COMPUTERS & LAW

JOURNAL FOR THE AUSTRALIAN AND NEW ZEALAND SOCIETIES FOR COMPUTERS AND THE LAW

Pnnt Post—PP233867/00008

Editors: David Standen, Simon Pollard and Kent Davey
Number 34

ISSN 08117225
December 1997

The Use of Copyright and Patents for Software Protection

©1997 Annelies Moens

Introduction

This article discusses the two forms of intellectual property protection for software with primary emphasis on the Australian jurisdiction. The patentability of software is discussed, in particular why software should be patented, rather than solely protected by copyright. The components of software which copyright and patents protect can be technically distinguished. Reasons for seeking software patent protection, in particular petty patent protection, instead of relying solely on copyright are espoused. The requirements for software patents are considered in light of Australian, US, and EC perspectives. Arguments made by the opposition against software patents are critically assessed. In particular it is espoused by the author that software patents have the potential capacity to increase freedom of software dissemination, through disclosure of source code within a regulated environment.

Forms of Computer Intellectual Property Law Protection

There are three forms of computer intellectual property protection, namely copyright, patent and circuit layout. Each of these forms of protection are best suited for particular purposes. The two forms of protection most relevant to software are patents and copyright. The Australian *Copyright Act* 1968 (Cth) and the

Continued on page 3 ☞

In this issue...

Continued from page 1

Australian *Patents Act* 1990 (Cth)[1] each cover distinct components of intellectual property. In regard to software this distinction has been muddied in the eyes of the public. There are in fact major differences between what the *Copyright Act 1968 (Cth)* protects on the one hand, and what the *Patents Act* 1990 (Cth) protects on the other. An analysis of their differences in respect of software ensues.

Copyright protects the expression of an idea, through an original literary, dramatic, musical or artistic work,[2] sound recording, film, broadcasts (TV and sound) and published editions of works.[3] A computer program, or software is considered a literary work by definition in s.10(1) of the *Copyright Act* 1968 (Cth):

literary work includes:

a) a table, or compilation, expressed in words, figures or symbols (whether or not in a visible form); and
b) a computer program or compilation of computer programs;

A computer program is in turn defined within the section as:

an expression, in any language, code or notation, of a set of instructions (whether with or without related information) intended, either directly or after either or both of...

a) conversion to another language, code or notation;
b) reproduction in a different material form; to cause a... [computer]... to perform a particular function;

The copyright owner of software has the exclusive rights to reproduce the work, publish the work, and make an adaptation of the work, *inter alia*.[4] An adaptation to a literary work, being a computer program in the *Copyright Act* 1968 (Cth) is defined as "a version of the work (whether or not in the language, code or notation in which the work was originally expressed) not being a reproduction of the work".[5] In practice adaptation requires a translation of the source code. The copyright owner also has the exclusive right to reproduce the work. Where a person were to reproduce the work, for instance in the same language, with substantial modifications, then this would infringe the copyright owner's exclusive right to reproduce the work.[6] However, back up copies of computer programs do not violate the exclusive right of the copyright owner.[7] In essence copyright protects the source code and object code, but not its function.[8] The source code is the programming language the software is written in, which is understandable by a person experienced in the language, and the object code is the executable code, output by the compiler, and is understandable by the computer.

The *Patents Act* 1990 (Cth) protects the function of the software. S.18(1) of the *atents Act* 1990 (Cth) regards a atentable

COMPUTERS & LAW

Editors

Simon Pollard
c/- Gilbert & Tobin
50 Carrington Street
Sydney 2000 Australia
Tel: (02) 9367 3103
Fax: (02) 9367 3111
e-mail: spollard@gtlaw.com.au

David Standen
c/- Gilbert & Tobin
50 Carrington Street
Sydney 2000 Australia
Tel:(02) 9367 3059
Fax:(02) 9367 3111
e-mail:dstanden@gtlaw.com.au

Kent Davey
Australian Government Solicitor
GPO Box 2853AA
Melbourne VIC 3001
Tel:(03) 9242 1273
Fax:(03) 9242 1481
e-mail:kent.davey@ags.gov.au

