# TRASH TALKING: THE PROTECTION OF INTELLECTUAL PROPERTY RIGHTS IN COMPUTER SOFTWARE

*Michael F. Morgan**

*Since the early 1980s the question of how to protect intellectual property rights in computer software has been the subject of intense debate. Early in this debate some academics suggested that computer software would be best protected under* sui generis *legislation. However, it has now become apparent that copyright will be the primary legal mechanism for the protection of intellectual property rights in computer software. In the United States a number of Circuit Courts have considered the scope of protection to be provided under copyright and come to different conclusions. Some of these decisions threaten to extend a* de facto *monopoly over key new technologies. Other decisions have sought to balance the need to reward developers with the need to provide access to these new technologies. It now appears that in the United States this less protectionist view may prevail.*

*While the question of scope has arisen in Canadian courts there has yet to be a detailed analysis of the type of protection that should be provided under the Canadian* Copyright Act*. An examination of Anglo-Canadian jurisprudence reveals that there has been a tendency to support strong copyright protection. However, this paper will suggest that strong copyright protection of computer software may impede technological innovation, and threaten the development of a strong*

*Depuis le début des années 1980, la question de savoir comment protéger les droits de propriété intellectuelle relatifs aux logiciels a fait l'objet d'un vif débat. Peu après le commencement de ce débat, des universitaires ont émis l'opinion que les logiciels seraient mieux protégés par une loi* sui generis*. Cependant, il est maintenant évident que le droit d'auteur sera le principal mécanisme juridique de protection des droits de propriété intellectuelle relatifs aux logiciels. Aux États-Unis, plusieurs* Circuit Courts *ont examiné la portée de la protection accordée par le droit d'auteur et sont arrivées à des conclusions différentes. Certaines décisions rendues par ces tribunaux menacent d'étendre un monopole* de facto *à de nouvelles technologies clés. D'autres décisions ont cherché à créer un équilibre entre la nécessité de récompenser les créateurs et la nécessité d'avoir accès à ces nouvelles technologies. Il semble maintenant que cette vision moins protectionniste pourrait prévaloir aux États-Unis.*

*Bien que la question de la portée de cette protection ait été soulevée devant les tribunaux canadiens, il reste à faire une analyse détaillée de la protection qui devrait être accordée par la loi canadienne sur le droit d'auteur et la jurisprudence afférente. Un examen de la jurisprudence anglo-canadienne révèle qu'on a eu tendance à favoriser une solide protection*

* Michael Morgan is a recent graduate of the University of Ottawa Law School. He also has a B.Sc. and M. Math in computer science and has worked as a member of scientific staff at Bell-Northern Research in Ottawa. The author would like to thank Glen Bloom of Osler, Hoskin & Harcourt for his helpful suggestions and criticisms. Any errors which remain are the result of the author's views prevailing.

This article was written before the British Columbia Supreme Court's decision in *Prism Hospital Software Inc.* v. *Hospital Medical Records Institute*, and the Ninth Circuit Court of Appeal's decision in *Apple Computer Inc.* v. *Microsoft Corp.*

"Trash Talking" has the distinction of being the prize-winning piece in the first annual Fasken Campbell Godfrey / OTTAWA LAW REVIEW student writing competition. The Toronto law firm Fasken Campbell Godfrey has generously underwritten the $500 prize with the hopes of encouraging the production of publishable works by University of Ottawa law students. On the occasion of the 25th anniversary of the OTTAWA LAW REVIEW, the Senior Board of the Review founded an annual writing competition for all students enrolled in the Common and Civil law sections of the Faculty. The establishment of two prizes, one in each official language, resulted from the idea of promoting scholarship in both French and English. Upon completion of the prize committee's deliberation, Mr. Morgan's article was chosen as winner of the English prize. Following subsequent consideration by the Senior Board of the Review, the winning submission was accepted for publication.

*software industry. This paper will also suggest that Canadian legislation and jurisprudence can support a less protectionist approach, and furthermore, that such an approach can both reward creators and preserve public access to key technologies.*

*par le droit d'auteur. Cependant, l'auteur de cet article estime qu'une solide protection par le droit d'auteur pourrait entraver l'innovation technologique et menacer l'essor d'une forte industrie du logiciel. En outre, l'auteur est d'avis que la législation et la jurisprudence canadiennes peuvent favoriser une approche moins protectionniste et que, par ailleurs, cette approche peut à la fois récompenser les créateurs et préserver l'accès du public aux technologies clés.*

TABLE OF CONTENTS

I don't want to end up with this case saying, "Oh, well, all we have to do is change a trash can or change this or that." That isn't what this case is involving. We didn't bring a small case; we brought a case of tremendous significance to the client and to the industry. It's virtually the most important – certainly the most important case I have ever tried – but it's important because of the impact, the importance of this intellectual property asset of Apple's that they created and they owned and the totality of that expression.

Counsel, Apple Computer, Inc.[1]

A simple comparison of the "Waste Basket" icons used in NewWave 3.0 and NewWave 1.0 reveals that each icon is dissimilar from the "Trash" icons used in the Macintosh and Lisa works. The "Trash" icon in Macintosh Finder is a two-dimensional side view of an outdoor alley-style cylindrical garbage can. The depiction includes a set of four vertical lines to simulate a ribbed or fluted surface, and a closed lid with a handle on top. The Lisa Desktop "Waste Basket" icon shows the same view of an alley-style garbage can, also with vertical lines to indicate a ribbed surface, but with an open lid and a handle on the front of the can.

In contrast, the "Waste Basket" icons in NewWave 3.0 and 1.0 are perspective views of an indoor kitchen-style rectangular flip-top container.

District Judge, Vaughn Walker[2]

## I. Introduction

The fact that it has taken five years of litigation and 30 million dollars in legal expenses[3] to determine, pending appeal, that two iconic representations of a garbage can are not similar for the purposes of copyright law suggests that the current legal regime is not fulfilling its mandate.[4] This conclusion is further supported by the increasing number of conflicting decisions which have been rendered in the various American Circuit Courts.[5] The lack of judicial consensus and the wide scope of the protection

[1] *Apple Computer, Inc.* v. *Microsoft Corp.*, 821 F.Supp. 616 at 624 (N.D. Cal. 1993) [hereinafter *Apple* v. *Microsoft*].

[2] *Ibid.* at 622.

[3] V. Slind-Flor, "Computer War Comes Down To Mere Pixels" *National Law Journal* vol. 15 no. 38 (24 May 1993) 8.

[4] The industry view of this litigation is reflected in J. Morrissey, "Copyright Suit Enters Fifth Year With No End in Sight" *PC Week* vol. 9 no. 33 (17 August 1992) S5.

Question: What Windows-related development has been under way for more than four years, cost millions of dollars and produced nothing more than reams and reams of documentation?

Answer: Apple Computer Inc.'s copyright-infringement lawsuit against Microsoft Corp. and Hewlett-Packard Co. (HP).

[5] See *e.g.* *Whelan Associates* v. *Jaslow Dental Laboratory*, 797 F.2d 1222 (3rd Cir. 1986) [hereinafter *Whelan*]; *Broderbund Software Inc.* v. *Unison World Inc.*, 648 F.Supp. 1127 (N.D. Cal. 1986) [hereinafter *Broderbund*]; *Plains Cotton Cooperative Association* v. *Goodpasture Computer Services, Inc.*, 807 F.2d 1256 (5th Cir. 1987); *Digital Communications Associates, Inc.* v. *Softklone Distributing Corp.*, 659 F.Supp. 449 (N.D. Ga. 1987) [hereinafter *Softklone*]; *Manufacturers Technologies Inc.* v. *Cams, Inc.*, 706 F.Supp. 984 (D. Conn. 1989); *Lotus Development Corporation* v. *Paperback Software International*, 740 F.Supp. 37 (D. Mass. 1990) [hereinafter *Lotus* v. *Paperback*];

granted in some of the earlier cases has caused concern in the software industry.[6] The perceived shortcomings of the current legal regime have been hotly debated in the academic literature.[7] Some academics have suggested that the best way of protecting computer software is through the development of *sui generis* legislation.[8] However, the

*Lotus Development Corporation* v. *Borland International Inc.*, 799 F.Supp. 203 (D. Mass. 1992) [hereinafter *Lotus* v. *Borland*]; *Computer Associates International, Inc.* v. *Altai, Inc.*, 982 F.2d 693 (2nd Cir. 1992) [hereinafter *Computer Associates*]; *Atari Games Corporation* v. *Nintendo of America Inc.*, 975 F.2d 832 (Fed. Cir. 1992) [hereinafter *Atari Games*]; *Sega Enterprises Ltd.* v. *Accolade, Inc.*, 977 F.2d 1510 (9th Cir. 1992) [hereinafter *Sega Enterprises*]; and *Apple* v. *Microsoft*, *supra* note 1.

