COPYRIGHT PROTECTION OF COMPUTER SOFTWARE AND USER INTERFACES: JUST WHAT SHOULD BE PROTECTED?

BY STEPHEN FRANCIS*

[*In this article the author examines the practical implications of applying the idea/expression dichotomy to software code and user interfaces. He argues that copyright protection of software code should extend beyond literal copying to encompass copying of the code's structure, sequence and organization. He also argues that user interfaces should attract copyright protection in their own right. In light of the practical ramifications of extending protection to user interfaces, the author concludes that the most equitable solution is to make such interfaces the subject of a compulsory licensing scheme.*]

1. *INTRODUCTION*

It is a fundamental principle of copyright law that copyright does not protect ideas, but only the form of their expression. However, as is clear from even a cursory examination of the cases, it is extremely difficult to define any line which separates an idea from its expression.[1] This dichotomy is even more contentious in the context of computer programs. The controversy arises because computer programs are both functional and expressive, and so require a resolution of the issue of whether the function of the program should be protected. The recent High Court decision of *Autodesk Inc. and Another v. Dyason and Others*,[2] while neatly sidestepping the question of the role of the idea/expression dichotomy in relation to computer software, highlights the need for a thorough analysis of this area.

The purpose of this article is to demonstrate the practical implications of applying the idea/expression dichotomy to software code and user interfaces. An application of copyright which reflects the realities of software development will, it is urged, extend copyright protection beyond literal code to encompass, to a limited extent, the structure, sequence and organization of a program. Further,

* B.Sc. (Mon.), LL.B. (Mon.). Information Support Specialist for Chubb Sovereign Life Insurance Company, Santa Barbara, California.

1 *Beck v. Montana Constructions Pty Ltd* (1963) 5 F.L.R. 298, 301; *Nichols v. Universal Pictures Corporation* 45 F. 2d 119 (1930), 121; *Lotus Development Corporation v. Paperback Software International* 740 F. Supp. 37 (D. Mass. 1990), 60; *Bauman v. Fussell* [1978] Reports of Patent Cases 485 (Court of Appeal).

2 (1992) 104 A.L.R. 563. The *Autodesk* case hinged on the question of whether a device which encoded and produced the sequence of signals sent by a hardware component was an infringement of copyright. The Federal Court at first instance (*Autodesk Inc. and Another v. Dyason and Others* (1989) 15 I.P.R. 1) held that the device was an infringement because it reproduced the function (or interface) of the component. On appeal, the Full Federal Court (*Dyason and Others v. Autodesk Inc. and Another* (1990) 96 A.L.R. 57) denied that the function of a hardware component was protected by copyright. The High Court (*Autodesk Inc. and Another v. Dyason and Others* (1992) 104 A.L.R. 563) reversed the Full Federal Court's decision and held that there was a breach of copyright, but did not determine the judgment by looking at the issue of function. Rather, the High Court held that the device, which embodied the sequence of signals in a table, had indirectly reproduced the table in the software which read the signals from the hardware component. See Part 4.1 below.

a consistent and practical application of copyright principles requires that user interfaces, although functional, attract copyright protection in their own right. After considering the practical ramifications of extending protection to user interfaces, it is suggested that the most equitable solution is to establish a compulsory licensing scheme for user interfaces.

2. *THE IDEA/EXPRESSION DICHOTOMY*

While copyright law prohibits the making of infringing reproductions of works, it has long been recognized that reproduction for copyright purposes refers only to copying the expression of a work — there is no prohibition on copying the work's underlying idea.[3] Clearly, copyright would be of little use if it afforded protection only to the literal expression of a work. However, it has been held that, even if there is no literal copying, the copyright in a fictional novel or play will be infringed where the 'combination of situations, events and scenes which constitute the particular working out or expression of the idea or theme' is copied.[4] In *Nichols v. Universal Pictures Corporation*,[5] Learned Hand J. recognized the difficulty of determining whether a variation from the work's literal text amounts to a reproduction of the work's protected expression or its unprotected idea:

> [A]s soon as literal appropriation ceases to be the test, the whole matter is necessarily at large, so that ... the [previous] decisions cannot help much. ... Upon any work, and especially upon a play, a great number of patterns of increasing generality will fit equally well, as more and more of the incident is left out. The last may perhaps be no more than the most general statement of what the play is about, and at times might consist only of its title; but there is a point in this series of abstractions where they are no longer protected, since otherwise the playwright could prevent the use of his [*sic*] 'ideas', to which, apart from their expression, his [*sic*] property is never extended. ... Nobody has ever been able to fix that boundary, and nobody ever can.[6]

Thus copyright protects 'levels of abstraction' beyond the literal expression of a work until the level of abstraction embodies the work's underlying ideas. This is necessary so that copyright protection cannot be evaded by simple alterations such as changing the names of characters in a play.

The level of abstraction which is protected by copyright, and thus the ambit of what is regarded as expression, differs from case to case, and is significantly narrowed in some situations. Where the idea is only capable of being expressed in a limited number of ways, the courts will accord only narrow protection to any particular representation of that idea.[7] Thus in *Kenrick v. Lawrence*,[8] Wills J. held that copyright in a drawing of a hand holding a pencil marking an X must be confined to what was special about the individual drawing over and above the idea, and so protection would extend only to preventing almost identical reproductions. A similar approach is taken for architectural plans: the protection

3 *Baker v. Selden* 101 U.S. 99 (1879); *Hollinrake v. Truswell* [1894] 3 Ch. 420.

4 *Zeccola v. Universal Studios* (1982) 46 A.L.R. 189, 192 *per* Lockhart J.

5 45 F. 2d 119 (1930).

6 *Ibid*. 121.

7 Indeed, it seems that the High Court has implicitly adopted the United States doctrine of merger, that 'when the expression of an idea is inseparable from its function, it forms part of the idea and is not entitled to the protection of copyright.' *Autodesk Inc. and Another v. Dyason and Others* (1992) 104 A.L.R. 563, 572 *per* Dawson J.; Mason C. J., Deane, Brennan and Gaudron JJ. agreeing.

