Analysis of software patentability in Europe

François PELLEGRINI

Associate professor in computer science,

ENSEIRB & LaBRI, Université Bordeaux I

351, cours de la Libération
33405 Talence, France
pelegrin@enseirb.fr

Introduction

The question of software patentability has been in recent years, and is still at the time being, the subject of much controversy within the institutions of the European Community. The extension of the patent system towards the immaterial realm, which ultimately leads to the granting and, most importantly, the enforcement of patents on mathematical, educational and business methods, has been a constant trend in the last thirty years, driven by some prominent users of the patent system as well as by a large number of patent offices themselves. In Europe, unlike in other areas such as the United States, software patentability is effectively prevented by law, namely by Article 52.2 of the European Patent Convention[1] (EPC), which draws a list of subject matters excluded from patentability, such that "*shall not be regarded as inventions* [and consequently should not be eligible for patent protection] [...] *(a) discoveries, scientific theories and mathematical methods; [...] (c) schemes, rules and methods for performing mental acts, playing games or doing business, and programs for computers; [...]*". In order to restrict the benefits of this exclusion to software and yet still allow for the patenting of devices that would be eligible to patent protection but comprise some software-operated components, Article 52.3 states that the provisions of Article 52.2 "*shall exclude patentability of the subject-matter or activities referred to in that provision only to the extent to which a European patent application or European patent relates to such subject-matter or activities as such*".

Much effort has been put in the last two decades by the European Patent Office (EPO), and more specifically by the chambers of its Technical Board of Appeal (TBA), to provide successive interpretations of the above sentences that made possible for them to accept claims on intellectual methods in an ever broader context. Yet, as the EPC still formally forbids patents on software and business methods, all of the software patents granted to date by the EPO[2] can still be challenged in court, with a high probability of being invalidated, depending on the inclination of the judges to stick to the letter and spirit of the EPC rather than to follow TBA case law. Therefore, proponents of software

1 http://www.european-patent-office.org/legal/epc/e/ar52.html .

2The FFII, a NGO opposed to software patentability, estimated that there might be about 30,000 such European software patents, basing on a statistical analysis of the EPO patent database. See: http://gauss.ffii.org/Statistics/State .

patentability concurrently explored ways to overturn the EPC. In November 2000, a diplomatic conference was convened to have signatory States of the EPC vote for some modifications to this latter. Among the items to be discussed was the deletion of the "programs for computers" term from the exceptions listed in Article 52.2[3], proposed by the EPO administrative council, but a majority of States finally decided not to vote for it. In February 2002, the General Directorate for the Internal Market of the European Commission released a draft Directive "*on the patentability of computer-implemented inventions*"[4] which, after heated debates between the European Commission, the Council of the European Union, and the European Parliament, leaded to a rejection of the directive by the European Parliament, on second reading, on July 6th, 2005[5].

The purpose of this paper is to address, in the European context, the subject of software patentability from legal, technical, systemic and economic perspectives, to discuss the inconsistencies in patent and copyright laws that result from the extension of the patent system towards software, and to propose a clarification of the frontier between the patentable and the unpatentable in the immaterial realm.

Legal analysis of software patentability

Why the question of software patents is badly put

One of the main arguments put forward by the proponents of software patentability was the need to protect the software components which are increasingly present within innovative tangible inventions, and for which, according to them, copyright does not provide sufficient protection. An example that is often considered by members of patent offices is the one of a new washing machine that washes the linen more efficiently, and such that the innovating washing sequence is implemented by means of a computer. The software carried out by this computer would be, within this framework, at the origin of the technical contribution brought by this new machine, and should thus be covered by patent claims.

This test case was used as a conceptual basis to allow claims on software features in patent applications, by arguing that since the subject matter of the patent did not relate to software "as such", the embedded software could receive patent protection. However, this first drift was not sufficient in itself to allow the patenting of software running on a generic purpose computers, because in European patent law, for a claimed invention to be patentable, the innovative part of an invention must have "technical character"[6]. Therefore, in other decisions, the TBA defined rules under which some software could be considered

3 http://www.european-patent-office.org/epo/dipl_conf/pdf/em00002.pdf .

