Software Patentability: a computer research scientist's view‡

Bernard Lang (INRIA – Rocquencourt)†

working draft version§ *- October 13th, 2005*

> *The businessman :* When you find a diamond that belongs to nobody, it is yours. When you discover an island that belongs to nobody, it is yours. When you get an idea before any one else, you take out a patent on it: it is yours. So with me: I own the stars, because nobody else before me ever thought of owning them.
>
> *The little prince :* I myself own a flower, which I water every day. I own three volcanoes, which I clean out every week (for I also clean out the one that is extinct; one never knows). It is of some use to my volcanoes, and it is of some use to my flower, that I own them. But you are of no use to the stars ...
>
> *Antoine de Saint-Exupéry, The little prince, ch. 13.*
>
> Man-made laws can be significantly helpful but not when they contradict fundamental truths.
>
> *Donald Knuth, Letter to the Patent Office, February 1994.*[1]

The issue of software patentability[2] raises all kinds of questions. It is however to be noticed that, usually, only lawyers seem to have a say on the matter, and sometimes economists. These are essential dimensions when the issue is to regulate and govern, and more specifically the economic system. But, without ignoring them, I thought this contribution would be of greater use if it left largely aside the purely legal and economic aspects, and focused more on presenting the vision of the research scientist. The research scientist is in principle the source of innovation and as such the first concerned by the economic and legal mechanisms that are supposed to regulate it.

---

‡ An earlier version is to appear in Actes du Colloque "Brevet - Innovation - Intérêt général" organized by the Chaire Arcelor de l'Université de Louvain la Neuve, 11-13 March 2004. http://www.chaire-arcelor.be/?rb=colloque

† Current address : Bernard Lang, INRIA, B.P. 105, 78153 Le Chesnay CEDEX, France. <Bernard.Lang@inria.fr> http://pauillac.inria.fr/~lang/ .

The author wishes to thank Areski Nait Abdallah, François Pellegrini, Cyril Rojinsky and Pierre Weis for comments and suggestions, Rosaire Amore and Bruno Berthelet for proofreading the French version, and James Leifer for his help with the translation, though the final form is the author's entire responsibility.
This document was produced with the free office suite OpenOffice.org.

§ This document is still being worked on. Some parts have to be extended or reorganized. The most recent version will be accessible from the following address: http://pauillac.inria.fr/~lang/ecrits/liste/software-patentability.sxw , or http://pauillac.inria.fr/~lang/ecrits/liste/software-patentability.doc . This document can be reproduced according to the terms of the Free Document Dissemination Licence FDDLv1, http://pauillac.inria.fr/~lang/licence/v1/fddl.html . However, you should not reproduce this version if a more recent version is available.

1 Donald Knuth is probably the world most recognized computer scientist. http://www.pluto.linux.it/meeting/meeting1999/atti/no-patents/brevetti/docs/knuth_letter_en.html .

2 Despite the hypocritical wording of the directive proposal by the European Commission [Proposal for a Directive of the European Parliament and of the Council on the patentability of computer-implemented inventions (2002/C 151 E/05), submitted by the Commission on 20 February 2002, *O.J. Of the European Communities*, 25 June 2002, p. 129, http://europa.eu.int/eur-lex/pri/en/oj/dat/2002/ce151/ce15120020625en01290131.pdf] as well as that of numerous press releases on this topic [*Patents: Commission welcomes Council agreement on Directive on computer-implemented inventions*, press release from the European Commission, IP/04/659, 19/05/2004, http://europa.eu.int/rapid/pressReleasesAction.do?reference=IP/04/659&language=EN], the question that is currently and actually put forward to Europe — and the world — is indeed to determine whether or how patents should be granted on pure software methods, independently of any innovation involving a new use of the forces of nature. This is the essence of the conflict opposing the European Parliament and the European Commission, conflict which is reproduced in several member countries of the Union between the local parliament and executive body.

1. The spirit of programming

The current trend seem to be, in all areas of creation, towards a continuous extension of the patentable domain, while asserting that there should be limits to what is patentable. The question is therefore to define and set such limits.

To this end, it is first necessary to understand the subject at hand. For WIPO, software is understood as "*a set of instructions which, once they are transposed through a medium which is decipherable by a machine, can indicate, realize or obtain a function, a task or a specific result thanks to a machine capable of processing information.*[3]"

If this definition was already simplistic when originally published in 1978, it is even more so with the increasing sophistication of computer science. For a computer scientist, for a programmer, there is no real difference between programs and data. It is only a question of context and presentation. Typically, the data for a program can become increasingly complex, to a point when they are themselves to be expressed with a specific language. The program may then be seen as an interpreter for that language, while the data for a program for this interpreter. But who could tell, in this evolution towards complexity, when the switch occurs between the data view and the program view.

The programming language may just simply describe complex data, for which a given reading, a given interpretation (which need not be unique), will produce results. To speak of *instructions* is therefore often false,

However, is it reasonable to think that the simple fact that one describes *entities* that, "*once they are transposed through a medium which is decipherable by a machine, can indicate, realize or obtain a function, a task or a specific result"* shows the existence of a process ? Actually no, because it is not necessary for the program to describe how the result is obtained. For example, it may simply describe the result that is being expected, and leave to a compiler or an interpreter the charge of finding the proper computation that will produce this result. Then the program no longer expresses a method for obtaining the result, but only describes the result or the functionality to be achieved. And a simple description is traditionally non patentable.

Actually, the mathematics of programming see all programs in that way, as specifications of functionalities and not as the description of the way they are achieved.[4] And how could it be otherwise, when most programs are intended to be compiled, that is translated from one language into another, from a *source language* used by the programmer to an *object language* intended to be interpreted by an information processing tool, while preserving the functionality which is the only thing that matters. For example, the *source program* may be simply the specification of the grammar of a natural language, which determines the structure of sentences in this language without making explicit in any way how this structure can be retrieved for a given sentence, while the *object program* will be a syntactic analyzer for the sentences in that language, which will be actually capable of retrieving the structure of sentences. In other situations, the source program may be written in an algorithmic language and the object program in "*machine language*". But there is no a priori reason to believe that the machine code actually simulates what is "intuitively" described in the algorithmic presentation of the source program, The role of compilers is to preserve

3 Dispositions types de l'OMPI relatives à la protection du logiciel, 1978. Dans : Michel Vivant, *Les créations immatérielles et le droit.* Ellipse, p. 51*, 1997.*

4 These mathematics of programming deal with the definition of the meaning of programs from formal definitions of programming languages and their semantics. These issues of semantics are usually distinguished from algorithmics, which is more concerned with the actual implementation of functionalities. But there seem to be a consensus that algorithms are abstract techniques which, as such, cannot be patentable. This viewpoint is actually confirmed by mathematical results that establish that algorithms and mathematical proofs are the same kind of entities (technically, they are isomorphic).

functionality, but they may do it in their own way without any identifiable structural correspondence between the two forms of the program because the source program is nothing more than a specification of the functionality, as in the previous grammar example. This is typically what happens, in increasingly sophisticated ways, with optimizing compilers[5] and various systems for the automatic generation of programs.

Hence we have a situation where the human readable code (the source code) is only a functionality specification and therefore not patentable. Criteria of readability, organization and style actually play an important role in the expression of this source code. The machine executable object code indeed meets well the WIPO definition and could be considered as a process (at least when executed). But this code is, for all practical purposes, unreadable by people, and no one knows what that process might actually be composed of.[6] This is even truer when you compound the fact that the executable code depends on the system it is intended for, and even more on the compiler that produces this code, given that this compiler is not necessarily known to the author of the program or to its users. And, being mechanically derived from the source code, this executable object code cannot meaningfully be the object of intellectual property other than by an extension of the rights associated exclusively to the source code.

This analysis of the foundations of programming activities justifies the position of the European Parliament, as voted on 24 September 2003, according to which the inventive step that would justify patentability cannot reside solely in the software itself, which remains an abstract entity to the same extent as mathematical formulas, but in " *the use of natural forces to control physical effects beyond the digital representation of information.*[7]"

2. The art of programming

Software generally consists of complex creations involving hundreds, possibly thousands, of components, of very diverse nature. This raises considerable problems of expression and organization. Programmers rather seldom encounter critical algorithmic problems, and they show

---

5 Compilers can use a wide variety of techniques, separately or together, to optimize the object code they produce. In addition, they may seek to optimize different characteristics of that code, depending on external constraints, such as the size of the memory required for execution, the speed of execution, or even the size of the compiled object code itself. In each case, the result can be object code that executes in different ways, sometimes very different. Furthermore, rather than producing object code, the compiler can produce the drawing of an electronic circuit that implements the same function (as with silicon compilers), and for this circuit it may choose to optimize its speed, its surface, its electric consumption or any balance of the three.