8 (1890) 25 Q.B.D. 99.

afforded to any 'element which may be described as common to all plans' must of its nature be very limited.[9]

Similarly, what is regarded as expression is narrower in factual works than in fictional works. Clearly, works dealing with the same factual data will resemble each other if it is not possible to express the data in a substantially different manner. In situations where there is no other way to express the facts, only the literal expression of the information is protected.[10] However, in such situations, the courts also confer protection on the skill and effort involved in the compilation of the factual data, and will not allow another simply to appropriate the efforts of the compiler.[11]

In the context of computer software, several factors complicate the identification of the protected expression and the unprotected idea. First, computer programs are functional, not artistic. Therefore, they are analogous to factual literary works which have a narrow scope of protection. Secondly, programs may be functionally equivalent in several forms. This means that without protection to an appropriate 'level of abstraction', copyright would be easy to circumvent. Thirdly, and most importantly, it is often overlooked that one program has two separate expressions — its code and its user interface. This factor is examined in Part 4 below.

3. *THE DICHOTOMY AS IT APPLIES TO CODE*

It is now accepted in most jurisdictions that the appropriate means to protect software code is to treat it as a literary work in which copyright can subsist.[12] The Australian legislature adopted this approach by amending the Copyright Act 1968 (Cth) via the Copyright Amendment Act 1984 (Cth),[13] and it is now reasonably settled[14] that software is protected as a literary work in both its object code[15] and source code[16] forms. The Australian legislature has not, however,

9 *Beck v. Montana Constructions Pty Ltd* (1963) 5 F.L.R. 298, 301 *per* Jacobs J.

10 *E.g.* in *Football League v. Littlewoods Pools Ltd* [1959] 1 Ch. 637 Upjohn J. held that the copyright in the plaintiff's annual football fixture was infringed by the defendant's weekly betting coupons. However, since there is no copyright in information, Upjohn J. queried whether the defendant would have infringed had it listed the matches in a different order from those in the plantiff's fixture.

11 *Ravenscroft v. Herbert and Another* [1980] Reports of Patent Cases 193 (Chancery Division); *Elanco Products Ltd and Another v. Mandops (Agrochemical Specialists) and Another* [1980] Reports of Patent Cases 213 (Court of Appeal).

12 *E.g.* 17 U.S.C.A. § 101 (U.S.); Copyright, Designs and Patents Act 1988 (U.K.) s. 3(1); Consolidated Act No. 453 of June 23, 1989 on Copyright in Literary and Artistic Works (Denmark); Japanese Copyright Law, Law No. 48 (1970) Arts 2(1)(x-ii), 10(1)(ix).

13 The Copyright Act 1968 (Cth) as amended, provides ın s. 32(1) that copyright may subsist in an original literary, dramatic, musical or artistic work. 'Literary work' is defined in s. 10(1) to include:
(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.

14 *Star Micronics Pty Ltd v. Five Star Computers* (1990) 18 I.P.R. 225 (Federal Court).

15 Object code refers to the actual numeric instructions in the computer's memory. Each number in the memory acts as an instruction to the processor.

16 Programmers write programs in source code form. Source code is much more 'human readable' than object or executable code because a significant amount of its expression is removed when the program is compiled into object code. For example, source code will contain helpful comments, and variables and subroutines will be referred to by meaningful names (such as 'Number_of_Files'). Further, source code is much more concise than object code because each source code statement may require dozens of object code instructions to implement.

considered the extent to which programs are protected by copyright beyond the literal copying of code. This issue has generated a substantial degree of controversy in the United States.

Whelan Associates Inc. v. Jaslow Dental Laboratory Inc.[17] is the leading United States case in this area. In that case, the plaintiff had created a program to aid in the administration of dental laboratories which could only be run on large computers. The defendant developed a program with similar functions and screen displays to run on personal computers. The Third Circuit Court of Appeals held that the structure, sequence and organization of the plaintiff's program was protected by copyright, and that the copyright protection of a program was not limited to the literal computer code. It reached this conclusion by analogy with various cases which had held that the copyright in a book or play encompassed the arrangement of dramatic incidents. Although the Court was probably correct in this conclusion, it is widely regarded that the extent of protection which the Court was prepared to grant was far too generous. The Court stated that the limitation on what was protected was to be found in the idea/expression dichotomy, but then held that

> the line between idea and expression may be drawn with reference to the end sought to be achieved by the work in question. In other words, *the purpose or function of a utilitarian work would be the work's idea, and everything that is not necessary to that purpose or function would be part of that expression of the idea.*[18]

In the case before it, the Court held that 'the purpose of the . . . program was to aid in the business operations of a dental laboratory.'[19] Clearly, with such a general purpose, the structure of the plaintiff's program was protected up to a very abstract level. The Court held that the defendant's program reproduced the 'structure, sequence and organization' of the plaintiff's program because there were other ways of structuring programs to perform the function of aiding the business operations of a dental laboratory. The Court reached this conclusion because, when run, both programs performed the same sequence of accounting functions. There was no comparison of the computing procedures or the code. This method comes perilously close to conferring a monopoly over the system of accounting used.

In my view, a preferable approach would have been to compare the computing procedures of the two programs at a level of generality independent of the two different computer languages used. This would be an equally valid comparison of structures, and, indeed, may have revealed that the programs used different methods to achieve the same results.[20] In an action for infringement of the copyright in the structure of code, the results of running the program should, at most, create a rebuttable presumption that the structure of the codes is the same, but should not constitute the basis for a finding of reproduction of the code. The relevant expression in such an action is not that produced by running the program, but the code itself, and the two expressions must not be confused.

[17] 797 F. 2d 1222 (3rd Cir. 1986).
[18] *Ibid.* 1236 (emphasis in original).
[19] *Ibid.* 1238.
[20] Greenleaf, G., 'Screens, Structures and Ideas on the Boundary of Copyright' (1988) 62 *Australian Law Journal* 630, 631.