[6] M.A. Goetz, "When Copyright Goes Wrong" *Datamation* vol. 38 no. 17 (15 August 1992) 110; E. Harding, "Unix Pioneer Ends BSD Research; UCal Berkeley Blames Lawsuit by USL" *Software Magazine* vol. 12 no. 14 (October 1992) 21; G. Groenewold, "Too Clever By Half?" *Unix Review* vol. 11 no. 5 (May 1993) 59; G. Buchan, "Protecting Competition" *Computing Canada* vol. 18 no. 24 (23 November 1992) 13; P. Samuelson, M. Denber & R.J. Glushko, "Developments on the Intellectual Property Front" *Comm. of the ACM* vol. 35 no. 6 (June 1992) 33; R. Green, "Apple's Look-and-Feel Legal Blues Make You Wanna Twist and Shout" *Government Computer News* vol. 11 no. 10 (11 May 1992) 61; S. Gibson, "Lotus Ruling Will Damage the Industry if it Protects Languages" *Infoworld* vol. 13 no. 37 (14 September 1992) 42; C. Van Brussel, "Borland-Lotus Suit Stirs Up Copyright Issue" *Computing Canada* vol. 18 no. 18 (1 September 1992) 1; P.S. Flanagan, "Lawsuit Mania!" *Computer Shopper* vol. 13 no. 2 (February 1993) 834; P. Samuelson, "The Ups and Downs of Look and Feel" *Comm. of the ACM* vol. 36 no. 4 (April 1993) 29.

[7] The major issue being debated in the academic journals is the scope of protection which should be provided under copyright. Wide copyright protection has been advocated by A.L. Clapes, P. Lynch & M.R. Steinberg, "Silicon Epics and Binary Bards: Determining the Proper Scope of Copyright Protection for Computer Programs" (1987) 34 U.C.L.A. L. Rev. 1493; C.A. Sundholm, "High Technology Jurisprudence: In Defence of 'Look and Feel' Approaches to Copyright Protection" (1992) 8 Computer & High Tech. L.J. 209; S.A. Dunn, "Defining the Scope of Copyright Protection for Computer Software" (1986) 38 Stan. L. Rev. 497; S.R. Englund, "Note, Idea, Process or Protected Expression: Determining the Scope of Copyright Protection of the Structure of Computer Programs" (1990) 88 Mich. L. Rev. 866; J. Sholkoff, "Breaking the Mold: Forging a New and Comprehensive Standard for Protection for Computer Software" (1988) 8 Computer L.J. 389; and A.L. Clapes & J.M. Daniels, "Revenge of the Luddites: A Closer Look at Computer Associates v. Altai" (1992) 9 The Computer Lawyer 11. The following articles have advocated a narrower scope of protection: P.S. Menell, "An Analysis of the Scope of Copyright Protection for Application Programs" (1989) 41 Stan. L. Rev. 1045; W.G. Duflock, "'Look and Feel': A Proposed Solution to the Diverging Views Between the Software Industry and the Courts" (1992) 8 Santa Clara Computer & High Tech. L.J. 447; K.S. Kovach, "Computer Software Design: User Interface – Idea or Expression?" (1991) 60 U. of Cin. L. Rev. 161; D.S. Karjala, "Copyright, Computer Software, and the New Protectionism" (1987) 28 Jurimetrics J. 33.

[8] P. Samuelson, "Creating a New Kind of Intellectual Property: Applying the Lessons of the Chip Law in Computer Programs" (1985) 70 Minn. L. Rev. 471; R.H. Stern, "The Bundle of Rights Suited to New Technology" (1986) 47 U. Pitt. L. Rev. 1229; and D.S. Karjala, "Lessons from the Computer Software Protection Debate in Japan" (1984) Ariz. St. L.J. 53. The Karjala paper provides a description of Japanese proposals for *sui generis* legislation. These proposals were never implemented. The WIPO "Model Provisions on the Protection of Computer Software" offered certain minimum provisions which constituted a *sui generis* system. For arguments against *sui generis* legislation, see L.J. Raskind, "The Uncertain Case for Special Legislation Protecting Computer Software" (1986) 47 U. Pitt. L. Rev. 1131.

weight of legislative change,[9] international agreements[10] and jurisprudential opinion[11] has made it clear that copyright will be the primary legal mechanism for the protection of intellectual property rights in computer software. This choice is not without its problems: the low threshold[12] and long period of protection[13] are features of copyright which make it less than ideal for protecting works of technology such as computer software. However, given that this is the framework within which protection is to be provided, the goal of the legislatures and courts should be to tailor the scope of protection so that public access to innovative new works is maximized, while a level of individual incentive which ensures a continuing supply of new works is maintained.

The major advantage of copyright as a legal means for protecting computer software is its inherent flexibility. Copyright has existed in some form for hundreds of years[14] and over that period of time it has had to adapt to the introduction of a number of new technologies.[15] The fact that computer software may be treated differently than other

---

[9] *Copyright Act*, R.S.C. 1985, c. 10 (4th Supp.). The definition of "computer program" was added (s. 1(3)), and the definition of literary work was amended to include computer programs (s. 1(2)).

[10] Article 2 of the *Berne Convention* (Rome Revision) provides that:

> The expression "literary and artistic works" shall include any production in the literary, scientific or artistic domain, whatever may be the mode or form of its reproduction....

This general description is followed by a non-exhaustive list of such works. It has generally been accepted that computer programs are writings for the purpose of the *Berne Convention.*

[11] *Apple Computer, Inc.* v. *Mackintosh Computers Ltd.* (1986), [1987] 1 F.C. 173, 10 C.P.R. (3d) 1 (F.C.T.D.), aff'd (1987), [1988] 1 F.C. 673, 18 C.P.R. (3d) 129 (Fed. C.A.), aff'd, [1990] 2 S.C.R. 209, 30 C.P.R. (3d) 257 [hereinafter *Apple* v. *Mackintosh* (trial decision) cited to 1 F.C.]; *F & I Retail Systems Ltd.* v. *Thermo-Guard Automotive Products of Canada Ltd.* (1984), 1 C.P.R. (3d) 297 (Ont. H.C.J.) [hereinafter *F & I Retail Systems*]; *Gemologists International Inc.* v. *Gem Scan International Inc.* (1986), 9 C.P.R. (3d) 255, 7 C.I.P.R. 253 (Ont. H.C.J.) [hereinafter *Gemologists International* cited to C.P.R.]; *Delrina Corp.* v. *Triolet Systems Inc.* (1993), 47 C.P.R. (3d) 1, 9 B.L.R. (2d) 140 (Ont. Gen. Div.) [hereinafter *Delrina*]; *Titan Linkabit Corp.* v. *S.E.E. See Electronic Engineering Inc.* (1993), 48 C.P.R. (3d) 62 (F.C.T.D.). See also *supra* note 5.

[12] *Copyright Act*, R.S.C. 1985, c. C-42, s. 5. This section outlines the four requirements for copyright: originality, citizenship, subject matter and authorship. *University of London Press Ltd.* v. *University Tutorial Press Ltd.*, [1916] 2 Ch. 601 at 608, 86 L.J. Ch. 107 at 111 [hereinafter *University of London Press* cited to Ch.], further specifies the originality requirement:

> The word "original" does not in this connection mean that the work must be the expression of original or inventive thought. Copyright Acts are not concerned with the originality of ideas, but with the expression of thought, and, in the case of "literary work", with the expression of thought in print or writing. The originality which is required relates to the expression of the thought. But the Act does not require that the expression must be in an original or novel form, but that the work must not be copied from another work – it should originate from the author.

[13] *Ibid.* at s. 6, provides that:

> The term for which copyright shall subsist shall, except as otherwise expressly provided by this Act, be the life of the author and a period of fifty years after the death of the author.

[14] J. Lahore, *Intellectual Property in Australia* (Sydney: Butterworths, 1977) at 22. The first copyright Act, *The Statute of Anne*, was passed in 1709.

