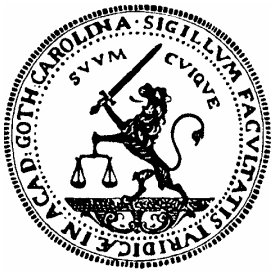

FACULTY OF LAW
University of Lund

Emma Lundgren

"Trivial" Software Patents

ABSTRACT:
Inventive step is the requirement for patentability under the European patent system which, applied at a sufficient level, is a way of ensuring quality patents. Lack of quality, with its negative effects on innovative activity and its detrimental implications for competition, is one main concern of opponents of software patents and of harmonization of software patent law in the EU. This thesis suggests that the solution to this problem is not to hinder harmonization, but to facilitate for better examination procedure and better patent applications, which may very well be better achieved through harmonization, than without it.

Master thesis
20 points

Supervisor
Hans Henrik Lidgard

Intellectual Property Law

Spring 2007

Contents

Summary

Computer programs are excluded subject matter “as such” under the EPC. However, once “technical character” can be established in a software invention, and it fulfills all the requirements for patentability – being an invention, which is novel, has inventive step and is industrially applicable – software is very much patentable. The technical character requirement was originally connected to the invention criteria, but has been appropriately moved towards being part of the assessment for inventive step. Inventive step is concerned with the level of novelty required for a good quality patent. A patent lacking inventive step, but which is nonetheless granted, is called a “trivial” patent – which means “novel but lacking inventive step”.

Patents should provide incentives for innovation and facilitate for diffusion of technology that offset the negative effects the exclusive rights of patents have on competition. Therefore it is essential that granted software patents uphold a certain level of quality. Trivial patents are detrimental to innovation as they skew innovation incentives by, e.g. facilitating for an excessive amount of patent applications for low quality inventions and thus crowding the patent offices with work. Something that, in turn, requires a high level of skill of the patent examiners to “sort” the good from the bad. These problems are heightened by the current inability at the patent offices to keep up to date with the fast moving pace of the development in the software industry and a somewhat inconsistent application of patent law to software.

The solution of the trivial patent problem, in my opinion, must be to look at the possibility of making amendments to the current assessment procedure of inventive step in software inventions rather than putting up a ban against them. Raising the standard of inventiveness required, but also granting the patent offices some more time to adjust to this fairly new field of technology would help. Furthermore, harmonization of software patent law on the EU level, possibly along the lines of what was proposed in the CII Directive, would make for more conform and predictable assessment of inventive step in software inventions, thus providing for legal certainty.

In conclusion, there is no need for a prohibition on software patents, which are already allowable, or for strong opposition towards harmonization. The best step forward is to facilitate for an improved examination procedure, which will be better achieved through harmonization, rather than without it.

Preface

Thank you, sister, for inspiring me to find this topic by talking to me on the phone for half an hour about World of Warcraft, in a very knowledgeable way, despite having no clue what it is. And thank you Lisa, for reading it all and providing invaluable comments, but most of all for just hanging out this year. Last, but not least, thank you Tess for providing the snacks.

Abbreviations

ABS	Anti-lock Braking System
CAFC	United States Court of Appeals of the Federal Circuit
CD	Compact Disc
CII(s)	Computer-Implemented Invention(s)
DVD	Digital Versatile Disc
EC	European Community
EP	European Parliament
EPO	European Patent Office
EPC	European Patent Convention
EU	European Union
ICT	Information and Communication Technology
IP	Intellectual Property
IPR(s)	Intellectual Property Right(s)
OS	Operating System
R&D	Research and Development
UK	United Kingdom
US	United States
U.S.C.	United States Code
USPTO	United States Patent and Trademark Office
WIPO	World Intellectual Property Organization

1 Introduction

Today, the patentability of software is a hot topic at the European Patent Office (EPO) and in the European Union (EU). Despite the rejection of the proposed Directive for computer-implemented inventions (CIIs) by the European Parliament (EP) in 2005,[1] the lively debate has persisted. The attention given to software patents is not surprising in light of a steady increase in claims for them, and that, over the past few years, it is the category of patent applications that has had the highest growth rate. The main aim of the EPO, when granting patents, is to distinguish genuine technological innovations from ones that are mere variations on existing products and processes. This makes it increasingly important to ensure the quality of claims for software patents.[2] High quality in patents in the information and communication technology (ICT) industry, including software, is also recognized as one of the top priorities of the European patent system.[3]

Demanding a certain level of inventiveness in patent-protected subject matter is essentially a question of demanding quality. Opponents of software patents are of the opinion that the nature of software, and the software industry, is not well suited for patent protection. Furthermore, it is claimed that it is not possible to maintain a high level of quality in software patents, leaving the market with an excessive amount of "trivial" patents, i.e. patents upholding an insufficient level of inventive step. That, in turn, will inevitably hamper innovation in the software industry, since it cannot provide appropriate incentives for innovation. This may also lead to competition concerns because, e.g. the exclusive rights provided by patents in combination with low quality software may block development of superior products and processes. Proponents, on the other hand, are convinced innovation will not be restricted by patents and that they in fact, as long as high quality is demanded, will help the industry progress. Opinions clearly differ greatly between both sides, and at this point no common ground seems to be in sight.

This thesis means to take a closer look at how software patents are treated under the European patent system today, especially with regards to the assessment for inventive step and its implications for the quality of granted patents. Furthermore, what may be done to improve the quality of patents

[1] European Parliament Legislative Resolution on the Council Common Position With a View to the Adoption of a Directive of the European Parliament and of the Council on the Patentability of Computer-implemented Inventions (CII Directive).
[2] http://www.epo.org/focus/issues/computer-implemented-inventions.html.
[3] Speech/06/453, Charlie McCreevy, European Commissioner for Internal Market and Services, *Closing Remarks at Public Hearing on Future Patent Policy*, public discussion on future patent policy in Europe, Brussels, July 12, 2006, available at: http://europa.eu/rapid/pressReleasesAction.do?reference=SPEECH/06/453&format=HTML&aged=1&language=EN&guiLanguage=en.

granted and the possible impact of EU-level harmonization in the field will be evaluated.

1.1 Purpose and delimitations

The purpose of this thesis is to evaluate whether the problem of granting "trivial" software patents provides a valid argument for the continued non-harmonization of patentability of computer software in EC law.

To find a possible solution to this problem, a number of questions present themselves:

- What are "trivial" patents, what is problematic about them, and what are the consequences of granting them?
- What implications do trivial software patents have on incentives to innovate?
- What is entailed in the concept of inventive step, why is it important, and how is a sufficient level of inventive step ensured by patent examiners at the EPO?
- What are the economic aspects that have to be considered?
- What possible properties does the software industry have that makes patenting more controversial than in other industries?

The discussion of this thesis will be limited to patents on computer programs/software, as defined in Section two, in a European context. US patent law will be used for comparative reasons and for illustrative purposes.

1.2 Method and material

Traditional legal method will be used for assessing the laws and case law concerning patents and computer programs/software on the European level, to paint a picture of how the state of software patent law has developed and what it looks like today. Moreover, it will be used in developing a sense of how inventive step is assessed. A descriptive analysis will be used for explaining key concepts and basic economic theories under which much of patent law should be evaluated. This approach will furthermore be used for the section on the proposed CII Directive.

A law and economics perspective will be taken on the laws concerning patentability of software and the economic implications of "trivial" software patents. Intellectual property (IP) law in general, and patent law in particular, is much affected by economic considerations, which makes it appropriate to take this perspective into consideration.

For the section on US software patent law a comparative perspective will be taken. However, this will only serve explanatory ends and does not intend to be anything more than an overview for illustrative purposes.

The European Patent Convention (EPC), its Guidelines (EPO Guidelines)[4] and the case law of the EPO Boards of Appeal provide the legal framework upon which this thesis is based. Furthermore, there is a large amount of articles written on different aspects of this topic. However, many of them are colored by strong, idealistic arguments, something that has been taken into consideration when assessing the material. For the section on US software patent law, legal texts and case law will be summarized, and the literature used has been chosen because it provides for general comments on patentability and software in the US system.

1.3 Outline

Section two will present certain definitions of specific terms used in the software field. The legal framework will then be provided for an overview of the requirements for patentability and the state of the law as it is today in the European patent system, as stated in the EPC and the way it has been developed through case law. Furthermore, the economics of software patents and the particular characteristics of the software industry will be presented.

Section three will go on to explain the concept of "trivial" patents and what problems may arise when they are granted. Moreover, the patentability criteria of inventive step will be presented and elaborated on in more detail.

Section four will give a brief summary of the proposed CII Directive for that was rejected by the European Parliament in 2005. Section five will then provide for a brief comparison with the legal development and current practice for software patents in the US.

Section six will analyze what has been found in the previous sections, from which the conclusions presented in Section seven will be drawn.

[4] Guidelines for Examination in the European Patent Office (EPO Guidelines).

2 Software Patents

2.1 Definitions

The terms computer program, software and computer-implemented invention may not always be clearly distinct from each other, creating some confusion on what is meant by what. This part will explain the different concepts, what they entail, and how they relate to each other.

2.1.1 Software and computer programs

Both computer programs and software are forms of what is a major part of technology, generally known as **information technology**. Software, however, is a larger concept than computer programs, because, e.g. software may be used on other hardware than computers.[5] A computer program, as the name indicates, is run on a computer. Examples of computer programs are e-mail programs, word processors and anti-virus programs, whereas examples of software are operating systems (OS), DVDs, CDs and databases. Notably, all the examples of computer programs are also software, but the software examples are not all computer programs.

Software is

> "a composite term comprising all the material required by a computer user to operate and control a computer including the program, program description and supporting material."[6]

Simply put, the software is used to give instructions to the hardware; the hardware reads the instructions provided by the software and performs the task assigned by the instructions. **Computer programs** may be defined 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."[7]

Computer programmers make software or computer programs by designing **algorithms**. An algorithm is a "detailed sequence of actions intended to perform a specific task"[8], i.e. a set of numeral instructions on how to carry

[5] Hardware is for example a computer or CD-player.
[6] WIPO Model provisions on the protection of computer software, Geneva 1978.
[7] Id.
[8] Proposal for a Directive of the European Parliament and of the Council on the Patentability of Computer-implemented Inventions, COM(2002) 92 final, *Explanatory memorandum*, p. 7.

out a task, which can be applied in many different ways.[9] After the algorithm is designed, the programmer writes computer **source code**, which describes the algorithm in purely *functional* terms.[10] Then the source code is provided to a computer, which transcribes the code into **object code** that is not human-readable.[11] Therefore, computer programs and software are, on their own, nothing more than representations of mathematical formulae, which makes them, as such, abstract and mental in nature.[12] These features add to the complexity of software because it gives it attributes that are clearly nothing more than numbers, but on the other hand, skillfully handled, these numbers made into code may perform tasks that provide a completely new dimension to them.

2.1.2 Computer-implemented inventions and software patents

According to the EPO definition a **computer-implemented invention** is an expression intended to cover claims that involve computers, computer networks or other conventional programmable apparatus. The novel feature of the claimed subject matter should be realized by means of a program, meaning that if there is a part of the invention that is implemented by means of a computer, but that part is not the novel one, it is *not* a computer-implemented invention.[13] Computer programs are a form of computer-implemented invention. The core part of a computer-implemented invention may lie in an apparatus, a system, an algorithm or method, the processing of data, a network or the software itself. Examples of products using computer-implemented inventions are: mobile telephones, DVD players, aircraft navigation systems, and anti-lock braking systems (ABS).