The fact that the Court in *Whelan* identified the idea as being 'to aid in the business operations of a dental laboratory' implies that what remained was the expression of the program. This is 'hopelessly overbroad in theory'[21] because, at this level of abstraction, there will almost always be other ways to effect a function, and so programmers would be prohibited from using or adapting any part of the program, including the *processes* in it.[22] The *Whelan* test possibly protects the program's data processing techniques to such an extent that the author could even prevent their use in a program which had a completely different application.[23]

Thus the *Whelan* decision must be criticized, not for conferring protection of programs at a level higher than literal code, but for failing to apply the idea/expression dichotomy correctly.[24] The Court regarded the purpose of the program as the only idea it embodied. Yet any program will involve many ideas, and most of these will be implemented in non-original, standard code. 'Top-down' design is probably the most popular systems design methodology used today. This involves breaking the original problem into smaller problems. These smaller problems are also broken down in an iterative process until the resulting problems are trivial to encode (*e.g.* 'sort the numbers into order'). Thus each program involves many ideas in order to arrive at a workable solution, and *none* of these ideas should be protected by copyright. The mere use of a bubblesort (or quicksort or hashsort or any other method of sorting) in a program should not confer any rights over the idea of that sorting system.

In my view, the preferable approach with regard to software code would be to apply the judgment of Learned Hand J. in *Nichols v. Universal Pictures Corporation*.[25] In other words, a court must identify the highest level of abstraction beyond the code's literal expression which does not embody the underlying ideas. This approach is particularly suited to software protection, given the top-down methodology generally applied to software development.[26] Clearly, functionally-dictated structures of common modules should be available to all, yet to grant protection at a level where the structure is not functionally-dictated may come close to protecting the idea. For this reason, it is suggested that the level of abstraction for copyright purposes should be one fairly close to

21 Spivack, P., 'Does Form Follow Function? The Idea/Expression Dichotomy in Copyright Protection of Computer Software' (1988) 35 *U.C.L.A. Law Review* 723, 747. However, it should be noted that the Australian High Court seems to have impliedly approved of the decision in *Whelan Associates Inc. v. Jaslow Dental Laboratory Inc. (supra* n. 17) and also the characterization of the unprotected idea being only the 'purpose . . . of . . . assist[ing] in keep[ing] records of the business operations of a dental laboratory', although the issue was not considered at length. See *Autodesk Inc. and Another v. Dyason and Others* (1992) 104 A.L.R. 563, 572 *per* Dawson J.; Mason C. J., Deane, Brennan and Gaudron JJ. agreeing.

22 Karjala, D., 'Japanese Courts Interpret the "Algorithm" Limitation on the Copyright Protection of Computer Programs' [1990] 7 *European Intellectual Property Review* 235, 240. It is interesting to note that, due to the fact that they viewed the protection conferred by the decision as too broad, Japanese courts have not followed *Whelan*, but instead have opted to deny protection of program structure except for literal or near-literal copying of code (Karjala, *op. cit.* n. 22, 235).

23 Spivack, *op. cit.* n. 21, 748.

24 Brinson, J. D., 'Copyrighted Software: Separating the Protected Expression From Unprotected Ideas, a Starting Point' (1988) 29 *Boston College Law Review* 803, 831.

25 *Supra* n. 6 and accompanying text.

26 Brinson, *op. cit.* n. 24, 854.

code level and of a narrow scope.[27] Standard routines, which may be common to all, would only attract protection in the literal code, if at all, because they are likely to lack the originality required to warrant further protection.[28] However, the sequence and structure of a *compilation* of these standard routines could be protected if it were sufficiently original and the skill involved in creating it was not negligible.[29] Compilations of computer programs should be afforded the same copyright protection as other literary compilations.[30] Indeed, the drafters of the 1984 amendment to the Copyright Act seem to have anticipated this result by defining a literary work as, *inter alia*, 'a computer program or compilation of computer programs'.[31] This would validly protect software authors against those who attempt to appropriate their efforts by simply writing the same code in a different language, or substituting alternate routines to achieve the same logical function (as in using an alternative sorting routine), but would leave the ideas of the program available for all to use.

4. *USER INTERFACES*

There has been a great deal of debate and academic comment as to whether, and the extent to which, the user interface of a computer program should be protected by copyright. The user interface of a computer program is clearly an expression of that program. Indeed, to all *users* of the software, it is the only expression of the program which they encounter. The expression of the code is only seen if the object code is disassembled or if the source code of the program is available. The fact that the user interface is a separate expression from that in the code (and thus deserving of separate copyright protection) is highlighted by the fact that the 'creative authorship in a program's audio-visual components [may be] altogether separate from the creative authorship in a program's code.'[32] Thus developers may employ graphic artists and others to design an interface, and then programmers will implement that design in code. This means that the programmer's creativity and expression in the code is separate from the creativity and expression in the visual display itself.[33]

As the expression of a program is found in two distinct manifestations, it is apparent that copyright protection for screen displays and interfaces should be

27 This is also the view taken by several other commentators. *Ibid.* 828.

28 Applying *Beck v. Montana Constructions Pty Ltd* (1963) 5 F.L.R. 298, 301 (where it was held that the protection afforded to elements of architectural plans which were 'common to all' must be very limited) and *Kenrick v. Lawrence* (1890) 25 Q.B.D. 99 (where it was held that copyright in a drawing of a hand would only be infringed by virtually identical reproductions).

29 This appears to be the approach taken by Falconer J. in an application for an interlocutory injunction: *M.S. Associates Ltd v. Power and Others* [1988] Fleet Street Reports 242, 247-8 (Chancery Division).

30 *E.g. Ravenscroft v. Herbert and Another* [1980] Reports of Patent Cases 193 (Chancery Division); *Elanco Products v. Mandops (Agrochemical Specialists) and Another* [1980] Reports of Patent Cases 213 (Court of Appeal).

31 Copyright Act 1968 (Cth) s. 10(1)(b).

32 Reback, G. L. and Hayes, D. L., 'A Modest Proposal for the Registration of Computer Displays' (1987) 4 *The Computer Lawyer* 1, 4-8 cited in Highley, R., 'Copyright Law and Computer Screen Displays' (1990) 48 *University of Toronto Faculty of Law Review* 48, 86.

33 *Ibid.*

separate and distinct from, and additional to, copyright in the program code itself.[34] Copyright attaches to the expression of the concept, not the medium. Thus, while reproducing a program which creates a screen display or user interface would infringe copyright in both the code and the interface, a program which did not copy the code of the original, but replicated its user interface, would infringe copyright in the interface only.[35] Such an approach to user interfaces has emerged in the United States, although the courts have not followed this path without controversy.