6 This code could be analyzed by means of decompilation techniques. However these techniques are very costly and hard to use, and are generally reserved for the analysis of small code fragments, for example when needed to ensure interoperability between programs. In the European Union, according to the 1991 directive on the legal protection of computer programs (see note 56 below), this analysis for interoperability purposes is actually the only permitted use of decompilation. Thus, even if some software constructs were patentable, one might imagine situations where the source code would not be infringing, while the object code produced by some compilers would be, unbeknownst to anyone (or maybe not). But who would then be the infringer? And what if the source code is infringing (whatever that might mean), but the object code is not?
We do not contend that this situation is currently the most common case, though it may be expected to become more common with the evolution of programming towards ever more sophisticated practices. But the simple fact that it may occur is enough to raise questions about the soundness of software patenting. Interestingly, the one case where we may expect object code to faithfully reflect whatever is described in the source code is for program fragments intended to communicate with other programs that are compiled independently, precisely the case covered by the 1991 directive. This is further discussed in section 7 below.

7 Position of the European Parliament adopted at first reading on 24 September 2003 with a view to the adoption of Directive 2003/..../EC of the European Parliament and of the Council on the patentability of computer-implemented inventions. European Parliament, Final Edition : 24/09/2003, Patentability of computer-implemented inventions, P5_TC1-COD(2002)0047, A5-0238/2003. http://www3.europarl.eu.int/pv2/pv2?PRG=CALDOC&TPV=DEF&FILE=20030924&TXTLST=2&POS=1&LASTCHAP=10&SDOCTA=2&Type_Doc=ANNEX&LANGUE=EN .

their mastery in their art through their management of complexity, in the choice of their means of expression (language) and in the clarity of that expression: in a word, through their style.[8] Beginners quickly learn that a good program is, before anything else, a readable program, and that it has a significant impact on the maintenance and evolution of programs, which constitute the major part of the cost of software.

Only a deep ignorance of the programming activity, and a gratuitous contempt for what is foreign, can explain the assertion by some authors of their belief that writing programs is a mundane activity, on a par with writing grammatically correct documents, rather than with literary creation, and conclude from it that the use of copyright would be inappropriate because software writing is not creative expression.[9] Of course, few programmers have true creative abilities, just as few written

---

8 There is a significant literature on programming style, of which we mention here some of the best known examples. Some languages or systems aim at promoting specific styles, but there always remain ample space for personal expression. In my own programming experience, I remember a colleague coming to me to learn the reason for a stylistic change in a program fragment I had written. This fragment was indeed atypical, for reasons unrelated to its functionality or the proper implementation of that functionality. Some classical references are:
Brian W. Kernighan et P. J. Plauger, *The Elements of Programming Style*, McGraw-Hill, Inc., New York, NY, 1982.
Donald Ervin Knuth, *Literate Programming*, CSLI Publishing, 05/1992, ISBN: 0521073806
Donald Ervin Knuth, Computer Programming as an Art, 1974 Turing Award Lecture, *Communications of the ACM*, Vol. 17, Nb. 12, December 1974. http://fresh.homeunix.net/~luke/misc/knuth-turingaward.pdf .

9 It must be remarked that the originality and creative character of a work is totally independent of the encoding used to represent the work. The argument sometimes heard, that there cannot be any literary or artistic value to a creation that amounts to a binary sequence of 0s and 1s is simply naive. One can encode in this way (and in many other ways) all non material creations, the Beethoven symphonies and their interpretations by orchestras, the choreographies of ballets and the plays of Shakespeare. Though not stating the point as trivially, Vincent Casiers asserts that a fragment of HTML code or a small binary sequence of 0s and 1s (each explicitly given in the paper) is "*manifestly not a literary work*" even though it is to considered as such "*in virtue of Article 1 LPO which transposes the directive of May 14, 1991 to Belgian law.*" His point is that "*it is not an original creation, because (i) it does not bear the mark of the personality of its author; (ii) it is exclusively functional and, above all, (iii) it is not communicable to the public. The likening of lines of code to a literary work is absurd and over-the-top.*" It is probably true that such an excerpt is probably much too short to represent an original creation bearing "*the mark of the personality of its author*" (though, for example, musical originality legally begins with only 8 measures), but this is quite unrelated to its encoding, whether binary or in HTML. However this would certainly not be true in general of longer fragments, which could possibly encode recognized literary work as we pointed out above. Whether it is "*exclusively functional*" actually depends on the way it is interpreted, i. e. on the information it is supposed to convey, which cannot be determined for the incomplete HTML fragment, and even less for a binary sequence (of any length) given in isolation. And even assuming its main purpose is functional, there is always more to it than the simple realization of that functionality for those interested in this mode of expression: people do read code. The exclusively functional character could be invoked against the used of copyright only if it fully determined the code produced, which is never true except for the most elementary programs. Furthermore, copyright applies to any written work, whether or not it is considered literary, such as for example a technical user manual or a mathematical textbook. Finally, asserting that "i*t is not communicable to the public*" is simply wrong. HTML is precisely a form intended to communicate with the public through an appropriate decoder (the browser), and mediation of a decoder to access artistic or literary creation is an old practice, whether simple glasses for the visually impaired reading small print, or a HI-FI set to listen to vinyl recordings. Ironically, publishers currently consider that binary sequences are the best way to communicate with the public. The digital rights management systems (DRMS), that they are trying to promote, aim at preventing the copy of the binary representation of a work, but they are much less concerned with more traditional analogical representations, considered to have less accuracy and reproducibility. This is mainly observable for music and films, but is also true for literary works.
Vincent Casiers, *Software, Business Methods & Patents*, page 13, 22 February 2004 (date of original French version). http://www.chaire-arcelor.be/fichiers/03-04/patinovpubint.pdf . This is actually an introductory report for the International Conference "Patents: why and to do what ? – Patents, Innovation & Public Interest", University of Louvain la Neuve (Belgium), 11-13 March 2004, organized by the ARCELOR Chair "Technology & Law".
Readers interested in the relations between numbers (of which binary sequences are but one representation) and intellectual property, can usefully explore the Gallery of CSS Descramblers at Carnegy-Mellon University, and specifically its exhibit on illegal prime numbers. David S. Touretzky (2000), *Gallery of CSS Descramblers,* accessed September 2005, http://www.cs.cmu.edu/~dst/DeCSS/Gallery .

works have true literary originality. However, even granting that such a line of reasoning is warranted, which is disputable,[10] it is the appropriateness of the medium for expressing an author's style and personality that justifies adequateness of copyright, rather than the proportion of authors who are able to take advantage of if it in their creations, or the proportion of the public who is literate enough to appreciate the stylistic qualities of these creations.

Most of the cost of software creation is indeed in this work of expression and organization, which falls quite naturally within the province of copyright law. If such a protection does not prevent a competitor from creating functionally equivalent software, it gives sufficient lead time to the first creator to let him set up his position in these fast evolving markets. This is actually a medium-term protection, which is not that much unlike some suggestions (legally hard to implement) to institute short lived patents. And, as it does not block independent creation, copyright has significantly less pernicious effects, most notably regarding legal uncertainty (see below the issue of inadvertent infringement) and the blocking of dependent innovation.

Moreover, if the innumerable logical elements of a program, individually or in diverse interacting combinations, can be the object of exclusive appropriation, there is a high risk of a fast increase of transaction costs. This can only result in anti-commons and situations that are economically inefficient and block new developments.[11]

Finally, if such exclusive appropriation of (combinations of) logical elements can be opposed to software authors, it is a major limitation of the rights normally granted to all authors regarding their exclusive control over what can and cannot be done with their works. Beyond the simple fact that this contradicts treaties on copyright,[12] it fully devalues the work that goes into the composition and actual writing of software, which is both the most expensive and the most creative aspect of software development. As we shall see below, this can only be a disincentive to software creation.

3. The patent religion

Answering a written question by a French member of parliament regarding the position on the French government on the Common Position of the European Union Council on the patentability of software,[13] the government wrote that "*only patents will enable an inventor to fully protect his rights*

10 The "*droit d'auteur*" was indeed created to allow authors to retain and protect rights over their original creations. Though the history of *copyright* is somewhat different, it is also based on the originality of the work. However, independently of any legal or philosophical considerations on the nature of originality, and on what may or may not be considered artistic creation, it seems legitimate to use copyright just as a regulation mechanism – based on some adequate definition of originality – endowed with specific characteristics that distinguish it from other intellectual property mechanisms. The only question that really matters, as always, is to determine whether this mechanism is appropriate to best regulate a domain of creation, in accordance with goals (enforce God given rights, foster innovation, reinforce the commons and the public domain, develop monopolies, ...) sought by the legislator. This is further discussed in note 68 below.

11 This is probably what led the chairman of Microsoft to state, in 1991, that "*if people had understood how patents would be granted when most of today's ideas were invented and had taken out patents, the industry would be at a complete stand-still today.*" William Gates III, *Challenges and Strategy memo*, May 16 1991, dans : Fred Warshofsky, *The Patent Wars*, 170-71 (NY: Wiley 1994). http://www.bralyn.net/etext/literature/bill.gates/challenges-strategy.txt . The fact that he changed his mind on this issue, at least publicly, is possibly related to his desire to preserve the current dominant position of his company, which developed without any software patent.

12 This curtailment of the rights of authors is analyzed in some detail in: Christian Beauprez, *In Defence of the Software Author: A Study of Copyright and Patent Law*, 18 August 2004, http://beauprez.net/softpat/defence.pdf .