What is in literature, and will be herein, referred to as **software patents**, are in essence computer program related inventions, but not strictly computer-implemented inventions. There is no formal definition of a software patent, however, it may be described as

> "patents that claims as all or substantially all of its invention some feature, function or process embodied in a computer program that is executed on a computer."[14]

The distinction between a computer-implemented invention and a software patent lies in that

---

[9] Zoracki, Allen Clark – *When is an Algorithm Invented? The Need for a New Paradigm for Evaluating an Algorithm for Intellectual Property Protection*, 15 Albany Law Journal of Science and Technology 579, 2005, at 581.
[10] As opposed to *structural* terms, in which patent claims are written.
[11] Downing, Robbie – *EC Information Technology Law*, Wiley 1995, p. 10.
[12] Paterson, Gerald – *The European Patent System – The Law and Practice of the European Patent Convention*, 2nd Edition Sweet & Maxwell 2001, p. 418.
[13] EPO Guidelines, C-IV 2.3.6.
[14] Gratton, Éloïse – *Should Patent Protection be Considered for Computer Software-related Innovations?*, Computer Law Review & Technology Journal, Vol. VII 2003, p. 224.

> "a software patent concerns an invention about a software-based computer implementation, while a computer-implemented invention is about an invention that *may* be implemented in software".[15]

2.1.3 Relation to copyright

Software has characteristics that make it susceptible to different types of IP protection; patents and copyright may both be utilized for protection of it. The two forms of IPRs do not, however, cover the same 'part' of the software. Copyright protects the source code in which the computer program is written, whereas patents protect the underlying idea or concept.[16] The Computer Program Directive clearly states that what is protected is merely the "expression" of the computer program and not the "[i]deas and principles which underlie any element of a computer program."[17] Patents therefore provide different and broader protection than copyright. Furthermore, the value of the software is often in the commercial potential of the product or process, which lies in what the program *does* and not in the code as such.[18] Legal protection by copyright and patents may exist in a complimentary manner in software.[19]

2.1.4 Summary

To summarize, the problem, essentially, is to make the distinction between the algorithms that the software is comprised of, which is protected by copyright, and the underlying idea, which is potentially covered by patent protection. It is also necessary to be aware of the difference between software and computer programs, the former being a larger concept covering more than the latter. Software patent is the term used for all CIIs and other patents covering software-related products.

---

[15] Bergstra, Jan A. and Paul Klint – *About "Trivial" Software Patents: The IsNot Case*, Science of Computer Programming 64 (2007) 264-285, p. 277.

[16] The code is protected by Council Directive of May 1991 on the legal protection of computer programs 91/250/EEC (Computer Program Directive) OJ L122, 17.5.1991, which provides that all Member States must facilitate copyright protection for computer programs as literary works within the meaning of the Berne Convention for the Protection of Literary and Artistic Works.

[17] Article 1(2), Computer Program Directive.

[18] Domeij, Bengt – *Patenträtt – Svensk och Internationell Patenträtt, Avtal om Patent samt Skyddet för Växtsorter och Företagshemligheter*, Iustus Förlag 2007, p. 61.

[19] It would also be possible to protect software by e.g. trade secret.

2.2 Software patents under the European Patent Convention

European patents[20] are granted by the EPO, which consists of the EPC[21] and the national patent laws of the member states of the EPO.[22] The EPO, through application of the EPC, accepts applications and grants patents on the European level. Once granted, the European patent will have the same effect as a nationally granted patent in the individual states.[23] The European patents, post-grant, are turned into national patents, with the national patent laws applying to them. It should be pointed out that the EPC is *not* an EU instrument.

2.2.1 Requirements for patentability

Articles 52 and 53 of the EPC govern the nature of patentable inventions in the European patent system.[24] The basic patentability considerations are in principle the same for software as for any other subject matter.[25] There are four statutory criteria for patentability under the EPC. The primary requirement, which is separate from the other three, is the existence of an **invention**[26]. Additionally, inventions must, according to Article 52(1) EPC, be **novel**,[27] include an **inventive step**[28] and be susceptible to **industrial application.**[29] Novelty means that the invention must not form part of the state of the art. Industrial application requires the inventions to, as the wording suggests, be industrially useable. The EPC does not define the word invention, however, in Article 52(2) it provides a non-exhaustive list of what is not to be regarded as inventions. The Article states what is *excluded* from patentability in the sense that the subject matter is either *not* regarded as an invention, or it is not an invention that is susceptible to industrial application, and therefore by definition not patentable.[30] The principle behind these exclusions is that abstract and intellectual methods

[20] "Patents granted by virtue of this Convention shall be called European patents." Article 2(1) EPC.
[21] The EPC is a special agreement within Article 19 of the Paris Convention and a regional patent treaty within the meaning of Article 45.1 of the Patent Cooperation Treaty.
[22] The contracting states of the EPO are all 27 EU Member States, Switzerland, Iceland, Lichtenstein, Monaco and Turkey.
[23] "The European Patent shall, in each of the Contracting States for which it is granted, have the effect of and be subject to the same conditions as a national patent granted by that State, unless otherwise provided in this Convention." Article 2(2) EPC.
[24] Article 53 EPC describes what is *exempted* from patentability, which contains subject matter that is considered as inventions but which, with regard to public policy issues, cannot be patentable.
[25] EPO Guidelines, C-IV 2.3.6.
[26] Article 52(1) EPC; for full text see Supplement.
[27] Articles 52(1) and 54 EPC.
[28] Articles 52(1) and 56 EPC.
[29] Articles 52(1) and 57 EPC.
[30] Paterson, p. 405.

should not be patentable.[31] Therefore, these excluded subject matters are only excluded where the patent application relates to them "as such".[32]

An additional requirement for patentability not explicitly stated in the EPC, is that for **technical character**. In the case of subject matter excluded "as such" this requirement provides for a distinction between that and acceptable inventions. This approach to patentability stems form the national laws of the EPO contracting states and the subsequent confirmation in the practice of the Boards of Appeal.[33] It may also be deduced from Rule 27(1)(c) EPC,[34] which states that the description must disclose the invention in such terms that the technical problem and its solution can be understood. Furthermore, Rule 29(1) EPC requires the claims to define the matter for which protection is sought in terms of the technical features of the invention. The concept of technical character has changed over time through its development in case law. When it was first mentioned in *Vicom,*[35] it was mainly a factor taken into consideration when deciding on the subject matter being an invention or not. However, that approach has essentially been abandoned in favor of the assessment being part of the inventive step, but also as inherent in the notion of industrial applicability, because it implies a physical rather than abstract or mental nature.[36] The technical contribution must in itself have inventive step.

2.2.2 Development through case law

Certain principles have been developed in the case law of the Boards of Appeal to help in the interpretation and understanding of the provisions of the Articles of the EPC that deal with excluded subject matter, and with the interpretation of the requirements for patentability. Of most importance here are the cases where the notions of "as such" and "technical character" have been elaborated on. In the field of software and CIIs, most case law concerns business methods, which is outside the scope of this thesis, but certain principles may be used in this area as well.

2.2.2.1 Vicom

How to understand and interpret the exclusion of computer programs from patentability was first tried in the *Vicom* case, although not strictly relating to computer programs, but a computer-implemented method of processing digitally encoded images. In *Vicom*, a distinction was for the first time made between subject matter providing an effect that is abstract (not patentable) and those with an effect that is technical (potentially patentable).[37] The Board of Appeal stated that it was inappropriate to make a distinction

[31] EPO Guidelines, C-IV 2.1.
[32] Article 52(3) EPC.
[33] Paterson, p. 409; For case law see e.g.: IBM*/Document abstracting and retrieving* (T22/85) O.J. EPO 1990, 12; [1990] E.P.O.R. 98.
[34] The Rules may be found in the Implementing Regulations to the European Patent Convention; For full text see Supplement.
[35] VICOM/*Computer-related invention* (T208/84) O.J. EPO 1987, 14; [1986] E.P.O.R. 74.
[36] Paterson, p. 410.
[37] VICOM/*Computer-related invention*, at reason 5.

between a process carried out in software or in hardware and that the decisive issue was whether there was a technical contribution made by the claim.[38] A function-over-form approach was taken and the notion of "technical contribution" was introduced.

The principles of the *Vicom* case was applied to computer programs in the *IBM/Computer-related invention* case:[39]

> "Even if the basic idea underlying an invention may be considered to reside in a computer program, a claim directed to its use in the solution of a technical problem cannot be regarded as seeking protection for the program as such within the meaning of Article 52(2)(c) and (3) EPC."

2.2.2.2 IBM/Document abstracting and retrieving

In the *IBM/Document abstracting and retrieving* case,[40] the Board of Appeal elaborated on what the exclusions in Article 52(3) had in common, and concluded that it was that they did not aim at technical results, but were of abstract and intellectual character.[41] The Board went on and settled that the requirement for technical character or technical contribution was inherent in the practice of the contracting states, and that this requirement was to be seen as implicit in the wording of the EPC.[42]

After the *IBM/Document Abstracting and Retrieving* case, the "contribution approach" began to be used for assessing technical character and contribution. The approach required the claimed subject matter's contribution to the art to be of a technical character. The Boards of Appeal have subsequently in principle abandoned this approach in regards to using it for fulfillment of the requirement of subject matter being inventions.[43]

2.2.2.3 IBM/Computer program product I

In the *IBM/Computer Program Product I*[44] it was concluded that computer programs cannot be considered as having technical character for the very reason that they are computer programs.[45] It was also held that claims which read "a computer program in which…" are not considered claims for computer programs "as such", and are thus allowable so long as the program, when running on a computer brings about, or is capable of bringing about, a technical effect which goes beyond the 'normal' physical interactions between the program (software) and the computer (hardware).

---

[38] Id., at reasons 12 and 16.
[39] IBM/*Computer-related invention* O.J. EPO 1990, 30; [1990] E.P.O.R. 107.
[40] IBM/*Document Abstracting and Retrieving*; See fn 33.
[41] Id., at reason 2.
[42] Id., at reason 3.
[43] Booton, David – *The Patentability of Computer-implemented Inventions in Europe*, Intellectual Property Quarterly 2007, 1, 92-116, at 102; See e.g. *PBS* (fn 47) and *Hitachi* (fn 50) cases.
[44] IBM/*Computer Program Product I* (T1173/97) O.J. EPO 1999, 609.
[45] Id., at reason 6.1.

"As such" was construed to mean programs that are "mere abstract creations, lacking in technical character."[46]

The case concluded that the ordinary interaction between software and hardware could not constitute technical effect because it would make *all* software patentable. Going beyond the 'normal' physical interactions means that there has to be a *further* technical effect, which may be in the form of the technical character of the effect, or if it causes the software to solve a technical problem. This case was the first in which the technical character of the invention was found in the computer program itself.

2.2.2.4 PBS Partnership and Hitachi

If the Board of Appeal in the *IBM I* case introduced the possibility of, and expressed its preference of that kind of proceeding, 'moving' the technical character assessment to that of inventive step, the *PBS Partnership/Controlling pension benefits* case[47] took that reasoning one step further. First of all the Board stated that an activity carried out by a program does not have to be technical in itself. Furthermore, that technical means are used for a non-technical purpose does not necessarily mean that the subject matter has technical character. The contribution approach was said to be very closely related to examination with regard to the requirement of inventive step,[48] so the contribution approach, although abandoned by the Boards of Appeal, took a new shape in the assessment of inventive step.[49] This case represents a shift in the reasoning of the Board of Appeal where it joined the requirements for inventive step and technical contribution together in a way that had not been done prior to this case.

The *Hitachi* case[50] took the reasoning from *PBS* one step further and the statement was made that all methods involving technical means are inventions within the meaning of the EPC. The Board of Appeal hereby reduced the requirement for an invention to any subject matter, or "any method", using technical means.[51] It was stated that in practice, the separate requirement for the claimed subject matter to be an invention is so easily satisfied that it is an insignificant bar to patentability. The only thing required is technical implementing means, which may very well be known in the prior art. Therefore, again it was pointed out that the assessment for inventive step was a more appropriate place for the technical character to be found.