Subscription: $32.00 per 4 issues

Advertisements: Inserts $300.00; for advertisements within the journal, rates and information will be provided by the Editors on request
Articles, news items, books for review and other items of interest may be sent to the Editors.
Journal contents may be reproduced if the source is acknowledged.

invention as being one that satisfies the requirements of manner of manufacture, novelty, inventiveness, usefulness and not having been secretly used before. The test applied for a manner of manufacture is that the invention "must be one that offers some advantage which is material, in the sense that the process belongs to a useful art as distinct from a fine art".[9] Novelty of the alleged invention is assessed in the light of the prior art base,[10] and inventiveness is assessed with regard to the skilled person in the relevant prior art.[11] There are two types of patents which can be sought, a standard patent and a petty patent. The prior art base for a petty patent is restricted to Australia, whereas the prior art base for a standard patent is the prior art within the world.[12] The software developer has two options of intellectual property protection. At first glance it may seem that copyright is sufficient. But, on a closer analysis patent protection, specifically petty patent protection is perhaps the optimal form of intellectual property protection for software, in more ways than one. In the next section the appropriateness of seeking patent protection for software shall be investigated.

Why Seek A Software Patent? — Why Not Rely On Copyright Alone?

Since 1994 Australian courts have recognized that software is patentable subject matter. [13] One of the main reasons why a patent should be sought for software, is because software is primarily functional in nature. Functionality is protected by patents, whereas only expressions of ideas or functions are protected by copyright, as can clearly be seen in the appeal decision of *Powerflex Services Pty Ltd* v. *Data Access Corporation*[14], which restricted the scope of copyright protection in relation to computer programs. There are several factors which need to be taken into account when determining whether the additional protection of a software patent is required.

Software patent protection under the *Patents Act* 1990 (Cth) should be seriously considered by the software industry, since software is inherently functional in nature. Software code is written not because of its literary authorship, but rather to serve a particular function.[15] Moreover, functionality is not the proper object of copyright protection. As Dawson J stated in *Autodesk Inc.* v. *Dyason (No. 1)*[16] "the idea of a utilitarian work is its purpose or function and that the method of arriving at that purpose or function is the expression of the idea". In that case the High Court of Australia held that AutoCAD which was a compilation of computer programs, one of which sent messages to a lock attached to the computer, was copyright protected. The respondents in this case had reverse engineered the lock, and had constructed a device with the same output as the AutoCAD lock, and were selling it at a substantially cheaper price. The court held that both the object code and source code were protected under the *Copyright Act* 1968 (Cth). This case widened the scope of copyright protection, since the software in question could today, quite suitably have been protected by a patent. However, the Full Federal Court of Australia in *Powerflex Services Pty Ltd* v. *Data Access Corporation*[17] has significantly retracted its interpretation of reproduction and adaptation in respect of computer programs.

The *Powerflex Services Pty Ltd* v. *Data Access Corporation*[18] appeal decision overruled substantially the decision of Jenkinson J sitting in the Federal Court of Australia. Black CJ, Hill and Sundberg JJ's decision in the Full Federal Court has extensive impact upon the future legal protection to be given to computer programs in Australia. The appellant, Dr Bennett, incorporated a company under the name of Powerflex Services Pty Ltd to sell software known as PFXPlus. It was argued by the respondent; Data Access Corporation, that PFXPlus infringed the copyright in Dataflex. Dr Bennett studied the documentation of Data Access Corporation's software (not code) and its operation in order to create compatible software. The source code which Dr Bennett wrote and its corresponding object code was quite different to Data Access Corporation's source code and object code. Dr Bennett used 192 of the 254 words in Data Access Corporation's language to invoke identical functional consequences. In essence the expression of the idea or function (the source code) was different, but the function was the same. Since copyright only protects the expression of ideas, Dr Bennett did not infringe Data Access Corporation's copyright in the 192 reserved words.