13 Common Position (EC) No 20/2005 of 7 March 2005 adopted by the Council, acting in accordance with the procedure referred to in Article 251 of the Treaty establishing the European Community, with a view to adopting a Directive of the European Parliament and of the Council on the patentability of computer-implemented inventions, O.J. of the European Union, Notice No 2005/C 144 E/02, p. 9, Vol. 48, 14 June 2005. http://europa.eu.int/eur-lex/lex/LexUriServ/site/en/oj/2005/ce144/ce14420050614en00090015.pdf

against competitors".[14] This tautological answer is very characteristic of the ideological – if not religious – approach of many actors regarding intellectual property and more specifically patents. As such, it is very enlightening as to the current debates.

This answer is tautological from a legal point of view because the use of the word "*inventor*" assumes *a priori* that there is an invention and thus a right to a patent. But this is precisely what is being debated.

More importantly, this answer is tautological quite simply because, like anyone else, creators have no other rights – especially economic rights – than those granted by laws and institutions. If the law makes no provision regarding patent rights for software, software creators have no such rights to protect against competitors. To argue about the protection of a right with no legal existence is to assume that it has an *a priori* existence, for some metaphysical reasons. One may then rightly wonder whether these reasons are anything but ideology or religion, which is all the more surprising since these hypothetical patent rights generally concern moral rather than physical persons. In a nutshell, patents are a right because they are a right.

Actually, the granting of patents on new techniques is a fairly recent idea, historically motivated mainly by the need to regulate the economy so as to encourage investment and foster technological development. It was not always linked to the concept of an invention, and the almost mystical link that patent professionals see today between an invention and a patent right is probably the source of their ideological and religious attitude on these issues, reinforced by a moral belief in a natural right of the inventor ... even though, for example, there may be several inventors, and the various legal systems do not agree on the rules determining which one should be the exclusive recipient of this allegedly natural right (*first to invent* vs *first to file*). And why should a *natural* right last 20 years, rather than 10 or 50 years ?

It might be wiser to go back to a more rational approach, particularly that of the patent as a tool for economic regulation. Of course this calls back into question the sacrosanct rules of patent offices that define patentable matter as necessarily pertaining to invention. This calls them into question both because it may be economically efficient to grant patents (i. e. exclusive monopolies) regardless of the existence of an actual invention, and conversely because granting exclusive monopolies may be economically inappropriate for some creative (one hardly dare use the word "inventive"[15]) activities.

The patent system classically answers three main purposes. It attempts to achieve them by granting temporary monopolies that ensure actors a better chance to obtain a return on their investments, but they could be achieved by other means. These purposes are:

---

14 "*Seul le brevet permet un inventeur de protéger pleinement ses droits vis--vis de compétiteurs*" This statement appears in the French Government reply to the written question N° 42439 from député Jean-Yves Le Déaut to the ministre de l'économie,des finances et de l'industrie. Journal Officiel, question : 29 juin 2004, p. 4847, reply : 26 octobre 2004, p. 8394. http://www.questions.assemblee-nationale.fr/visualiser-questions.asp?K2DocKey=..\..\..\Questions\Q12\html\12-42439QE.htm@QST . This reply was also sent by M. Philippe Braidy (directeur du cabinet au ministère français de l'industrie) to several people who had written to the minister about the position of France regarding software patentability. It is likely that this text was actually written by officials from the French patent office INPI (Institut National de la Propriété Industrielle) which usually represents France on these issues. This is corroborated by the reappearance of the very same sentence, a year later, in two press releases from a different minister, M. François Loos (secrétaire d'état à l'industrie) : "Logiciels et propriété industrielle : François Loos favorable à un juste équilibre", 5 July 2005, http://www.industrie.gouv.fr/cgi-bin/industrie/sommaire/comm/comm.cgi?COM_ID=5411&_Action=200 , and "Brevetabilité des inventions mises en œuvre par ordinateur: François Loos regrette que les questions juridiques restent sans réponse", 7 July 2005, http://www.industrie.gouv.fr/cgi-bin/industrie/sommaire/comm/comm.cgi?COM_ID=5449&_Action=200 .

15 See the discussion on "*What is an 'invention'?* " by judge Peter Prescott, in Campbell vs UKPO, 21 July 2005, [2005] EWHC 1589 (Pat), http://www.bailii.org/ew/cases/EWHC/Patents/2005/1589.html

- to foster technical innovation (before the patent is granted)
- to encourage its publication (when the patent is granted)
- to promote its exploitation, especially industrial exploitation (after the patent is granted)

The third objective is essential. It typically justifies that public laboratories may give to private enterprises the exclusive exploitation of some results, since the role of a public laboratory is not to optimize its return on investment, but rather to optimize the public usefulness of the exploitation of its results. Without industrial use of innovation, or more generally without actual use of innovation in the economy and society, whatever the means, the other two objectives of the patent system remain only virtual progress.

It is thus altogether remarkable that, despite its major importance, this third point is not taken explicitly into account by the existing legislation, which again reinforces the ideological view that nowadays regards patents as rewards that should be necessarily attached to innovation.

Paraphrasing the little prince, what is important is to be of use to innovation and the economy. This position is actually more in favor of industrial property than the words of the United States Constitution which state that Congress can " *promote the Progress of Science and useful Arts, by securing for limited Times to Authors and Inventors the exclusive Right to their respective Writings and Discoveries.*[16] " Indeed, this constitutional clause allows the granting of a monopoly only to creators, and only inasmuch as it is useful to innovation. The French Constitution differs by excluding, as a principle, any private monopoly.[17] Thus the *privilege* of such a monopoly should only be allowed when justified by the common good, independently of any other consideration.

4. The price of sin

One common characteristic of most analysis of intellectual property is their silence regarding the effects of infringement suits. It is implicitly taken for granted that the infringer is by essence guilty and that he must make up for it at whatever price. How infringement and infringers are dealt with would be somehow a question unrelated to the role of patents and thus not worthy of discussion

This systematic omission might be justified if infringement were always the result of a deliberate intent to circumvent intellectual intellectual property, which could be taken as exceptional and abnormal behavior. This is, for example, the case with copyright. It protects original creations inasmuch as they carry in their form the personal mark of their authors, and they are thus not very likely to be imitated unwittingly.[18] The same generally goes in various technological domains where

16 *The Constitution of the United States*, Art. I, Sect. 8, Clause 8, http://www.archives.gov/national-archives-experience/charters/constitution_transcript.html .

17 « Tout bien, toute entreprise, dont l'exploitation a ou acquiert les caractères d'un service public national ou d'un monopole de fait, doit devenir la propriété de la collectivité. » — " *Any property, any enterprise, the running of which has or acquires the character of a national public service or of a de facto monopoly, must become public property* " — Preamble of the Constitution of the French Republic of 27 October 1946, included in the 1958 Constitution, http://www.legifrance.gouv.fr/html/constitution/const02.htm . Actually, the institutional role of the French "*bloc de constitutionnalité* " is different from that of the United States Constitution. It is a reference for the law making process but cannot be invoked directly in a legal proceeding.

18 This may not be completely true in some domains of creation, mainly because of the extension of copyright to all kinds of unexpected domains, or because of unconscious reuse by a creator of memorized information. However the history of software shows that litigation caused by inadvertent copyright infringement is essentially non existent for software. One may memorize a tune or even a part of a text. It is seldom done, especially by accident, for a program, and the complexity of programs is another protection with respect to inadvertent "inspiration". In particular inadvertent copyright infringement of software is also rare because of the usual separation of source code and object code. A good musician will know the score from hearing the music, and a good cook may guess the recipe of his food, but it is much harder – actually not plausible in most cases – to imagine its source code from simply using a program. Software copyright infringement is established, at least in part, on circumstantial evidence, and access to the source code is one such evidence. Proposing to impose "*general and mandatory disclosure of source codes for computer programs*" as

products and processes are simple to identify and make use of a small number of components that might be patentable or already patented.

But software programs are usually complex systems, often combining thousands of elements, all susceptible of interactions producing effects that are often unexpected, even for the authors of the software.[19] Given this situation, how can one hope to identify, among all these combinations and potential and hard to categorize effects, possibly not yet observed, which could have been patentable ? And how, at what cost, can one actually find out whether it is actually patented?[20]

Indeed, on can easily observe that numerous software authors are unknowingly infringing patents, and often many patents at the same time.[21] Inadvertent infringement is commonplace. As a consequence, creators find themselves in considerable legal uncertainty, even when they are not actually infringing, since they have no way to ascertain it.[22]

This situation also foster a variety of predatory behavior such as *poaching*. This idea is to apply for a number of patents, with no intention of exploiting them, then to wait for a software creator to inadvertently infringe, and make him pay ransom.[23] Such practices run precisely against the third and main objective of patents: promote exploitation of innovation to create economic value. The patent applicant does not exploit anything, and thus produces no benefits for society. Worse, he acts as a predator at the expense of entrepreneurs, who take risks and invest so that the economy can actually take advantage of the exploitation of innovations.