---

[46] Id., at reason 5.2.
[47] *PBS Partnership/Controlling pension benefits* (T931/95) O.J. EPO 2001, 10; [2000] E.P.O.R. 441.
[48] Id., at reason 7.
[49] Booton, at 103.
[50] Hitachi/*Auction method* (T258/03) O.J. EPO 2004, 12; [2004] E.P.O.R. 575.
[51] Id., at 4.7.

2.2.3 Summary

For patentability under the EPC, there has to be an invention, which in turn is novel, contains an inventive step and that is industrially applicable. Furthermore, it has in the case law of the Boards of Appeal, been established that there is also need for the invention to have technical character. With regards to software, this technical character is essential for patentability, much because computer programs are not patentable "as such" under the EPC. The technical character has to be more than the mere interaction between computer program and computer. This assessment has been moved from being essential for the invention criteria to being a part of the inventive step requirement, a development that has had great implications especially in the field of software patents.

2.3 The economics of software patents

> "Patent law essentially is economic law, and its merits must be judged primarily on economical terms."[52]

The patent system has economic considerations as its foundation and objective: to foster innovation and growth through encouragement of invention and diffusion of technology. This makes an economic approach to the patent law, and software patents, both reasonable and necessary. The economic perspective is a utilitarian view where patents should only be granted if they are beneficial to society. Moreover, the patents that are granted should be designed as to minimize the costs and maximize the benefits society can obtain.[53] Strict economic theory may seem, at times, a bit separate from reality as it does not take into consideration such things as fairness or the inventors' 'natural' rights to their inventions, rather than it being more efficient in terms of the welfare of society as a whole to grant exclusive rights.

The question whether software patents provide incentives to innovate is important. The entire patent system is built on the idea that patents, despite their detrimental effects on competition and static efficiency, are needed to provide incentives to invent and by that bring dynamic efficiency. However, there is an important balance to be struck between providing these incentives and the promotion of competition on the market. The balance may be found in different adjustments to the patent system, and the way in which they are provided. One can adjust the duration or the breadth of the patents. However, what this thesis is concerned with is patent height; the proper level of novelty required from a software invention. The economic

[52] Bakels, Reinier and P. Bernt Hugenholtz – EP Directorate-General for Research Working Paper – *The Patentability of Computer Programmes Discussion of European-level Legislation in the Field of Patents for Software* JURI 107 EN, p. 5.
[53] Guellec, Dominique and Bruno van Pottelsberghe de la Potterie – *The Economics of the European Patent System – IP Policy for Innovation and Competition*, Oxford University Press, p. 114.

theory at the foundation of patent law makes certain assumptions – first and foremost it is all about decision-making. The socially optimal level of any activity lies where the costs and benefits outweigh each other and the market is in equilibrium – i.e. where supply and demand are aligned, something that will only happen under specific conditions.[54] However, the perfect condition – **free competition** – under which equilibrium may exist, never occurs in reality. There is always some level of **market failure** to take into consideration.

The problem with the patent system first of all is that it provides the patent holder with a monopoly. Monopoly is the direct opposite of free competition, and is considered a market failure. However, there are also efficiency problems with having no patent system, problems which will be dealt with in the following. It is the task of the policy maker to weigh the negative and positive effects of the different sides and decide on how to make regulation correspond to what is optimal in regards to what society wants to obtain. It is also, to a great extent, a question of static vs. dynamic efficiency. Static efficiency is promoted by free competition in the market and is a here-and-now type of policy, whereas dynamic efficiency is a more long-term efficient use of policy, where some short-term restrictions may have to be incurred to gain the long-term goal of e.g. innovation. A tradeoff has to be made in this respect, as to what is to be regarded as more important, or at least how to balance the two.

Applied to the software industry, what is sought after is an efficient level of innovation and a market where supply of innovation and demand of it is in equilibrium, which will, in turn, lead to maximized profit for companies and welfare for consumers, and society as a whole. This is, however, an unobtainable goal because perfect competition cannot exist and market failures are always present. It is a matter of making the best of the situation at hand. Incentives must be given to inventors to continue their endeavor if technological development is to progress. The problem is to know what those incentives are and how to balance them properly, to come as close to a perfectly competitive market as possible.

2.3.1 Information as a public good

A **public good** is a good that is non-rivalrous and non-excludable in nature. The classic example of a public good is military services, everyone can 'make use' of it, but no one wants to pay since it will be there whether or not the individual contributes to it. Non-rival means that consumption by one person does not hinder the consumption of that very same good by someone else.[55] An example of this would be a copy of the word processing computer program Microsoft Word; one person buys the program, runs it on his computer and uploads it to a file-sharing site for others to download without him losing any of the availability or utility of the program since it

[54] Cooter, Robert and Thomas Ulen – *Law and Economics*, 3rd Edition 2000, pp. 11-12.
[55] Guellec and van Pottelsberghe, p. 49.

has already been installed on his computer.[56] Non-excludable means that it is next to impossible, or extremely costly (not worth it) to exclude anyone from making use of the good once it is known. E.g., it is almost impossible, and definitely very costly, for the developer of the computer program in the previous example to prohibit the user of it from sharing it with others.

Software is an **information good**, which essentially has the same characteristics as public goods, plus the additional favorable feature of no congestion.[57] Information is often expensive to produce and cheap to transmit. Buyers cannot determine the value of the good until they have it, and when they do they will not want to pay for it. Moreover, the buyer can easily transmit the good at transmission cost (and perhaps the cost of a guilty conscience).[58] Look, e.g. to the example of non-rivalry in the previous paragraph where transmission cost is zero, or possibly the cost of five minutes worth of Internet access. This type of good therefore is prone to attract **free riders**; consumers who consume the good, but are not willing to pay for it. Conversely, producers cannot predict or appropriate the social value of the idea. They are additionally unable to exclude the free riders, who do not pay their share of the cost of the product. The producer will thus provide too little of the good, which creates deadweight loss and in turn causes inefficiencies that are detrimental to both the producer and the consumer.[59]

2.3.2 Information goods under free competition and monopoly

Free competition has had to be supplemented with the patent system because, due to their public good nature, information goods will be supplied at an inefficiently low level under free competition.[60] However, inefficiencies also occur under the **monopoly** provided by the same patent system that is there to mend inefficiencies, by 'privatizing' the public goods and making them excludable. It is not a simple task for the lawmaker to weigh the different aspects and interests to decide on a policy that has the least harmful effects on efficiency.

The inefficiencies under free competition occur because the marginal cost of using information is zero. Therefore, the cost of invention is a sunk cost, one which must be incurred once, before the production of the invented good starts. This sunk cost cannot be returned to the inventor if competitors are free to take the information and start competing with the original

[56] Obviously Microsoft has taken precautions against this type of illegal copying of their products by requiring complicated keys, but when searching a file-sharing site for 'Microsoft Word' you will, in addition to the product, find files for pass keys etc. to be able to use the program on your computer.
[57] Cf. a public road, which is a public good, but will at a growing rate of users be 'filled up', thus making it less valuable to old and new users after a certain point.
[58] Cooter and Ulen, p. 126.
[59] Id., p. 128.
[60] Guellec and van Pottelsberghe, p. 49.

inventor as soon as the product is put on the market. This would happen under free competition because information is so easily copied once it has been made public. A further problem for the original inventor is that he must charge a higher price than his competition to recoup the sunk costs, and will thus be driven out of the market by his competitors. Hence, he will not make the innovative effort in the first place, anticipating not being able to gain anything from it. However, re-invention of public goods is a waste of resources and unlimited and free access to them is socially preferable. 'Positive spillovers' arise since the existing knowledge can be beneficial to others than the inventor. The social return of inventions is higher (an unlimited number of people can use it without it losing its value) than the private return, which will be too low (the inventor cannot appropriate the value of the invention to him). Due to the insufficient private returns, the innovation rate will be lower than socially optimal. So, the patent system is in place to make excludable the goods invented and to make it possible to, to some extent, have costs returned to the inventor to make him inclined to invent in the first place. However, the exclusive rights provided by patents generate costs, e.g. by reducing the positive spillovers.[61]

Furthermore, market equilibrium cannot be reached in a monopolistic market. When there is only one supplier, the holder of the patent, he can set prices that will not be subject to competition from others for that specific product, for the term of the patent, and thus the product will be produced at an inefficiently low level. Consumers are made to pay a price higher than the marginal cost of production, and some consumers who would be willing to pay at marginal cost will not be able to buy the good, as it will be supplied in too low quantities. Thus deadweight loss is created.[62] Seemingly, neither the patent system with the provision of exclusive rights, nor the situation without it with information as a public good, are able to bring efficiency on their own. It is therefore the not so simple task of the regulator to create a policy that makes the situation as good as it can be, by providing incentives for efficient behavior on part of both the inventor and the consumer.

2.3.3 Summary

Considerations of an economic nature have to be taken into account when assessing software patent law. First of all, patents makes for market failure in the form of monopoly, which should be avoided generally. However, because of the specific nature of IP, i.e. its public good nature, there would be market failure even without the patent. The crux is to find the right balance between negative effects inherent in the provision of the exclusive rights to the patent holder and the incentives the exclusive right gives for innovation.

[61] Id., pp. 49-50.
[62] Id., p. 78

2.4 The software industry

One of many arguments in the software patent debate is that software is far from suitable for patent protection due to the nature of the industry in which they exist. Innovation in the software industry runs at a fast pace, and the life cycle of a software product or process is often, but not always, fairly short and replaced every few years with an updated version.[63] Product updates are constantly demanded and produced. Think for example on the frequency at which Microsoft makes available a new version of its Windows OS. Windows XP was released in 2001 and was recently 'replaced' by Windows Vista in 2007, leaving a mere six years of life before being replaced by a successor. And these six years are fairly long compared to the one year that passed between Windows XP and its predecessor Windows 2000. One to six years is not a very long life, and it may seem disproportionate to the 20-year patent life. The new software is often built on the knowledge and the code of the older versions, making the developments in the software industry incremental and sequential with contributions made in small steps by different researchers. Guellec points out that it is mainly the production of software that works in this way where pieces are put together into a finished product, but that innovation is different and should be separated from the production.[64] This reasoning has implications for how the incentives mechanism works in relation to software development, which will be further discussed in the following.

Standardization is an important feature in the software industry. Standards are welfare enhancing where they allow economies of scale to lower costs of production. Standardization is often a result of **network effects** and need for interoperability in the software industry. Network effects are present when usability of one consumer is increased by a growing number of other users. Standardization is beneficial in many ways to consumers, because it e.g. provides for software products that work on different hardware, and facilitates for easier after-sales services where all software may be compared. It may, however, result in competition problems when, for example one software manufacturer holds patents for many products and processes made into industry standards, thus shutting out competition for fear of infringement or too high licensing costs. The standards may then be used as a barrier to entry for competition. Standards are also detrimental when they are set by a sub-optimal product or process, and the market is then locked-in. Or because is gives less technical diversity to consumers. Many would argue that Microsoft Windows, which is as much a standard as you may find, is not the optimal operating system on the market, but due to it being set it is difficult to change, and consumers are 'locked in' to a standard which is not the best possible.[65]

---

[63] Burk, Dan and Mark Lemley – *Designing Optimal Software Patents*, Stanford Law School – Public Law & Legal Theory Working Paper Series Research Paper No. 108, 2005, available at: http://ssrn.com/abstract=692044, p. 90.
[64] Guellec and van Pottelsberghe, p. 127.
[65] Id., pp. 105-106.

For the sequential improvements and the standards to work properly, the software industry requires openness and **interoperability** of standards. Interoperability concerns the ability of two pieces of technology to work together e.g. the multitude of software pieces that make up what is Microsoft Windows and its applications. Interoperability, or compatibility is an essential characteristic of software – compatibility with hardware, with other software etc. A word processing computer program that does not work on the OS would be useless and worthless. So would a word processing program which is not compatible with other users' word processing programs, an illustration of the network effects.