In addition Dr Bennett developed three macros; Report.pfa, Entgroup.pfa and Enter.pfa. Dr Bennett gave evidence that he intended to produce the same results, as the equivalent three macros in Data Access Corporation's language, Dataflex. However, the expressions he used were his own independent creations, even though the source codes were similar. The Full Federal Court of Australia held that "a process of devising a source code to perform the same function as is performed in some other source code expressed in original language does not involve creating a version of the original source code", hence does not infringe Data Access Corporation's exclusive right to adapt the work. Dr Bennett also incorporated a Huffman compression table into the software. The Full Federal Court did not overrule Jenkinson's decision in regard to this aspect, and held that Dr Bennett had infringed Data Access Corporation's copyright in their Huffman compression table, since it was indistinguishable from the one used by Data Access Corporation. The Huffman compression table came under copyright jurisdiction not as a computer program, but rather as a table or compilation.[19] The file structure and flex keys were held not to infringe Data Access Corporation's copyright, for the same reasons that the PFXPlus language was held not to infringe Data Access Corporation's copyright in Dataflex. In regard to the error text table, Jenkinson J's decision was upheld, because in regard to the error messages the idea and expression thereof were inseparable, and therefore not protected by copyright. The implication of the decision is that software developers, such as Data Access Corporation need to patent their software, if they wish to secure legal protection in Australia for their software. This decision is consistent with US reasoning in the courts in regard to the scope of copyright protection in relation to software.[20]

Several practical factors need also to be considered when deciding which form of intellectual property protection is best suited for particular software. Factors such as distribution, competition within the software industry, patent costs, licensing options, *inter alia* need to be weighed. Highly competitive industries such as the software industry, are best protected by petty patents (known as utility models in some nations). This is especially the case in countries which treat the best form of intellectual property protection for software to be patents, which now includes Australia. In particular, the courts in the USA have been reading the US Code, Title 17, Copyrights narrowly in regard to software. In the USA and Japan more and more software is being protected by patents.[21] Patents also require exploitation of the invention , unlike copyright where a work doesn't need to be exploited to be protected by copyright. Where a patent is not exploited commercially, the patent is revoked. Under copyright, there is no requirement for commercial exploitation. The existing petty patent system in Australia is particularly suited to software patents, especially since the software industry has a high turn over rate of software. The duration of patent protection for a petty patent is six years, whereas for a standard patent it is twenty years.[22] Petty patents are also cheaper to obtain than standard patents, and the prior art base for petty patents is limited to within Australia.[23] In contrast the *Copyright Act* 1968 (Cth) gives software code protection for the life of the author plus 50 years.[24] The number of claims for a petty patent is also limited to three (one independent claim and two dependent claims), in contrast with six for standard patents.[25]

Requirements For Software Patents

Requirements for patents differ slightly in each country. In Australia petty or standard patents can be obtained. In the US there is no petty patent system, hence several prominent academics have suggested that a *sui generis* approach be taken in the US to patent software.[26] The *sui generis* approach has similarity with the petty patent system in Australia. Several EC countries have utility models which are analogous to petty patents in Australia. It is important to target the countries that would be most affected by a developer's software. Patents can be obtained in the countries where a developer's software is being exported. This section covers the Australian, US and European guidelines on software patents. In addition, the Australian, US and European disclosure requirements are also addressed.

The Australian Industrial Property Organization (AIPO) publishes guidelines for computer intellectual property protection.[27] Software patents need to satisfy the s.18 requirements enunciated earlier. Algorithms, which are procedures for solving given types of mathematical problems,[28] are not patentable as such, unless they are applied in practical situations, and are not merely implemented in a computer program. Even where one of the claims in the specification is for something that is inherently unpatentable, the entire specification is not necessarily dismissed, since the specification is judged as a whole. Software patent inventions follow the same rules as their mechanical counterparts. In today's information technology age, information engineering is rapidly replacing the mechanical counterpart. Hence, it is only to be expected that patents are issued for software.

