Journal of Information, Law and Technology

Much Pain for Little Gain?
A Critical View of Software Patents

Christian Koboldt
DotEcon Ltd*
christian.koboldt@dotecon.com
<mailto:christian.koboldt@dotecon.com>.

I would like to thank Réka Horváth, Dan Maldoom, Richard Marsden and an anonymous referee for helpful comments, but point out that all errors are mine.

This is a **refereed** article published on: 4 July 2003.

Citation: Koboldt, C,'Much Pain for Little Gain: A Critical View of Softwre Patents', 2003 (1) *The Journal of Information, Law and Technology (JILT)*.
<http://elj.warwick.ac.uk/jilt/03-1/koboldt.html>

Abstract

The question whether access to patent protection for computer software should be made easier (for example by removing the restrictions that would allow a classification of computer programs 'as such' as inventions) would be in the overall benefit has exercised policy makers for quite some time. 'Better protection' of software-related innovations (compared to copyright protection) as well as 'better disclosure' of the underlying ideas and principles have been cited as the main benefits.

This paper takes a critical view of these arguments, taking into account that in many cases the underlying ideas and principles may be most effectively be protected as trade secrets (in combination with copyright protection of the 'expression', i.e. the computer program as it is made available to the user). Giving software producers the option to apply for patent protection may not make much difference in terms of the information generated for the benefit of other innovators. Patent protection may be most attractive for ideas and principles that are to a large extent obvious or become apparent to the user. This might lead to a raft of patents for rather obvious 'inventions' (even if patent office searches were improved and patent applications were assessed more rigorously), which might cause little benefit but much friction in the process of innovation.

Keywords: Economic analysis of intellectual property rights, patents, copyrights, computer software, innovation

1. Introduction

The European Commission's proposal for a Directive on the patentability of computer related inventions brought a period of intense consultation and lobbying to a preliminary close. In the eyes of many it ended with a whimper rather than a bang. Despite having initially identified a considerable need for action, the Commission finally was of the view that 'the Directive should harmonise protection for computer-implemented inventions while avoiding any sudden change in the legal position, and in particular any extension of patentability to computer programs 'as such'.

In essence, the proposed Directive's main effect would be to require harmonisation of national patent laws on one particular issue (and little else), namely that 'it is a condition of involving an inventive step that a computer-implemented invention must make a technical contribution' (Article 4(2)). In practice this means that in order to obtain a software patent, some clever drafting of the application is required - but this appears not to have in any way prevented software producers from obtaining patents in the past, nor is it likely to do so in the future. Indeed, a more radical step of aiming to remove the restrictions arising from Art 52(2) of the EPC, which stipulates that computer programs 'as such' are not inventions in their own right and are therefore not patentable, might have made drafting of patent applications slightly simpler, but changed little else in practice.

Against this background, it is interesting to note that one of the studies prepared for the Commission concluded that 'any move to strengthen IP protection in the software industry cannot claim to rest on solid economic evidence'. (Intellectual Property Institute, p36). Given that software patents are with us (and that even removing the 'as such'

restriction might have made only little, if any, difference), does this suggest that we have gone to far, made it too easy to obtain patent protection for computer software (and in other jurisdictions even 'mere' business methods)?
In order to answer this question, one has to look back at the economic arguments for and against intellectual property rights in their various guises, and in particular:

1 the trade-off between providing incentives for innovation and creativity on the one hand, and the inevitable restrictions imposed on the usage of the products of innovation and creativity on the other hand; and

2 the relationship between the scope of protection and the disclosure obligations associated with the intellectual property right.