The initial cases of screen display protection in the United States related to video games. The owners of these games had registered copyright (as is necessary in the United States) in the screen displays as audio-visual works, a category of protection not available under Australian copyright legislation. The courts held that the element of fixation was satisfied by permanent storage in Read Only Memory (ROM).[36] The various decisions on the matter have left no doubt that copyright subsists in video game display screens.[37]

As noted above, the Court in *Whelan Associates Inc. v. Jaslow Dental Laboratory Inc.*[38] regarded the similarity of the screen displays of the programs as indirect evidence of similarity of the structure, sequence and organization of the underlying code. This concept was misinterpreted by the Court in *Broderbund Software, Inc. v. Unison World, Inc.*,[39] which extended it by holding that *Whelan* stood for the proposition that the copyright in the program protected the structure, sequence and organization of the program's *screen displays* in addition to the structure, sequence and organization of the code.[40] The Court in *Digital Communications Associates, Inc. v. Softklone Distributing Corporation*[41] refused to follow *Broderbund* for this reason and held that the copyright in a computer program conferred no protection on the screen displays.[42] It should be noted, however, that these decisions were influenced by the fact that registration is required for copyright protection in the United States. The *Softklone* Court held that registration of copyright in the code did not confer protection on the screen displays. This is, I believe, the preferable position. Copyright in the user

34 *Manufacturers Technologies, Inc. v. CAMS, Inc.* 706 F. Supp. 984 (D. Conn. 1989), 993; This is also the view adopted by several commentators: see *e.g.* Barkume, A., 'Proprietary Protection of Computer User Interfaces' (1990) 64 *St John's Law Review* 559, 572; Rudnick, R., '*Manufacturer's Technologies, Inc. v. CAMS, Inc.*: A False Hope for Software Developers Seeing Copyright Protection for their Generated Screen Displays' (1991) 17 *Rutgers Computer and Technology Law Journal* 211, 230; Pratt, T., 'A Legal Test for the Copyrightability of a Computer Program's User Interface' (1991) 39 *University of Kansas Law Review* 1045, 1068; Smith, B., 'Copyright Protection of Computer Software' (1989) 6 *Auckland University Law Review* 245, 253.

35 *Digital Communications Associates, Inc. v. Softklone Distributing Corporation* 659 F. Supp. 449 (N.D. Ga. 1987), 455-6. This could have the result that a licence to reproduce the code of interface routines could still infringe copyright in the interface. This, however, is merely something for the drafters of assignments and licenses to be wary of.

36 *E.g. Williams Electronics, Inc. v. Artic International Inc.* 685 F. 2d 870 (1982), 874.

37 Barkume, *op. cit.* n. 34, 568.

38 *Supra* n. 17.

39 648 F. Supp. 1127 (N.D. Cal. 1986).

40 Barkume, *op. cit.* n. 34, 570.

41 659 F. Supp. 449 (N.D. Ga. 1987).

42 The *Softklone* courts refusal to follow *Broderbund* was endorsed in *Manufacturers Technologies, Inc. v. CAMS, Inc.* 706 F Supp 984 (D. Conn. 1989), 992-3, where the Court also stated that it believed that the *Broderbund* decision was erroneous.

interface is separate and distinct from that in the code. This is consistent with the realities of programming because identical user interfaces can be produced from completely different programs.[43] It is also consistent with decisions which have held that to make an article in accordance with instructions does not infringe copyright in those instructions.[44] A user interface is made according to instructions in the software code and thus should not be held to have reproduced that code.[45] However, this does not mean that copyright should not subsist in the product of following those instructions.

Confusing the two separate copyrights of the code and the user interface often leads to additional errors when determining infringement. The Court in *Broderbund Software, Inc. v. Unison World, Inc.*[46] did not recognize the separation of the idea in the program and the idea in the screen. The dispute in that case was over similarity in *screen displays*; however, the Court focused on the idea of the *program*. The Court should have identified the idea behind the screen displays and determined the scope of protection in light of how how originally those ideas had been expressed (or, in United States terms, whether the expression was the only means possible for that idea so that the idea 'merged' with the expression and lost protection).[47] This was the approach of the Court in the *Softklone* case — it focused on the idea of the computer display because the computer display was the relevant copyrighted work. It was explicitly recognized in the later decision of *Manufacturers Technologies, Inc. v. CAMS, Inc.*[48] that a court must avoid 'the mistake of identifying a program's idea with the idea of a particular screen display or some element therein.'[49]

In the United States, the Copyright Office has adopted the practice of accepting only one registration for a program to cover both the code and the screen displays. However, in *Manufacturers Technologies, Inc. v. CAMS, Inc.*[50] the Court held that even though there was a single registration, there was a 'legal fiction' of two separate copyright registrations, one in the code and one in the screens. In the later decision of *Lotus Development Corporation v. Paperback Software International*,[51] it was held that the categories of registration overlap and that the registration of the software copyright covered the separate copyright in the user interface.[52] Thus, it seems that the United States courts have arrived at the correct conclusion that copyright in the code and interface are separate because they are expressions of different ideas.

[43] Barkume, *op. cit.* n. 34, 570; *Manufacturers Technologies, Inc. v. CAMS, Inc.* 706 F. Supp. 984 (D. Conn. 1989), 991.
[44] *Computer Edge Pty Ltd and Another v. Apple Computer Inc. and Another* (1986) 161 C.L.R. 171; *Cuisenaire v. Reed* [1963] V.R. 719.
[45] Neely, R., 'Form or Function: What is Look and Feel?' (1990) 8:4 *Copyright Reporter* 9, 10.
[46] 648 F. Supp.1127 (N.D. Cal. 1986).
[47] Pratt, *op. cit.* n. 34, 1045, 1056-7. It now seems that the Australian High Court has also implicitly adopted the U.S. doctrine of merger: *Autodesk Inc. and Another v. Dyason and Others* (1992) 104 A.L.R. 563, 572 *per* Dawson J.: Mason C. J., Deane, Brennan and Gaudron JJ. agreeing.
[48] 706 F. Supp. 984 (D. Conn. 1989).
[49] *Ibid.* 993.
[50] *Ibid.*
[51] 740 F. Supp. 37 (D. Mass. 1990).
[52] *Ibid.* 81.

