. Final Report and Recommendations", June 2008, page 5.
[24] Andrew Murray, page 246; EPO Case T 0208/84 at: https://www.epo.org/law-practice/case-law-appeals/recent/t840208ep1.html.

The **second approach** is to look for an invention "**inside**" a computer program, in those functional elements which remain after we eliminate copyrightable part and other unprotected elements excluded by Article 52. This approach helps to justify why computer program having technical character (producing "further technical effect") deserves patentability, as already recognized by the EPO.

The EPO's Board of Appeal's decisions in the *IBM cases (1998 and 1999)* state that computer programs "as such" are "mere abstract creations lacking in technical character", while computer programs having technical character (producing "further technical effect") are not "as such" and can be patentable.[25] These decisions distinguish computer programs "as such" from "not as such".

However, technical character is a separate requirement applicable to any invention, so it does not seem a good criterion here. The "further technical effect" concept only reflects the specifics of technical character requirement for software. It seems more correct to conclude (as discussed above) that computer program "as such" is a set of instructions (narrow sense), while computer program "not as such" also includes patentable functional processes (broad sense).

It should be noted that interpretation suggested in this paper allows to conclude that there is no conflict between the *IBM* and *MICROSOFT* cases (as discussed above). The potential discrepancy between *EPO Cases T 0424/03* and *T 1173/97* was considered by the EPO's Board of Appeal in 2008-2010 following the President's of EPO referral concerning patentability of programs for computers (referral was found inadmissible). The EPO's Board of Appeal Opinion of 12 May 2010[26] states that there is no conflict between the cases, as the difference between the positions is a legitimate development of the case law.

Indeed, the *IBM case (1999)* distinguishes computer program "as such" (narrow sense) from "not as such" (broad sense), while *MICROSOFT case* distinguishes computer program "as such" (narrow sense) from CII. CII is based on the narrow understanding of computer program, while "computer program having a technical character" is based on the broad understanding of computer program. Thus, it is not only the case law development, where CII emerged in addition to "computer program", but also two different meanings of the same term ("computer program").

That said, "computer program having a technical character" is patentable not because of "further technical effect", which is an undisputable pre-requisite, but because this concept is based on a broad understanding of computer program "not as such" which might contain patentable elements. Such patentable elements produce "further technical effect". In order to figure out such elements, we need to analyze computer program in a broad sense.

2.2. Computer Program in a Broad Sense

In addition to literary work, which is protected by copyright, computer program has functional elements. The problem is that in a "computer program" expressive and functional

[25] EPO Case T 0935/97 at: https://www.epo.org/law-practice/case-law-appeals/recent/t970935eu1.html; EPO Case T 1173/97 at: https://www.epo.org/law-practice/case-law-appeals/recent/t971173ex1.html; Andrew Murray, page 248.
[26] EPO Case G 0003/08, see above.

elements are often inseparable. This is called "idea-expression dichotomy".[27] Thus, as it is noted in doctrine, "while courts continue to try to distinguish between program expression and program functionality, this distinction has proven elusive".[28]

In the case *C-406/10* (*SAS Institute Inc. v. World Programming Ltd.*) ("**Case C-406/10**"), considered by the European Court of Justice, there is a conclusion (described in detail in the Opinion of Advocate General) that functional elements of a computer program can be copyrightable to the extent they constitute the expression or a substantial part of the expression of the author's own intellectual creation (to be ascertained by national court in each particular case).[29]

That said, it can be concluded that to the extent they do not constitute such expression or its substantial part, they can be patentable (provided that patentability requirements are met). The question is where are these functional elements.

A broad approach to a "computer program" can be found in the Software Directive. Recital 7 contains a definition of computer program. For the purpose of the Directive, the term "computer program shall include programs in any form, including those which are incorporated into hardware". The deficiency of this definition is that it uses the defined term itself ("program"), therefore, it does not actually explain the meaning of a "program". However, if read together with Recitals 10 and 11, the Directive seems to approach "computer program" broadly.

According to Recital 11, only the expression of a computer program is protected by copyright and not the ideas and principles which underlie any element of a program. Firstly, it implies that computer program is something broader than its expression. Secondly, it mentions that computer program has elements.

Furthermore, Recital 10 defines the function of a computer program, which is to communicate and work together with other components of a computer system and with users. This is ensured by logical and, where appropriate, physical interconnection and interaction between elements of software and hardware, known as "interoperability" (ability to exchange information and mutually use the information which has been exchanged). The parts of the program which provide for interconnection and interaction are known as interfaces.

That said, functional elements of a computer program are about communicating, working together, interconnection and interaction, exchange and mutual use of exchanged information. These processes are caused by the set of instructions, result in a program's behavior and can be called "computational processes".