[15] Broadcasting, cinematographic films, and cable rediffusion are but a few examples.

utilitarian works[16] is not an incorrect application of copyright doctrine. It would, however, be incorrect to treat software the same as other literary works simply because it bears a superficial resemblance to these works.[17]

To date, the bulk of software copyright litigation has been in the United States. While most of this type of litigation will remain in the United States, the issue of copyright protection of computer software has arisen in Canadian courts[18] and it is not unreasonable to expect that there will be a growing number of such cases. In dealing with the issues that these cases raise, Canadian courts and legislatures will face two challenges. The first will be to avoid the tendency to apply American decisions, based on the American statute and American case law, without consideration of the Canadian statute and Canadian case law.[19] The second will be to determine Canadian needs and goals with respect to the protection of computer software.[20] The first challenge is primarily legal in that it concerns the proper application of Canadian law by the courts. The second challenge is more policy-oriented in that it concerns the overall direction that Parliament should take in drafting a Canadian law for protection of computer software. This paper will attempt to analyze the first issue by examining the treatment that Canadian courts have given to copyright protection of computer software thus far. Where a particular question has not come before the Canadian courts, a suggested treatment under existing law will be given. Analysis of the second issue will focus on

---

[16] For example, maps, charts, and schematic drawings.

[17] Some have argued for strong protection of computer software by analogizing programs to other literary works: for example, Clapes, Lynch & Steinberg, *supra* note 7 at 1524-25:

> A program is a literary work. It is a particular kind of literary work, to be sure, as is a musical composition or a "shooting script" for a movie, but one that has attributes in common with more familiar kinds of literary works. Those attributes are structure, flow, logic, design, naming conventions, commentary, and resultant style.
> Large programs are usually organized into sections that are called modules. Typically, each module will be devoted to one of the major capabilities of the program. Modules are roughly equivalent to chapters in a book. A module may be broken down into recognizable submodules, just as a chapter may have subchapters or paragraphs (footnotes omitted).

It is certainly possible to make these analogies, but their value in terms of understanding computer software is questionable. On a sunny day it is possible to look at a cloud and imagine that it looks like a rabbit or an elephant; however, this does not help one understand cloud composition or the physics of cloud formation. What is needed is a thorough understanding of the science of software engineering and the economics of the computer software industry.

[18] *Supra* note 11.

[19] *Compo Co. Ltd.* v. *Blue Crest Music Inc.* (1979), [1980] 1 S.C.R. 357, (1979) 545 C.P.R. (2d) 1. In this case, Estey J. commented on American decisions that interpret American copyright laws and their applicability to problems arising under the Canadian *Act*. Estey noted that while there exist points of similarity, there also exist fundamental differences. It is already possible to see some evidence of the unquestioned application of American jurisprudential concepts in Canadian software copyright litigation. See *Delrina, supra* note 11.

[20] D. Vaver, "Copyright in Foreign Works: Canada's International Obligations" (1987) 66 Can. Bar Rev. 76 at 79, points out that the United States is a net exporter of copyrighted material while Canada is a net importer of copyrighted material. While it is in the interest of the United States to provide strong intellectual property protection for copyrighted material, the same may not be true for Canada.

the creation of a policy which best suits the development of a vigorous software industry in Canada. As part of this analysis, proposals will be made with respect to the treatment of those issues that are currently viewed as the most problematic.

Current Canadian copyright law seems to favour strong protection of intellectual property rights in computer software. From a policy perspective this may not be the best choice. The *Copyright Act* has traditionally had two goals: to encourage the creation of works for the "advancement of learning", and to protect and reward the intellectual effort of the author (for a limited period of time) in the work.[21] Historically, copyright has

[21] *Apple* v. *Mackintosh*, *supra* note 11 at 213 (per Reed J.). J. Lahore, in J. Lahore, G. Dworkin & Y.M. Smyth, *Information Technology: The Challenge to Copyright* (London: Sweet & Maxwell, 1984) at 3, takes the view that:

> The rights of the copyright owner have never been absolute. The purpose of copyright law is to reward the author, to encourage the development of cultural, scientific and educational works, and to encourage dissemination and exploitation of this material by protecting the investments of those, such as publishers, who enable this dissemination and exploitation to take place.

H.G. Fox, *The Canadian Law of Copyright and Industrial Designs*, 2d ed. (Toronto: Carswell, 1967) at 2 states:

> The primary object in conferring the monopoly lies in the general benefits derived by the public from the labours of authors; it is the equivalent given by the public for the benefits bestowed by the genius and meditation and skill of individuals and the incentive to further efforts for the same important objects (footnotes omitted).

Fox cites *Millar* v. *Taylor* (1769), 4 Burr. 2303 at 2335 as authority. The cases of *Donaldson* v. *Beckett* (1774), 4 Burr. 2408, 1 E.R. 837 (H.L.) and *Jeffreys* v. *Boosey*, (1854), 4 H.L.C. 815, 10 E.R. 681 (H.L.) also support this proposition.

A contrary view has been expressed by McLachlin J. in *Bishop* v. *Stevens*, [1990] 2 S.C.R. 467 at 478-479, 31 C.P.R. (3d) 394:

> [The] distinction between the right to perform and the right to record a work is unsurprising in light of the object and purpose of the Act. As noted by Maugham J., in *Performing Right Society, Ltd. v. Hammond's Bradford Brewery Co.*, "the *Copyright Act*, 1911, was passed with a single object, namely, the benefit of authors of all kinds, whether the works were literary, dramatic or musical...." (citations omitted).

The passage quoted by McLachlin J. is that of a judge of the English Chancery Division. *Donaldson* v. *Beckett* and *Jeffreys* v. *Boosey* are decisions of the House of Lords. The major copyright texts refer to *Performing Right Society* v. *Hammond's* only on the issue of public versus private performance. Given these factors, it seems unlikely that this case can stand for a proposition which alters the philosophical underpinnings of copyright law. In *Bishop* v. *Stevens*, McLachlin J. seems to have taken the comments of Maugham J. out of context. Maugham J.'s comments were in relation to the 1911 *Copyright Act*. This *Act* did extend significant new rights to authors; however, this was done in reaction to an imbalance which had developed because of the invention of new technologies which allowed greater exploitation of existing works. While the changes in the 1911 *Act* may have been solely for the benefit of authors, the overall purpose of copyright remained the two-fold goal of rewarding authors and advancing learning.

The proper identification of the purpose of copyright is extremely important, because this will affect the way in which all statutory provisions are to be interpreted. In the rest of this paper it will be assumed that the purposes stated by Reed J. in *Apple* v. *Mackintosh* are the correct ones.

sought to balance these two purposes; however, the unmodified application of the traditional literary standard of copyright protection to computer software will skew this balance in favour of authors, to the detriment of the general public. This paper will suggest that it is possible to implement a less protectionist scheme that still provides the incentives that authors need in order to create new works. The key to developing such a law is the understanding that software is different from the traditional literary works currently covered by copyright. This difference arises out of the fact that software is technological and utilitarian.

While it is important to develop a Canadian solution to the problem of software copyright protection, the experience of other jurisdictions should be drawn upon, and, if found to be suitable for Canadian purposes, adopted. An example of how Canadian copyright law can benefit from foreign experience is the treatment of utilitarian works. Copyright law in the United States has recognized the principle that utilitarian works should receive only limited protection.[22] Historically, this distinction has been weak in Canadian copyright law, although that may now be changing.[23] This paper will take the position that this distinction is crucial to the development of a workable scheme of copyright protection for computer software.

Just as there are certain aspects of foreign copyright law which should be adopted in the development of Canadian copyright law, there are other aspects which should not be adopted. Two examples of principles which should not be adopted are the protection of non-literal elements of computer programs, and the protection of the "look and feel" of computer user interfaces.[24] The American cases have articulated the principle that copyright protection of computer programs extends beyond the literal instructions to the non-literal aspects of a computer program. This paper will attempt to show that this type of protection is overly broad and in fact beyond the type of protection which has traditionally been provided by copyright. It will also be suggested that the experience in the United States has demonstrated two points: first, as a practical matter, the courts are unable to determine effectively what is, or is not, protected under this principle;[25] and second, that the industry is unable to determine what is, or is not, copyright infringement under this principle.[26]

The development of a viable copyright protection scheme for computer software must start with an understanding of the unique nature of computer software and the way in which computer software is developed and marketed. The following section will

---

[22] *Baker* v. *Selden*, 101 U.S. 99 (1879); *Cooling Systems & Flexibles, Inc.* v. *Stuart Radiator, Inc.*, 777 F.2d 485 (9th Cir. 1985); and *Landsberg* v. *Scrabble Crossword Game Players, Inc.*, 736 F.2d 485 (9th Cir. 1984).