A legal system that can easily be infringed unwittingly, often without any reasonable way of knowing, is a bad system. Furthermore, whatever means are chosen by enterprises to manage this risk, it always amounts to sizable expenses or cash provision, which are out of proportion with the

---

suggested by Vincent Casiers (see page 48 of the reference in note 9 above) would introduce more legal insecurity (in addition to that software patenting might cause) since programmer would no longer be able to use source code unavailability as a defense. It is true that anyone can choose to make his source code public, and that some software authors do actually release their source code – most notably free software authors, – but experience show that these authors are not prone to litigation. This may seem in contradiction with our statement in section 6 below, that there is much to be learned from reading source code. But several points have to be considered. First there is already a considerable amount of source code available, it is better to read source code from people who are not prone to copyright infringement suits and many programmers will not read the code of proprietary software, even if it is available. .....xxxxxxxxxx.....xxxxxxxxxx....

19 One of the oldest examples is a theorem prover program that produced a proof about isosceles triangles by means of a proof technique that the author of the program did not know to be legitimate.

20 "Avoiding infringement can also be fraught with uncertainty, because the metes and bounds of software patent claims are often ambiguous." Report by the Federal Trade Commission, *To Promote Innovation: The Proper Balance of Competition and Patent Law and Policy*, Ch. 3, p. 52, October 2003. http://www.ftc.gov/os/2003/10/innovationrpt.pdf . One example among many, on the official site of the Joint Photographic Experts Group (JPEG), the Frequently Asked Questions (FAQ) has a section on the JPEG 2000 standard: "*What is the patent situation with JPEG 2000?* ". The answer starts by stating that "*in today's world it is impossible to create a multimedia standard of the complexity of JPEG 2000 without the possibility that some company would claim it infringed one or more of their patents.*" The rest of this long statement describes the policies followed by the standard group to avoid patented technology. http://www.jpeg.org/faq.phtml?action=show_answer&question_id=q3f042a68b1081 .

21 The case of patents concerning the design of web sites is quite illuminating, though it concerns rather simple situations from a technological viewpoint: Jonas Maebe, *Patented European webshop*, March 2004, http://webshop.ffii.org/ . Quite obviously, the pervasiveness of inadvertent infringement all over the Internet is a clear indication of the degree of non-obviousness of these alleged inventions for the proverbial "*person having ordinary skill in the art.*"

22 The doctrine of some legal systems, such as that of the United States, is to adjust the court decision according to whether one can prove that the infringement was intentional or inadvertent. This does not meet our problem here. To begin with, this doctrine has the undesirable side-effect – from the patent system viewpoint – of being an incitement to avoid any investigation on patented art, since any hint of such investigation might increase the financial penalty in case of litigation. But more importantly, we are not concerned here with a lack of proper care, i. e. inadvertent or deliberate neglect, but rather with an actual impossibility for reasons both technical and economical. REFERENCES

23 Actual infringement is not even necessary. It is often enough for it to be plausible to make the victim pay rather than litigate, because of the high cost of litigation.

average financial capitalization of this kind of activity based more on human intellectual resources than on financial investment.

It is consequently absurd to claim that "*it is up to each economic agent to choose his strategy*,[24] " as if the only economic effect of software patents were to set up a simple regulation of competition at the disposal of those who find it convenient, while harmless for all others, as if infringement were easily avoided and thus by nature just a behavior to be reproved. One must observe that, while there is much talk about the cost of patents for applicants, it is nearly always forgotten that non-applicants are "*passive smokers*" who suffer also from parasitic costs that may reveal major and even lethal.[25]

The first victims are SMEs, for lack of strong financial resources and of thick enough patent portfolios, though they are known to be the innovators and the risk takers, while the software industry already tends far too much to become monopolistic.[26]

Of course, programming is not the only domain where such complex systems are created. But in the physical world, this complexity will only be found in products or industrial processes that are highly sophisticated, requiring major financial commitments which can hardly be envisioned by SMEs. The costs induced by patentability, whether active or passive, have then a proportionally weaker impact. A regulation system can be useful if the economic distortions it causes by simply existing are minimal, and it is most likely an economic perversion otherwise.[27] It seems fair to think that extending patentability to software innovation would be an example of such perversion.[28]

In contrast, copyright is for software a form of appropriation with low and predictable regulation costs. It is thus economically better suited for activities requiring relatively low financial commitment.

---

24 « il revient à chaque acteur économique de choisir sa stratégie » in an official reply to a written question from député Jean-Yves Le Déaut, *see note 15 supra*.

25 A known example is the French company Getris Images. As they were attempting to expand their market in the United States, they were sued for infringement of 4 patents. Lacking the cash necessary for their legal defense, they were lead to bankruptcy and finally resold. All four patents were later declared invalid in a legal proceeding were the defendant was Adobe, a larger and richer company. There is actually a clear unbalance between the parties in this kind of law suits: the patent owner who decides to sue (or not) has much to win and little to lose, while the defendant has everything to lose and nothing to gain. Hence, it is much harder to find financial resources for defense than for suing. Or, where permitted by the legal system, the pursuant can probably retain a lawyer on a contingency fee, while the defendant cannot.

26 "*But other publishers and the majority of service companies rather see software patents as an unwarranted interference with the running of their business, given the risk of inadvertent use of the invention of a third party. Those in favor of patenting, for legitimate reasons, are mainly major corporations in informatics and telecommunications, with confirmed experience in the management of patent portfolios.*" This statement appears appears in the section "Is software patentability favorable to SMEs ?" of an official statement by the Syntec Informatique, the French Association of the Software and Computing Services Companies. *Position du Syntec Informatique sur la brevetabilité du logiciel,* Chambre Syndicale des SSII et des Éditeurs de Logiciels, 13 December 2000. http://www.syntec-informatique.fr/information/page.asp?article_id=36&Theme_id=1 . A large study on intellectual property and its protection was mounted in Britain "*to learn how the prevailing system for protecting intellectual Property is working, particularly for small and medium-size companies (SMEs).*" In its final reports it is remarked that, outside biotechnologies, "*the patent system gives SMEs no help in innovating*" as concluded "*from surveys and interviews involving over 2,600 firms*", adding further that " *it neither fosters nor protects their innovation. The patent system is at best an irrelevancy for most small firms.*"
Ron Coleman et David Fishlock, *Background and Overview of the Intellectual Property Initiative*, http://info.sm.umist.ac.uk/esrcip/background.htm . But, though SMEs have little use for patents, they have no protection against the risks and the costs of infringement suits.

27 Some regulations may actually be intended to introduce economic distortions in order to meet other, probably non economic, constraints, or simply to force internalization of externalities. But this does not seem to apply here.

28 As we shall see in the next section, these economic distortions, while already a significant burden on SMEs, are fully incompatible with new modes of non-commercial, not-for-profit production of resources that are specific of non-material creations. A major example is the open cooperative creation of free resources.

5. On secularism

The concept of secularism in its common understanding concerns the separation of the religious sphere from public affairs in which it used to interfere. Nowadays, we observe a similar phenomenon of interference of regulations intended for commercial activities in other areas of human activity.

The traditional criteria for patentability (novelty, non-obviousness and industrial use) were worked out at a time when innovation was naturally restricted to new products and processes involving " *the use of natural forces to control physical effects.*[29] " Actually this is not quite accurate because there were already innovations in non-material domains like mathematics. These patentability criteria could thus have been applied to mathematical innovation, even though enforcing such patents would have been hard to enforce.[30]

Legal texts usually extend the applicability of patents to all activities of a commercial character. French law states however that " the rights granted by a patent do not extend to activities accomplished privately and for non-commercial purposes.[31] ". This type of limitation is actually provided for by the TRIPS agreement, section 5, article 30, about patents.[32]

This setting seems fairly adequate for material innovations, involving physical phenomena. Indeed, outside the private sphere, the exploitation of material processes or the dissemination of material products at a significant scale necessarily requires investments and operating costs, which in practice are only affordable in the context of a commercial activity. The effects of the patent system are thus limited to the commercial sphere, and the cost of patent based regulation can usually be covered also by the commercial activity.

However things change completely when we are dealing with non material creations in a digital world where copy and dissemination can be performed at nearly no cost, as has been the case for about twenty years. This context give rise to new modes of wealth creation such as *free software*, the best known and most striking example, but not the only one.[33] With an innovative use of copyright, and taking advantage of the non rival character of immaterial creations, new resources are created through open cooperation and made freely available to all, quite often without any direct (or even indirect) commercial interest being involved.

---

29 Position of the European Parliament, see note 7 above.

30 Yet trade secret was sometimes used to "protect" mathematical innovation. Without going as far back as Pythagoras (6th century BC), whose motivations for keeping mathematical knowledge secret are more difficult to ascertain, we know that in the 16th century, mathematicians used to keep secret some of their algorithmic innovation so as to challenge each other with problems to solve. At the end of the 18th century, descriptive geometry was for a while a French military secret. It already had industrial applications at the time. Alain Bru (Jean Baptiste Margeride), Histoire de la guerre à travers l'armement, Ch. VI, section 1, 2001. http://www.stratisc.org/act/bru/act_bru_hisguerre_44.html .