4.1 *The* Autodesk *Decisions*

It may be argued that user interfaces should not be protected because they are a function of the program. However, there is no reason not to protect function in the context of copyright law, provided that the extent of this protection is limited to that which is expression.[53] Indeed, copyright law protects function in other areas (*e.g.* sound recordings). The expression of the sound recording on the medium is not protected, but the function achieved by the interaction of that medium with a playback device is protected. Therefore, it is irrelevant if a sound recording is expressed on a normal audio cassette or in digital tape format. Both may be played on the same machine, and it is the function that these recordings effect — or rather the expression at the human interface (*i.e.* speakers in the case of a sound recording) — which attracts copyright protection. Similarly, computer user interfaces are a function of a program, but function at the user interface is a relevant expression of the program. This function should attract copyright protection regardless of whether or not an infringing program reproduced any of the code of the protected program.

In my view, only computer-human interfaces should be capable of supporting the subsistence of copyright. Hardware interfaces, while sharing the same element of functionality, can make no claim to being expressive, and functionality alone is not sufficient to support copyright. This issue was addressed in the *Autodesk* decisions. Here, Autodesk Inc. owned the copyright in an expensive computer program ('AutoCAD') which was sold with a hardware lock ('AutoCAD lock') which had to be attached to the computer running the program. The AutoCAD lock, which could not be purchased separately, was designed to respond in a predetermined way whenever it received a signal from a routine in the AutoCAD program ('Widget C'). Widget C sent periodic signals to the lock and compared the returned response with the expected results which were stored in a table in Widget C. The AutoCAD program would fail to operate if the correct response was not returned (*i.e.* if the lock was not attached to the computer). This ensured that the AutoCAD program could only be used on one computer at a time. The defendants designed and marketed a lock ('Auto Key lock') which performed exactly the same function as the AutoCAD lock in a completely different way. Therefore, users could run pirated copies of the AutoCAD program on other computers if they purchased the substantially cheaper Auto Key lock.

Northrop J. at first instance[54] correctly decided that, in certain situations, function is the only significant form of expression of software for copyright purposes, but did not limit this recognition of function as expression to user interfaces, nor did he recognize that the copyright in the expression in the interface should be separate and additional to that in the program itself. He

[53] The Australian High Court accepted that purely functional objects can be protected as artistic works in *S. W. Hart & Co. Pty Ltd v. Edwards Hot Water Systems* (1985) 159 C.L.R. 466; for a House of Lords decision to the same effect see *British Leyland Motor Corporation Ltd v. Armstrong Patents Co. Ltd* [1986] 1 A.C. 577.

[54] *Autodesk Inc. and Another v. Dyason and Others* (1989) 15 I.P.R. 1 (Federal Court).

incorrectly held that similarity in function between the two devices meant that the Auto Key lock infringed the copyright in the AutoCAD lock, when at most such similarity could have been evidence of infringement of the copyright (if any) in the interface used by the AutoCAD lock.

The Full Federal Court denied that copyright law extended protection to the function of computer programs.[55] This is the correct conclusion on the facts because hardware interfaces, which lack any expression, should not be accorded copyright protection. However, the decision should not be interpreted as denying the copyrightability of user interfaces.

The High Court held that there was a breach of copyright, but this was not attributable to any reproduction of the expression or function of the interface.[56] Rather, the court held that the defendant's software table, used to ensure the Auto Key lock responded as the plaintiff's program expected, was a reproduction of the table in the Widget C program which checked the interface responses. The Court impliedly adopted the reasoning of the Full Federal Court that 'the mere fact that [the Auto Key lock] performed the same function as the AutoCAD lock did not mean that it had a sufficient degree of objective similarity to AutoCAD to amount to a breach of copyright.'[57] This is a sensible statement of the law because it does not preclude the conclusion that the copyright in an expressive interface (as distinct from the copyright in the code) may be infringed where its function is reproduced. In this case, the function was a hardware interface which was not capable of supporting copyright. Thus, although the High Court made no direct decision on the protection of user interfaces, it did not eliminate the possibility of such protection.

4.2 *Scope of the Protection*

Assuming that copyright protection extends to user interfaces, determining the extent of this protection remains a clouded issue, even in the United States. As Keeton J. observed in *Lotus Development Corporation v. Paperback Software International*,[58] the phrase 'look and feel' is of little use in determining if a program has infringed the copyright in the interface of another. A determination that the 'look and feel' of a program has been infringed is a conclusion, 'and the usefulness and applicability of a precedent depends on the *reasons* the conclusion was reached in a particular context, not on the *conclusion* itself.'[59]

An action for infringement of copyright will be established (assuming the existence of such copyright) if there is both objective similarity and a causal connection between the copyright material and the allegedly infringing material.[60] Due to the fact that copying is rarely provable by direct evidence, it is acceptable to establish a causal connection by establishing close similarities

55 *Dyason and Others v. Autodesk Inc. and Another* (1990) 96 A.L.R. 57.
56 *Autodesk Inc. and Another v. Dyason and Others* (1992) 104 A.L.R. 563.
57 *Ibid.* 572 *per* Dawson J.
58 740 F. Supp. 37 (D. Mass. 1990).
59 *Ibid.* 63 (emphasis in original).
60 *Computer Edge Pty Ltd and Another v. Apple Computer Inc. and Another* (1986) 161 C.L.R. 171.

between the allegedly infringing work and the plaintiff's work, and that the defendant had access to the plaintiff's work.[61] Establishing access will rarely be a problem with regard to published software. Thus the crux of establishing copyright infringement is whether it can be shown that there is sufficient similarity between the works.

It should be noted, however, that it is only substantial similarity between the copyrighted portions of the plaintiff's work and the defendant's work which is relevant. The issue is not if there is a substantial similarity between the look and feel of the defendant's work and the plaintiff's work, but if there is a substantial similarity between the protected expression in the plaintiff's work and the comparable parts of the defendant's work.[62] Thus, in order to establish infringement, it is necessary to define the scope of the protection of the original work.

The United States courts have held that copyright will not protect a choice to implement functions in a certain manner if the choice is restricted by hardware. Therefore, there can be no protection of the choice to use the arrow keys to move through a list, or a '/' key to call up a menu, when most other keys have assigned functions.[63] Australian courts would probably deny copyright protection to such choices on the basis that they lack originality or are too insubstantial to warrant protection. Similarly, the *ideas* in interfaces are not protected. Thus, in *Telemarketing Resources v. Symantec Corporation*,[64] copyright protection for the use of 'pull-down menus' was denied.[65] Again, Australian courts are likely reach the same conclusion.