[23] *Bayliner Marine Corp.* v. *Doral Boats Ltd.* (1985), 5 C.P.R. (3d) 289, [1986] 3 F.C. 346 (F.C.T.D.), rev'd (1986), 10 C.P.R. (3d) 289, [1986] 3 F.C. 421 (Fed.C.A.) [hereinafter *Doral Boats*]; and *Spiro-Flex Industries Ltd.* v. *Progressive Sealing Inc.* (1986), 32 D.L.R. (4th) 201, 13 C.P.R. (3d) 311 (B.C.S.C.) [hereinafter *Spiro-Flex Industries* cited to D.L.R.]. But see *Act, supra* note 9, as amended, S.C. 1988, c. 10 adding s. 64.1. This section provides that the copying of the solely utilitarian features of a copyrighted work is not an infringement.

[24] Protection of "non-literal elements" refers to an extension of copyright to features such as modularity, flow of control, data structures, command sets, and menu structures. The "look and feel" doctrine is based on the principle that the whole is greater than the sum of the parts; even if all of the parts are not copyrightable, it is possible to have copyright in the "feel" or the "whole" of the work.

[25] Karjala, *supra* note 7 at 55.

[26] *Supra* note 6.

discuss the features of computer software which make it different from traditional literary works. Once this foundation has been established the paper will proceed with the legal analysis.

## II. The Nature of Computer Software

Computer software is the technology which causes computers to perform the complex and varied tasks that make them useful. As Professor Randell Davis of MIT stated, "Software is a 'machine' whose medium of construction happens to be 'text'".[27] While textual in form, computer software is inherently functional. Copyright has been used to protect other functional textual works;[28] however, computer software differs from these works in that its purpose is not to convey information, but to cause processes to be performed. A second feature of computer software which distinguishes it from other textual works is that in its object code format[29] it must be copied in order to be studied or used.[30] A final feature of computer software which distinguishes it from other works is the ease with which it can be copied.

The unique nature of computer software presents two fundamental problems for copyright. The first is that an overly protective scheme could grant patent-like protection without having met the stringent requirements of novelty and non-obviousness. The second is that unless reverse engineering (disassembling), and thus a certain degree of literal copying, is allowed, the public will be denied the opportunity to read the work and extract the ideas and processes embodied in it. The remainder of this section will focus on certain specific features of computer programs and how these features contribute to these two basic problems. The discussion will assume that the reader is familiar with basic computer and software engineering terminology.[31]

### A. *Behaviour*

Computer programs may appear, at first glance, to resemble other literary works. They differ, however, in the fundamental sense that they behave. While programs must

[27] R. Davis, "Intellectual Property and Software: The Assumptions Are Broken" in *WIPO Worldwide Symposium On The Intellectual Property Aspects Of Artificial Intelligence* (Geneva: World Intellectual Property Organization, 1991) 101 at 110 (conference proceedings).

[28] Copyright protects maps, charts, and schematic diagrams. However, the primary purpose of these works is to convey information. This purpose is fundamentally different from the purpose for which computer programs are used.

[29] This is the form of the work which is understood by computer hardware. When viewed by a human being, it looks like a long random sequence of 0's and 1's. It is possible for someone who is trained in the machine language of a particular processor to interpret these sequences. However, the task of hand disassembly is very difficult, time-consuming, and error prone. As a result, it is common to use another computer program to translate machine language into another level of language known as assembly code. Assembly code is an intermediate level language which is closely linked to the underlying processor, but still symbolic in form.

[30] This process is known as disassembling or decompiling. The latter term is a misnomer since the product of the reverse engineering is not the original high-level source code.

[31] For a good description of these concepts, see R. Davis, "The Nature of Software and its Consequences for Establishing and Evaluating Similarity" (1992) 5 Software L.J. 299; C. Walter, "Defining the Scope of Software Copyright Protection for Maximum Public Benefit" (1988) 14 Rutgers Comp. & Tech. L.J. 1.

exist in a textual format in order to be created and modified, they must also exist as an executing process in an active processor in order to perform the task for which they were initially created. Computer programs resemble literary works only in their formal representation. Because of this distinction, analogies to other literary works are of questionable value. Behaviour is a trait normally associated with machines, and as a textual work, software is unique in possessing this characteristic. The creation of software involves both authorship and invention.[32] Traditionally, there has been a legal dichotomy between authorship and invention: one has been the subject of copyright law, and the other has been the subject of patent law. The question of how to protect computer software blurs this distinction. As a result there are two potential problems: first, an overly protective copyright scheme could potentially grant patent-like protection over a program's functionality without first having met the rigorous requirements of novelty and non-obviousness. The second problem is that sole reliance on a patent scheme could result in the underprotection of the authorship aspects of computer software. Since the development of computer software involves both authorship and invention, it may be reasonable to expect that a legal regime for the protection of intellectual property rights in computer software should be based both on copyright and patent law. However, this paper will attempt to show that an effective scheme of protection can be developed without significant reliance on patent law.

## B. *Interoperability and Standardization*

Traditional literary works are valued for their variety: there is little social value in a clone of a popular novel or play.[33] However, variety is not a characteristic that is valued in software development; in fact, the opposite is true. The ability to interoperate with or conform to industry standards is a valuable characteristic of computer products. While strong copyright protection serves the goal of encouraging diversity in literary expression, it does not serve the goal of encouraging standardization and interoperation of computer products. The need for standardization spans the whole range of computer products: application software must interact with other application software, operating system software, and ultimately hardware.

Computer systems consist of layers which meet at interfaces.[34] It is essential that entities which are on the opposite sides of an interface understand the format in which they are expected to communicate with each other: this is the issue of compatibility. Compatibility becomes an important consideration when one manufacturer wants to design a product that interacts with an existing manufacturer's product. In order for the products to interact, it is necessary for the second manufacturer to adopt the interface protocol used by the first manufacturer. Such use has been alleged to be a violation of copyright.[35] The danger in allowing this type of copyright protection is that it can result

[32] Davis, *ibid.* at 315.

[33] Karjala, *supra* note 7 at 49.

[34] Some common interfaces are: the machine-operating system interface; the application program-operating system interface; and the human-machine interface (often called the user interface). Another type of interface which is becoming increasingly important is the communications interface. This is the interface that allows connection to computer communication networks.

[35] *Sega Enterprises, supra* note 5. Sega manufactures a device called the "Genesis" console. This device is a specialized computer that runs video game programs, which are stored on cartridges. Before

a *de facto* monopoly in the underlying product. By allowing copyright in interface protocols, other designers are effectively prevented from developing competitive products.

The issue of interoperability and standardization becomes much more contentious in the area of computer user interfaces. Because of the apparent analogy to artistic works, most courts have tended to view the features of computer user interfaces as expressive.[36] This view, referred to by Bill Curtis as the expressive view,[37] is incorrect, because it is based on the assumption that users are attracted to computer software because of its aesthetic appeal. The proper view is the functional perspective,[38] which is based on the assumption that users are attracted to computer software because of the operations that they need to perform and the efficiency with which those operations can be performed. The creation of more efficient computer-human interfaces is an area of active research: both universities and private corporations devote large amounts of money and effort towards this type of development. An indication of the importance of this field is given by the number of journals and professional organizations that deal with issues in the area of computer-human interactions.[39]

An examination of the "look and feel" cases shows that plaintiffs have attempted to stress aesthetics. For example, in its copyright infringement complaint against Microsoft and Hewlett-Packard, Apple claimed:

> [T]he Macintosh computer programs generate a series of *fanciful* visual displays and images....*artistic, aesthetically pleasing* visual displays and graphic images that enhance the value and commercial appeal of Apple's products....the *distinctive expression* represented by the visual displays and graphic images....is widely recognized as a hallmark of the Macintosh computer system. [Emphasis added][40]

The wording of the brief shows a desire on the part of Apple to avoid the possible legal consequences associated with a finding that functional considerations directed the development of their user interface.

The design of user interfaces is an engineering task as opposed to an artistic endeavour. The discipline of human factors analysis has been developed to identify users' needs; once these needs are known, specific techniques can be used to facilitate computer-human interaction. The primary objective of human factors analysis is to

running a game, the Genesis console checks the cartridge for the presence of certain data: if a specific sequence is not present, the console will not run the game. Accolade, through a program of reverse engineering, determined what this sequence was and included it in cartridges which they manufactured. The result was that Accolade games could run on the Genesis console.