Adopting a sometimes Panglossian attitude, economists have argued that the various forms of intellectual property protection - trade secrets, copyrights and patents - have developed in such a way as to guarantee a welfare-maximising outcome. In very simplistic terms:

3 Trade secrets are just that - innovations that are kept secret and where the law does not afford any specific protection other than that competitors trying to find out the secret must not engage in illegal behaviour. Trade secrets are appropriate for innovations that can be exploited without being disclosed in the process - things that can be done behind closed doors in a workshop without anyone else being able to find out in sufficient detail about the innovation to be able to replicate it. Glass-making in 13th century Venice is an example - although the death penalty for glass makers leaving the Republic may appear to be a rather extreme way of protecting a trade secret. -

4 Copyright protects a particular expression rather than underlying idea. What is prohibited is copying, not re-creating another, perhaps even similar, expression based of the same idea. Copyright is appropriate where effort goes mainly into expressing an idea rather than the idea itself. Literary works (in the most literary meaning) are perhaps the best example of creations that are suited to copyright protection. The object of copyright protection being the expression rather than the idea, there is of course full disclosure out of necessity - there would be little point in writing poetry that nobody is allowed to read.

5 Patents, at the other extreme, protect ideas and concepts. Even if someone other than the patent holder came up with a sufficiently similar invention completely independently, using this invention would infringe the patent granted to the first inventor (which can lead to so-called patent

races). Given the considerable scope of protection, a patent will only be granted if the invention meets certain conditions), is limited in duration and requires the disclosure of the invention which is thought to facilitate further innovation (although in practice the importance of patents as a source of information about the state of technology may be rather limited).

The legal framework - in particular patent and copyright law - may indeed have evolved over time to deal in the most appropriate manner with different forms of human creativity. It should, therefore, not be surprising that trying to shoehorn new types of innovation that do not necessarily fit existing categories into the system of intellectual property rights must inevitably create tensions and perhaps even bring the system to its breaking point. Computer software is arguably a case in point.

2. Protection of Computer Software

Copyright protection of computer software deals with one of the major issues facing software producers, namely piracy (assuming, of course, that enforcement works sufficiently well). After all, it is the use of unlicensed copies that is often portrayed as the main threat to software producers. It is the ease with which software can be copied that potentially undermines the software producer's ability to recover the cost of the considerable amount of effort going into creating and - with unfortunately more than a few notable exceptions - thoroughly testing a piece of software.

However, because copyright protects the specific expression rather than the underlying idea it appears to be unsuitable for areas where the value is in the idea rather the expression. Whilst, for example, a considerable proportion of the books that one reads over time are variations of a handful of different themes - different expressions of similar basic storylines - presumably only limited enjoyment can be had from perusing a number of different versions of spreadsheet software. People are unlikely to match their collections of music with a collection of e-mail clients, and so on.

Where variety of expression is valuable in areas such as literature, music or film, the source of value in the case of software is different - it is finding the most efficient algorithm dealing with a particular problem, and coding it up in a way that works. Poetry in Perl competitions notwithstanding, users are unlikely to care particularly about the inherent beauty of the source code or the specific sequence of zeros and ones that constitute the object code. What they care about is that the software does what it is supposed to do quickly, efficiently and reliably, and that it is easy to use.

As copyright protection covers the expression, but not the idea, it would not protect the developer of a clever piece of software from others writing software that performed exactly the same function. However, algorithms, principles and ideas underlying a specific piece of software may not be particularly obvious from inspecting the binary files

that will have to be made available to the user, or analysing the way in which the software works. Thus, without disclosing the source code or providing a detailed description of the underlying ideas and principles, these are often effectively protected as trade secrets.

Moreover, network effects may limit the incentives of others to develop similar products, even if they include slight improvements, once there is an established base. Whilst there may be a huge marketing opportunity for a better mousetrap, experience suggests that this does not necessarily hold for a better computer operating system.

It is not copyright protection of software, but rather the fact that the underlying algorithms essentially remain 'trade secrets' that could potentially stifle innovation, particularly incremental innovation. Moreover, it is often not disclosure of the methods and algorithms themselves that matters most, but rather disclosure of the interface information that allows other developers to achieve interoperability. Interoperability can be expected to promote competition between software producers in the same way as interconnection does in telecommunications.