4 http://eur-lex.europa.eu/LexUriServ/site/en/com/2002/com2002_0092en01.pdf .

5 http://www.europarl.europa.eu/sides/getDoc.do?objRefId=98041&language=EN .

6 DBA case T 0641/00, http://legal.european-patent-office.org/dg3/biblio/t000641ex1.htm : "*An invention consisting of a mixture of technical and non-technical features and having technical character as a whole is to be assessed with respect to the requirement of inventive step by taking account of all those features which contribute to said technical character whereas features making no such contribution cannot support the presence of inventive step*".

“technical”, as opposed to software “as such” which cannot be patented because it does not produce any “technical contribution”. It is also for this purpose that the EPO forged the term “Computer Implemented Invention”, to be able to refer to software as a patentable subject matter without having to use the word “software”, but on the opposite to insist on the term “invention”, which conveys an idea of statutory patentability. This term appeared for the first time in Appendix 6 of a report entitled “*Examination of "business method" applications*”, in which the EPO instructed the other members of the Trilateral Cooperation (the US Patent and Trademark Office and the Japan Patent Office) on how it was granting patents on business and software methods provided that they were described in a “technical” way[7].

However, postulating the possible existence of a discrimination between “technical” and “non-technical” software based on the examination of their effects, which we call the “doctrine of software technicality”, cannot hold vis-a-vis a rigorous analysis. In the above-mentioned case of the washing machine, the technical effect is the improvement of the washing of linen by the machine. This effect is obtained by application of an innovating washing process, consisting in the dispensation of detergent at such or such moment of the washing cycle, by the realization of better mixing of water and of the linen, etc., all operations carried out by the machine in the material world. The software operating the machine does control this innovating washing process, but is not, in itself, the source of it. It is the process which is innovative and produces the technical effect wished for, not the software.

Any software, carried out by a computer, does nothing but handling symbolic quantities according to a pre-established program, independently of the way in which these symbolic quantities are materialized within the computer which carries out the program (electrons, photons, magnetic moments, quantum states, etc). The computer, in order to interact with the physical world, needs peripherals, which materialize the symbolic systems quantities into physical actions in the outside world: activation of a servo unit, displaying of a particular luminous information on a screen, etc. The software which drives the washing machine can thus be carried out on a simulator, and run exactly as it would do within the washing machine, but without producing the expected technical effect. It is because the technical effect is independent from the considered software, but resides in the effective carrying on of the process.

The legislator had already analyzed this situation to draw the conclusions which it imposes, and to authorize the claiming of industrial processes. The manufacturer of innovative washing machines can thus claim his process of innovative washing as well as the machine which carries it out without needing software patents at all.

Let us notice to conclude this section that software which would sum the values of two quantities of reagents that are present in the tanks of a chemical reactor could be regarded as “technical” according to the doctrine of software technicality, while it would not if it were used to compute the sum of the balances of two bank accounts, while both software perform exactly the same process (however, both

7 http://www.trilateral.net/projects/other_project/business_method/appendix6.pdf . As the EPO writes: “*The expression "computer-implemented inventions" is intended to cover claims which specify computers, computer networks or other conventional programmable digital apparatus whereby prima facie the novel features of the claimed invention are realised by means of a new program or programs*”.

are patentable according to current EPO practice of “technical considerations” in the implementation).

Similarly, one cannot consider that some software produces a technical effect when it allows one to “perform computations more efficiently” or it “improves the efficiency of a computer”. Let us remark at first that this doctrine is even more radical than the one of software technicality, because it would allow one to consider as “technical” any software that provides better processing performance than some existing software, without any reference to any given problem, or rather by considering that any use of a computer is in itself a technical application.

The actual efficiency of a given algorithm being completely dependent on the architecture of the computer onto which it has been implemented and is running, the allegedly technical features of an algorithm are not linked to it, but rather are based on the adequacy between the underlying hardware and implementation considerations that belong to the realm of copyright, since they are specific to each individual implementation of software on a given architecture. The consideration of increased efficiency is therefore not effective to define a stable frontier between allegedly “technical” and “non-technical” software.