While it may be argued that a pull-down menu is in fact an expression, and the idea is to give options via a menu, the fact that interfaces are functional means that what amounts to expression should be construed narrowly so as to avoid giving a monopoly over user-friendly interface techniques. However, while individual elements of an interface (*i.e.* the menu type, key assignment, window shape *etc.*) should receive little protection due to their inherent necessity in a program, the overall arrangement of these elements may still amount to protected expression as a compilation. Indeed, this was the result in *Lotus Development Corporation v. Paperback Software International*.[66] Similarly, the use of icons as on-screen 'buttons' should be regarded as an idea, and thus not susceptible to copyright.[67] If, however, an icon is sufficiently distinctive in its expression (such as an elaborate picture of a garbage compacter) then that *image* may be protected; otherwise, the only possible protection for the icon would be as part of the user interface compilation.

[61] *Corelli v. Gray* (1913) 30 T.L.R. 116.

[62] Barkume *op. cit.* n. 34, 575.

[63] *Manufacturers Technologies Inc. v. CAMS, Inc.* 706 F. Supp. 984 (D. Conn. 1989), 994-5; *Lotus Development Corporation v. Paperback Software International* 740 F. Supp. 37 (D. Mass. 1990), 66-7.

[64] 12 United States Patent Quarterly 2d 1991 (N.D. Cal. 1989).

[65] 'Pull-down menus' are used in many user interfaces. When the cursor is pointing to an entry in the main menu, a sub-menu is revealed (or 'pulled down') giving a series of more specific options relating to a main topic. *E.g.* pointing to a main menu entry of 'file' may reveal a pull-down menu of 'file save, file load, file copy, file print *etc.*'

[66] 740 F. Supp. 37 (D. Mass. 1990).

[67] Rudnick, *op. cit.* n. 34, 244.

In the *Lotus* case, the creators of a commercially successful spreadsheet program, Lotus 1-2-3, sued the developers of a competing program which looked, from a user's point of view, practically identical to the Lotus program. The defendant's program reproduced the Lotus interface exactly but it contained a few additions to the menus. Both programs used exactly the same key assignments, the same words in the menus, and the same hierarchy of menus. The Court found that the entirety of the menu-command structure, including the choice of terms, the structure and order of those terms, their presentation on the screen *etc.*, was capable of copyright protection. The Court found this by drawing an analogy with the cases on dramatic works which had held that 'copying of nonliteral expression, if sufficiently extensive, has never been upheld as permissible copying; rather, it has always been viewed as copying of elements of an expression of creative originality.'[68] The Court held that the defendants had infringed the plaintiff's copyright by directly reproducing the menu-command structure. Similar infringements were found in the *Softklone*[69] and *Manufacturing Technologies*[70] cases. It should be emphasized, however, that the copyright in the sequencing and structure of screen displays is completely different from the copyright in the sequencing and structure of the underlying program.[71]

In my view, this approach should be adopted by the Australian courts. User interfaces should be protected as compilations, but, applying the judgment of Jacobs J. in *Beck v. Montana Constructions Pty Ltd*,[72] they should be accorded narrow protection because they will contain elements which are common to all interfaces.

4.3 *Arguments Against Protecting User Interfaces*

As the *Lotus* case illustrates, there are several strong arguments against extending copyright protection to user interfaces. The main argument is that protecting interfaces effectively confers a monopoly over a type of application program on the developers of an innovative interface. The more user-friendly an interface is, the more popular it will be. Indeed, it may become a *de facto* standard.[73] This may result in a monopoly over the *application* because designers of competing software would be forced to use less-popular *user interfaces* to avoid infringement of the copyright in the interface. Therefore, it is argued that allowing competitors to take advantage of the market success of a program's user

[68] *Lotus Development Corporation v. Paperback Software International* 740 F. Supp. 37 (D. Mass. 1990), 52.

[69] *Digital Communications Associates, Inc. v. Softklone Distributing Corporation* 659 F. Supp. 449 (N.D. Ga. 1987).

[70] *Manufacturers Technologies, Inc. v. CAMS, Inc.* 706 F. Supp. 984 (D. Conn. 1989) 993.

[71] *Ibid.* 992; *Johnson Controls v. Phoenix Control Systems* 886 F. 2d 1173 (9th Cir. 1989), 1175; *cf.* Hunter, D., 'Protecting the "Look and Feel" of Computer Software in the United States and Australia' (1991) 7 *Santa Clara Computer and High Technology Law Journal* 95, 121.

[72] (1963) 5 F.L.R. 298.

[73] The fact that the interface of Lotus 1-2-3 had become such a *de facto* standard led Paperback Software to replicate it: *Lotus Development Corporation v. Paperback Software International* 740 F. Supp. 37 (D. Mass. 1990), 69.

interface is preferable to granting an arguably undeserved monopoly over a type of application.[74] However, the Court in the *Lotus* decision denied that protecting user interfaces conferred such a monopoly, pointing to the commercial success of another spreadsheet package with a substantially different interface.[75]

A related argument is based on commercial necessity. In *Lotus*, the defendant claimed that the only way to compete effectively against the commercial success of Lotus was to make a compatible product.[76] The Court rejected this argument because the defendant also had the options of selling it as an 'add-in' product,[77] of writing a routine to convert Lotus commands to whichever structure Paperback Software adopted, and of licensing the interface from Lotus.[78]

Another argument raised is that, since standardization of interfaces would save training time and improve productivity in the computer industry, the public interest in promoting standardized interfaces should prevail over the interface designer's right to remuneration.[79] The defendant in *Lotus* pleaded standardization as a defence to the infringement, but this was rejected on the basis that there was no room for such a defence under the United States Copyright Act.[80] Lewis argues, however, that the Court failed to address this argument adequately because it relates to the innovation and public interest goals of copyright law.[81] However, within the constraints of the United States Copyright Act, it is difficult to see how the Court could have considered these policies more carefully and given them more weight than it did.[82] Nevertheless, it has been argued that instead of rejecting standardization as playing any role in the decisions, the Court should have used it as a factor without disturbing the idea-expression analysis. This could have been done by the Court not defining the idea of the program as being a spreadsheet, but as a saleable, marketable spreadsheet. In this context, as there may have been no other system available that would satisfy the idea, the interface would not attract copyright protection because it would lack an original expression which was separate from the idea.[83] This argument, however, is susceptible to the criticisms the *Lotus* Court made against other arguments for standardization — it turns the concept of copyright upside down because it

[74] Barkume, *op. cit.* n. 34, 583. It is also argued that the interface warrants little protection because it is largely 'cosmetic' (Smith, *op. cit.* n. 34, 258). Such suggestions, however, cannot be regarded seriously in light of the major effort devoted to user interfaces, the impact they have on program popularity, and the effort spent in protecting them.