It is by comparison with the trade secret protection of copyright that the patent system might be seen as an improvement. Do software patents lead to better disclosure of information that would overall improve social welfare? Would incremental innovation be made easier if more software patents were granted and therefore more information had to be disclosed? Is there a case for providing incentives for a more widespread use of patents to protect software innovations? Is there a case for software patents at all?

3. Is there a case for software patents?

Against the background described above the case for software patents must be based on a presumption that:

a) patents afford more effective protection than would be available through a combination of copyright (preventing unauthorised copying) and non-disclosure of the underlying ideas (the specific algorithms and their implementation in the source code), thereby stimulating investment in software innovations to the benefit of the economy; and/or
b) patent protection would encourage disclosure of the underlying ideas and thereby stimulate or facilitate further (incremental) innovation.

The decision to apply for a patent, thereby having to disclose the underlying ideas (in line with the requirements of the patent system), is made by comparison to the alternative of non-disclosure and reliance on trade secret protection. Therefore, it is far from clear that making access to patent protection easier would lead to more openness and transparency (albeit at the cost of making use of these underlying principles and ideas for further innovation more difficult). Accepting the obligation to disclose underlying ideas and principle would most likely be the preferred option where trade secret protection were insufficient, i.e. where the underlying ideas and algorithms could easily established through using a piece of software.

In particular, if patent protection were to be combined with requirements to disclose, and license, information that would allow other programmers to release interoperable software or exploit the information disclosed through incremental innovation, software producers may well prefer not to apply for a patent, provided that the underlying ideas are not obvious to any user. Put differently, if the main benefits from patent protection were seen to arise from the second of the effects described above, and patent applications were to be required to disclose information that reduces the search costs for other innovators, this might undermine the incentives for seeking patent protection in the first place for software whose underlying ideas and principles can be kept secret to a sufficient degree to eliminate the threat from competitors developing similar software products independently.

In this context, it is important to recognise that the crucial piece of information that is required in order to ensure interoperability of software (and thus promote effective competition) is not detailed description of the algorithms and procedures that drive any particular piece of software, but interface information. It is the holding back of such interface information that may distort competition in innovation and limit the extent to which network benefits can be realised. This problem has been identified as one of the main issues in *US v* Microsoft where it has been called the 'Applications Barrier to Entry'.

There may be good reasons for supporting a general obligation not to restrict use of such information - essentially the same arguments that support the interconnection and access obligations that are common in the telecommunications sector or other network industries. Indeed, the proposed Directive on the patentability of computer related innovations expressly states that the right to decompilation for the purpose of establishing interface information as set out in the Software Directive would not be affected by patent protection.

However, the right to decompilation provides only a backstop, and given the difficulty and costs of reverse engineering this threat may ultimately be ineffective as an incentive to disclose and license such information. If the main benefit of software patents were to arise from improved disclosure, then it may be necessary to consider other measures to promote competition (such as compulsory licensing of interface information). Unfortunately, this would make seeking patent protection less attractive.

Thus, trade secret protection, coupled with first mover advantages in markets with network externalities, appears to be a very attractive option for ideas that are not entirely obvious and where reverse engineering is too costly or ineffective because of the time delay involved. It would be even more attractive if the disclosure required to obtain a patent were to have the effect of making life easier for other innovators (producing not only potential substitutes, but also complementary products that might ultimately have the effect of weakening the applications barrier to entry).
By contrast, seeking patent protection would appear to be attractive for those ideas where the obligation to disclose the underlying ideas and principles has little impact because they are obvious for all to see. One might suppose, for example, that the patent granted

for one-click shopping does not provide much additional (if any) insight into the underlying ideas compared to seeing how this invention works in practice.

As a result it may well be possible that software patents gives rise to an adverse selection process that results in a raft of - perhaps unenforceable patents - for rather obvious inventions. Casual observation appears to support this view - one only needs to look, for example, through the list of patent applications and patents issued for computer-supported auction systems where much more creativity seems to have gone into drafting these applications than into developing the underlying systems.