The US Patent and Trademark Office published guidelines for computer-implemented inventions on the 1st June 1995. These US guidelines were published in response to a spark of US Federal Circuit cases involving software. Thousands of software patents have since been awarded in the United States.[29] The guidelines recognize that computer-implemented inventions may fall into three statutory recognized categories:

> a process:- a series of specific operational steps to be performed on or with the aid of a computer
>
> a machine:- a computer or other programmable apparatus whose actions are directed by a computer program or other form of software
>
> an article of manufacture:- a computer-readable memory that can be used to direct a computer to function in a particular manner when used by the computer

Essentially the US Code, Title 35 on Patents has the same requirements of utility, novelty and inventiveness as the Australian *Patents Act* 1990 (Cth).[30]

The European Patent Office requires that inventions must relate to a technical field, be concerned with a technical problem and be characterized in the claims by means of technical features.[31] Article 52(2) lists subject-matter which is not to be regarded as an invention, this includes programs for computers. However, the current interpretation of article 52(2)&(3) is that subject-matter can be protected where there is a technical contribution to the known art, even if computer programs are involved. This has been confirmed by a significant number of decisions of the EPO Boards of Appeal. Since the EPO has been founded, 11000 patents for software related inventions have been granted. Fewer than 100 applications in this area were rejected.[32] For instance, patents have been granted for image processing, computer simulations, neural networks, processing of natural languages, speech recognition, *inter alia*.[33]

The disclosure requirements are also of importance in software patents. In Australia, s.40(2) of the *Patents Act* 1990 (Cth) requires that a complete specification must: "describe the invention fully, including the best method known to the applicant of performing the invention." What is of importance in the specification is that it can be understood, hence complete program listings are not necessarily required. Flowcharts, or pseudo code descriptions would be adequate. This is especially so under the EPO guidelines, where flowcharts and pseudocode descriptions are welcomed.[34] Under the EPO Guidelines, computer inventions have to be described in terms of structure and function.[35] Terminology and signs must be clear and consistent throughout the entire application.[36] The technical problem which the invention aims to overcome needs to be understood.[37] Hence, complete program listings cannot be relied upon as the sole disclosure of the invention. Under the US Code, Title 35, Patents, a patent application includes a specification (s.112), a drawing if necessary (s.113), and an oath by the applicant stating the inventor's belief that the invention is original and that she or he is the first inventor (s.115). The specification, under s.112 needs to contain a written description of the invention, the manner and process of making and using it in clear terms, which a person skilled in the art would understand.

In the author's opinion, complete source code listings in addition to descriptions of function should be included in the specification. This revelation of source code would greatly enable future development of software, since other programmers would be able to freely use the code after expiration of the patent. In fact some US software patents have included full source code listings, including, patent No. 5299121: "Non-prescription drug medication screening system" obtained by Medscreen in 1994. It contains 2300 lines of C code and 134 expert system rules with weights and confidences. Another patent including full source code listings is patent No. 5261041: "Computer controlled animation system based on definitional animated objects and methods of manipulating same" obtained by Apple Computer in 1993. It contains 1500 lines of C++ code.[38] It would be theoretically possible for patent publications to act as some sort of software library. This is specifically of use in relation to petty patents, since the contents of pending applications are not published until the date of the grant of patent.[39] Users of the library, during the patent term, no doubt would need to pay license fees to contributors (patent holders) to the library, to avoid infringement proceedings. This idea will be further explored in the next section.

The Opposition To Software Patents

Some developers believe that software patents will stifle innovation and competition within the industry,[40] whereas others think it will do exactly the opposite. Financial software patents are especially on the rise in the banking industry. Bankers argue that patents encourage development of new applications, through disclosures and promote investment. In this highly competitive industry, companies which normally had trade secrets, are now more likely to patent.[41] Some people involved in the software industry believe that programming freedom will be curbed with the introduction of software patents. In addition it has been expressed that Patent Examiners will have difficulty determining novelty and inventiveness based on the prior art. Also of consideration is the possibility of unknowingly infringing a software patent. The author gives a counter view to all these fears.

One of the largest bodies against software patents is the League for Programming Freedom. This body believes that software patents will inhibit programming freedom.[42] However, it is quite possible that software patents may increase dissemination of source code. This is through the author's suggested amended disclosure requirements for software patents. It is quite feasible that the software patent system may be able to develop into a software library. This would especially be the case if specifications required disclosures of source code. This collection of source code by the Patent regime can be likened to a software library. Holders of software patents are contributors to software libraries and borrowers who reuse the code in new applications (hence saving enormous development and reverse engineering costs) would need to pay fiscal compensation to the contributors or patent holders, at least for the duration of the patent. This could take the form of some type of license. The *Patents Act* 1990 (Cth) includes a section on compulsory licensing requirements.[43] After expiration of the patent, the software can freely be used. If software developers wish to reuse the code, or alter it, then fiscal compensation can be made in the form of copyright licenses or assignments, to avoid copyright infringement proceedings. For all practical purposes, patents establish a trade within industries. Patents can and are used as bartering tools for the use of other company's patents.