The impossible formalism of software technicality

The conceptual impossibility to discriminate between “technical” and “non-technical” software condemns in advance to failure any attempt to embody this discrimination into legislative texts. An obvious example of this failure is the text proposed by the Irish presidency of the European Union[8] after the European Parliament voted in first reading a text explicitly rejecting software patentability[9]. The “Irish” text, which was presented by its proponents as preventing the patentability of “pure” software (Recital 7b), stated however that software alone could carry out a “technical contribution” (Recital 13 and Article 2b), and finally allowed the patenting of such software (Article 5b). These articles would have allowed allow one to patent any type of software, because those are always created to solve a given problem.

Any doctrine that enables the granting of any single software patent, by considering either that some software is intrinsically “technical” or that the use of “technical means” to implement some otherwise non patentable algorithm constitutes a “technical invention”, opens to patentability the whole field of intellectual methods, including business methods as well as mathematical methods, because there is no substantial difference between all of them. Business and educational methods are basically algorithms, and all algorithms equate to mathematical proofs, according to the Curry-Howard isomorphism[10].

---

8 http://register.consilium.europa.eu/pdf/en/04/st09/st09713.en04.pdf . As its Article 2b stated, the “technical contribution” test, a mix of the traditional subject-matter and novelty tests, would have been performed on the patent claim as a whole, such that an innovative software running on a non-innovative “technical” hardware could have been considered both innovative and technical, and thus could have been eligible to patent protection.

9 http://www.europarl.europa.eu/pv2/pv2?PRG=DOCPV&APP=PV2&DATE=240903&DATEF=030924&TPV=PROV&TYPEF=TITRE&POS=1&SDOCTA=2&TXTLST=2&Type_Doc=ANNEX&PrgPrev=PLAGE@1&LANGUE=EN .

10 http://en.wikipedia.org/wiki/Curry-Howard_isomorphism .

The choice of the legislator thus cannot be a half-measure: *either he fully endorses the enablement of a regime of total patentability of software, business, educational and mathermatical methods, or he entirely rejects it*, as did the European Parliament when amending the draft directive during the first reading, so as to reject claims on software while still allowing the patenting of material inventions and of industrial processes that make use of software for their control, which was the usual practice before the EPO drift since it is the very meaning of Articles 52.2 and 52.3 of the EPC.

This binary choice, of considerable economic and strategic consequences, cannot be done lightly, all the more than the extension of patentability would be practically irreversible.

Are software patents legally mandatory?

Some players referred to Article 27 of the TRIPS Treaty, which requires the existence of patents for any field of technology, to justify the legal need for software patents. As underlined by Mr. Paul Hartnack, General Comptroller of the United Kingdom Patent Office, this interpretation is not the only possible one[11].

One can consider, like the Commission and the Council do in their proposals for a directive, that software is intrinsically technical, and that Article 27 of TRIPS requires to have software patents. Else, one can also consider, like the European Parliament did, that software does not reside within a field of technology but within the one of intellectual works, and thus beyond the range of Article 27 of TRIPS. Here again, policy makers are free to decide, because the choice between the two doctrines is still possible.

Software Patents and copyright are exclusive

Until now, the copyright and patent regimes had coexisted without conflict, their purposes being different, since copyright dealt with the protection of individual works of authors, while the purpose of patents was to protect inventors of innovative processes implemented within (non-unique) apparatuses designed to solve a given problem.

The extension of the patent system to software constitutes the first major blow to copyright law, into which the legislator had, after deep thinking, incorporated software. This assimilation was altogether natural because software, just like books, consists in the production of a textual original work (the source code of the program), resulting from the combination of elementary ideas. For an adventure novel, these elementary ideas can be: “love scene on a balcony”, “twins being mistaken one for another”, etc. For software, it can be: “alphabetical sorting in a list”, “displaying a progress bar instructing the user to be patient”, etc.