[36] *Lotus* v. *Paperback, supra* note 5; *Lotus* v. *Borland, supra* note 5 . The *Lotus* decisions are in sharp contrast to *Apple* v. *Microsoft, supra* note 1, which focused on the functional aspects of user interfaces.

[37] B. Curtis, "Engineering Computer 'Look and Feel': User Interface Technology and Human Factors Engineering" (1989) 30 Jurimetrics J. 51 at 54.

[38] *Ibid.*

[39] The Association for Computing Machinery (ACM) has a special interest group devoted to issues in computer-human interactions (SIGCHI). The following are some of the professional journals which give coverage to this area: Communications of the ACM, Ergonomics, Human Factors, IEEE Computer, IEEE Computer Graphics and Applications, and The Journal of Applied Psychology.

[40] Curtis, *supra* note 37 at 54.

enhance user productivity. In order to meet this objective, knowledge and techniques from sciences such as anatomy, physiology, and psychology are used to develop the best possible product. The use of such knowledge and techniques means that the design of computer user interfaces is an engineering task like any other engineering task. The characterization of this process as an art ignores the fact that the principles of human factors analysis have developed through scientific research on human capabilities and performance. The role played by aesthetics is described in the following passage:

> Aesthetics do not appear on our list of important user interface objectives for commercial systems because they are not a significant market issue. Aesthetics are valuable in videogames but they are far less important in the development, marketing, or use of software such as word processors, spreadsheets and data bases. The incremental value added to a software package by user interface aesthetics alone is negligible since sales will be determined by the software's ability to help users work more productively. The important criteria in buying software are breadth of functionality, ease of learning and use, speed of performance, and ability to integrate with other software packages.[41]

Even an item as innocuous as the now infamous Macintosh "Trashcan" possesses functional properties which can be measured. The following criteria give an indication of how the effectiveness of a proposed icon can be evaluated:

> · glance legibility – the ability to recognize an icon at a brief glance
> · distance legibility – the ability to recognize an icon at a distance
> · comprehensibility – the ability to understand the semantics implied by an icon
> · preference – a user's subjective choice among alternative icon designs
> · discriminability – the ability to distinguish the icon from other icons.[42]

The point to be taken from this analysis is that if something appears in a user interface, it is there for a reason.

The granting of strong copyright protection for user interfaces is potentially problematic for another reason. Once users become accustomed to a particular user interface there is a cost associated with switching to a non-compatible system.[43] This cost

---

[41] *Ibid.* at 63.

[42] *Ibid.* at 71. An example of the analysis used in determining whether to choose a particular icon is given at 73. The following passage describes the considerations used in the selection of the document icon for the Xerox 8010 Star Workstation:

> [T]he one chosen to represent a document was....the one with the folded edge on the upper right-hand corner. On one hand this is the edge that is folded over on most documents. However, the most important reason for choosing this icon is that the folded edge is at the top, leaving the bottom of the icon for written material....using the bottom portion for written material is standard across most of the icons....[w]hen users search for a particular item by its title, they will be able to search through the icons faster when they can inspect the same region of each icon for the title regardless of its shape. Scanning a row of icons is easier when the eye can stay on the same vertical level rather than having to move in a zig-zag.

[43] The standard QWERTY keyboard arrangement is an example of user reluctance to switch products once one format has become dominant. The original reason for choosing the QWERTY arrangement was to solve the problem of keys jamming. The letters were arranged on the QWERTY keyboard in an inefficient manner in order to reduce the number of keystrokes that a skilled typist could execute. This reduction "solved" the jamming problem; however, technology has long since eliminated

arises from the need to retrain. Unless the production of compatible rival products is allowed, copyright will grant a *de facto* monopoly to any user interface that gains a dominant market position.[44] Unlike the non-literal elements of a computer program, an equally good user interface cannot be developed independently, because "equally good" in the context of a standardized user interface means exact duplication.[45] The result is that copyright protection becomes tantamount to patent-like protection of the underlying human factors research. This paper will take the position that copyright protection is not appropriate for computer user interfaces and that they should instead be protected by *sui generis* legislation.

## C. *Reverse Engineering*

Reverse engineering is the process of starting with a finished product and working backwards in order to discover how that product operates or how it was produced. Traditionally, the law has allowed reverse engineering of technological works for the purpose of determining the unpatented processes used in making a work, and also for extracting the unpatented features of a work. Software and traditional technologies share a vulnerability to reverse engineering. The primary technique for reverse engineering a computer program is disassembly. Disassembly refers to the use of a computer program to read the machine language version of another program and to then display the assembly language version of that program. Unlike the reverse engineering of traditional technological works, disassembly of a computer program requires literal copying of the portion of the work which is being examined.[46] This copying is a technical violation of copyright.

The result of disassembly is not a high-level source code version of the copied program, but instead a simple mechanical conversion of the machine code version of the

the need for this inefficient arrangement. It is now user reluctance that prevents a switch from this long-established standard to a more efficient arrangement.

[44] The Lotus 1-2-3 litigation provides reason for the concern that corporations will seek to take advantage of potential monopolies. B. Johnson, "An Analysis of the Copyrightability of the 'Look and Feel' of a Computer Program: *Lotus* v. *Paperback Software*" (1991) 52 Ohio State L.J. 947 at 948, describes the financial incentives for such behaviour. In 1987, Paperback had the fourth leading spreadsheet package (VP-Planner) with sales of approximately $2.8 million. Lotus had the leading product with sales of $198 million, Microsoft was second at $38 million and Supercalc was third at $12.5 million. The sale price per unit of Lotus 1-2-3 was $495 while VP-Planner sold for $99. Two days after receiving a favourable decision in *Lotus* v. *Paperback*, Lotus filed suits against Borland (Quattro Pro) and The Santa Cruz Operation (SCO Professional). Lotus and SCO settled out of court, with SCO agreeing to stop manufacturing, distributing and licensing SCO Professional. Lotus and Borland eventually went to court with the result that Quattro Pro's emulation interface was found to infringe Lotus 1-2-3's user interface. Trade Publications were of the opinion that Quattro Pro was a technical improvement over Lotus 1-2-3, allowing users to simulate the Lotus interface as an option, while offering a unique interface of its own. See L. Picarille, "Lotus Settles with SCO, but Continues Battle with Borland" *Infoworld* vol. 13 no. 25 (24 June 1991) 152.

[45] Menell, *supra* note 7 at 1097.

[46] This requirement is similar to the requirement that a program be copied into RAM before it is executed. In the case of execution, the program must be copied in order to be put in a format which can be understood by a processor. In the case of reverse engineering, the program must be copied in order to be put in a format which can be understood by a human being.

program into the assembly code version of the program.[47] High-level source code programs are converted into object programs through a process known as compiling. During this process certain information is removed,[48] and because of this loss, it is mathematically impossible to reconstruct the original high-level source code from the resultant object code. Thus the process of disassembly is not a shortcut by which a competitor can appropriate an original developer's high-level source code.

Once it is understood that disassembly is simply a mechanical step that allows a programmer to obtain a limited, but now readable, version of a computer program, it becomes evident that the value in this process results from the software engineer's analysis of the disassembled code. From this perspective, it can be seen that reverse engineering of a computer program is equivalent to the studying of any other copyrighted work. The one difference is that a certain amount of literal copying is necessary to make the work human-readable. Another important observation is that reverse engineering is not a shortcut or means of avoiding the effort of development. Reverse engineering is a labour-intensive task, and the cost of reverse engineering a product will not be substantially less than the original cost of development. It must also be remembered that the first author has the significant advantage of being first to market with a product. This lead time, along with patent protection for those developments that warrant it, has worked in other technological disciplines; there is no reason to expect that it will not also work for the software industry.[49]

---

[47] S. Johnson-Laird, "Reverse Engineering of Software: Separating Legal Mythology from Actual Technology" (1992) 5 Software L.J. 331 at 343:

> Deciphering computer-executable programs is extremely tedious and error prone; it can take up to a minute or so for each computer instruction (a typical program might contain 500,000 instructions – days' worth of deciphering); so most programmers use another specialized form of program called a disassembler to create listings of the instructions and the numerical data on which they operate. Disassemblers only mechanize the deciphering process. They cannot do more, as the computer-executable form simply does not have any more information in it.