In sum, the software industry is characterized by short market lives, importance for standards and network effects for necessary interoperability of products on the market.

3 ”Trivial” Software Patents

Triviality in software patents is a question of quality; what standard should have to be achieved to offset the negative consequences of being granted an exclusive right to exploit a product or process, which in an optimally efficient market would be provided for free? Much concern is directed towards the possibility of there being too many trivial software patents granted in Europe, thanks in large parts to the development in the US (see Section five). With the objective of the patent system in mind, and the economic idea that patents should only be granted if beneficial to society, the quality problem is given an economic foundation. When trivial patents are granted, they will incur costs on society that are not offset by their potential benefits – which makes them undesirable from an economic perspective.

3.1 What is a “trivial” patent?

Depending on the perspective, the definition of a “trivial” patent differs somewhat. The legal definition in the European patent system, which will be used in the following, is “novel, but lacking inventive step.” More informally it is “a patent that describes a small but insignificant advance over the state of the art”.[66] There is general consensus that trivial patents are common in the software industry, something that may be explained by different factors. First of all, the patent examiners at the patent offices are not familiar with this relatively new field of technology and are possibly not entirely equipped to deal with it yet. It is difficult for the patent examiners to keep up to date with the state of the art in this fast moving industry, and effectively compare computer programs with each other to find the inventive step required.[67] Combined with the nature of the process by which software is developed, risks of granting patents to inventions lacking inventive step may be increased. There may also be a problem in the rules to be applied, if they are not sufficient it is rather impossible for the examiner to apply them. These two problems are brought together by the fact that assessment of inventive step does, as current law looks, leave room for interpretation and when the examiner is not sure what is to be expected, the rules will not be applied properly, leaving room for less inventive software to ‘slip through’.

Guellec describes three different main perspectives that can be used in assessing patent quality: the legal, technical and economic notions of a “quality patent”. The legal sense of a quality patent is one that looks to the extent to which there is legal certainty, i.e. if it is probable that the patent will be upheld in court because the examiner has interpreted and applied the law correctly. The technical sense is that the subject matter upholds the right

[66] Bergstra and Klint, p. 264.
[67] Gratton, p. 240.

level of novelty and height; no prior art should be missed, and inventive step must be carefully assessed. Legal uncertainty has negative effects on competition and investment and low technical quality undermines competition and increases business risk. The economic notion of a quality patent takes the assessment one step further. Although an economically qualitative patent requires both legal and technical quality, it also requires the patent to be efficient. A patent, which is in line with the law, hence offering legal and technical quality, may be seen as low quality from the economic perspective if it does not encourage innovation or limits diffusion of knowledge, due to a too low inventive step or too broad scope.[68] A patent that provides the wrong incentives for innovation, despite being legally correct, is not socially efficient to grant. The three notions of quality are rather intertwined and should be taken in assessment together.

3.2 Inventive step

The requirement for **inventive step** is set out in Article 52(1) EPC. What is signifying an inventive step is that the invention is not **obvious (**meaning that which "follows plainly or logically" from the prior art) to a **person skilled in the art**[69] having regard to the **state of the art**[70]. Before assessment for inventive step comes into question, it has to already have been established that the subject matter is in fact an invention, in the sense of Article 52(1), and that it is novel.[71] Where novelty requires the claims to distinguish the subject matter from separate parts of the state of the art, there is inventive step if the invention is non-obvious on basis of the state of the art as a whole. The interpretation of the term inventive step and the standard to which it is to be used has been left up to the EPO.[72] In general, the patentee should be able to rely on an invention being granted a patent to have sufficient inventive step to be, with "a fair degree of certainty", upheld in court.[73]

The extent of the difference between the invention and the state of the art determines the inventive step, a difference that should be significant and essential to the invention. The examiner should see to how technically difficult it was on the date of the claim to take the step from the state of the art to the invention.[74]

The requirement for inventive step emerged as a response to competition concerns. The purpose of the assessment is to avoid granting patents to

[68] Guellec and van Pottelsberghe, p. 115.
[69] Article 54(2) EPC.
[70] The 'state of the art' is defined as: "everything made available to the public by means of a written or oral description, by use, or in any other way, before the date of filing of the European patent application". See Article 54(2) EPC.
[71] Paterson, p. 536.
[72] Visser, Derk – *The Annotated European Patent Convention*, 13th revised edition 2000, p. 77.
[73] AECI/*Thermoplastics sockets* (T1/81) O.J. EPO 1981, 439; [1979-85] E.P.O.R.: B: 273.
[74] Domeij, p. 82.

predictable and natural progressions. If simple novelty is required, any routine improvement could be monopolized, it would block the ordinary business development of the company that did not come up with the solution first.[75] If patents on these simple improvements that any person in that industry could make, troublesome lock-ups will arise in the industry, and the patent system will have failed its goal of incentivizing innovation. The type of improvements one is trying to avoid granting patents to, correspond to those that the skilled man in the art would have made on the date of the application.[76]

3.2.1 Person skilled in the art

For an invention to have inventive step, the **person skilled in the art** should not be able to, at the date of the application and with basis in the state of the art, arrive at anything falling within the claim. If he can, the invention is to be considered obvious.[77] The person skilled in the art is a fictitious person, with certain knowledge and abilities, who will determine the standard for obviousness.[78] Many characteristics have been attributed to this person. He is not the public at large; neither is he a high level scientist. He is a specialist in his field,[79] and at the same time his knowledge may not be extended unrealistically.[80] He has no fantasy or creativity;[81] he simply knows what is **common general knowledge** in his field. Furthermore,

> "generally accepted definitions of the notional 'skilled person in the art' […] have one thing in common, namely that none of them suggests that he is possessed of any inventive capability".[82]

Essentially, he is an ordinary practitioner in a specific technical field with the prior art in that field as his source of information. The only clarity brought to the level of skill of the skilled man in the technical field of software is that he should not be a "programming specialist".[83] When the knowledge of the skilled person falls short, he should go outside the specific field of technology to which the invention belongs, and search patent literature on the subject and related fields, where similar problems occur. In especially research intense industries, such as the software industry, the man skilled in the art is possibly a group of people rather than one single person; otherwise the assessment would be too simplistic and too many patents would be granted.[84]

---

[75] Guellec and van Pottelsberghe, pp. 37-38.
[76] Domeij, p. 83.
[77] EPO Guidelines, C-IV 9.4.
[78] Visser, p. 78.
[79] EPO Guidelines, C-IV 9.3.
[80] Domeij, p. 84.
[81] EPO Guidelines, C-IV, 9.3.
[82] Paterson, p. 546; quoting ALLIED COLLOIDS/*Polymer powders* (T39/93) O.J. EPO 1997, 134; [1997] E.P.O.R. 347.
[83] EPO Guidelines, C-II 4.14a.
[84] Domeij, p. 84.

3.2.2 Problem and solution approach

The only established procedure for evaluating inventive step in the EPO is the problem and solution approach, which has its legal basis in rule 27(1)(d) EPC.[85] It is the official policy of the first instance departments of the EPO and is almost always applied by the Boards of Appeal[86] - the EPO Guidelines state that in practice, "the examiner should normally apply" the approach.[87] The procedure is aimed at standardizing the assessment to some extent, by bringing objectivity and predictability to it, and reducing the potential use of hindsight. It was thought that simply asking whether the invention was obvious to a person skilled in the art was too general and subjective a question.

The problem and solution approach is based on three steps: first, the technical field and the closest prior art are determined; second, the technical problem to be solved is established; and lastly, starting from the closest prior art and the technical problem, it is ascertained whether the claimed invention would have been obvious to the person skilled in the art.[88] The steps effectively bring together the separate requirements for inventive step into one single test.

The closest prior art is, in practice, generally that which corresponds to a similar use and has the most relevant technical features in common with the invention. It may in many cases be what is most closely concerned with the problem underlying the claimed invention. The closest prior art must be selected from the objective practical view of a skilled person in the art at the filing date.[89] When finding, and starting, from the closest prior art, there is no need for consideration of the inventiveness compared to any other prior art. This approach is used to find a consistent standard, which is highly desirable goal for the public confidence in the inventiveness of subject matter being granted a patent.[90]

To establish the objective technical problem, the examiner must first determine it, examine how the inventor solved it, and then decide on the obviousness. If no objective technical problem can be found, there can be no contribution to the art and thus, no inventive step.

> "When assessing inventive step […], it is not a question of the subjective achievement of the inventor, so that the case history of the invention […] is irrelevant […]."

[85] "[D]isclose the invention, as claimed, in such terms that the technical problem and its solution can be understood." For case law see, e.g., T20/81 O.J. EPO 1982, 217 and BASF/*Metal refining* (T24/81) O.J. EPO 1983, 133; [1979-85] E.P.O.R.: B: 354.
[86] Paterson, p. 536.
[87] EPO Guidelines, C-IV 9.8.
[88] Id.
[89] Paterson, p. 540.
[90] Id., p. 539.

The subjective problem formulated in the claim may need to be reformulated in light of objectively more relevant elements originally not taken into account.[91] Important in assessing inventive step is to take the starting point at the closest prior art and not at the invention, to avoid hindsight. However, using the knowledge of the invention is inevitable when deciding the closest prior art. Extreme care must be taken in this regard. The requirement for the invention to make a technical contribution to the art has a close relation to this step of the problem and solution approach, because technical contribution entails a concrete solution to a certain practical problem. If the claim does not have a *prima facie* technical character the claim should be rejected – lack of inventive step have been established sufficiently.[92]

The objective technical problem is the problem that a skilled person would objectively recognize as the problem when comparing the closest prior art to the technical effect of the claimed invention.[93] In the last step, if the person skilled in the art *would* have been prompted to modify or adapt the closest prior art to anything falling within the claim, there is obviousness.[94] For the purposes of this test, the invention must be considered as a whole.[95] The skilled person may only be defined after the objective technical problem has been established, otherwise the technical field may be wrong.

3.2.3 Secondary indicia

There are a number of 'objective circumstances', that can be perceived as indicators of inventive step, which have been established in case law.[96] A general opinion in the field that the presented solution was not possible can be one of those indicators. So can commercial success, as long as the success is due to the technical qualities of the invention and not e.g. marketing.[97] New and surprising technical advantage of an invention, which is of significant technical value, may be yet another indicator of inventive step.[98]

In the assessment for inventive step, it is of great importance how significant the problem was. A considerable technical step indicates inventive step, especially in research-intense fields.[99] Inventive step is found lacking when the invention appears to be a predictable modification of something known and there is no surprising effect. An example is when something, which has previously been performed manually, is done by

[91] Paterson, pp. 541-542; quoting BASF/*Metal refining* (T24/81) O.J. EPO 1983, 133; [1979-85] E.P.O.R.: B: 354 and ALLIED COLLOIDS/Polymer powders; see fn 82.
[92] EPO Guidelines, C-IV 2.3.6.
[93] Paterson, p. 543.
[94] *Would* is to be distinguished from *could*, which would give less weight to the last step. See EPO Guidelines, C-IV 9.8.3.
[95] EPO Guidelines, C-IV 9.8.
[96] Id., C-IV 9.10; Domeij, p. 89.
[97] Id., C-IV 9.10.4; Domeij, p. 91.
[98] Id., C-IV 9.10.2.
[99] Domeij, p. 87. See T192/82 O.J. EPO 1984, 415.

automatic means, unless some specific difficulty has been overcome or there is a surprisingly great result.[100]

3.3 Disclosure and diffusion of technology

One of the benefits of the patent system in general is that it promotes the disclosure of technology. The patent holder trades the publication of his invention for the exclusivity granted by the patent, thus disclosing inventions to other inventors and enhancing efficiency by reducing the risk of re-invention of already invented products and processes. Furthermore, disclosure facilitates further development of the patented products and inventing around of them for provision of substitutes, thus bringing about positive effects for consumers, such as reduction in market prices and greater product variety.[101]

Critics of software patents argue that in the software industry, the disclosure effect of the patent system may be less important because most product and processes, once put on the market, will, due to the public good characteristics, be easily reverse-engineered, decompiled and invented around without need for disclosure. The problem with this argumentation is that if inventions are kept secret instead of patented, the competitors of the original inventor are more likely to simply duplicate the invention rather than coming up with new inventions by inventing around the original invention.[102] Patents are therefore desirable in that they force competitors to provide original solutions due to the risk of otherwise infringing existing patents.[103] The requirement for disclosure may conversely have negative effects when inventors prefer to keep their inventions secret for as long as possible due to the knowledge that once put on the market, the software product or process will be easily invented around. For example, Google has stated that "[a] lot of our best ideas don't get filed as patents because patents eventually become public"[104] providing a picture of how the companies may think in regards to disclosure. What may be concluded is that conflicting results arise from the disclosure that follows with a patent application.