Unlike copyright, which protects final works, software patents, which protect against the imitation of features, allow one to monopolize these elementary ideas, and thus prevent anybody from realizing a

11 http://www.patent.gov.uk/softpat/en/1000.html .

program implementing a patented idea. This would amount, in the case of books, to be allowed to claim elementary ideas such as the balcony love scene, although the same idea can lead to very different implementations, such as the balcony love scenes of *Romeo and Juliet* and of *Cyrano de Bergerac*, which are not plagiarisms one of the other.

Software patents, by allowing their holders to claim elemenary ideas, thus constitute an extremely powerful monopoly-building tool, because the holder of a single patent can prevent the selling of all software implementing this idea, whatever their application domains can be.

In this extent, software patentability can be regarded as violating the TRIPS and WIPO copyright treaties. Indeed, Article 10 of TRIPS[12] stipulates that software belongs to the realm of copyright, and Article 13 of TRIPS, as well as Article 10 of WIPO[13], that one cannot systematically prejudice the legitimate interests of the rightholders. However, considering software patentable indeed prejudices the rightholders, since the author of an original software can be prevented from marketing it for the reason that this software infringes on one or several software patents, irrespective of legitimate exceptions such as harm to public morals. As more and more artistic creations make use of software (such as video games, for instance), the occasions for litigation can only increase, at the detriment of creation in the considered industries.

Although one can argue that these articles in TRIPS do apply only to the exceptions which the legislator could introduce in the copyright regime only, it is clear, considering the spirit of these treaties, that *the forging of software patentability, which has no legal necessity, tends to have a destructuring, rather than structuring, effect on existing intellectual property regimes that exist to date*, and no one knows what could be the long-term consequences of the extension of the patent regime to fields for which it had indeed not be designed for, neither for which it has economic justification.

The decision to introduce software patents amounts to answering implicitly the question of knowing which system, between the patent or the copyright regimes, must have preeminence, without any possibility of debate nor to check the validity of this approach.

The copyright regime is oriented towards the protection of existing works, already accessible to the public since already created, the existence of the protection making it possible to regulate by subsequent contracts the way the public can access these works. The software patent regime allows one to claim an idea, irrespective if there exist or not concrete achievements implementing these ideas and being able to benefit to the public. In the case of software, *privileging the patent regime compared to the copyright regime thus amounts to privileging the defensive and anti-competitive behaviors which are inherent to the patent system, based on exclusion and rent seeking, as opposed to the active behavior of the creator of original works putting them at the disposition of the public in a context of free competition.*

12 http://www.wto.org/english/tratop_e/trips_e/t_agm3_e.htm .

13 http://www.wipo.int/treaties/en/ip/wct/trtdocs_wo033.html#P83_10885l .

Systemic analysis of software patentability

The legal analysis of software patentability presented above evidenced that there exists no legal obligation by international treaties, such as TRIPS, to create software patents. Even quite the opposite, the existence of software patents leads to inconsistencies in the legal system and legal insecurity for software rightholders. The decision to have software patents or not within the European Union can thus be freely decided on the basis of economic and strategic criteria only.

Another consequence of this analysis is that, contrary to what the writers of the draft directive have claimed, the operations of software patentability in Europe will indeed be similar to the ones in the United States, such that the current situation in the United States reflects exactly what the situation in Europe would be if a legislation close to the one proposed by the Irish presidency had been accepted.

Let us also notice that, with respect to this analysis the conceptual bases of which are simple and accessible to any person who has some knowledge of the domain, one can legitimately wonder about the the apparent ingenuity of the writers of the directive who pretend that there exists a discrimination between “technical” and “non-technical” software. If one gives them a minimum of credit, one will arrive at the conclusion that they are fully aware of this impossibility, which leads to wonder about their motivation to legitimate software patentability even at the price of a distortion of the truth.

If simple ideological motivation cannot be neglected, it is not sufficient to account for all of the observed behaviors. One has therefore to acknowledge the existence of a collusion of economic interests between the leaders of the patent offices, the members of the various commissions of experts involved in the drafting of the directives, and the people in charge of the legal departments of some big companies, with the deliberate purpose of providing these latter the most efficient anti-competitive weapons as possible, considering that the legal services of these companies are the biggest customers of the patent offices, which themselves live on the fees of granted patents, and considering that there exists an important professional porosity between these three communities.