The Court in *Sega Enterprises, supra* note 5 at 1525, took a somewhat different view, focusing on the need to maintain a record of the work:

> Sega makes much of....[the] statement that a reverse engineer can work directly from the zeros and ones of object code but "[i]t's not as fun." In full, [the] statements establish only that the use of an *electronic* decompiler is not absolutely necessary. Trained programmers can disassemble object code by hand. Because even a trained programmer cannot possibly remember the millions of zeros and ones that make up a program, however, he must make a written or computerized copy of the disassembled code in order to keep track of his work (emphasis as in original).

[48] When object code is created, high-level features such as procedure and variable names are lost, and all in-line source code comments are stripped out (these do not generate executable code). Furthermore, optimizing compilers may significantly modify the resulting object code. For example, an optimizing compiler may decide that it does not need to store a value (variable) in memory; that value is instead maintained in a general purpose register. It is also common for a compiler to alter the order in which certain statements are executed in order to enhance efficiency.

[49] This proposition assumes that there is sufficient anti-piracy protection. Copyright in the literal elements of a computer program and its underlying object code should provide this protection.

This paper will take the position that reverse engineering for the purposes of extracting the underlying ideas and processes should be allowed. If reverse engineering is disallowed there will be a negative impact on both software designers and consumers. Designers will no longer have access to the latest ideas and developments; this, in turn, will decrease the opportunity for innovation. Consumers will be affected because a ban on reverse engineering will increase the incidence of monopolistic behaviour.[50] If the right to reverse engineer is removed, the software industry will be deprived of the primary technique by which the functional elements of copyrighted works are examined. An original developer will thus have the means to prevent competitors from gaining access to the functionality embodied in his or her copyrighted work. By preventing others from gaining access to the functionality embodied in a given work, an original designer can gain an effective monopoly over that functionality.

## D. *The Software Development Process*

As with other areas of technology, software advances through incremental and cumulative improvements on an existing knowledge base. This process was metaphorically described by Sir Isaac Newton in his famous statement, "If I have seen further it is by standing on the shoulders of Giants." The position taken by several American courts has had the effect of narrowing these shoulders:

> The metaphorical "shoulders of giants" on which successors may legally stand are not as broad as defendants contend. The legally relevant shoulders of programming giants are their ideas – and do not extend to all of their expressions. The encouragement of innovation requires no more.[51]

Unfortunately, when expression includes the non-literal elements of computer programs and the human factors components of user interfaces, innovation is likely to suffer. When the scope of copyright protection is this wide it covers some of the functional and technological aspects of programs and interfaces. Copyright protection of these features effectively excludes them from the natural process of incremental and cumulative improvement. The logical conclusion is that the pace of innovation will be affected.[52] The force of this conclusion is strengthened when one considers the increasing trend towards software re-use.[53] Once again, industry practice and the public

---

[50] *Sega Enterprises*, *supra* note 5, is a good example. Sega tried to maintain a monopoly over the supply of Genesis-compatible computer games.

[51] *Lotus* v. *Paperback*, *supra* note 5 at 78.

[52] Samuelson, Denber & Glushko, *supra* note 6, gives the results of a survey of industry reaction to copyright protection as measured at SIGGRAPH, 1991 and SIGCHI, 1989. The results indicate a low level of industry support for copyright protection of such features as: user interface commands, user interface "look and feel" and user interface functionality. There was, however, significant support for copyright protection of source and object code. When asked about the predicted effect of "look and feel" protection, a large majority felt that it would have a negative or strongly negative impact on the industry.

[53] Johnson-Laird, *supra* note 47 at 338, describes the costs associated with bug fixing. The implication is that a piece of software that has proven to be reliable is a valuable commodity. Increasingly, there is a trend towards software re-use as a way to reduce costs and speed the development process. Software re-use is an important part of the object-oriented design methodology. Many experts feel that this design paradigm will replace the top-down design process. For a discussion

interest suggest that the functional elements of computer software and user interfaces should not be subject to wide copyright protection.

Some academic writers have taken the position that strong copyright protection of computer programs is justified because the process of software development is analogous to the process of literary writing.[54] It is true that some people are very good programmers even though they have had very little training, while others may only be mediocre programmers despite having had extensive training. This, however, is true of any engineering discipline, and is indicative of a superior skill and understanding of the relevant technology. When a programmer is designing and implementing a piece of software, his or her thoughts are dominated by practical considerations.[55] Software is an inherently functional work; the creation of expression is not among the considerations of a programmer when developing a program. This paper will take the position that any judicial investigation of software based on the idea/expression dichotomy is inappropriate because it involves a search for something (expression) which is inconsequential to the work under consideration. It will be proposed that a more effective scheme[56] can be designed by taking into consideration the practical factors which determine how software is created and distributed.

## E. *Ease of Copying*

The primary difference between computer software and other technologies is the ease with which software can be copied. With other technologies, reverse engineering

of object-oriented design, see D.M. Barkan, "Software Litigation in the Year 2000: The Effect of Object-Oriented Design Methodologies on Traditional Software Jurisprudence" (1992) 7 High Tech. L.J. 315.

[54] Clapes, Lynch & Steinberg, *supra* note 7, quoting from F.P. Brooks, *The Mythical Man – Month: Essays on Software Engineering* (Reading, Massachusetts: Wesley Publishing Co., 1975) at 7-8:

> The programmer, like the poet, works only slightly removed from pure thought-stuff. He builds his castles in the air, from air, creating by exertion of the imagination. Few media of creation are so flexible, so easy to polish and rework, so readily capable of realizing grand conceptual structures....
>
> Programming then is fun because it gratifies creative longings built deep within us and delights sensibilities we have in common with all men.

[55] The following are some of the considerations that may affect the design of a computer program:

- efficient use of resources (memory and cpu);
- required throughput;
- required user response time;
- compatibility with the hardware, operating system, and other application software;
- maintenance requirements;
- the possibility of later upgrades to add new functionality;
- reliability and recoverability constraints.

[56] In this context, "effective scheme" implies a scheme that is both beneficial to creators (in that they receive a reasonable return on their efforts) and to the public (in that they receive a continuing stream of reasonably-priced innovative products).

requires a period of study, followed by time to "tool up", followed by manufacturing time. Under these circumstances, the initial developer still has the significant advantage of being first to market. However, a pirate with a personal computer and some diskettes can create a competing product in a matter of minutes.[57] There is no need to study, no need to tool up, and negligible manufacturing overhead. Furthermore, a pirate can sell his or her product at a much lower price, since no developmental costs were incurred. The problem is that the pirate can appropriate all of the advantages of the creator's development and suffer none of the disadvantages. The result is anti-competitive behaviour. The goal of a protective regime should be to ensure that there is sufficient incentive to encourage developers to create new products. This incentive will be provided by allowing a reasonable return on investment. When pirates are allowed to operate, the reward, and consequently the incentive, are removed. Conversely, when the protective scheme is too extensive, legitimate competition is eliminated and the public suffers. This paper will suggest that copyright protection of computer software should be aimed at preventing anti-competitive behaviour. Once a competitive environment is established, the balance between the need to innovate and the need to be rewarded should operate as efficiently as it has for other technical disciplines.

Later in this paper a two-fold protective scheme will be proposed. The first level will be based on copyright protection of the literal elements of computer software. The purpose of this level of protection is to combat piracy and "slavish" copying.[58] The second level of protection is designed to prevent anti-competitive behaviour. This level of protection is based on the premise that reverse engineering is a legitimate activity that does not give rise to an anti-competitive advantage.

While a scheme based on the prevention of anti-competitive behaviour is feasible for computer programs, it is not feasible for user interfaces. The first problem with user interfaces is that there is no process that is equivalent to the disassembly of computer programs. In order to extract information from a computer program it is necessary to go through the labour-intensive process of disassembly and analysis; with a user interface one can extract information simply by looking at it. The second problem with user interfaces is that it is not possible to come up with an equally good implementation of a user interface. For compatibility purposes, only exact duplication will do. Because of these difficulties this paper will suggest that copyright should not be used to protect computer user interfaces, instead, a *sui generis* scheme should be adopted. Under this scheme designers would be able to register new user interfaces, and receive a short period of monopoly protection.[59]

## III. International Obligations

Canada adheres to two major international copyright conventions, the *Berne Convention for the Protection of Literary and Artistic Works* (*Berne Convention*) and the *Universal Copyright Convention* (*UCC*). By its assent, Canada has agreed to maintain

[57] Coincidentally, this happens to be the same equipment that potential customers will have.