Ideas, principles and algorithms that are not obvious might still be better protected as trade secrets, in particular if an effective regime for the compulsory licensing of interface information for patented innovations were in place. As a result, crucial information about interfaces required in order to achieve interoperability might not be available, and distortions of competition in innovation may remain.

Unfortunately, the overall effect would not be neutral. Having a raft of patents for rather obvious ideas would still create a considerable amount of friction and increase the cost of innovation because one would have to establish whether a particular idea, however obvious, is protected by patent. For example, one might well have thought that the idea of keeping a customer's details on file in order to make further purchases easier is an obvious one, and one that could be used freely in designing e-commerce systems. But then, it might be covered by the one-click shopping patent - or might it not? Doubts about the enforceability of such patents only create more uncertainty and costs. Making patent applications easier (by removing the need for clever drafting, for example) might unleash a wave of applications for the most spurious patents in the hope of effectively blackmailing innovators by threatening to sue for infringement. Defensive patenting - the natural response - might add a further pile of worthless, but ultimately costly patents.

In summary, software patentability - and moves towards making it easier to obtain software patents - might not make much difference as far as disclosure is concerned, but add a considerable amount of friction to the system. Whilst some of the worst excesses might be curbed by improvements in patent office searches and a more restrictive interpretation of the requirements that have to be met for patentability, this would not address the problem of the distortion of incentives that might result from the disclosure obligation (in particular if this were designed in a way to maximise the benefits for other innovators). It might well be much pain for little gain, and that sounds like a rotten deal.

Bibliography

Besen S M and Raskind L J (1991) 'An Introduction to the Law and Economics of Intellectual Property', Journal of Economic Perspectives 5.
Boone J and van Dijk T (1998) 'Competition and Innovation', De Economist 146.
Breyer S (1970) 'The Uneasy Case for Copyright: A Study of Copyright in Books, Photocopies and Computer Programs', Harvard Law Review 84.
Farrell J (1995) 'Arguments for Weaker Intellectual Property Protection in Network Industries', Standard View 3.

Farrell J and Saloner G (1985) ‘Standardization, compatibility and innovation’, Rand Journal of Economics 16.
Friedman D D, Landes W M and Posner R A (1991) ‘Some Economics of Trade Secret Law’, Journal of Economic Perspectives 5.
Gilbert R and Shapiro C (1990) ‘Optimal Patent Breadth and Length’, Rand Journal of Economics 21.
Hurt R R and Schuchman R M (1966) ‘The Economic Rationale for Copyright’, American Economic Association Papers & Proceedings 56.
Katz M L and Shapiro C (1985) ‘Network externalities, competition and compatibility’, American Economic Review 75.
Kitch E W (1977) ‘The Nature and Function of the Patent System’, Journal of Law & Economics 20.
La Manna M (1993) ‘New Dimensions of the Patent System’, in Norman G and La Manna M (eds.) The New Industrial Economics (Aldershot:Edward Elgar).
Landes W M and Posner R A (1989) ‘An Economic Analysis of Copyright Law’, Journal of Legal Studies 18.
Menell P (1987) ‘Tailoring Legal Protection for Computer Software’, Stanford Law Review 39.
Menell P (1989) ‘An Analysis of the Scope of Copyright Protection for Application Programs’, Stanford Law Review 41.
Merges R P (1999) ‘As Many as Six Impossible Patents Before Breakfast: Property Rights for Business Concepts and Patent System Reform’, Berkeley Technology Law Journal 14.
Plant A (1934a) ‘The Economic Aspects of Copyright in Books’, Economica 1.
Plant A (1934b) ‘The Economic Theory Concerning Patents for Inventions’, Economica 1
Schmidtchen D and Koboldt C (1993) ‘A Pacemaker that Stops Halfway: The Decompilation Rule in the EEC Directive on the Legal Protection of Computer Programs’, International Review of Law and Economics 13.
Scotchmer S (1991) ‘Standing on the Shoulders of Giants: Cumulative Research and Patent Law’, Journal of Economic Perspectives 5.