31 Article L613-5 of the French intellectual property code:

Les droits conférés par le brevet ne s'étendent pas :

a) Aux actes accomplis dans un cadre privé et à des fins non commerciales ;

b) Aux actes accomplis à titre expérimental qui portent sur l'objet de l'invention brevetée ;

c) A la préparation de médicaments faite extemporanément ...

http://www.legifrance.gouv.fr/WAspad/UnArticleDeCode?commun=CPROIN&art=L613-5 .

32 TRIPS Agreement (Annex 1C of the Marrakesh Agreement), Section 5, Article 30, Exceptions to Rights Conferred: " Members may provide limited exceptions to the exclusive rights conferred by a patent, provided that such exceptions do not unreasonably conflict with a normal exploitation of the patent and do not unreasonably prejudice the legitimate interests of the patent owner, taking account of the legitimate interests of third parties. " http://www.wto.org/english/docs_e/legal_e/27-trips_04c_e.htm#5 .

33 Another economic model permitted by non-materiality is *shareware*, where the lack of cost is used to freely disseminate a creation, for example via existing customers or as free supplement to magazines, while legally (or otherwise) requiring payment of a fee for protracted use.

Twenty years of actual experience show that this, so-called, *free*[34] model of creation has its own incentives and competition mechanisms, and is indeed efficient both for the economy and for innovation. This is not too surprising since it is precisely the model evolved by scientific research, and which is at the source of its fast and effective development, especially after the invention of the printing press and the creation of the first scientific journals that, like the Internet today, significantly improved communications and exchange between creators.[35] It can be particularly observed that the freedom to modify and republish granted to everyone leads to a significant enlargement of the pool of potential contributors and innovators[36] and foster innovation through the direct involvement of users[37]. Additionally, in the context of copyright, this model remains compatible with, and most likely complementary to, the more traditional mode of software production usually termed "*proprietary*". However, free development is fundamentally based on the absence of constraints and financial requirements[38]. It is therefore incompatible with the existence of software patents for at least two reasons:

- contributors to free development cannot usually assume passive costs of patents, particularly those related to the risk of inadvertent infringement on patented methods[39] not identified as such,
- and even assuming that it is possible to identify all patented methods used, it would be impossible in practice to get licenses for freely modifiable and reproducible software.[40]

34 Various references are accessible from the various "*free software*" related articles in Wikipedia: http://en.wikipedia.org/wiki/Free_software . See also : Bernard Lang, The battle for free software, *Swiss Yearbook of Development Policy 2003*, Information Society and International Cooperation, Vol.22, n°2, iuéd, Geneva, November 2003, http://www.unige.ch/iued/wsis/DEVDOT/01725.HTM . Free software is also often called *open source software* (*OSS*) or sometimes Free/Libre/Open-Source Software abbreviated as *FLOSS*. In a nutshell, software is said to be free if anyone owning a copy can technically and legally use it, study it, modify it and disseminate it without restrictions. Technically, this implies at least availability of the source code. Legally, this requires that it is provided with a license meeting a set of criteria for which there is a consensus corresponding to the definition given by the Open Source Initiative (OSI) : *The Open Source Definition*, http://www.opensource.org/docs/definition.php . Software is said to be *copylefted* if the license requires that the software or any derived software can only be disseminated as free software. The most widely used *copyleft* license is the GNU General Public License (GPL): http://www.gnu.org/copyleft/gpl.html . In France, another copyleft license called CeCILL was officially created by three public laboratories for free software produced by the public sector: http://www.cecill.info/ . Other licenses are used to account for more complex situations regarding components available in software libraries.

35 This was actually the explicit intent of the two visionaries who initiated the research about the Internet: Robert W. Taylor and Joseph C.R. Licklider, *The Computer as a Communication Device*, Science and Technology, April 1968. http://gatekeeper.dec.com/pub/DEC/SRC/publications/taylor/licklider-taylor.pdf .

36 Patenting is promoted mainly as an incentive for innovators, but it actually also have opposite effects. For example, by discouraging or even blocking the development of free software, software patents may significantly reduce the pool of people who might contribute to innovation (outside traditional commercial or state organizations), simply because of legal and technical obstacles to their involvement in the creation or the evolution of software resources that might have benefited from their ideas.

37 «Open Source Software Projects as User Innovation Networks», Eric von Hippel, Conference "Open Source Software : Economics, Law and Policy", 20-21 June 2002, Toulouse, France. http://www.idei.fr/activity.php?a=2493 or http://www.idei.fr/doc/conf/sic/papers_2002/vonhippel.pdf .

38 In his letter to the Patent Office (USPTO) on February 1994 (*see reference in footnote 1 above*), Donald Knuth recalled: « I developed software called TeX that is now used to produce more than 90% of all books and journals in mathematics and physics and to produce hundreds of thousands of technical reports in all scientific disciplines. If software patents had been commonplace in 1980, I would not have been able to create such a system, nor would I probably have ever thought of doing it, nor can I imagine anyone else doing so. » The importance of this system is still steadily increasing in international scientific communication. http://www.softwarepatenter.dk/baggrund/letter-uspto-donaldknuth/view . From an economic point of view, this is easily understood. The viability of free development hinges fundamentally on the non rival character of software, on the fact that the marginal cost for producing and disseminating one more copy is essentially zero. If patent royalties have to be paid, more or less proportionally to the number of copy produced, the marginal cost is no longer zero.

39 As shown earlier, these methods are essentially specification methods anyway.

40 A simple reason is that the royalties are usually based on the number of units produced that actually use the patented methods, which is of course undefined and without any a priori limit in the case of free software. To compute the

We thus have an unprecedented situation due to technological evolution: massive public exchange of resources is possible independently of any commercial activity, with positive effects on the economy and innovation,[41] but it is not compatible with older practices that were evolved to regulate commerce and investment.[42]

The question raised today is then to understand how to extend existing legal practice in this new technological and economic context to allow these original and effective modes of wealth creation.[43]

Actually, there is not much room for maneuver. There seems to be little hope of changing national and international instruments that determine the limitations to the rights granted by patents. Hence, if we do wish to preserve this original mode of creation, with its increasing success and popularity due to reasons both economical and political, the only path left open is not to extend patentability to software, since no existing text requires such extension.

More generally, this conclusion applies as well, for precisely the same reason, to all areas of immaterial creation. The "*technical character*", as necessarily involving a material or physical aspect in innovations, is thus once more a meaningful criterion as to the effects of patentability. It seems sensible to adopt this criterion for its economic consequences and, indirectly through free

---

royalties on an income basis (e. g. a percentage of the gross income) make no good sense either. Free licenses have been granted by some patent owners for free software development, for example: Raph Levien's patents in image processing http://www.levien.com/patents.html (11 May 2000); Dirac patents, freely usable in practice according to the BBC, though their precise legal status is not too clear, http://www.bbc.co.uk/rd/projects/dirac/licences.shtml (2004). But this remains rather marginal compared to the mass of patent applications, and it does not solve in any way the issue of inadvertent infringement.
See also patent releases by IBM and SUN. .... projects stopped by patents

41 Though beyond the scope of this paper, there are many other essential aspects of free software regarding for example its social and economic effects through more open competition and geographic devolution of technological competence, or regarding the control of information and communication infrastructure and its consequences on the sovereignty of nations over their culture and their national security. See for example the work of Lawrence Lessig on constitutional aspects of information infrastructure: Lawrence Lessig, *Code and other laws of cyberspace*, Basic Books, October 1999, ISBN 0-465-03913-8. http://www.code-is-law.org/ .

42 Actually, existing laws themselves reveal a form of inconsistency with the economics characteristics of software. In France, the Versailles court of appeal, having to decide on a case of massive forgery of Microsoft software licenses, pronounced fine and jail penalties for forgery, but rejected claims of compensation for lost business – there are no punitive damages in France – on the basis that Microsoft could not justify loss of business, nor could give any indication of how the price of a license is determined. However, because of the weakening of the trademark caused by the forgery, the court did grant some damage for "préjudice moral." *Joël B. et autres / Microsoft et autres*, Cour d'appel de Versailles, 9ème chambre, Arrêt du 09 septembre 2005, http://www.legalis.net/jurisprudence-imprimer.php3?id_article=1489 . This decision could still be broken by the higher Cour de Cassation, but only on a procedural basis, not on the assessment of the facts, namely that the prejudice is as undefined as the value of a license. Indeed, it is well known in the profession that the price of licenses – even commercial ones – is arbitrary and can even go down to zero, as economic theory would predict given the absence of marginal cost. But is that an inconsistency of the law, or a sign that it is attempting to regulate an inconsistent economic model ?