Hence, the patent regime has the capacity to increase programming freedom through a regulated software library. This concept can be of great importance, since quality software can be reused with minimal cost after expiration of the patent, thus reducing expenditure in developing software. Now is the time to make these decisions of software dissemination, since the software industry is still young. The patent system, rather than copyright can serve the ulterior purpose of freedom of source code dissemination, through source code disclosure in patent applications, within a regulated environment. Ironically, freedom of software code dissemination can only occur in a regulated environment.

Another concern of the software patent opposition is the practical problem of examination, to determine whether an alleged software invention is novel and inventive. Naturally, this will be a difficult task in the beginning, when software patents begin to be granted, since previous unpatented software will be difficult to locate. The US Patent Office in January 1996 issued, however, examination guidelines for software, which will help examiners determine whether particular software is patentable subject matter.[44] The guidelines indicate that examiners need to focus on statements that identify practical uses for the invention. The examiner must determine its functionality, how it is configured to provide that functionality, and the relationship of the software with subject matter outside the computer.[45] However, once a library of software patents is established, the problem of examination will diminish over time.

Other fears that have been raised, include the fact that an infringement of a software patent can occur unknowingly. However, s.123(1) of the *Patents Act* 1990 (Cth) stipulates that:

[a] court may refuse to award damages, or to make and order for an account of profits, in respect of an infringement of a patent if the defendant satisfies the court that, at the date of the infringement, the defendant was not aware, and had no reason to believe, that a patent for the invention existed.

Conclusion

There will never be a hard and fast boundary dictating whether to rely on copyright alone or use patents for software protection. To maximize software protection both copyright and patents can be used. Copyright rests on top of patents, as it is an automatic right of intellectual property protection in Australia (in the US, registration of copyright provides for additional benefits). In particular, the existing petty patent system in Australia is suitable for software protection within. The patent system can be used to establish a software library, wherein source codes for software patents are published. In this way, patents, contrary to commonly held beliefs, can increase the freedom of software dissemination, and hence aid the development of new software technologies, whilst maximizing reuse of quality code, if regulated licensing arrangements are made.

Bibliography

Aharonian, Greg, http://www.lpf.org/Patents/1995-patents.html

Aharonian, Greg, "Two Interesting Patents with Source Code" http://sunsite.unc.edu/patents/txt/081294.txt

Albert, Jennifer, "New Patent Examination Guidelines for Computer Software" http://204.4.64.2/patent/corpsurviv/softpat.html 1996

Australian Industrial Property Organization, "Guidelines for Computer Intellectual Property Protection" http://www.aipo.gov.au/vault/patents/computer.pdf

Beresford, Keith, "The Patenting of Software in Europe and the UK" *Patent World* Issue 91, 1997

Bierce B William and Harold C Michael, "New US Patent Guidelines Offer Hope to Software Developers in Era of Diminishing Copyright Protection" *Computer Law & Practice* Vol. 11, No. 4, 1995

European Patent Office, "Patenting Computer Software" *1994 Annual report of the European Patent Office*

Graham D Lawrence and Zerbe D Richard, "Economically Efficient Treatment of Computer Software: Reverse Engineering, Protection and Disclosure" *Rutgers Computer & Technology Law Journal* Vol. 22, No. 1, 1996

League for Programming Freedom, "Against Software Patents" http://www.lpf.org/patents/against-software-patents.html 1991

McQuaker, Ron, "Should Patents be Granted for Computer Programs as Such?" *The Computer Law and Security Report* Vol. 11, 1995

Paley, Mark Aaron, "A Model Software Petite Patent Act" *Santa Clara Computer & High Technology Law Journal* Vol. 12, No.2, 1996