[75] *Lotus Development Corporation v. Paperback Software International* 740 F. Supp. 37 (D. Mass. 1990), 78.

[76] *Ibid.* 69.

[77] *I.e.* a program which would run concurrently with, and provide additional features to, Lotus 1-2-3.

[78] 740 F. Supp. 37 (D. Mass. 1990), 69.

[79] Stern, R., 'The Centre will not Hold — Recent U.S. Developments in Protecting "Idea" Aspects of Computer Software' [1987] 5 *European Intellectual Property Review* 125, 129; Rudnick, *op. cit.* n. 34, 249.

[80] 740 F. Supp. 37 (D. Mass. 1990), 69, 79.

[81] Lewis, G., '*Lotus Development Corporation v. Paperback Software International*: Broad Copyright Protection for User Interfaces Ignores the Software Industry's Trend Toward Standardization' (1991) 52 *University of Pittsburgh Law Review* 689, 713.

[82] *Lotus Development Corporation v. Paperback Software International* 740 F. Supp. 37 (D. Mass. 1990). 52-3, 57, 77-9. For further discussion of the weakness of the standardization argument, see Hunter, *op. cit.* n. 71, 105.

[83] Lewis, *op. cit.* n. 81, 705.

denies protection for more creative interfaces.[84] Such an abuse of copyright law would change the weight of values reflected in the current history of copyright law, and should not be undertaken by the courts. The argument also fails to address the denial made by the *Lotus* Court that reproduction of the user interface was not the only way to create a saleable, marketable spreadsheet because the alternatives of add-in software, licensing the interface and conversion utilities were possible.[85]

Lastly, the defendant in *Lotus* alleged that advances in technological fields are incremental, with subsequent program versions improving slightly on those already released. The defendant argued that copyright protection would unduly restrict this development. However, the Court rejected this argument because, notwithstanding copyright protection of a program's *expression*, subsequent developers are free to borrow the *ideas* of existing programs.[86]

4.4 *Protection of User Interfaces Under the Copyright Act 1968 (Cth)*

It is likely that protection of user interfaces can be achieved under the Australian Copyright Act 1968 (Cth) as it currently stands. It may be argued firstly that copyright protection of the interface is inherent in the protection of the underlying program, although this is not appealing from either a legal or industry standpoint.[87] This approach, however, is unlikely to succeed, given that the High Court has held that in order to constitute an infringing reproduction there must be objective similarity between the infringing work and the work protected.[88] If the work protected is the code, a court is unlikely to accept that there is any objective similarity between an interface and the code.[89]

It is possible, however, that a program which recreates the user interface of another program may be held to be an adaptation of the protected program.[90] Sub-section 10(1)(b)(a) of the Copyright Act 1968 (Cth) defines an adaptation of a computer program as 'a version of the work . . . not being a reproduction of the work'. The fact that the definition of adaptation excludes reproductions of the work means that 'adaptation must be something more of a transformation, like the turning of words into pictures. . . . [T]he new definition of "adaptation" of computer programs, by excluding reproductions, apparently leaves only "look and feel" versions of a computer program [*i.e.* those with a similar user interface] within the definition.'[91]

[84] *Lotus Development Corporation v. Paperback Software International* 740 F. Supp. 37 (D. Mass. 1990), 79.

[85] *Ibid.* 69.

[86] *Ibid.* 78.

[87] Although there has been no consideration of the issue at trial, this is the approach adopted by the only two Commonwealth decisions which have considered this question. In the interlocutory decisions of *F & I Retail Systems Ltd v. Thermo-Guard Automotive Products Canada Ltd* (1984) 1 Canadian Patent Reporter (3d) 297 and *Gemologists International Inc v. Gem Scan International Inc.* (1986) 7 Canadian Intellectual Property Reports 225 the Ontario High Court assumed, without stating reasons, that screen interfaces were protected by copyright in the underlying code.

[88] *Computer Edge Pty Ltd and Another v. Apple Computer Inc. and Another* (1986) 161 C.L.R. 171.

[89] Smith, *op. cit.* n. 34, 259.

[90] *Ibid.*

[91] Knight, P., 'Current Issues in Copyright and Computer Technology' (1989) 7:2 *Copyright Reporter* 3, 9; *Computer Software Protection*, Issues Paper, April 1990, Copyright Law Review Committee 13; Knight, P., '*Autodesk*: An Alternative View' (1990) 8:2 *Copyright Reporter* 12, 21.

However, the better argument is that interfaces attract copyright protection in their own right either as artistic works, or as literary works, being a compilation of literary or artistic elements.[92] It has been argued that user interfaces may not warrant protection under these categories because they are analogous to blank forms, which have been held not to attract copyright.[93] However, it was held in *Kalamazoo (Aust.) Pty Ltd v. Compact Business Systems*[94] that a compilation of accounting forms was sufficiently original to support the subsistence of copyright. Since interfaces of even simple application packages are inherently more complex and convey more information than forms, copyright protection should not be denied on this ground.[95]

A more difficult hurdle to overcome before user interface screens may attract protection is the requirement of material form. However, it is suggested that the Australian courts should follow the United States courts and decide that a screen display is in a material form if it is embodied within stored object code from which the screen may be reproduced.[96] In my view, this approach accords with the Australian statutory definition of 'material form' in sub-s. 10(1) of the Copyright Act 1968 (Cth) of 'any form of storage from which the work or adaptation . . . can be reproduced.'