[100] Domeij, p. 88.
[101] Guellec and van Pottelsberghe, p. 75.
[102] Campbell-Kelly, Martin – *Not All Bad: An Historical Perspective on Software Patents*, 11 Michigan Telecommunication Technology Law Review 191 (2005), at 199.
[103] Id., at 202.
[104] Guellec and van Pottelsberghe, p. 74.

4 Proposed CII Directive

The European Commission was induced to take action in the field of software patents after a Green Paper on the Community Patent[105] in 1997 and its follow-up in 1999.[106] The Commission launched a round of consultations in 2000 and the response showed diverging opinions, however, the need for action was clear. A proposal for a Directive on patentability of computer-related inventions (CII Directive) was presented by the Commission in 2002.

> "The proposal for a Directive on the patentability of computer-implemented inventions aims at harmonizing the provisions of national patent law dealing with inventions which rely on computers for their performance. The Directive will bring under the supervision of the European Court of Justice the rules applicable by national courts and patent offices charged with assessing the validity of patents and applications in this field."[107]

Seemingly, the CII Directive sought to create a uniform set of rules for patentability for computer-related inventions in order to avoid divergences in the interpretation of software patents granted by the EPO and by national patent offices.

> "[…] a computer-implemented invention may be protected in one Member State but not in another one, which has direct negative effects on the proper functioning of the internal market. This Directive addresses this situation by harmonizing national patent laws with respect to the patentability of computer-implemented inventions and by making the conditions of patentability more transparent."[108]

The Directive was thus to provide for legal certainty for patent applicants, patent holders and potential infringers. Moreover, it would bring positive effects to the Common Market. The idea was to codify the current practice of the EPO and the Boards of Appeal case law, not to make substantial material changes to software patent law. The proposal included definitions of the essential terms "computer-implemented invention" and "technical contribution", however notably not for "technical".[109]

---

[105] Promoting Innovation through patents: Green Paper on the Community Patent and the Patent System in Europe COM(1997) 314 final, 24 June 1997.
[106] Promoting innovation through patents: the follow-up to the Green Paper on Community Patent and the patent system in Europe COM(1999) 42 final, 5 February 1999.
[107] 2002/0047 (COD) Communication from the Commission to the European Parliament pursuant to the second subparagraph of Article 251(2) of the EC Treaty concerning the common position of the Council on the adoption of a directive of the European Parliament and Council on the patentability of computer-implemented inventions.
[108] Explanatory Memorandum, pp. 2-3.
[109] Article 2, Proposal for a directive of the European Parliament and of the Council on the patentability of computer-implemented inventions.

After the Commission drafted its first version of the CII Directive in 2002 and a long first reading, during which substantial amendments were made to the Commission proposal, the final resolution was taken in September 2003. In May 2004 the Council came to an agreement on a common position, which was not adopted until early 2005. The EP then called for a restart of the consultation process by the Council, and the second reading was performed later that same year, after which the CII Directive was dismissed by a large majority, with 648 out of 732 MEPs voting against it.

As for the parts of the CII Directive important for inventive step, the definition of "technical character" was proposed to be

> "a contribution to the state of the art in a technical field which is not obvious to a person skilled in the art".

Furthermore, this requirement was not to be a substitution for Article 56 EPC, but a qualification of it. The Explanatory Memorandum also states that the assessment is to be made under the search for inventive step.[110]

There was definitely a great divergence in opinions on the appropriateness of a CII Directive, and it was given more attention than most proposals for Directives in the past. According to the consultations made by the Commission, the largest group opposed to software patents was the open source movement, however, about half the respondents were in fact in favor of software patents.[111] It seems as though in the end, the major lobbying of the anti-software patent groups, such as the open source movement, was successful and influenced the EP to vote against the proposal. Thus, the EU stands today without uniform rules for patentability of software, but there is by no means silence on the subject. The dismissal of the CII Directive has certainly had a great impact on the attention given to it in the EU in recent years. There has been debating on the subject consistently since 2005; however, the efforts to regulate the area on the EU level have been put on ice for now, as the Commissioner for Internal Market and Services put it in a speech in 2006:

> "The vote and the far-reaching debate which preceded it demonstrated, in my opinion, that the time is not ripe for a meaningful piece of legislation to be adopted on this issue."[112]

[110] Explanatory Memorandum, p. 13.
[111] *The results of the European Commission Consultation Exercise on the Patentability of Computer Implemented Inventions*, Final Report by PbT Consultants, under contract number PRS/2000/A0-7002/E/98, p. 4.
[112] Charlie McCreevy, *Closing remarks at public hearing on future patent policy*, (see fn 3).

5 Software Patents in US Law

There has been much concern expressed in Europe over trivial software patents and its implications for the patent system as a whole. The opponents of software patents in Europe tend to look at the development in the US in recent years, where the patentability criterion of obviousness (the corresponding requirements of inventive step in US law) for software has been reduced to insignificance. This has led to an over-flow of trivial patents and a significant loss of trust in the patent system. However, there are differences in US and EPO law and practice, and there is no reason to believe that EU legislation would provide for a path down the same road as the US.

5.1 General patentability criteria

The statutory requirements for patentability in US law are similar to the ones in the EPC. The US Patent Act requires first of all that the subject matter be statutory, that means basically that it is a product or process and that it does not fall into any of the excluded categories. Furthermore, subject matter has to fulfill the requirements of utility, novelty and non-obviousness. Title 101 of 35 U.S.C. states that

> "[w]hoever invents or discovers any **new** and **useful** process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title."[113]

Titles 102 and 103 provide the conditions for patentability. Novelty is meant as something that is distinguishable from prior art[114] and the subject matter has to be non-obvious to a person skilled in the art.[115]

The excluded categories of subject matter have, in case law, been said to be laws of nature, scientific phenomena and mathematical formulae. These categories have been excluded due to it, according to courts, granting unreasonable control to individuals to grant exclusive rights such fundamental "scientific truths".[116] The general understanding that "anything goes" in the US may in large parts be because of such cases as *Diamond v. Chakrabarty*[117] in which it was stated that "anything under the sun, made by man" should be patentable.

The corresponding requirement for inventive step in US patent law is that

[113] US Patent Act 35 U.S.C. §101.
[114] Id. §102.
[115] Id. §103.
[116] Mackay Radio & Telecommunication Co. v. Kalo Inoculant Co., 333 U.S. 127 (1948) at 94.
[117] Diamond v. Chakrabarty, 447 U.S. 303 (1980).

for non-obviousness.[118] Title 103 states:

> "(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains [...]."

This requirement asks the invention not to be obvious to a person skilled in the art, much like the requirement for inventive step in the EPC. The difference, essentially, is the requirement for technical contribution that has been developed in EPO practice, which is not in any way present in US law. The differences does not lie so much in the legal texts, but in the way they are applied by the courts, which will be shown in the following more specifically regarding software.

5.2 Patentability of software

Patentability of software in the US patent system has been developed through case law. Hesitation was at first shown towards this particular subject matter in the early seventies. In the *Gottschalk v. Benson* case,[119] the Supreme Court stated that a mathematical algorithm itself was not patentable. However, they did not intend to exclude computer programming from patentable subject matter. Some ten years later, in 1981, in the *Diamond v. Diehr* case[120] it was concluded that an invention could not be denied a patent solely on the grounds that its claims contained mathematical formulae.

In the *State Street Bank* case[121] the Court found that the previously applies exceptions from patentability for mathematical formulae and business methods were to be abandoned. The Court quoted the *In re Alappat* case[122] that had stated that an invention was patentable if it produced "useful, concrete and tangible result[s]" and thereby made mathematical formulae patentable under these conditions. This because it was found that it was more appropriate to focus on the practical utility of the claimed subject matter rather than on its category. The *WMS Gaming* case[123] somewhat limited *State Street* by stating that algorithms are patentable when they limit a general-purpose computer to a specific use, performing functions pursuant to the software. This was furthered by the statement that

[118] See section 3.2.
[119] Gottschalk v. Benson, 409 U.S. 62 (1972).
[120] Diamond v. Diehr, 450 U.S. 175 (1981).
[121] State Street Bank & Trust Company v. Signature Financial Group 149 F.3d 1368, (Fed. Cir. 1998).
[122] In re Alappat, 33 F3d 1526 (Fed. Cir. 1994).
[123] WMS Gaming Inc. v. International Game Technology, 184 F.3d 1339 (Fed. Cir. 1999) quoting In re Alappat, fn 122.

> "the scope of [title] 101 [is] to be the same regardless of the form – machine or process – in which a particular claim is drafted"[124]

in the *AT&T Corp. v. Excel Communications Inc.* case.[125] Two recent cases in the CAFC have brought up the problem of obviousness in connection to software patents: *Lockwood v. American Airlines*[126] and *Amazon.com v. Barnesandnoble.com*[127] where obviousness was viewed as a substantial hurdle for software patenting, however not insurmountable.

When making a comparison, however shallow, with the EPO system, one may conclude that there are stricter criteria in the European system, especially because of the requirement for technical character, which adds another dimension to the assessment. To say that, because laws are less strict in the US and because the practice at the USPTO may need revision, there is cause for hindering software patents in the EU is wrong. Caution should definitely be taken, and there is reason to learn from the 'mistakes' of the US, but it does not automatically mean that the EPO or a similar EU body would be unable to make correct assessments in line with what should be required of a software invention to ensure quality. It is surely so that the US legislator did not mean for trivial software patents to flood the market, it has happened due to other circumstances, outside the scope of this thesis. Neither does it mean that the US rules are necessarily bad, simply badly applied. Thus, to make the assumption that Europe will inevitably go the same way as the US cannot be correct if one has any faith in the ability of the European patent examiners to learn from experience and to adjust to a new market.

[124] AT&T Corp. v. Excel Communications Inc., 172 F3d 1352 (Fed. Cir. 1999).
[125] Schechter, Roger and John Thomas, *Principles of Patent Law*, 2nd ed. Concise Hornbooks, pp. 42-48.
[126] Lockwood v. American Airlines, 107 F.3d 1565 (Fed. Cir. 1997).
[127] Amazon.com v. Barnesandnoble.com, 239 F.3d 1343 (Fed. Cir. 2001).

6 Analysis

> "To determine if software patents should exist, one must balance the incentive to innovate that software patents provide versus the frequency with which potentially obvious software patents may arise."[128]

Looking to the rationale of the patent system as a whole, its two clearly stated main objectives are to provide incentives for innovation and to facilitate diffusion of technology. It is a generally accepted understanding that patents do provide these incentives in most industries – "a high patent activity is generally considered a sign of economic health",[129] it is merely a question of it being more or less pronounced.