43 For some time, nearly all documents aiming at promoting software patentability insist that this patentability would in no way hinder the development of free software. This is certainly a tribute to, and a recognition of, the success and the importance of this mode of wealth creation. Nonetheless, it does not make this assertion any less false. A number of free software projects have been blocked by patents in the United States. For example, the Sorenson patents on the codecs used in Quicktime have prevented the development of a free interpreter of Quicktime files. A confidential memorandum from Microsoft stated as early as 1998: "The effect of patents and copyright in combatting Linux remains to be investigated." Vinod Valloppillil and Josh Cohen, *Linux Operating System - The Next Java VM?*, v1.00, Aug 11, 1998, http://www.opensource.org/halloween/halloween2.php . Since then, Microsoft has applied for numerous patents on formats and communication protocols, some being clearly pointed at free software projects like, for example, Samba. Bruce Perens, *The Microsoft penalty that isn't*, CNET News,15 avril 2002, http://news.com.com/2010-1071-882846.html . These patents have so far kept a low profile, giving possibly the illusion that they are not a major threat. It is more likely that, as long as software patentability is not legalized in a significant part of the world, and particularly in Europe, the fear of blocking its further extension to the planet by frightening legislators has so far discouraged any legal action whose effect would be geographically limited anyway.

software, for its political consequences.

6. And scientific research ?

It is quite remarkable that, in this debate where each and everyone pretends to regulate to increase innovation, very few actually mind the opinion of research scientists,[44] of the very people who actually practice a computer research activity rather than those who claim to be speaking in their name.[45] Whether legal or economical, the adverse consequences of patentability on software innovation have been analyzed by other authors[46], and my intent here is to give the viewpoint of the research scientist in his professional practice.[47]

The price in return for the monopoly privilege granted by a patent is often supposed to be the disclosure of the invention, as a contribution to the public heritage of knowledge and know-how. My experience as a research scientist is that this does not work for software, quite the contrary, and for many reasons.

The first such reason is that software patents, as already granted in the United States, or with doubtful legality by the European Patent Office (EPO), are most often trivial from a researcher point of view. Additionally they are often written in a gibberish quite impossible to understand in any reasonable amount of time by a sane professional.[48] Therefore, to look for useful teaching in patent applications is *a priori* a very low yield activity in a scientific universe already overcrowded with useful information and results. I never met a software research scientist claiming to have used patent claims as a source of information. It would certainly be useful to have more precise statistics

44 *Petition to the European Parliament*, May 2003, signed by 30 reputable research scientists. http://www.cs.chalmers.se/~bengt/petition.pdf .

45 This distinction is essential. The "research scientists' point of view" is too often stated by people in charge of technology transfer who tend to be more concerned with short-term micro-economic returns (possibly better regarded by decision makers), and who have often no personal experience of research, or an experience limited to a scientific domain that may not be relevant. Thus, like many legal specialists, they naturally tend to have a global vision that takes no account of the economic, technological – or even political and social – parameters that characterize the various research areas, and more generally the various aspects of scientific and technical activities.

46 Regarding the legal effects of patents, see for example the article by Bessen and Maskin. Though it does not prove, because of the excessive simplicity of the sequential innovation model, that software patentability slows down innovation, it at least shows that the even more simplistic model based on patents as incentives for innovation has no better credibility in this context: James Bessen and Eric Maskin, Sequential *Innovation, Patents, and Imitation*. MIT Dept. of Economics, Working Paper No. 00-01January 2000. http://www.researchoninnovation.org/patent.pdf .
From a purely economic viewpoint, the study of Bessen and Hunt shows that the funding of industrial property appears to be managed at the expense of research and development activities, competitive investment remaining constant: James Bessen and Robert M. Hunt, *An Empirical Look at Software Patents*. FRB of Philadelphia Working Paper No. 03-17, mars 2004. http://papers.ssrn.com/sol3/papers.cfm?abstract_id=461701 .
The FFII site proposes a bibliography on these issues: *Research on the MacroEconomic Effects of Patents*. http://swpat.ffii.org/vreji/miroir/sisku/index.en.html .

47 Bernard Lang, Liberté et barrières dans l'univers des logiciels. *SPECIF*, No. 50, décembre 2003. http://pauillac.inria.fr/~lang/ecrits/liste/specif2003.pdf .

48 Martin Campbell and Patrick Valduriez, " *A technical critique of fifty software patents* " (January 2005) http://ssrn.com/abstract=650921 . Though they study a selection of "*good*" patents, the authors note that " *the level of disclosure is often less than optimal, indicating the need for reform.*" This is corroborated by Dan L. Burk and Mark A. Lemley, " *Is Patent Law Technology-Specific?* ", Berkeley Tech. Law Journal, Vol. 17, p. 1155, 2002, http://ssrn.com/abstract=349761 : " *The Federal circuit has essentially excused software inventions from compliance with the enablement and best mode requirements [...]* ". The former document (Campbell and Valduriez) is interesting in that it aims at establishing that, from the point of view of classical patentability criteria, software could be an adequate domain of patentability. We have shown in section 1 above that, even from a purely technical viewpoint, the issues can be more subtle than they claim. Furthermore, their whole analysis could be conducted as well on the proof techniques used in a good set of mathematical publication, though no one is suggesting (yet) that mathematical proof techniques ought to be patentable. Note further that in this document, we have generally avoided the use of the words "*technique*" or "*technical*" to prevent abuses of interpretation or quotation. We deliberately use it here, as would mathematicians, about mathematical proofs which are not patentable subject matter.

on this topic. But, anyway, it can also be objected that the most interesting innovations are also published elsewhere once the patent application is made. This too would be worth an investigation.

Besides, patented methods are not the only useful information in software. For many programmers, there is often more to be learned from the study of the source code of programs, of their architecture and their style, than from the "*innovations*" that one might want to patent.[49]. Furthermore, this source code is essential for the maintenance and evolution of programs, while the useful life of the program may exceed that of the companies or organization that produce them. However, the software methods used (at least to specify what is to be obtained) are more readily analyzed by studying the source code if it is available, but much harder to find when only the executable object code may be examined.[50] Since, because of its high complexity, any software runs a high risk of inadvertently infringing patents, the best of protection of the authors is to prevent easy identification of the methods used by giving access only to the executable code. Far from fostering dissemination of useful knowledge, patentability would result in incentives for secrecy, regardless of whether the software creation is innovative or not.

In particular, there is a high risk of discouraging or hampering all modes of creation and innovation based on open dissemination of source code. This matters of course for free software, previously discussed, whose development relies fundamentally on the availability of source code. But it matters as much for the research work of many software research scientists. Collaboration between scientists relies on publication of theoretical articles, but also on the exchange of data and experimental realizations which, in the case of software, are actual programs. Proper scientific development requires open communication, so that everyone can contribute, analyze, critic, take up or modify the work done by colleagues, and minimize the global investment for partners as well as for competitors.[51]

There is of course, in various legal systems, a research exemption[52] that places research work outside the patent protected domain, but this is restricted to research bearing specifically "on the subject matter of the patented invention." Obviously, this condition cannot be met by all the components of a research software system, most of the components being only the necessary context for the few more experimental ones. Thus the scattering of ownership and the transaction costs it entails may handicap technical developments, and even more the exchanges between

---

49 If software research scientists took the time to patent all of their "software ideas" at the level of what is commonly accepted by patent offices, they would not have much time left to get down to serious work and actually build useful systems. The difficulty is not to imagine solutions, but to actually build them and even more to fine tune them. This explains for example why software projects are almost systematically delayed. Even experienced professionals always underestimate the difficulties of actual realization. To describe the principles is always much easier than to make it actually work, even when one is certain of the end result. There is no lack of ideas, but of time to implement them. – Given the shortage of good programmers and research scientists, patent applications are as much wasted time and energy that adds to the toll exacted by the patent system on the strained resources available for innovation and development. This limitation on resources might be construed as a natural regulation on the number of patents that should improve quality. However such a conclusion is unwarranted for at least two reasons: *a)* there is not lack of mediocre professionals who can patent mediocre results as the best they can come up with, while good people have better things to do, and *b)* the purpose of patenting, from the applicant's point of view, is not to disclose useful information (quite the contrary actually) but to hinder the progress of competitors; hence there is more to be gained by patenting trivial techniques of wide applicability rather than real innovations.

50 Jugement britannique : illegal to look at object code to find previous art - see reference in note 15 *supra*. ...etc...

51 Fostering competition and contradiction is an essential ingredient for the advance of scientific knowledge.

52 Article L613-5 from the French Intellectual Property code, see note 31 above: « Les droits conférés par le brevet ne s'étendent pas [...] aux actes accomplis à titre expérimental qui portent sur l'objet de l'invention brevetée ; », which translates as "*The rights granted by a patent do not extend [...] to activities performed for experimental purpose and bearing on the subject matter of the patented invention.*"

US – UK law

researchers[53]. This kind of situation has already been observed in the area of automatic natural language processing. It actually resulted of an abuse of copyright and data-base rights,[54] used more extensively than is usual in most of the other areas of computer science, which caused a significant slowdown of innovation and useless duplication of much research and development.[55]

What is worse is that these pernicious effects can propagate to many scientific domains, even outside the realm of computer science, because of the increasing role of software in conducting experiments and analyzing experimental results. With the low cost of electronic publication, it is nowadays possible to annex to any publication of scientific results the raw experimental data and the software that was used to produce and to analyze it, so as to create the best conditions for the critical analysis or the reenacting of these experiments. Is it still possible to improve in this way our reporting practices if it is at the risk of being sued for patent infringement on methods that may be used in the software ?