It is quite surprising for instance to have EPO people specifically in charge of relations with the European Parliament and other European institutions[14], acting as lobbyists such that Members of the European Parliament vote for directives that increase the power of the EPO and limit democratic control on patent and therefore innovation policies. In almost all countries, patent offices are under the political control of industry or justice ministries, because innovation policies are decided at the political level. Having the EPO using parts of its resources towards “promoting the European patent system in Europe and increasing the role of patents in the internal market” poses a threat to the separations of powers, as an agency pertaining to the executive power is trying to get legislative power through the case law issued by its own boards of appeal and through lobbying towards. Attempts to get judicial power through special courts such as the one proposed in the context of the European Patent Litigation Agreement[15] are also a concern.

---

14 http://www.european-patent-office.org/news/pressrel/2003_08_07_e.htm .

15 http://patlaw-reform.european-patent-office.org/epla/ .

This pledges for a deeper structural reform of the patent system in Europe, to ensure that innovation policies remain under elected political control, and that patent offices are accountable for wrongfully issued patents. Disconnecting the income of patent offices from the number of patents they grant may also help to weaken parts of this vicious circle.

Claims on intellectual methods, interoperability, and monopolies

The analysis of the version of the directive endorsed by the Irish presidency shows that most of this text was designed to favor the creation and enforcement of monopolies on whole sectors of the data-processing industry, without any counterpart for the consumer. Article 5b of this text introduced program claims, which would have make it possible to sue any software creator, vendor, or user, provided that a patent has been granted on any of the elementary ideas that the software implements. This would have allowed big companies with large legal departments to scare potential users of competing software, well before resorting to courts, thus distorting competition in an insidious way.

This text also rejected any right to interoperability. Article 6 does nothing but maintaining the exceptions of retro-engineering and decompilation allowed by European directive 91/250 CE regarding the protection of software by copyright[16]. But whereas these exceptions are sufficient in the context of copyright, they are worthless in the context of patents, since any manufacturer who reimplements, for the sake of interoperability, processing methods for a patented file format, would be infringing of this patent.

The mode of creation of software, based on cumulative innovation, as well as the very short development cycles of this industry, do not allow any longer to account for the existence of a patent system which is expensive, heavy and slow. Indeed, unlike in the material world, the time necessary for a competitor to imitate some software is hardly any less than time taken by the development of the original software, because the development itself constitutes the most expensive activity in comparison to the simple fact of having the idea of the software, which yields that, in this very reactive market, initial innovators have time to profit from their products before any competitor appears. Moreover, any such competitor can gain market shares only if its product is of better quality, brings additional innovations, or is less expensive than the original product, which can only be, as development costs are almost identical by what precedes, if the market price of the original product were too high. A proof of this is that, on any given market segment, a product of good quality and at a reasonable price has little competitors.

Therefore, without software patents to build monopolies, competition, innovation, and consumers win. These consumers are both private individuals and companies of all sizes and all fields of activity, because the impact of information technologies on productivity affects every sector of the economy.

To conclude, let us note that patents do not bring any information which is not already public, since all features, whether patented or not, of any software, naturally appear when using the software, thus

16 Articles 5 and 6 of: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31991L0250:EN:HTML .

allowing any "person skilled in the art" to understand its principles of operation. Let us also remark that the jargon used to write these patents is in general incomprehensible to any computer professional, and of a technical level which is so low that no information is usable in practice, since the goal of the patent filer is to be as generic as possible in order to block as many competitors as possible. Therefore, the benefit of publication that is generally associated with patents does not make much sense in the context of patents on intellectual methods.

The economic threat of the software patents to the European data-processing industry

According to a report by the French *Commissariat au Plan* issued in 2002, the French software sector is essentially made of small structures. Indeed, 58 % of the 28,500 companies of the sector have at most two employees. These figures are almost equivalent in Germany and in the other countries of the Union. Small and medium-sized enterprises of the software sector represent more than 70 % of jobs and wealth produced by this sector in Europe, not considering the software departments of large industrial and service companies (such as banks, insurance companies, etc) which contribute to a great extent to the development of "in house" software which will never be marketed but could nevertheless be subject to software patent infringement lawsuits.