As for the software industry, it is likewise undisputed that innovation in software is essential and wanted for technological and economical growth. In today's society, there is an obvious demand and need for an endless number of products using software, such as aircrafts, cars, mobile phones and personal computers, to name a few. Software patenting is however controversial. Certain groups claim that patents are directly harmful to innovation in and the quality of software.[130] Furthermore, it has been stated that current IP law has not taken software into consideration, and therefore the law itself may not be suitable.[131] On the other hand, others say that the software industry is in principle no different than other industries and that it just needs to mature and will then adjust to the conditions of the patent system.

With the concerns for the quality of software patents being so pronounced, it becomes increasingly important to make sure that the expressed goal of maintaining high quality in European patents is in fact lived up to also with respect to software patents. The question of inventive step is particularly interesting in this relation because, where most claims concerning software will pass the invention criteria without substantial problems, or fail on the ground that it is a mathematical method, rather than a computer program "as such",[132] the standard-setting for software patentability falls on the examiners in determining novelty and inventive step.

---

[128] Yang, Grant C. – *The Continuing Debate of Software Patents and the Open Source Movement*, 13 Texas Intellectual Property Law Journal 171, at 205.
[129] Bakels and Hugenholtz, p. 5.
[130] E.g. the Free and Open Source Software movements.
[131] Zoracki, at 581.
[132] The computer program 'as such' is as good as nonexistent.

6.1 Incentives for innovation

The requirement for inventive step, as it is set out in the EPC,[133] provides no room for an evaluation of an appropriate level for its assessment – it is simply a question of, "yes, there is" or "no, there is not" inventive step. However, in a law and economics setting such as patent law, there is almost always a tradeoff to be made, making 'yes and no' questions too simplistic.

There is in this context of economic law a level of inventive step that is socially efficient, and that level should be pursued because it is the level at which the best possible incentives for innovation are facilitated for. Since there is a myriad of different interests and circumstances influencing incentives, intricate work is required to weigh and balance these interests and circumstances against each other in the determination of whether particular software is sufficiently inventive. The level of inventiveness must be adequately high not to grant patent protection to trivial software inventions that is detrimental to innovation. On the other hand, the extent of the inventive step assessment cannot be so rigorous as to incur costs that are higher than the costs of a few trivial patents.

6.1.1 Appropriate level of inventive step

When a low level of inventive step is required for software patents, many patents will be granted and there will be increased competition in the market. Increased competition may at first be seen as a good thing as it may create incentives for providing the best product or process. Assuming that the consumer knows what he wants and knows how to pick the best product (this is probably especially true for software), then low quality software will be pushed out of the market. With the massive flow of information available about software products, it is increasingly difficult for a low quality software product to make it in the market. On the other hand, increased competition leaves the inventor with a lower expected return on investment made, since there will be more competing products to share the market with. This effect will lower incentives for innovation. Hunt points out that this is an effect that is more pronounced in high technology industries,[134] thus providing an argument against the relaxing of patentability criteria of inventive step for software. This argument may however be reversed for a different perspective: where an invention lacking inventive step, and of low quality is granted a patent, it may not be so harmful, because the older product will prevail due to being the one of higher quality. Yet another problem with a low level of inventive step is that the lower the quality of the granted patents, the more likely software developers may try to file their low quality products, disregarding the negative implications this has for the patent system as a whole, because they know that other poorly inventive products have obtained patent protection. This increase in low quality patent

[133] See section 3.2 and Supplement.

[134] Hunt, Robert M. – *You Can Patent That? Are Patents on Computer Programs and Business Methods Good for the New Economy?*, Business Review Q1 ((2001), p. 11.

applications will put further strains on an already burdened patent office and there is an increased risk of trivial patents being granted, leading to a downwards spiral of lower quality patents being granted and thus creating more negative incentives for quality of the inventions being claimed.

Conversely, with a high level of inventive step required, fewer patents will be granted and the expected returns on software innovation will be lower as well, but the incentive to produce quality products is higher since the one with the best product will gain the patent and the market.[135] Supposing that an invention only brings returns as long as it is not subject to competition by new technology, by which it will most likely be pushed out of the market, the economic life of the patent is shorter the heavier the innovation rate is in the specific industry. Since innovation in the software industry is fairly quick, the expected life of a patent should be short and there is less time for the inventor to recoup his investment costs. It is therefore, e.g. important that what is patented is truly living up to a high level of inventive step so that no invention can out-maneuver another without it rightfully doing so and gain a market advantage that is unfair in terms of pushing out a previous high quality invention.[136] A high level of inventive step may also be warranted due to the signaling effect of patents, it is important that other inventors do not get the signal from granted patents that low quality is acceptable. Furthermore, because of the diffusion of technology that takes place when patents are granted, it could be significant whether the technology spread is of high or low quality. This is so because if other inventors see the grant of a patent as a signal of quality, and believes that the patented software maintains a certain quality standard, they may use that as a starting point for further development or inventing around, and will thus end up inventing on a sub-optimal level of inventiveness, due to their starting point being set by the wrong standard.

In the legal sense, it is important to set a quality level of software patents that is possible to maintain for the respect of the patent system not to be lost. The exact optimal level of inventive step to require is a matter for economists to calculate. Arguments can obviously be made in both the directions of requiring low or high levels of inventive step, but the general consensus seems to be that lower inventive step makes for lower quality in patents, which in the long run is detrimental to the incentives mechanism that the patent system is instated to facilitate. The long run advantages of requiring a high level of inventive step for software products and processes are more convincing to me. A fairly high level of inventive step is required to maintain quality in software patents and it seems the bad effects of a low level threshold for inventive step are worse than the possible incentive-raising effect of lowering the standard – which could make things worse still, and reduce incentives in the long run because of the lowered chance of recoupment of R&D investment costs where there are more patented products to compete with.

---

[135] Hunt, Robert M. – *Economics and the Design of Patent Systems*, 13 Michigan Telecommunications and Technology Law Review 457, 2007, at 459.
[136] Id., at 459.

In sum, a low level of inventive step may, in the short run provide for a quicker process of patentability with more patents being let through and possibilities of quickly recouping R&D investment costs. Something that is positive for the inventor as the fast pace of innovation in software makes for uncertainty as to for how long a patent is valuable to the market. However, in the long run a high level of inventive step required, the process may be more dubious, but the patents that come out of it will be of a higher quality, which provides for better incentives for further innovation because of higher prospective returns due to a better product that has a better chance of prevailing on the market for a longer time. These incentives for innovation in turn make for a more efficient patent system in the long run, and are what should be pursued in my opinion.

6.1.2 Incremental innovation

It is often argued that it is difficult to maintain quality in software patents because software is developed in increments and often built by the programmers on the work of each other. Even if an entirely new product fulfills the inventive step criterion, subsequent improvements may not be sufficiently inventive to warrant patent protection. I do not see, however, why this is a convincing argument against the patentability of all software. Rather, I agree with Domeij, who makes the point that the patent system itself is adjusted to the general incremental development of technology, where inventors build on the results of each other.[137] In fact, most inventions in most fields are built on previous knowledge, so that the software industry would be much different in this respect is not well founded.[138] Campbell-Kelly argues, correctly in my view, that software patents are not so different from those in other technologies and therefore, the software industry will adjust, like other areas of technology has, to a patent regime. As long as the level of inventive step is kept at a sufficiently high level, which I believe it should (see section 6.1.1), it should be possible for any inventor who comes up with a technically sufficiently advanced invention, thus contributing to the art, to be granted a patent on his product, no matter what that product is. There is no reason in my mind to keep an inventor of new and useful software from gaining the same benefits of his innovative activity as any other inventors in any other field of technology. It may very well be so that no incremental developments will obtain patents, if the requirements are stricter than today, but the patent system should definitely be there for truly innovative inventions..

6.2 Clarity of rules for legal certainty

For the incentivizing function of software patents, discussed above, to work properly and for the standard of inventive step required to be applied in an appropriate and optimal manner, it must be made clear what that standard *is*

[137] Domeij, p. 52.
[138] Guellec and van Pottelsberghe, p.127.

and *how* to apply it. Legal certainty is of great importance for a qualitative patent system. Pre-grant, patentees have to be able to know that their invention is patentable and what is expected of it. Most importantly, the inventor will want to, to the extent possible, be able to predict whether investment costs may be returned, to be incentivized to invest in R&D in the first place. Thus, if there is not a sufficient level of predictability to what may happen with the patent and the invention, there will be less innovative activity. Further, it is essential to know what to expect post-grant, how the patent will be treated, e.g. in case of opposition or infringement litigation.

There are, as may be deduced from the previous chapters, certain areas where the software patent law is not particularly clear. What may constitute patentable subject matter is particularly blurred in the software field and furthermore, the way in which inventive step is assessed is not always particularly reliable, even though there is an adopted approach to this – the problem and solution approach.

Although the notion of "technical character" is neither an express requirement for patentability, nor necessarily a part of the assessment of inventive step, it certainly has great implications for the patentability assessment. As the notion was first construed, the "technical character" was easily attained and made little qualitative difference to the assessment for patentable subject matter. It was simply too easy to fulfill and was therefore abandoned. The way the notion of "technical character" influences patentability in its current form is through its close relationship to the assessment for inventive step, in which technical matters are taken very much into consideration.

Technical character is furthermore very much influential in software patents because it is the field of technology in which it was developed and is essentially the only one where it is given any real weight.[139] Because of the abstract and mental nature of the software code, and the possibly very tangible result of its implementation, the technicality requirement is used to make a distinction between patentable software (with "further technical effect") and non-patentable software ("as such"). As a result, the inventive step requirement becomes increasingly important, as it is possible to show technical effect in almost any software product (not least by simply being clever in the patent application). Carefulness is called for, due to the risk of simply seeing the technical contribution made clear in the claim and taking that for the only 'proof' needed for inventive step.

There is to some extent confusion as to what "technical character" means. The EPO itself has expressed that it is not clear, despite it being the EPO who 'made up' the term.[140] It is therefore undoubtedly difficult for the patent applicants to know what is required of them, and for the examiner to

[139] Booton points out that "technical character" is a so-called policy lever for the software industry, and is, despite it being given 'universal' application, in essence only applied in this field; See Booton, at 112.
[140] Booton, at 93.

know what to look for in claimed subject matter. Something that is difficult enough in the fast moving software industry, with new sub-fields emerging all the time. This is an interpretation problem, which leads to legal uncertainty that needs to be resolved. For more certainty and transparency of the system, codification of the concept of "technical character" is needed, whether it is to be applied as a separate requirement for patentability or as an added part to inventive step.[141]

How to exactly specify "technical character" is difficult to answer, but it needs to be made sufficiently clear for the inventors to know what to expect, and for the examiners and the Boards of Appeal to apply the law in the same manner. This could be at least in part helped by harmonization of software patent law in the EU. This was also one of the aims of the proposed CII Directive – to provide a definition of technical character for the uniform application of what is considered inventions and the patentability requirement of inventive step. The potential difficulty that one uniform, rule-based application of the term "technical character" is not optimal for all industries should be mitigated by the fact that this concept is in reality only used in connection to software related products and processes.

6.3 Examination procedure

It is not only important that the inventive step requirement is set at an optimal level, and that the rules to be applied are sufficiently clear; for an efficient system the examination procedure has to be in conformity with the legal rules. The set standards for inventive step will be meaningless if the patent examiners do not know how, or do not have the right means, to apply them properly. Although the problem and solution approach is established for assessing inventive step that in itself is no guarantee for proper outcome in all cases since there are inherent problems in the approach, such as the probable, almost unavoidable, use of hindsight when setting out to find the technical problem. The approach has further been criticized because it may make the examiner lose track of the bigger picture, i.e. finding the overall technical contribution of the invention, instead of simple fitting the pieces together in the step-by-step method the approach is built on.[142]

6.3.1 The examiner

Excessive strains on the patent offices are possible when new technologies emerge. Because the software industry is still relatively young, and because innovation runs at a fast pace the patent examiners have not yet had time to become as skilled as may be required in the long run. Along with the difficulty in keeping up to date with fast technological advances, the examination process may not be as good as it should optimally be – yet.