As it does for free software, which is actually a very similar mode of creation, software patentability runs the the risk of strongly hampering in many fields research activities that are useful to innovation and the economy, and which are very often conducted outside the commercial world.

7. The freedom to communicate

Software is at the core of what is often called ICT, Information and Communication Technologies. Thus it comes to no surprise that software communication protocoles (procedures) and formats (data organizations) are pervasive. Communication appears in several ways:

- internal communication between the various software components used on a given system,
- external communication, through the Internet or other means (CD, DVD, removable memory, etc.), between different computerized systems, especially for human communications (email, web, chat),
- man-machine communication interfaces.

It is essential for software to be able to communicate according to interfaces and communication protocoles of other existing systems, which are often official or *de facto* standards.

This is essential for man-machine interfaces because users are generally extremely reluctant to switch to a new mode of communication different from the one they have become used to, often with some learning effort. Similarly, enterprises tend to use only software with the most common man-machine interfaces so that employees are more easily found or replaced, and without training costs.

But the ability of software to communicate with other software, whether to connect components or to exchange between systems, is even more critical. Without this ability, it is generally useless. What would be the use of a text processor unable to read the documents sent to you by another user?

53 More generally, these remarks imply that the research exemption is essentially ineffective to help innovation in complex systems. One should however note that this is less of a problem for complex material systems, again because of the greater cost to simply implement and run such systems, independently of the intellectual property costs.

54 Directive base de données

55 This problem with research in Natural Language Processing is probably due to historical circumstances and a significant early involvement of private enterprise in this area, combined with the high cost of creating resources “by hand”. Currently, one may observe an evolution towards more openness and sharing, which should speed up the maturation of this domain and, in the end, be beneficial to enterprises. This kind of situation is quite similar to the economic slowdown cause by excessively protectionist policies. We should note that the current trend towards less protectionism results from the individual decisions of some actors who choose to make the resources they create available to all, without running the risk of being blocked by those who wish to keep their own for themselves. While this is possible with a copyright regime, it would be impossible with a patent regime since it does not allow independent re-creation.

And how would you create documents with it if it cannot follow the protocoles needed to run the printer? This ability to communicate is what is generally called *interoperability*.

The need to communicate is the source of network externalities which give a major competitive advantage to software using the leading communication standards. To control these standards, by whatever means, is the best route to set up a monopoly by excluding competitors' software, with the usual negative consequences that may have on the economy, and even more on innovation.

One way to ensure this control is secrecy of protocoles and formats, which can be effectively preserved by the unavailability of the software source code. This is what motivated the European legislator, in article 6 of the 1991 directive on the legal protection of computer programs, to authorize analysis and partial decompilation of computer programs " *to achieve interoperability.*[56] " At the time, no other means than secrecy was available to enable appropriation of the methods used to communicate. Thus this major stipulation by the legislator was a conscious and calculated decision to prevent such appropriation and preserve interoperability, and thereby preserve free competition and freedom for independent innovations. To allow this appropriation through patenting would amount to making the directive stipulation null and void in practice.[57]

The importance of the interoperability principle is also indirectly acknowledged in article 4 of the recent French law on the digital economy, which defines an "*open standard*" as a " any protocole for communication, interconnection or exchange and any inter-operable data format whose technical specifications are public and without any restriction of access or use."[58]

Forbidding private appropriation of communication modes, and specifically the patentability of any part of a communication standard, is fully justified by the anti-competitive practices it would encourage.[59] It is also justified because the economic value of a standard is primarily in the number

56 Council Directive 91/250/EEC of 14 May 1991 on the legal protection of computer programs, *Official Journal* L 122 , 17/05/1991, p. 0042 - 0046, http://europa.eu.int/smartapi/cgi/sga_doc?smartapi!celexapi!prod!CELEXnumdoc&numdoc=31991L0250&lg=en .

57 Sir John Sulston, Nobel laureate in medicine, give a remarkable example of how software control can interfere with research work, when equipment providers try to keep control on the uses these equipments are put to. *In* John Sulston and Georgina Ferry, *The Common Thread: Science, politics, ethics and the Human Genome* (NAS, Bantam Press, 2002). See the excerpt: *Why software - in particular embedded software - should not be patented*, http://www.debatpublic.net/Members/paigrain/blogue/embedded .

Voir aussi le chien robot japonais AIBO et les procès aux créateurs de programmes. ... ou la reprogrammation d'autres appareils et les innovations que cela permettrait .. von Hippel

58 « On entend par standard ouvert tout protocole de communication, d'interconnexion ou d'échange et tout format de données interopérable et dont les spécifications techniques sont publiques et sans restriction d'accès ni de mise en oeuvre. » LOI n° 2004-575 du 21 juin 2004 pour la confiance dans l'économie numérique, *J.O.* N° 143, 22 June 2004, page 11168. http://www.legifrance.gouv.fr/WAspad/UnTexteDeJorf?numjo=ECOX0200175L .

59 Even with compulsory licensing, the time lost by the licensee to obtain the license would be prohibitive on these fast evolving markets. In addition, as previously shown, the price system for granting licenses is incompatible with the free dissemination of software, whether for free software, shareware or research software. Of course this remains true of so-called RAND licenses (i. e. *reasonable and non-discriminating*) which have been also suggested for patents on standards. For the second parliamentary reading of the directive on the patentability of computer-implemented inventions, the amendment 70, by Erika Mann, Arlene Mc Carthy and Dagmar Roth-Behrendt, suggested to impose a legal definition of RAND licensing specifying that "there should be no de facto discrimination between different development and licensing models, and the licensor should therefore offer, at least as an option, terms and conditions compatible with proprietary as well as open-source development models." The comments further insist that "the terms of a license should not be used to exclude any particular software development model from use of a technique essential for interoperability." If nothing else, this amendment does recognize two facts: that patents on standards are a real problem, especially where interoperability is concerned (and interoperability is indeed constantly referred to in this amendment and the sister amendment 68), and that RAND licensing, as commonly defined, is not a solution. While the new requirements of this amendment on RAND licensing might possibly alleviate to some extent the problems we have identified regarding patenting, there is no indication on how they could actually be implemented, and their practicability remains doubtful. *Texts of amendments and motions for resolutions: A6-0207/2005: Amendments*, EUROPARL, http://www2.europarl.eu.int/

of its users, much rather than in any innovation it might bring.[60]

This viewpoint was adopted by major Internet standard organization – and the Internet is the primary communication medium – such as W3C[61] and IETF[62], who developed and promoted standards unencumbered by any intellectual property constraint, and even produced free software components implementing these standards. These components could be used to test the standards, or could be embedded in applications using the standards.

8. The inflexibility of the dogma

This analysis of copyright versus software patents attempted to avoid dogmatic discussions, based on doctrine, jurisprudence, laws and treaties, while remembering some principles that seem to still meet a general consensus. We tried to stick to an agnostic approach, based exclusively on usefulness and effectiveness principles for both innovation and the economy, given existing economic, technological and sociological constraints. In a way, we even adopted a cynical point of view, since we did not consider political, cultural or ethical arguments – though they are unquestionably significant aspects of the general welfare – which should normally be taken into account in such a debate.

Many arguments we developed[63] boil down to the inescapable fact that non-material creations can be developed in a not-for-profit, non commercial setting because they are non-rival, i. e. because they can easily be duplicated, disseminated and modified at no cost, for better of for worse. If we

---

sce/server/internet/amend_motions_texts/sce_amend_motions_texts_main_02.jsp?ref=A6-0207/2005 – alternatively, see *Software Patents Directive Plenary Amendments: FFII Analysis and Guide*, FFII, http://www.ffii.org/~amacfie/plen0507/plen0507En/Plen0507.pdf

60 This is implicitly recognized by amendment 70, see note 59 above, which states that "fees, if any, should reflect innovative contribution only, and not monopoly rent or the strategic value or revenues deriving merely from an ability to exclude rivals from marketing inter-operating products." Of course, such a vague statement is largely inoperative. But it also runs contrary to much of the patent infringement jurisprudence that awards damages based on market loss for the patent holder. It is however to be noted that the authors of this amendment, that underlines the risks of software patentability, were actually known supporters of software patentability. It was also made known at the time that this amendment had the support of the European Commission (though not officially or in writing, for obvious reasons).

61 The position of the World Wide Web Consortium (W3C) is quite explicit : "In order to promote the widest adoption of Web standards, W3C seeks to issue Recommendations that can be implemented on a Royalty-Free (RF) basis. Subject to the conditions of this policy, W3C will not approve a Recommendation if it is aware that Essential Claims exist which are not available on Royalty-Free terms." Daniel J. Weitzner ed., *W3C Patent Policy*, 5 February 2004. http://www.w3.org/Consortium/Patent-Policy-20040205/ .