As it is noted in literature for software engineers, programs are patterns of rules that direct computational processes. Such computational processes execute programs, perform tasks

[27] Rosa Maria Ballardini, page 34.
[28] Pamela Samuelson, "Functionality and Expression in Computer Programs: Refining the Tests for Software Copyright Infringement", page 2, at: https://ssrn.com/abstract=2909152.
[29] SAS Institute Inc. v. World Programming Ltd., C-406/10, Judgment of the European Court of Justice and Opinion of Advocate General.

and manipulate abstract things called data.[30] There are therefore valid reasons to protect such processes contained in computer programs.

3. Technical Effect

3.1. Definition of "Technical"

According to the EPC, in order to be patentable an invention must have a "technical character". This is a traditional approach to inventions, which initially were represented primarily by machines. As it is noted in doctrine, many years have been spent attempting to define the meaning of "technical effect" for software.[31] As of now, "further technical effect" is needed for computer programs (in addition to normal technical effect).

Indeed, one of the most common problems is that technical effect is a characteristic of a machine, while software should not be treated as a machine. Software should be treated as software. That said, we should either invent a new criterion for software, which would replace "further technical effect", or understand what "technical effect" (including "further technical effect") could be with regard to software. For example, a possible alternative could be something similar to "useful, concrete and tangible result" as has been considered in the US case *State Street Bank & Trust Co. v. Signature Financial Group, Inc.* ("**State Street Case**").[32]

There are at least two arguments in support of "technical effect". Firstly, technical effect does not necessarily mean machine effect. The word "technical" has a much broader meaning. Secondly, the *State Street Case* test has proved to be ambiguous, and incapable of being the sole test of determination of patent eligibility.

The term "technical" is not defined by the EPC or EPO Guidelines. Moreover, in 2010 EPO has already noted that they do not attempt to define this term.[33] Probably the reason is that this definition is not about "technical considerations" present in almost any computer program. As long as "further technical effect" is required for a computer program, it is about some "further technical considerations" of a programmer, something beyond "merely finding a computer algorithm to carry out some procedure".[34] In each case this will be something specific depending on circumstances.

However, the EPO specifies that "further technical effect" of a computer program is a technical effect going beyond the "normal" physical interactions between the program (software) and the computer (hardware) on which it is run. The normal physical effects are, for example, the circulations of electrical currents in the computer.

Nothing in this definition suggests that software is treated as a machine. The only purpose here is to find some outcome of intangible processes, which is capable of industrial application. Indeed, "a computational process cannot be seen or touched, it is not composed

[30] Harold Abelson and Gerald Jay Sussman, Structure and Interpretation of Computer Programs, 2nd edition, 1996, pages 1-2.
[31] Philip Leith, page 8.
[32] Andrew Murray, page 248.
[33] EPO Case G 0003/08, page 23, see above.
[34] EPO Case G 0003/08, pages 53-54, see above.

of matter. However, it is very real. It can perform intellectual work, it can answer questions, it can affect the world".[35]

It is therefore very different from traditional tangible inventions, which the patent law was initially designed to protect. However, not all tangible inventions are machines. New technologies have created a great variety of new inventions. For example, biological material is also not a machine.

Moreover, in case of tangible inventions, patent law does not protect a particular physical object, but rather an idea of that invention, which is as much intangible as the idea of software invention. The whole essence of legal protection by patents lies in the non-rivalrous nature of inventions. As opposed to tangible assets, possession of idea by one person does not prevent others from acquiring it, that is why specific mechanisms of patent law are required.[36]

The only difference is that historically, as initially described by Plato in his theory of forms (ideas), there were physical things of material world and non-physical ideas of such things. In other words, an idea was represented in a certain physical object. With the development of technology, certain ideas are no longer represented physically. Instead, they are represented in things, which are as intangible as ideas themselves.

This intangible nature of software might create certain difficulties in assessment and examination of patent claims, including in terms of novelty and inventive step, but this does not affect its patentability per se. Any new technology has its specifics, so software is not unique. For example, "biological material cannot always be described sufficiently in writing for patent disclosure purposes, and must be deposited physically".[37]

That said, a new technology should definitely be "viewed as a technology on its own merits rather than via the legal fiction that it is something else".[38] Software is not a machine, and is not a tangible asset. Thus, it cannot produce machine technical effect. This means that "further technical effect" should be interpreted broadly, not in a traditional meaning of machine effect, but rather as a certain result that can be industrially applicable. The EPO's definition of "further technical effect" is fully consistent with this approach.

A good example of broad interpretation can be found in copyright law. Computer program is not a literary work, but it is protected by analogy as literary work within the meaning of the Berne Convention. In *Case C-406/10* Advocate General has compared computer program with a novel, saying that formulae and algorithms used by a programmer are the equivalent of the words by which the poet or the novelist creates his work of literature. The way in which all of these elements are arranged represents the style in which the computer program is written.[39] That said, as long as computer program has a literary style as a novel, it can have a "technical effect" as a machine. However, in both cases this is a kind of analogy, because these are not a novel's style and machine effect in their traditional sense.