[141] The legal foundation of such a practice is a matter of discussion not given room in this thesis.
[142] Reed, Chris and John Angel – *Computer Law*, Oxford University Press, 5th edition 2003, p. 171.

There are many who argue that the software industry is still young, and it will adjust, like most other industries have.

The problems the patent examiners face are possibly enhanced by the increased workload provided by more and more filed patent applications. With increased workload comes the need for more examiners and more training for them, increasing the costs, and where resources are scarce, this factor may add on to the problems currently facing the patent offices. To keep training and hiring of patents examiners up to speed with the innovation and patent application speed may be difficult at this time. However, the examiner must at the very least be capable of 'learning on the job', so to speak, providing a starting point for better examination procedures. When patent examiners are not sufficiently trained in what they do, they will inevitably let at least a certain amount of trivial patents 'slip through', providing the incentive problems partially described in section 6.4.1. Additionally, the patent offices are more likely to make mistake in fields of technology in which they have less experience, such as the software industry, thus increasing the risks of trivial software patents.[143]

However, it seems a weak argument against patentability in itself, to deny any patents in this field due to lack of experience of the patent office, they should get up to date at least at some point not too far in the future. It seems to me that the short-term cost of training the patent examiners to better understand the technological field of software would be offset by the gains of, in the long run, more innovative activity due better incentives for innovation where patents granted hold a sufficient level of inventiveness and thus quality.

6.3.2 The man skilled in the art

Another required task of the patent examiner is to define and put himself in the place of the man skilled in the art. Simply defining this character may be an obstacle; this person cannot be defined until the technical field has been established. Otherwise the person may not be skilled in the right art. The patent examiner then faces the task of putting himself in the place of this skilled person and evaluating what he would have done with the knowledge of the prior art and the problem at hand. Furthermore, reconciliation between the skilled man and an examiner at the patent office may bring certain difficulties. The former being devoid of any human capabilities it seems, at least if one looks to the characteristics described in section 3.2.1, and the latter being a very human being who may be neither sufficiently skilled nor sufficiently objective. To attribute all the expected characteristics to this fictional person and then expect an actual person to think like that seems deemed to fail at the get-go. Considering the skilled man may be construed of a group of scientists in the software field may mitigate the problems to a certain extent.

[143] Hunt (2001), p. 12.

Furthermore, there is often a discrepancy between the level of skill of the person skilled in the art and the real life computer programmers. Something that may infuse the idea of certain programmers that all software patents are trivial, because in their level of skill nothing is really significantly inventive or non-obvious. The common general knowledge at the basis of the assessment is therefore possibly incorrect, where the common general knowledge of the average computer programmer is significantly higher than that of the person skilled in the art personified by the patent examiner.

A tradeoff has to be made here between the level of skill to demand of the skilled man and the risk of granting trivial patents – what is economically justifiable? To require the man skilled in the art to be at the level of the most skilled computer programmer would clearly not be so – the cost of recruiting computer programming specialists to the patent offices or to educate the examiners would be too high, and would most likely not be offset by the gain in level of quality in patents granted.

6.3.3 Search for prior art

Easier search for prior art needs to be facilitated for the examination procedure to go smoother. The examiner needs, e.g. for the purposes of finding the closest prior art when utilizing the problem and solution approach but also for determining novelty. Furthermore, the patent applicants would be helped and could to a greater extent be required, to include further prior art searches in their applications. Care should be taken though in requiring too much of the applicants since an extensive requirement for them to add prior art searches in their application could be too costly on their part, and also lead to an over-flow of work for the patent examiners in going through all the extra material. It may effectively be so that the current practice of requiring the applicant to include the known art, i.e. what he knows of prior art at the time of the application, is the best practice for the applicant.

To help in the prior art search at the EPO level, granted and rejected patents may be a good starting point alongside private prior art databases used by both patent offices and software manufacturers.[144] A prior art database for software patents is a costly project to set up, but would in all likeliness be advantageous to the examination procedure, but as long as resources are not found to develop one uniform database on the European level, the private databases should be sufficient. The crux is to induce all the software patent holders to submit their patents to the database, and also for the patent offices to submit denied patent information for a more complete search facility.

[144] See, e.g. http://www.ip..com providing a prior art database, which is currently used by the EPO.

6.3.4 Ex ante vs. ex post control

Because there is a risk of lower quality patents 'slipping through' the examination process, due to the factors discussed in the previous sections, it is important to look at whether there are circumstances that make this risk less serious. It can be said that the really low quality patents that do come through the examination process are rarely enforced, the holders of these patents knowing the lack of quality in them will be less inclined to aggressively assert their rights, and would not likely hold up in case of litigation. This provides an ex post mechanism of ensuring at least that bad quality patents are not blocking better products from emerging. Moreover, the opposition procedure facilitated for in the EPC may further mitigate the negative impact of some trivial patents in the software industry. However, all forms of ex post control must be weighed against ex ante mechanisms. It seems to me that for upholding a good level of quality in software patents, it is better to ensure the quality in granting the patent rather than letting them through the examination process, with the detrimental signals that sends to the software industry, for subsequent invalidation. In sum, the ex post control mechanisms should only be utilized in the, hopefully with a better examination process, rare cases where an insufficiently inventive invention is granted a patent. This type of ex post control, with the exception for opposition, which is dealt with within the EPO, will be unpredictable as well because of the differences in practice of the national patent courts.

6.4 Competition concerns

Competition law and IP law are closely intertwined fields that affect each other to a great extent – they are opposites in a way – patents provide monopolies that competition law tries to prevent. When an exclusive right is provided in the shape of monopoly, obvious competition concerns arise. How to strike a balance between the two is no easy task. The monopoly in the case of patents, have been thought to be justified for the positive effects of patents to come forward. However, care should still be taken – if the monopoly is enhanced and imposes costs that are not offset by the stimulation of innovation and the diffusion of technology, it should possibly be reconsidered.

Due to the network effects and the standardization inherent in the software industry, a first inventor who obtains a patent on a broad software invention may set a standard with the effect that the monopoly provided by the patent will not easily be amended at the end of the patent period. Hence, the previous monopolist retains an advantageous position towards competitors also after the end of the patent term. However, this calls for the competitors of the first patent holder to be even more inventive and find other solutions to the same problems. The point being that in industries where standards are easily set, as in the software industry, it is important that granted patents live up to a standard of inventiveness that justifies them in light of the fact they may be used for anti-competitive purposes and furthermore, may block

other, better inventions from reaching the market. In the end, this would be detrimental to consumers who will have less access to high quality products. Moreover, consumers may hesitate in switching to the newer and better products because the network effects of the first, sub-optimal but standardized product has already 'kicked in', making it would be costly to switch to the new and better product.

That trivial patents enhance the negative effects of the exclusive rights granted to a patent holder is unquestionable. However, there is a natural tendency towards monopoly in software even without patents because of the network effects displayed.[145] Because there is a need for interoperability between different types of software and between software and hardware, it is most efficient to have one provider of all software and potentially hardware as well. These natural tendencies towards monopolies may be amplified by the patent system. For example, with personal computers, the one who controls the interface between the different pieces of software controls the market This is the current position of Microsoft, which controls the OS, the applications etc. and they use that power to gain a market advantage, sometimes in a way that puts them in a lot of trouble competition law wise.

Notably, a concern of economic nature – the preservation of competition – resulted in the technical examination inherent in inventive step.[146] This is an example of when competition concerns have contributed in the development of the rules for assessing inventive step. I do not believe that the competition problems connected to software patents are of such great consequence as to require a ban on these patents. However, care needs to be taken. Again, if the other parts of the process of assessing inventive step are kept up to par, the granting of software patents would most likely not worsen the competitive environment.

6.5 The role of harmonization

> "Effective, transparent and harmonized protection of computer-implemented inventions throughout the member states is essential in order to maintain and encourage investment in this field."[147]

Harmonization of software patent law is highly desirable from the point of view of legal certainty.[148] When an inventor applies for a patent with the EPO he may not be certain that the patent will hold up in the same way in all the contracting states of the EPO, or that it will even be enforceable at all, since national laws apply to the patents post-grant and the contracting states, who are not bound to the decisions of the EPO, do not all recognize

[145] Bakels and Hugenholtz, p. 19.
[146] See section 3.2.
[147] Murphy, David – *European Industry and Software Patents*, Patent Eye, No. 1 pp. 17-25, 2005, p. 24.
[148] This is an opinion that has been expressed by the Board of Appeal in the IBM/*Computer Program Product I* case, see fn 44.

software as patentable subject matter, even if all the statutory requirements for patentability are the same.[149] For example, the UK is fairly strict in their application of patent laws to software related inventions, especially in comparison to Germany, where practice is more relaxed. A patent granted by the EPO, which is recognized in one state but not in another is a problem because a granted patent that cannot be enforced in the whole area for which it is sought, cannot be an optimal patent in the eyes of the patent holder, thus it certainly should be in the interest of all patent applicants that the effect of granted patents are the same in all of the contracting states. Divergences in practices of the national patent court leave the market with around 30,000 software related patents whose enforceability cannot be guaranteed, and European industry may suffer for it.[150] An international company faced with having to pick between a patent system where their patents are uniformly applied over the entire area, and a system where the application process may be uniform, but there is significant uncertainty as to what will happen to the patent after it has been granted, will most likely pick the former – thus *not* the European system.

Legal certainty also extends to the interpretation of the laws, the 'technical character' requirement would benefit greatly from being rule-based rather than based loosely on EPO practice. Since the Boards of Appeal do not have any law-making powers as such, an actual codification of the term "technical character" would be impossible to obtain on the EPO level. This is another area where harmonization could facilitate for better solutions. Booton makes an important and good point in my opinion when he points out that unfortunately, due to the turbulent recent past of the debate over software patentability, it is unlikely that consensus could be reached, at this point, between all the bodies of the EU that are involved in the legislative process.[151]

Furthermore, harmonization would be beneficial from an economic perspective. First of all because economic considerations would be given more explicit weight on an EC law level, due in part to the fact that the EPC effectively denies the boards of appeal the right to make economic argumentation part of their assessment, but also because EC law has, or is at the very least moving towards, a practice of giving great weight to market conditions and other economic factors in several fields of law. Considering the inherent economic nature of software patent law, this would certainly be a step forward. The single market imperative of the European Common Market makes harmonization critical in an area, such as the software market, where trade between states is abundant.

There are many positive aspects to be found in favor of harmonization. It is desirable from the viewpoint of legal certainty, it could help with conform interpretation of patentability requirements and it is economically efficient,

[149] Explanatory memorandum, p. 9.
[150] Murphy, p. 20.
[151] Booton, at 113-114.

thus providing for all three quality notions for software patents.[152] It seems as though going actively against harmonization of software patent law is rather 'dumb' since software patents are already being granted on a grand scale today. It would, in my mind, make more sense to put energy into making the harmonization process go in the right direction, rather than fighting a system that will not likely be changed to a ban on software patents, and which could be made better than what is now a software patent practice that grants trivial patents and is unpredictable where national rules differ greatly, at least in application.

[152] See section 3.1.

7 Conclusions

No matter what was originally intended with regards to patents on software when the EPC was drafted, interesting as that question may be in itself, computer programs and software are highly patentable subject matters under the European patent system today.