62 The first 30 years of development of the Internet have produced a wealth of innovations in communication technologies. In that period, the Internet Engineering Task Force (IETF) had a policy of open standards validated with free (open-source) implementations, which later inspired the policy of W3C (see note 61 above). This was of course made easier while software was not patentable in the United States or anywhere else. After a period of time during which the IETF had chosen to more or less ignore the problem of standards encumbered with patents (RFC 2026, October 1996, http://www.ietf.org/rfc/rfc2026.txt), the issue became suddenly inescapable, particularly for the standardization of anti-*spam* technology by the MARID group. Despite the extremely high cost of *spam* abuse to the world economy and the urgent need of a new communication standard dealing with that problem, the project MARID had to be abandoned after a year work, because of a patent that blocked the necessary consensus. This blocking of an essential innovation by a patent is even more appalling when one reads, at the very beginning of the charter of the *Intellectual Property Rights (ipr)* working group, version dated 27 September 2005 : "*The IETF and the Internet have greatly benefited from the free exchange of ideas and technology. For many years the IETF normal behavior was to standardize only unencumbered technology.*" http://www.ietf.org/html.charters/ipr-charter.html . — The problems raised by software patents in the design and use of software standards are discussed by Bruce Perens in *The Problem of Software Patents in Standards*, November 2004, to appear in the collection "*The Standards Edge*", Sherrie Bolin ed., Sheridan Books. http://www.perens.com/Articles/PatentFarming.html .

63 The arguments developed at the beginning of this paper are of a different type. They rather attempt to document the conceptual inadequacy of patents to deal with creations of a logical or mathematical nature. We do not consider them here so as to simplify this part of the discussion which is more concerned with the effects of infringement than with the meaning of patentability.

recognize that this non-material character is at the heart of the issue, we might consider a new approach based on the definition of the patent infringement domain – to be circumscribed to material uses which cannot be dematerialized – rather than on the definition of what is patentable.

This approach would provide a simple answer to a very recurrent question: why can one get exclusive ownership on the material implementation of a process by means of circuits, but not on an equivalent implementation by means of software ? This question is all the more acute since both versions (circuit and software) may be derived from a single source program, thanks to silicon compilers. The answer is quite simple: because factory produced circuits pertain to the material economy while software does not. Though there may be a functional equivalence, there is no equivalence where economic and innovation networks are concerned. But the latter are precisely what the patent system is intended to regulate.

The principle of a new computing circuit, produced by assembling transistors in a factory, could be claimed for appropriation. But a software simulation of the same circuit should not be included in this appropriation, not because it is fundamentally different, but because it is no longer of a material nature, provided the software is on a programmable carrier. By "*programmable carrier*", we mean here any medium whose content may be modified (RAM, disk, ...) or any removable commodity carrier, simply writable by anyone, even if not rewritable (for example a CD or DVD, or possibly a PLA). Conversely, a non removable carrier whose content cannot be changed, might be considered as material, irrespective of the fact that it actually carries software, exactly like any industrial electronic circuit, and the behavior it defines could be claimed for exclusive appropriation or sued for infringement.[64]

A control component of a system could not be infringing (other than infringing copyright) if it were not material, that is as long as it would be possible to transmit it electronically and to inscribe it on a programmable carrier within the system, or alternatively to inscribe it on a commodity carrier which would then be inserted in the system.[65] Of course, any application of the system that would be an innovation regarding the use of natural forces could be claimed in a patent, independently of the means used to control this application.

The advantage of this change of paradigm is that it gives a perspective that can also account for some predictable evolutions of technology. One interesting example is the development of physical components that can be cheaply synthesized on commodity equipment from a non-material specification. This is not science-fiction since, for example, various laboratories and companies are developing printing techniques for electronic chips, using standard commercial ink-jet printers.[66] Inasmuch as such a circuit can be inserted in a removable way in a system, like an engraved CD, it follows the techno-economic logic of non-material creations, and it becomes legitimate, in our paradigm, to consider it as such.

At this stage, our proposal is still speculative, and it certainly requires a much deeper and detailed analysis to understand its effects and consequences. Whatever these may be, even if this approach or any other based on a redefinition of infringement were found to provide an adequate answer to the objectives sought by the legislator, it seems unfortunately doubtful that it could actually be

---

64 Assuming of course that the exact nature of this behavior can be precisely determined, which is not necessarily easy, as we have seen, when the code or the circuit is produced by a compiler.

65 Though reached independently, this viewpoint is similar to one expressed by Richard M. Stallman regarding freedom of access to software source code, a topic on which he is usually adamant: « You know, I'm not concerned with the computer inside my microwave oven. Yes, I know that there's a computer in there, but there's no facility for loading programs onto it, so the issue doesn't arise in practical life. » in *GNUisance: Pigdog Journal Interviews Richard Stallman*, 15 June 2004. http://www.pigdog.org/interviews/stallman/interview_with_stallman3.html .

66 Toby Howard, Plastic makes chip printing possible, Personal Computer World, 23 May 2001, http://www.pcweek.co.uk/Features/1122215 .

implemented because of the inflexibility of the legal instruments actually in force, since they are careful to leave almost no margin to adapt the definition of what constitutes infringement.[67]

9. Conclusion

For many reasons the economy of non-material creations differs considerably from the material economy. For many reasons the processes and networks for non-material innovation can follow paths that differ significantly from those of innovation in the physical world. The purpose of patents and more generally of intellectual property was always to provide incentives that foster innovation and its actual use, given economic and other constraints. The border between material and non-material is thus a natural discontinuity that justifies reassessing our incentive policies, even without taking argument of the fact that this border was always respected in the past, for mathematics to begin with, though may be sometimes for other reasons than those we are concerned with.

On the 24th of September 2003, the European Parliament voted an amended directive that instituted this border as the limit to the patentability of innovations by asserting that what is technical must be material. It is rather ironic to see some law specialists complain of this identification between technical and material. It is indeed a linguistic device to preserve existing legal texts at minimal cost within the changing context of their application. But this device is used for political, economic and others reasons which owe nothing to chance, history or ignorance of the stakes. This way of abusing language or reality is, for quite respectable reasons, a practice to which the legal profession was never averse.[68]

In its Common Position,[69] the Council of the European Union tried to back up on the precise wording of the Parliament by not defining the technicality constraint, which would of course open the door to unlimited patentability. Some legal experts justify this by arguing that too precise a definition of the law prevent its adaptability to evolving needs. This arguments seems specious at best. Our analysis shows that it is actually the inflexibility of existing legal instruments that threatens economic evolution, and that the flexibilities these experts want to preserve are only one way and intended to facilitate the elimination of whatever breathing space is left to take account of new economic evolutions.

The final rejection of the directive by the European Parliament in its second reading, a well as the

---

67 Another suggestion to (re)define infringement was proposed by Jean-Paul Smets-Solanes, Stimuler la concurrence et l'innovation dans la société de l'information, publication du Conseil Général des Mines, 30 August 2000, http://www.cgm.org/rapports/brevet.pdf , translated as *Stimulating competition and innovation in the information society*, English Version 1.0, 23 March 2001, http://pauillac.inria.fr/~lang/ecrits/autres/smets/brevets_plan-en.pdf . It is most likely as incompatible as this one with the rigidity of existing legal texts.

68 An amusing example is the legal statute of the homing pigeon. Though animals are usually considered movable property, which seems natural enough, the homing pigeon is considered as real property (as in real-estate) in the civil code. Livre II du code civil, article 524. http://www.legifrance.gouv.fr/WAspad/VisuArticleCode?commun=&code=&h0=CCIVILL0.rcv&h1=3&h3=2 .
Another example, closer to our topic, is article 10 of TRIPS which states: "*Computer programs, whether in source or object code, shall be protected as literary works under the Berne Convention (1971).*" TRIPS Agreement (Annex 1C of the Marrakesh Agreement), Section 1, Article 10, Computer Programs and Compilations of Data, http://www.wto.org/english/docs_e/legal_e/27-trips_04_e.htm#1 . Following our own arguments in section 1, unlike source code, object code might possibly not be considered as "*literary work*" from a common sense point of view. This is even strenuously argued by V. Casiers on page 14 of the reference in note 9 above. But it is all irrelevant, since legal definitions are intended to reflect the intent of the legislator, rather than natural or logical classification. The legislator does not wish to assert that programs, whether source or object code, are literary work, but only that they shall receive the same legal treatment, for whatever reasons. As we remarked in note 10, his only concern should be whether the chosen regulation mechanism is adequate for his purpose.

69 *Common position adopted by the Council with a view to the adoption of a Directive of the European Parliament and of the Council on the patentability of computer-implemented inventions*, 11979/1/04 REV 1, PI 61, CODEC 962, Brussels, 7 March 2005. http://www.europarl.eu.int/commonpositions/2005/pdf/c6-0058-05_en.pdf .

current pressure of developing countries on WIPO,[70] is evidence that the issues are not as clearcut as large ICT corporations would make it. It also gives us more time to better understand the differences between material and non-material innovation and economy, to assess emerging economic models that are demand driven and appear more competitive, and to design legislation that takes these changes into account to regulate without distorting the market or hampering evolution.

70 *Proposal to establish a Development Agenda for WIPO: an elaboration of issues raised in document WO/GA/31/11*, Group of Friends of Development, 5 April 2005, submitted for the first session of the Inter-Sessional Intergovernmental Meeting on a Development Agenda for WIPO, Geneva, April 11 to 13, 2005. http://www.wipo.int/edocs/mdocs/mdocs/en/iim_1/iim_1_4.pdf .