[35] Harold Abelson and Gerald Jay Sussman, pages 1-2.
[36] Amy L. Landers, Understanding Patent Law, 3rd edition, Carolina Academic Press, 2017, page 11.
[37] Justine Pila & Paul Torremans, "European Intellectual Property Law", Oxford University Press, 2nd edition, 2019, page 111.
[38] Philip Leith, page 156.
[39] SAS Institute Inc. v. World Programming Ltd., C-406/10, Opinion of Advocate General.

3.2. Ambiguous "Usefulness"

It seems that any simplified substitute to "further technical effect" requirement will be a very low threshold, making almost any computer program patentable. The practice of the US shows that it is not easy to develop a simple criterion.

In addition, there is no guarantee that liberalization is needed. In particular, in 2010 the EPO had an opportunity to collect opinions of business and public, which showed that only 10% argued for wider patentability. Others were either arguing for roughly the same conditions (around 30%), or saying that granting practice should be restricted.[40]

The US practice developed at least three main tests to determine software patentability: technological arts test, machine-or-transformation test, and usefulness test.

In *Diamond v. Diehr (1981)* the court ruled that if a process is performing a function of transforming or reducing an article to a different state or thing, then the claim satisfies the requirements for patentability.[41]

In 1998 the *State Street Case* became a point of "diversion of paths between the US and Europe".[42] The court ruled that anything producing a "useful, concrete, and tangible" result is patent eligible, and is not an abstract idea. Thus, "usefulness" became the primary requirement for patentability in the US.[43] This was a very low threshold, and was criticized in literature.[44]

Later in *E. Bilski v. Kappos (2010)* case the Supreme Court overruled the State Street outcome. The "machine-or-transformation test" (similar to the one in Diehr case) was said to be "a useful and important clue, an investigative tool, although not the sole test for deciding whether an invention is a patent-eligible process". According to this test, a process is patentable only if it is tied to a particular machine or apparatus, or it transforms a particular article into a different state or thing.[45] Usefulness was no longer a sufficient criterion.

In *Alice Corp. Pty. Ltd. v. CLS Bank Int'l et al (2014)*, the Supreme Court set forth the two-part test for patent eligibility. The court should first identify an abstract idea, take it out of the claim and then see if the remaining elements are directed to "significantly more" than the previously identified abstract idea.[46] In particular, the court added that claims "do not purport to improve the functioning of the computer itself or effect an improvement in any other technology or technical field".

Thus, it can be concluded that with *Bilski* and *Alice* cases the US approach is not totally different from the EPO's "further technical effect". Also, the two-part test in Alice case is very similar to the conclusions of this paper about taking out copyrightable and other parts

[40] EPO Case G 003/08, pages 1-3, see above.
[41] Eric Sutton, pages 68-69.
[42] Philip Leith, page 19.
[43] Philip Leith, page 19; Andrew Murray, page 248; Eric Sutton, page 72.
[44] Robert E. Thomas, "Debugging Software Patents: Increasing Innovation and Reducing Uncertainty in the Judicial Reform of Software Patent Law", Santa Clara High Technology Law School, Vol. 25, Issue 1, Article 7.
[45] Eric Sutton, pages 73-74.
[46] Eric Sutton, page 89.

excluded from patentability, and looking at the remaining computational processes and their "further technical effect".

Conclusion

A proposal to adopt the EU Directive on patentability of CII failed in 2005, and since that time there has been no similar attempt. If there will be another attempt in the future, the new Directive should cover not only CII, but also "computer program having a technical character".

In addition, it should be noted that EPO is also an important level of European intellectual property harmonization.[47] The conclusions of this paper might be useful in the EPO's practice, including in case of revision of the EPO Guidelines.

The key outcomes that might help to resolve the existing issues can be summarized as follows:

- "Program for computers as such" in Article 52 of the EPC shall be understood as a set of instructions, a literary work protected by copyright (in the narrow sense);
- The CII concept is based on the narrow understanding of computer program;
- The concept of "computer program having a technical character" is based on the broad understanding of computer program ("not as such") as including not only set of instructions, but also functionalities (computational processes);
- Computational processes represent functional elements of a computer program and can be patentable to the extent they (a) do not constitute the expression or substantial part of expression of the author's own intellectual creation, (b) do not fall under the exclusions of Article 52, and (c) produce "further technical effect", which should be interpreted broadly as a certain result that can be industrially applicable.

To conclude, computer programs are patentable in Europe, and justification for patentability is not only in "further technical effect", but also inside a computer program, in its structure and organization. To the extent such elements represent processes or other patentable categories, nothing in the existing regulation prevents them from patentability, provided that other general requirements (including technical character) are duly met.

[47] Laurent Manderieux, "Towards a Unitary European IP Architecture", Winning with IP, edited by Adam Jolly, November 2019.