Small companies are on average more innovative, more reactive, and more inclined to base their development on the creation of new products than on an established system of monopolistic revenues. In fact, *since SMEs of the software sector represent its economic majority, as well as a pool of creativity that is necessary to the development of new technologies, all new regulations should be developed for their benefit*.

Regarding the introduction of software patentability in Europe, this same report of the *Commissariat Général au Plan* sees in the "*abrupt decision to extend patentability to software* [...] *real dangers for the European industry, because of the considerable unbalance between the United States and Europe on the matter*", and insists that "*Only the armed peace prevailing to date, because of the legal uncertainty that remains around the notion of software patent, indeed explains that existing patents* [the about 30,000 doubtful software patents already granted by the EPO] *are not used more often*.".

A system of incentives for innovation cannot be effective unless it encourages the production of final goods that benefit to citizens and thus to the society at large. On the opposite, the monopolization of upstream concepts disadvantages the production of such goods, because it is then more profitable for an economic actor which has patented a concept to seek rents from the entities taking the commercial risk to carry out the final products, rather than to make such products by itself and be under the risk of patent infringement lawsuits from its likes. Consequently, the more the extent of the patent is broad and lasting, the more behaviors of predation and risk avoidance are favored, to the detriment of innovation. In the case of software, the minimum duration of 20 years, which would be imposed by TRIPS if one would consider software as patentable, represents more that 10 times the average duration of the

commercial life of any software, which is an economic nonsense, as this blocks further developments by other innovators who must, in order to enter the market, pay fees for basic ideas the development cost of which, if any, has been paid back many times already.

Patent inflation in the immaterial realm, in the US as well as in the EU, leads to a change of scale, where value does no longer reside in individual patents, but rather on patent portfolios, that is, strategic collections of patents covering the same problem space. Consequently, at constant R&D expenditure rate, patent filing intensity increases, as well as the pressure on patent offices, patent thickets develop, and patent lawsuits become more complex and expensive because of possible counter-claims[17]. At best do large players agree on patent portfolio exchanges, which allow them to pretend they live in a patentless world, but this has several perverse effects. First, what economic interest is there for these companies, and therefore for their customers and by extension for the society at large, to finance the filing of patent being used to counter the existence of other patents, whereas the absence of patents would amount to exactly the same? Then, after such exchanges, the only entities having to support the cost of the licenses are the innovative small and medium-sized enterprises, which cannot enter the markets where they could reimburse their investments. The system thus functions exactly the opposite as it should, wrongly penalizing the SMEs.

Software patents are moreover a danger for investors. Without patents, any investor is certain to be able to market the software he financed, and thus to have a return on investment if these products fulfill a need. With software patents, any economically worthwhile software may be attacked, and the return on investment is more dubious. The very recent exhaustive study carried out in the United States on more than 130.000 software patents and 1600 US companies between 1976 and to date showed that the expenditure in software patents replaced more than 10 % of the expenditure of R&D of the companies, at the national level[18]. During Federal Trade Commission hearings on the subject, some company CEOs reported that they were going to devote up to 35 % of their R&D expenditure[19] to the building of software patent portfolios, in order to try to protect themselves from infringement lawsuits, although these costly portfolios could not be of any help to protect them from the very high legal expenses that a competitor would like them to spend, at the detriment of their middle and long-term competitiveness.

In this context, it is legitimate to think that, because the market allowing the best return on investment will be a Europe without software patents, the companies that will succeed better will be European companies, that will be more close to their customers, and more able to provide the necessary R&D effort. Any company basing its software development in the United States can be subject to infringement lawsuits on the basis of US software patents, since software developers may implement, knowingly or not, algorithms that are patented in the United States. On the opposite, European developers would be free to develop their software without constraints. Thus, an Europe without software patents is likely to succeed in attracting US companies willing to secure their developments,

17 *Patent Portfolios.* R. P. Wagner and G. Parchomovsky. Research Paper 05-25, Nov. 2005, University of Pennsylvannia Law Review, Vol. 154, No. 1, http://ssrn.com/abstract_id=874445 .