[58] Slavish copying refers to the process of line-by-line copying or translation of source code (as opposed to disk copying).

[59] This scheme would be similar to the scheme used for the registration of industrial designs.

certain minimum levels of protection with respect to works covered by these *Conventions*.[60] Any changes to Canadian copyright law will have to be made with regard to the obligations imposed by these agreements. Canada is currently bound by the Rome (1928) revision of the *Berne Convention* and the Geneva (1952) revision of the *UCC*. In addition, by Article 1701 of the *North American Free Trade Agreement* (*NAFTA*), Canada has agreed, at a minimum, to give effect to the substantive provisions of the Paris (1971) revision of the *Berne Convention*.

The purpose of these international conventions was to replace the system of bilateral agreements which had proven inadequate as the international market in copyrighted works expanded. The *Berne Convention* established three basic principles for the international protection of copyright: national treatment, automatic protection and independence of protection. National treatment means that authors enjoy, in respect of works that are eligible for protection under the *Berne Convention*, in countries of the Union other than the country of origin, the rights that are granted to nationals of that country. Automatic protection means that enjoyment and exercise of protection under the *Berne Convention* is not conditional on the fulfillment of any formalities. The final principle, independence of protection, means that protection under the *Berne Convention* is independent of the existence of protection in the country of origin of the work. In addition to these principles, the *Berne Convention* also requires certain minimum levels of protection. These minimums deal with the types of works that are protected, the scope of protection, and the duration of protection. With respect to computer programs, the most important *Berne Convention* minimum is the term of protection. The *Berne Convention* requires copyright to subsist for the life of the author plus 50 years.

## IV. The Legal Issues

In the following sections, some of the legal issues related to copyright protection of computer software will be examined. The areas that are currently the most contentious are reverse engineering for the purpose of developing competitive products, copyright protection of non-literal elements of computer programs, and copyright protection of computer user interfaces. The legal analysis will start with an examination of reverse engineering. Reverse engineering is critical to the industry in that it is the principal means by which the latest ideas and techniques are disseminated. The development of a rational basis for the protection of all computer technology is predicated on the ability to engage in legitimate reverse engineering activity.

The discussion of reverse engineering will be followed by an examination of the issues related to the protection of non-literal elements of computer programs. The primary issue here is one of scope, the major concern is that protection of non-literal elements will provide a scope of protection not previously contemplated under copyright. The final area to be examined is the protection of computer user interfaces. The

[60] The focus of this discussion will be on the *Berne Convention*, since it imposes higher standards of conduct on its member nations. B. Torno, *Fair Dealing: The Need For Conceptual Clarity On The Road To Copyright Revision* (Ottawa: Consumer and Corporate Affairs, 1981) at 49-50:

> [T]he Geneva text of the UCC requires only that member states provide "adequate and effective protection" for works to which the Convention applies. The only specific minimum standards are with respect to [the] term of protection and the right of translation.

development of a computer user interface requires significant effort and expense. Because of this expense, developers are very interested in protecting their investment. This paper will suggest that copyright is not properly suited to solving the unique problems posed by computer user interfaces.

In each of the major sections, the legal analysis will begin with an examination of the current American position, followed by a discussion of how these issues might be treated under Canadian law. Once the Canadian treatment of these issues has been discussed, proposed solutions will be offered.

## A. *Reverse Engineering*

Reverse engineering refers to a process of study in which a finished product is analyzed in order to determine how it operates or how it was produced. This type of analysis is commonly applied to technological works for two reasons: first, to determine compatibility requirements; and second, to obtain functional information for the development of competing products. Since a computer program is a functional, technological work, the right to reverse engineer is of critical importance to the software industry. However, as compared to other technological works, software differs in that the work itself must be copied in order to be studied. This replication raises questions about potential copyright violation. This paper will take the position that the unique nature of computer software, and the importance of reverse engineering to the industry, justify a finding that this type of copying is not copyright infringement. The copyright *Acts* of Canada and the United States do not contain any express provisions dealing with reverse engineering; as a result, reverse engineering must be justified under the provisions on "fair dealing" and "fair use". While the American jurisprudence with respect to fair use is extensive, the Canadian jurisprudence with respect to fair dealing is not. This lack of jurisprudence raises some doubt as to the status of reverse engineering in Canada.

### 1. *The American Position on Reverse Engineering*

The American position on reverse engineering has developed under the doctrine of fair use.[61] The extent to which a competitor can reverse engineer a copyrighted product

[61] *Copyright Act*, 17 U.S.C. § 117:

Notwithstanding the provisions of sections 106 and 106A the fair use of a copyrighted work, including such use by reproduction in copies or phonorecords or by any other means specified by that section, for purposes such as criticism, comment, news reporting, teaching (including multiple copies for classroom use), scholarship, or research, is not an infringement of copyright. In determining whether the use made of a work in any particular case is a fair use the factors to be considered shall include –

(1) the purpose and character of the use, including whether such use is of a commercial nature or is for nonprofit educational purposes;
(2) the nature of the copyrighted work;
(3) the amount and substantiality of the portion used in relation to the copyrighted work as a whole; and
(4) the effect of the use upon the potential market for or value of the copyrighted work.

has been considered recently in *Sega Enterprises*[62] and *Atari Games*.[63] Both of these cases accepted that under certain circumstances, reverse engineering is a fair use.

Sega developed and marketed a video entertainment system which consisted of a "Genesis" console and assorted game cartridges.[64] When one of the cartridges is inserted in the game console, it allows the user to play the particular game that is encoded on the cartridge. Sega wanted to control access to its console and, in an effort to prevent other manufacturers from designing games for its system, developed a lock-out mechanism. Sega placed a certain byte pattern in the initialization code of all of its game cartridges. If the console did not detect this byte pattern it would not run the game.[65] Accolade reverse engineered Sega's video game programs in order to discover the Genesis console compatibility requirements. This was done by purchasing commercially available copies of Sega's game cartridges and disassembling the object code contained in them. Once the interface specifications for the Genesis console were discovered, this information was used in the creation of Accolade products that could be run on the Genesis system. Sega challenged the reverse engineering process used by Accolade on the grounds that the intermediate copying during disassembly was a violation of Sega's copyright in the game cartridge object code.

The Ninth Circuit Court of Appeal held that absent the defence of fair use, "[t]he intermediate copying done by Accolade therefore falls squarely within the category of acts that are prohibited by the statute."[66] The court then considered the application of the fair use doctrine. This analysis involved an examination of the four factors enumerated in section 107 of the statute.[67] The first of these factors was the purpose and character of the use. At the District Court level, this factor weighed heavily against Accolade; however, the Court of Appeal's treatment was quite different. After noting that copying for a commercial purpose weighs against a finding of fair use, the court concluded that the fact that Accolade eventually used the information obtained to produce a competing product is not conclusive of the issue of fair use:

> Sega argues that because Accolade copied its object code in order to produce a competing product, the *Harper* v. *Row* presumption applies and precludes a finding of fair use. That analysis is far too simple and ignores a number of important considerations. We must consider other aspects of "the purpose and character of the use" as well. As we have noted, the use at issue was an intermediate one only and thus any commercial "exploitation" was indirect or derivative....
>
> [A]ccolade copied Sega's software solely in order to discover the functional requirements for compatibility with the Genesis console – aspects of Sega's programs that are not protected by copyright....

---

[62] *Sega Enterprises*, *supra* note 5.

[63] *Atari Games*, *supra* note 5.

[64] The console consists of a specialized computer and operating system, which is designed to run programs that are stored on game cartridges. See *Sega Enterprises*, *supra* note 5 at 1514.

[65] The initialization code also caused the display of certain trademark information. While this was also at issue in the case, it is not relevant to the question of reverse engineering. *Ibid.* at 1515.

[66] *Ibid.* at 1518.

[67] *Supra* note 61.