A note-worthy question is whether software is really less suitable for patenting or just more controversial. A closer examination of what has been written about software patents in the past, it is striking how the controversy of the software patent in many ways is an idealistic war fought mainly by the (very loud) open and free software movements against the 'big bad wolfs' that are the handful of large software manufacturing corporations, predominantly Microsoft, that may be more visible in their patent activities than the multitude of smaller software developing firms. It is persistently held that because there is such a controversy, software patents must be bad. Claims of software patents being unable to hold sufficient quality and how all software is nothing but abstract mathematical formula and that it is built in increments, thus never providing large enough leaps of technological advancement to warrant patent protection. Not to say that there is nothing wrong with software patents, but one needs to look past the heavily subjective argumentation of certain critics that is often lacking substance.

There are indeed apparent problems with software patents that need to be addressed for the future prosperity of software innovation. "Trivial" patents are clearly a problem that is especially pronounced in the software industry. Such patents are granted on a regular basis by poorly educated patent examiners and make the inherent inefficiencies of the patent system worse because they provide insufficient or wrong incentives for innovation and for patent applications. But rather than banning software patenting, there are amendments to make within the existing system.

In line with what has been discussed in the previous, I find it strange, unreasonable even, that a risk of granting "trivial" patents should be anything other than a call for stricter examination or stricter and clearer criteria for patentability of software, possibly best achieved by harmonization of examination criteria – not a call for a ban on software patents. Granted, harmonization will not change much in the *examination* procedure since that is basically harmonized as it is on the European level. Harmonization would, however, make it possible to outright take into consideration economic concerns that are very much at the heart of patent law. Most importantly, harmonization would help in that it could bring national courts in line with each other and the European system, and make them bound by the same rules, thus improving legal certainty and predictability of the way in which a granted patent will be treated. Arguably, since computer software is in essence already patentable in the EU, with the exception of computer programs "as such". Something that, with the

addition of the requirement for technical character, makes them as patentable as any other invention fulfilling the requirements of the EPC, it would help with a uniform legal instrument which brings with it legal certainty in the area of software patents. Furthermore, concerns that a harmonized EU legislation would bring the EU closer to the US in terms of bringing down the standards for patentability to a level where algorithms would be patentable in themselves are unfounded. Upholding standards in the EU assessment, in the same way that standards need to be raised for all other purposes as well, for granting high quality patents, could prevent this.

Finally, I return to the question put at the beginning of this thesis, to bring it all together: is the problem with granting "trivial" patents a valid reason for the continued non-harmonization of software patent law in the EU? The simple and succinct answer would be "no". However, as simply as the answer may be put, the road to it has been anything but easy.

I cannot see the rationale behind leaving out a major part of what is technology, and important technology at that, for the reason that the specific industry is especially prone to lower standards at this point in time. Give it time to adjust, I propose, and put extra efforts into making the situation at hand better. This should be done by increasing the level of inventiveness required, setting a uniform standard for the assessment of inventive step by formalizing the "technical character" requirement so that it may be applied uniformly and provide legal certainty for the patent applicants. Discourage potential misuse of the patent system by overzealous computer programmers and abusive software manufacturers by requiring a level of inventiveness that makes natural progressions and small improvements to existing art unpatentable by not fulfilling the requirement for inventive step - not by failing on the ground of being unpatentable subject matter. Society as a whole benefits from software innovation, and I truly believe because we have a patent system that is already providing for software patents, it is a backwards move that would likely fail, to ban them now.

I opt for higher demands for level of inventiveness in regards to software inventions and rely on the adjusting capabilities of the patent system and the examination process to this new field of technology. This would in my opinion be helped by EU level harmonization for software patents, an important field of technology whose inventors should not be deprived of the same protective functions as those in other fields of technology simply because the market has not yet quite found its way.

As this field of law is still largely unexplored it is important that there is further study into what may be done in this interesting and developing area. For the future I would like to propose certain questions that may make for interesting research in the future:

- What are the actual economic implications of software patentability for innovation and investment in R&D in the software industry? Is ex post or ex ante protection more economically efficient?

- Is the general patent system sufficient for software patent protection, or would a *sui generis* regulation be more appropriate, or simply redundant?
- What level of skill should be required of the man skilled in the art when it comes to assessment of software inventions to provide for better levels of inventive step than can be found today?

Supplement

Relevant EPC Articles:

Article 52 – Patentable inventions

(1) European patents shall be granted for any inventions which are susceptible of industrial application, which are new and which involve an inventive step.

(2) The following in particular shall not be regarded as inventions within the meaning of paragraph 1:
(a) discoveries, scientific theories and mathematical methods;
(b) aesthetic creations;
(c) schemes, rules and methods for performing mental acts, playing games or doing business, and programs for computers;
(d) presentations of information.

(3) The provisions of paragraph 2 shall exclude patentability of the subject matter or activities referred to in that provision only to the extent to which a European patent relates to such subject matter or activities as such.

Article 54 - Novelty

(1) An invention shall be considered to be new if it does not form part of the state of the art.
(2) The state of the art shall be held to comprise everything made available to the public by means of a written or oral description, by use, or in any other way, before the date of filing of the European patent application.
(3) Additionally, the content of European patent applications as filed, of which the dates of filing are prior to the date referred to in paragraph 2 and which were published under Article 93 on or after that date, shall be considered as comprised in the state of the art.
(4) Paragraph 3 shall be applied only in so far as a Contracting State designated in respect of the later application, was also designated in respect of the earlier application as published.
(5) The provisions of paragraphs 1 to 4 shall not exclude the patentability of any substance or composition, comprised in the state of the art, for use in a method referred to in Article 52, paragraph 4, provided that its use for any method referred to in that paragraph is not comprised in the state of the art.

Article 56 – Inventive Step

An invention shall be considered as involving an inventive step if, having regard to the state of the art, it is not obvious to a person skilled in the art. If the state of the art also includes documents within the meaning of Article

54, paragraph 3, these documents are not to be considered in deciding whether there has been an inventive step.

Article 57 – Industrial Application

An invention shall be considered as susceptible of industrial application if it can be made or used in any kind of industry, including agriculture.

Articles of the Implementing Regulations to the EPC:

Rule 27

(1) The description shall
(a) specify the technical field to which the invention relates;
(c) disclose the invention, as claimed, in such terms that the technical problem (even if not expressly stated as such) and its solution can be understood, and state any advantageous effects of the invention with reference to the background art.

Rule 29

(1) The claims shall define the matter for which protection is sought in terms of the technical features of the invention. […]

Bibliography

Literature

Cooter, Robert and Thomas Ulen – *Law and Economics*, Addison-Wesley 3rd edition 2000

Domeij, Bengt – *Patenträtt – Svensk och Internationell Patenträtt, Avtal om Patent samt Skyddet för Växtsorter och Företagshemligheter*, Iustus Förlag 2007

Downing, Robbie – *EC Information Technology Law*, Wiley 1995

Guellec, Dominique and Bruno van Pottelsberghe de la Potterie – *The Economics of the European Patent System – IP Policy for Innovation and Competition*, Oxford University Press 2007

Paterson, Gerald – *The European Patent System – The Law and Practice of the European Patent Convention*, 2nd edition Sweet & Maxwell 2001

Reed, Chris and John Angel – *Computer Law*, 5th ed. Oxford University Press 2003

Schechter, Roger and John Thomas, *Principles of Patent Law*, 2nd ed. Concise Hornbooks 2004

Visser, Derk – *The Annotated European Patent Convention*, 13th revised edition 2000

Articles

Bakels, Reinier and P. Bernt Hugenholtz - EP Directorate-general for Research Working Paper – *The Patentability of Computer Programmes - Discussion of European-level Legislation in the Field of Patents for Software*, available at: http://www.ivir.nl/publications/other/softwarepatent.pdf, last visited: July 29, 2007

Bergstra, Jan A., and Paul Klint – *About "trivial" software patents: The IsNot case*, Science of Computer Programming 64 (2007) 264-285

Booton, David – *The Patentability of Computer-implemented Inventions in Europe*, Intellectual Property Quarterly 2007, 1, 92-116

Burk, Dan and Mark Lemley – *Designing Optimal Software Patents*, Stanford Law School – Public Law & Legal Theory Working Paper Series,

Research Paper No. 108, 2005; available at: http://ssrn.com/abstract=692044

Campbell-Kelly, Martin – *Not All Bad: An Historical Perspective on Software Patents*, 11 Michigan Telecommunication Technology Law Review 191 (2005), available at: http://www.mttlr.org/voleleven/campbell-kelly.pdf

Gratton, Éloïse – *Should Patent Protection be Considered for Computer Software-related Innovations?*, Computer Law Review & Technology Journal, Vol VII 2003

Hunt, Robert M. – *Economics and the Design of Patent Systems*, 13 Michigan Telecommunications and Technology Law Review 457, 2007

Hunt, Robert M. – *You Can Patent That? Are Patents on Computer Programs and Business Methods Good for the New Economy?*, Business Review Q1 2001

Murphy, David – *European Industry and Software Patents*, Patent Eye, No. 1 pp. 17-25, 2005

Yang, Grant C. – *The Continuing Debate of Software Patents and the Open Source Movement*, 13 Texas Intellectual Property Law Journal 171

Zoracki, Allen Clark – *When is an Algorithm Invented? The Need for a New Paradigm for Evaluating an Algorithm for Intellectual Property Protection*, 15 Albany Law Journal of Science & Technology 579, 2005

Other

Convention on the grant of European Patents (European Patent Convention) – Published by the European Patent Office, 12th Edition, 2006

Council Directive of May 1991 on the legal protection of computer programs (91/250/EEC) OJ L122, 17.5.1991

Communication from the Commission to the European Parliament pursuant to the second subparagraph of Article 251(2) of the EC Treaty concerning the common position of the Council on the adoption of a directive of the European Parliament and Council on the patentability of computer-implemented inventions, 2002/0047 (COD)

EPO – *Computer-implemented Inventions and Patents, Law & Practice at the EPO*

European Parliament legislative resolution on the Council common position with a view to the adoption of a directive of the European Parliament and of

the Council on the patentability of computer-implemented inventions (11979/1/2004 – C6-0058/2005 – 2002/0047(COD))

Guidelines for Examination in the European Patent Office, published by the European Patent Office adopted by the President of the EPO in accordance with Article 10(1)(a) EPC, effective on June 1, 1978

Speech/06/453, Charlie McCreevy, European Commissioner for Internal Market and Services, *Closing remarks at public hearing on future patent policy*, public discussion on future patent policy in Europe, Brussels, 12 July 2006, available at: http://europa.eu/rapid/pressReleasesAction.do?reference=SPEECH/06/453&format=HTML&aged=1&language=EN&guiLanguage=en, last visited August 1, 2007.

Proposal for a Directive of the European Parliament and of the Council on the Patentability of Computer-implemented Inventions, COM(2002) 92 final, *Explanatory Memorandum*.

The Result of the European Commission Consultation Exercise on the Patentability of Computer Implemented Inventions, Final Report by PbT Consultants under contract number PRS/2000/A0-7002/E/98

Promoting Innovation through patents: Green Paper on the Community Patent and the Patent System in Europe COM(1997) 314 final, 24 June 1997

Promoting innovation through patents: the follow-up to the Green Paper on Community Patent and the patent system in Europe COM(1999) 42 final, 5 February 1999

US Patent Act 35 U.S.C.

WIPO Model Provisions on the Protection of Computer Software, Geneva 1978

Internet Resources

Website of WIPO: http://www.wipo.int

Website of the EPO: http://www.epo.org

http://www.epo.org/focus/issues/computer-implemented-inventions.html, last visited: August 1, 2007

Table of Cases

EPO case law

US Supreme Court and CAFC case law

Lockwood v. American Airlines, 107 F.3d. 1565 (Fed. Cir. 1997)

Mackay Radio & Telecommunications Co. v. Radio Corp. of America, 306 U.S. 86 (1939)

State Street Bank & Trust Company v. Signature Financial Group, 149 F.3d. 1368 (Fed. Cir. 1998)

WMS Gaming Inc. v. International Game Technology, 184 F.3d 1339 (Fed. Cir. 1999)