18 http://www.researchoninnovation.org/patent.pdf .

19 The whole report of these FTC hearings is available at: http://www.ftc.gov/os/2003/10/innovationrpt.pdf . Regarding the above comments, see in particular pages 153 to 165 of this report.

reversing the direction of the flow of emigration of computer professionals.

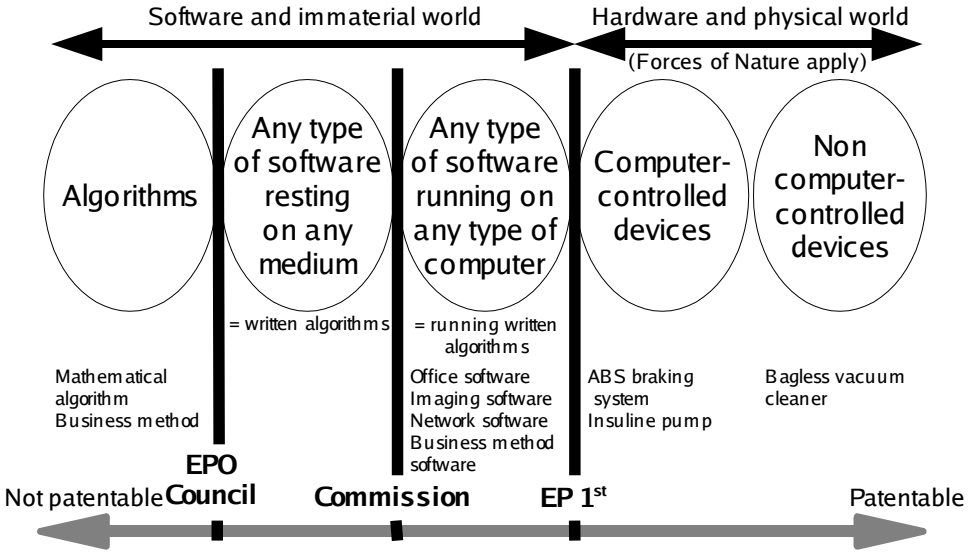
Let us notice by the way that nothing prevents a European company from filing patents in the United States to try to improve its chances to penetrate this market. This investment is small in comparison of the one to obtain European patents, and without common measure with the tribute which European SMEs should pay to large US companies if software patents were legalized in Europe.

How to restore some consistency in the patent doctrine

Although the EPO claims it does not grant patents on computer programs nor on source code, and literally does not, the effect is the same as the current EPO doctrine allows a patent holder to sue anybody running on a computer, called "apparatus" in the patent jargon, a computer program implementing some claimed "method", that is, a given algorithm. Therefore, with respect to software patentability and from a monopoly-building point of view, algorithms are the objects at stake.

Where to set the limit?

The figure below shows the different objects that patent and copyright laws have to deal with, with respect to software and intellectual processes.



The three vertical bold bars show the limits to patentability that were set-up by the three different texts produced by the EU bodies during the legislative process, namely the original draft directive proposed by the European Commission, the amended version voted by the European Parliament at the end of its first reading, and the text issued by the Irish presidency of the Council. In fact, the Commission and Council versions are similar in effect, because both allow one to monopolize features of a computer

program running on a general purpose computer, therefore enabling total software patentability.

The arguments developed in this paper show that software patentability poses many legal and economic problems, as well as ethical problems since code is also speech. The above discussion strongly suggests that software patentability should not be allowed, and on the contrary that clear exclusion rules must be drafted, that will effectively prevent the granting of such patents and, in case patent offices act of their own like for the EPO, make it possible to challenge these patents in court in an as unambiguous and cheap way as possible.

What should be considered “technical”?