Samuelson, Pamela *et al*, "A Manifesto Concerning the Legal Protection of Computer Programs" *Columbia Law Review* Vol.95, No.8, 1994

The Economist, "Fickle Innovation: Smart Money Banks of Patents" *The Australian* Tues 20 Feb 1996, p.43

1 Herein after all references to the Australian patent and copyright legislation will be referred to as the *Copyright Act* 1968 (Cth) and the *Patents Act* 1990 (Cth)
2 *Copyright Act* 1968 (Cth) Part III - copyright in original literary, dramatic, musical and artistic works
3 *Id.* Part IV— copyright in subject-matter other than works
4 *Id.* s.31(1)
5 *Id.* s.10(1)
6 *Id.* s.14(1)
7 *Id.* s.43A(1)
8 William B Bierce & Michael C Harold, "New US patent guidelines offer hope to software developers in era of diminishing copyright protection" *Computer Law & Practice* Vol.11. No.4., 1995, 94-96, 96
9 *NRDC* (1959) 102 CLR 252, 275 per Dixon CJ, Kitto J, Windeyer J
10 *Patents Act* 1990 (Cth) s.7(1)
11 *Id.* s.7(2)
12 *Id.* Schedule 1
13 *CCOM* v. *Jiejing* (1994) 122 ALR 417
14 [1997] 490 FCA (4 June 1997)
15 Lawrence D. Graham & Richard D. Zerbe Jr., "Economically Efficient Treatment of Computer Software: Reverse Engineering, Protection and Disclosure" *Rutgers Computer & Technology Law Journal* Vol.22. No.1., 1996, 61-142, 92
16 (1992) 22 IPR 163, 172
17 [1997] 490 FCA (4 June 1997)
18 *Ibid.*
19 *Copyright Act* 1968 (Cth) s.10(1)
20 *Lotus* v. *Borland* 49 F.3d. 807 (1st Cir. 1995)
21 Ron McQuaker, "Should Patents Be Granted For Computer Programs As Such?" *The Computer Law And Security Report* Vol. 11., 1995, 259-261, 261
22 *Patents Act* 1990 (Cth) s.67 (standard patent) s.68 (petty patent)
23 *Id.* Schedule 1
24 s.33
25 *Patents Act* 1990 (Cth) s.40(2)c
26 Pamela Samuelson *et al*, "A Manifesto Concerning the Legal Protection of Computer Programs" *Columbia Law Review* Vol. 94. No.8., 1994, 2308-2431, 2312.
27 http://www.aipo.gov.au/vault/patents/computer.pdf
28 Mark Aaron Paley, "A Model Software Petite Patent Act" *Santa Clara Computer & High Technology Law Journal* Vol.12. No.2., 1996, 301-380, 318
29 http://www.lpf.org/Patents/1995-patents.html
30 US Code, Title 35, Patents ss.102,103
31 Rules 27 & 29 EPC
32 "Patenting Computer Software" 1994 *Annual Report of the European Patent Office*
33 Keith Beresford, "The Patenting of Software in Europe and the UK" *Patent World* Issue 91, April 1997, 18-20
34 EPO Guidelines C-II, 4.14a
35 *Id.* C-II, 4.9a
36 Rule 35(13) EPC
37 Rule 27(1)c EPC
38 Greg Aharonian, "Two Interesting Patents with Source Code" http://sunsite.unc.edu/patents/txt/081294.txt
39 *Patents Act* 1990 (Cth) s.62
40 William B Bierce & Michael C Harold, "New US patent guidelines offer hope to software developers in era of diminishing copyright protection" *Computer Law & Practice* Vol.11. No.4., 1995, 94-96, 95
41 The Economist, "Fickle innovation: smart money banks on patents" *The Australian* Tues 20 Feb 1996, 43
42 League for Programming Freedom, "Against Software Patents" http://www.lpf.org/Patents/against-software-patents.html
43 *Patents Act* 1990 (Cth) s.133
44 Jennifer A. Albert, "New Patent Examination Guidelines for Computer Software" http://204.4.64.2/patent/corpsurviv/softpat.html 1996.
45 *Ibid.*

Annalies Moens is a computing science/law student at the University of Queensland.