Equitable solutions will generally be reached if standard copyright principles are applied to computer interfaces. The *elements* of any interface (things such as key assignments, pull-down menu structures, the use of icons, *etc.*) will not attract copyright protection because they lack originality or are too insubstantial. However, the specific compilation of these facets of any interface may attract protection in the way that other compilations of factual information do.[97] It is generally recognized that the creation of a user interface is more difficult, and requires greater creativity, than the reduction of that interface to code. To protect only the less creative encoding of the work would be, it is submitted, fundamentally inconsistent with the objectives of copyright.[98]

The protection afforded to compilations must necessarily be narrow to ensure that the elements of the compilation are not protected. Applying these concepts to user interfaces, no software house could gain a monopoly over any menu structure or menu terms, but literal recreation of a user interface could be prevented.

4.5 *The Ramifications of Extending Protection to User Interfaces*

There is some concern that extending copyright protection to user interfaces will stifle innovation in the computer industry. There are two aspects to this issue: whether creativity in the development of interfaces themselves will be

92 *Computer Software Protection*, Issues Paper, April 1990, Copyright Law Review Committee 13.
93 Smith, *op. cit.* n. 34, 260.
94 (1985) 5 I.P.R. 213 (Supreme Court of Queensland).
95 Hunter, *op. cit.* n. 71, 140.
96 *Stern Electronics Inc. v. Kaufman* 669 F. 2d 852 (1982).
97 Bixby, M., 'Synthesis and Originality in Computer Screen Display and User Interfaces: The "Look and Feel" Cases' (1991) 27 *Willamette Law Review* 31, 49.
98 *Cf. Lotus Development Corporation v. Paperback Software International* 740 F. Supp. 37 (D. Mass. 1990), 56.

stifled, and whether the granting of proprietary rights in an interface will curb development of new techniques in the underlying program which utilizes the interface.

As to the first question, it has been urged that without copyright protection of user interfaces, companies which develop market dominating screen formats would be under constant pressure to improve them, or lose their market share to rivals copying the interfaces, and that this development would be beneficial to society.[99] This argument does not answer the fact that allowing the free use of interfaces diverts resources from those who have demonstrated that they are prepared to undertake research and development, and reduces the incentive of companies to undertake this development of improved user interfaces. Further, such an approach would make the development of new interfaces the sole province of large software houses with extensive resources. If an individual did develop an innovative interface, the larger software companies could reproduce it fairly quickly and rapidly market it on a much wider scale than the individual could hope to compete with.

The development of interfaces is far more likely to suffer without copyright protection than with it because there would be no incentive to develop. Indeed, the very purpose of copyright is to provide an incentive for the development of new ideas by granting limited rights over the expression of those ideas.[100] Those who invest intellectual effort in the development of user interfaces are no less deserving of copyright protection (and remuneration) than those who gain copyright protection for any written text.

With regards to the second question, however, a different perspective is necessary. Copyright over user interfaces is unique, in that the same interface may be used for several programs that perform their underlying computations in markedly differing ways. Each program must use an interface of some sort. While each of these underlying programs warrants copyright protection, the only expression that the ordinary user will see is that of the interface, and it is, generally, on this interface that a decision to use the program or not will be based. As was pointed out above, preventing the reproduction of user interfaces could lead to the creation of *de facto* monopolies in any particular software application, due to public refusal to accept variations of the command structure used by the market leader. Thus, there is a very real danger that granting copyright protection over user interfaces could stifle development not in the interfaces themselves, but in the underlying area of applications. Whoever controls the copyright in an interface which has become popularly accepted for an application also controls the market for that application, at least until a more powerful interface is developed. Therefore, I believe that such interfaces should be the subject of compulsory licences.

Compulsory licences would not deprive the developer of the interface of the fruits of his or her efforts in creating and marketing the interface. Yet at the same time they would allow competition in the market for that application, and would

[99] Spivack, *op. cit.* n. 21, 754.
[100] Ricketson, S., *The Law of Intellectual Property* (1984) 7; Hunter, *op. cit.* n. 71, 104.

not prevent the development of improved ideas and techniques in the substantive area of the application. As Osborne states, 'The software developer needs the freedom to use the interface of another program, just as a watchmaker might borrow the concept of a watch's face while revolutionizing its clockworks.'[101] With a compulsory licence, those who develop an improved technique (*e.g.* for handling spreadsheet calculations) would not be prevented from effectively distributing their improved software simply because their development was not in the user interface, but in the 'substantive' area of the program.

Compulsory licences are generally unpopular in Berne Convention countries because they are contrary to the accepted basis of copyright that the rights conferred are the author's exclusive rights to dispose of as he or she wishes.[102] However, a policy decision must be made, and, in light of the strong arguments both for copyright protection and allowing free use of user interfaces, I believe that a compulsory licensing scheme is the only equitable compromise.

5. *CONCLUSION*

With the growing intrusion of computers into our everyday lives, a clear legal position must be adopted regarding the intellectual property rights which attach to software. Australia has not yet addressed these issues in the courts, but I believe that the current statutory framework is adequate to reach sensible results. The courts must follow the American decisions and realize that the one program supports two separate copyrights: one in the code and another in the user interface. A realization that the ideas of the interface and the underlying program are separate will lead to rational decisions which accurately reflect the realities of the software development industry.

Further, it should be realized that the protection of both these expressions extends beyond literal copying to protection of the compilation of the structural elements of each. Due to the functional nature of computer programs, however, this protection should be defined quite narrowly. The protection of interfaces, especially, should not extend greatly beyond literal copying, so as to avoid giving an undeserved monopoly in a type of interface.

In order to allow competition and prevent the creation of *de facto* monopolies, the Copyright Act 1968 (Cth) should be amended to regulate the compulsory licencing of user interfaces. This will allow developers with new ideas in the underlying application to effectively market them in a form which is acceptable to the public. Any other solution would either unjustly deprive the interface developer of his or her work in creating and marketing a popular interface, or prevent the distribution of valuable ideas.

[101] Osborne, A., (former head of Paperback Software) quoted in 'A Divisive Lotus "Clone" War', *New York Times*, 5 February 1987 as cited in Lewis, *op. cit.* n. 81, 696 (n. 21).

[102] Dworkin, G., 'The Concept of Reverse Engineering in Intellectual Property Law and its Application to Computer Programs' (1990) 1 *Australian Intellectual Property Journal* 164, 176.