The EPO justified its diverging practice with respect to the EPC by two kinds of rhetorical arguments. The first one concerns the “*as such*” term discussed above, while the second one regards “technicality” as a requirement for patentability. In a series of drifts on patent granting rules[20] [21] [22], the EPO TBA built a doctrine that allows a patent to be granted provided that “technical considerations” are taken into account in the implementation. This represents a complete twist of patentability rules, of tremendous consequences: patentability criteria are not evaluated on the patentable object itself, but on its immediate environment. As the EPO TBA notes in section 4.6 of decision T 0258/03 : “*The Board is aware that its comparatively broad interpretation of the term "invention" in Article 52(1) EPC will include activities which are so familiar that their technical character tends to be overlooked, such as the act of writing using pen and paper. Needless to say, however, this does not imply that all methods involving the use of technical means are patentable. They still have to be new, represent a non-obvious technical solution to a technical problem, and be susceptible of industrial application*”[23]. However, as EPO states in its trilateral document on the “*Examination of "business method" applications*”: “*according to Sohei* [decision T 0769/92]*, the computer implementation of a, for example, business method, can involve "technical considerations", and therefore be considered the solution of a technical problem, if implementation features are claimed*”, therefore making any computer implementation of any intellectual method patentable. As written in DBA T 0931/95, which enabled business method patents: “*An apparatus constituting a physical entity or concrete product, suitable for performing or supporting an economic activity is an invention within the meaning of Article 52(1) EPC*”.

In order to prevent the granting of patents on intellectual methods, while still allowing the patenting of computer-controlled innovations such as better washing machines or a better-braking ABS systems, one has to define clearly “technicality” in the sense of patent law, as well as the way it should be evaluated, that is, provide a clear delimitation of patentable subject matter. Technicality has nothing to do with the problems being considered, because patenting any solution to the problem of curing depression on the assumption it is a “technical problem”, could lead to the patenting of playing music as a possible cure

20 DBA case T 0931/95, http://legal.european-patent-office.org/dg3/biblio/t950931ep1.htm .

21 DBA case T 1173/97, http://legal.european-patent-office.org/dg3/biblio/t971173ep1.htm .

22 DBA case T 0769/92, http://legal.european-patent-office.org/dg3/biblio/t920769ep1.htm.

23 DBA case T 0258/03, http://legal.european-patent-office.org/dg3/pdf/t030258ex1.pdf .

as well as medicines for which patent protection can be desirable. Therefore, the only viable criterion for technicality is the technicality of the solution being proposed. Consequently, to prevent the patenting of solutions the novel part of which belongs to the immaterial realm, one has to implement criteria that are equivalent to the ones voted by the European Parliament during the first reading of the software patent directive, which are similar to the ones of the classical "core theory": in order to be considered "technical", a solution must provide "a new teaching on the use of controllable forces of nature". According to this definition, new washing machines or new ABS systems could be patentable, even if the only change in the devices is embedded in the software, because the solution they provide takes place in the physical world, based on a novel physical process. However, the embedded software that controls these new processes cannot be protected by patents because these software are, "as such", excluded from patentability, but instead can be protected by copyright. This does not hinder at all the efficiency of the patent, as an infringer would have not only to copy the software, but also the hardware, in order to create an infringing product. On the other hand, this criterion prevents the granting of patents on software and business methods, as the teaching carried by such patents would not concern forces of nature but organisational rules.

During the examination process, every patentability criterion, especially novelty and technicality, have to be evaluated separately in the examination process. The novel features have to be evaluated according to a "paper and pencil" test: if the innovative part of a claim could be performed intellectually by a human being, using the only resources of his mind (although at a different speed, but speed does not change the nature of the process in itself), then it is an intellectual claim, which must not be accepted.

Conclusion

Patent law, as a means of implementing innovation policies, has to remain under democratic control, and the extension of patentability to new areas should be carefully examined with respect not only to the expected benefits for the users of the patent system, but also for society at large. The extension of patentability to the immaterial realm poses many problems in terms of consistency of the legal system, but also provides no economical benefit to the majority of its potential users, that is, the SMEs, which contribute to about two thirds of the jobs and wealth produced in the software sector. It is therefore up to the European policymakers to take actions in order to prevent the granting and enforcement of such patents in the European Union, to regain control over the EPO, and to convince our international partners that they would also benefit from a ban on such patents.