> On these facts, we conclude that Accolade copied Sega's code for a legitimate, essentially non-exploitive purpose, and that the commercial aspect of its use can best be described as of minimal significance. [Citations omitted][68]

The second statutory factor is the nature of the copyrighted work. In considering this factor the court noted that not all copyrighted works are entitled to the same level of protection. In particular, the protection provided by copyright does not extend to the ideas underlying a work or to the functional or factual aspects of the work. The fact that computer programs cannot be examined without a certain amount of literal copying was decisive in the view of the court:

> The unprotected aspects of most functional works are readily accessible to the human eye. The systems described in accounting textbooks or the basic structural concepts embodied in architectural plans, to give two examples, can be easily copied without also copying any of the protected, expressive aspects of the original works.[69]
>
> Disassembly of object code necessarily entails copying. Those facts dictate our analysis of the second statutory fair use factor. If disassembly of copyrighted object code is *per se* an unfair use, the owner of the copyright gains a *de facto* monopoly over the functional aspects of his work–aspects that were expressly denied copyright protection by Congress....
>
> Because Sega's video game programs contain unprotected aspects that cannot be examined without copying, we afford them a lower degree of protection than more traditional literary works. [Citations omitted][70]

The third statutory factor is the amount of the work copied. Since Accolade had disassembled entire programs, this factor weighed against them. The court, however, noted that the fact that an entire work has been copied does not preclude a finding of fair use.[71] The final statutory factor is the effect on the potential market for the copyrighted work. In considering this factor, the court examined whether the copying would adversely affect the potential market for the copyrighted work by diminishing potential sales. The court said that this examination must be done with caution, because it has the potential to render all of the other considerations irrelevant. The following approach was proposed:

> The *Harper & Row* Court found a use that effectively usurped the market for the copyrighted work by supplanting that work to be dispositive. However, the same consequences do not and could not attach to a use which simply enables the copier to enter the market for works of the same type as the copied work....
>
> [A]ccolade did not attempt to "scoop" Sega's release of any particular game or games, but only sought to become a legitimate competitor in the field of Genesis-compatible video games....

---

[68] *Sega Enterprises, supra* note 5 at 1522-23.

[69] *Ibid.* at 1525.

[70] *Ibid.* at 1526.

[71] *Sony Corp. of America* v. *Universal City Studios, Inc.*, 464 U.S. 417 at 449-50 (1984); *Hustler Magazine, Inc.* v. *Moral Majority, Inc.*, 796 F.2d 1148 at 1155 (9th Cir. 1986).

By facilitating the entry of a new competitor, the first lawful one that is not a Sega licensee, Accolade's disassembly of Sega's software undoubtedly "affected" the market for Genesis-compatible games in an indirect fashion....In any event, an attempt to monopolize the market by making it impossible for others to compete runs counter to the statutory purpose of promoting creative expression and cannot constitute a strong equitable basis for resisting the invocation of the fair use doctrine. [Citations omitted][72]

The court concluded that the fourth factor favoured Accolade despite the "minor economic loss" that Sega might suffer.

After consideration of the four statutory factors, the court held that Accolade's reverse engineering activities were within the doctrine of fair use. The Court stated that the following type of activity would constitute legitimate reverse engineering of a computer program:

We conclude that where disassembly is the only way to gain access to the ideas and functional elements embodied in a copyrighted computer program and where there is a legitimate reason for seeking such access, disassembly is a fair use of the copyrighted work, as a matter of law.[73]

The court in *Atari Games* reached a similar conclusion about the legality of reverse engineering;[74] however, since the issue arose in the context of a preliminary injunction, the analysis was not as detailed as that in *Sega Enterprises.*

The American position under the fair use doctrine seems to favour a wide scope for reverse engineering. According to the current case law, reverse engineering is allowed for the creation of competitive products as well as for purposes of interoperability. This position is in sharp contrast to that adopted in Europe under the Directive on the Legal Protection of Computer Programs.[75] Under this directive there is a limited right to reverse engineer to achieve interoperability of independently created computer programs.[76] Of the two positions, the American approach would seem preferable.

---

[72] *Sega Enterprises, supra* note 5 at 1523-24.

[73] *Ibid.* at 1527-28.

[74] *Atari Games, supra* note 5 at 843:

When the nature of a work requires intermediate copying to understand the ideas and processes in a copyrighted work, that nature supports a fair use for intermediate copying. Thus, reverse engineering object code to discern the unprotectable ideas in a computer program is a fair use....

Fair use to discern a work's ideas, however, does not justify extensive efforts to profit from replicating protected expression. Subparagraphs 1 and 4 of section 107 clarify that the fair use in intermediate copying does not extend to commercial exploitation of protected expression. The fair use reproductions of a computer program must not exceed what is necessary to understand the unprotected elements of the work. This limited exception is not an invitation to misappropriate protectable expression. Any reproduction of protectable expression must be strictly necessary to ascertain the bounds of protected information within the work (citations omitted).

[75] Council Directive 91/250 on the Legal Protection of Computer Programs, 1991 O.J. (L 122) 42.

[76] P.G. Hidalgo, "Copyright Protection of Computer Software in the European Community: Current Protection and the Effect of the Adopted Directive" (1993) 27 International Lawyer 113 at 139.

### 2. *The Status of Reverse Engineering in Canada*

The question of whether reverse engineering is permitted under the Canadian doctrine of fair dealing has not yet been answered. The issue of reverse engineering has not arisen in any Canadian litigation; thus, any insight into how reverse engineering might be treated must come from the existing law on fair dealing. Unfortunately, this area of copyright law is very underdeveloped. The result is that any analysis of how reverse engineering might be treated in Canada is, by necessity, speculative.

#### (a) *Canadian Jurisprudence*

Fair dealing is a statutory defence to an action for copyright infringement. Section 27(2) of the *Copyright Act* provides that:

> 27(2) The following acts do not constitute an infringement of copyright:
>
> > (a) any fair dealing with any work for the purposes of private study, research, criticism, review or newspaper summary;[77]

Fair dealing is not defined in the *Act*; thus, the only guidance provided by section 27(2) is that fair dealing is limited to acts which fall within the five enumerated categories. The question of what constitutes fair dealing has been left to judicial determination. As illustrated by the following passage from *Hubbard* v. *Vosper* (per Denning M.R.), the result in a given case will depend on the facts:

> It is impossible to define what is "fair dealing." It must be a question of degree. You must consider first the number and extent of quotations and extracts. Are they altogether too many and too long to be fair? Then you must consider the use made of them. If they are used as a basis for comment, criticism or review, that may be a fair dealing. If they are used to convey the same information as the author, for a rival purpose, that may be unfair. Next, you must consider the proportions. To take long extracts and attach short comments may be unfair. But, short extracts and long comments may be fair. Other considerations may come to mind also. But, after all is said and done, it must be a matter of impression.[78]

The lack of a statutory definition has resulted in the development of criteria which may be examined in determining whether a particular use is fair dealing. The following list was suggested by J. Lahore with respect to the U.K. *Act*:

- the purpose and character of the use;
- the nature of the copyrighted work;

---

Article 6 provides that a person having a right to use a copy of a program may perform acts of reproduction and translation of the program's literal code without authorization of the right holder, but only when they are "indispensable to obtain the information necessary to achieve interoperability of an independently created computer program with other programs" (footnotes omitted).

[77] *Copyright Act*, *supra* note 12.

[78] *Hubbard* v. *Vosper*, [1972] 2 Q.B. 84 at 94, [1972] 1 All E.R. 1023 at 1027 (C.A.) [hereinafter *Hubbard* cited to Q.B.].

- the amount and substantiality of the portion in relation to the copyrighted work as a whole both quantitatively and qualitatively;
- the effect of the use upon the potential market or value of the copyrighted work;
- the absence of the intent to plagiarize.[79]

The first four factors are the same as those in section 107 of the U.S. *Act.* However, fair dealing is more limited in scope than fair use.

Since its introduction, the doctrine of fair dealing has arisen as a significant issue in only one Canadian case, *Zamacois* v. *Douville.*[80] This case dealt with the reproduction, in a Quebec newspaper, of an article that had appeared in a French newspaper. The article itself was about the Second World War, and much of the discussion in the case related to the characterization that should be given to this piece. The defendants claimed that it was an article of current interest on an economic or political topic. The plaintiff characterized the article as a literary work. The defendants also claimed that another article published in the same edition of their paper criticized the Zamacois article, and that the printing of the Zamacois article was necessary for this criticism. The court ruled that the article was not of current interest on an economic or political topic. With respect to the claim of fair dealing, it was held that:

> The right to quote is permitted by the Act; to refuse this would in effect suppress the right of literary criticism. However, a critic cannot, without being guilty of infringement, reproduce in full, without the author's permission, the work which he criticizes.[81]
>
> ....
>
> In my opinion, "Le Bien Public" had no right to reproduce the complete text of the article by Zamacois together with the comments of their contributor, Leon Dufrost. In doing so they went beyond the right to quote recognized by the Act.[82]

The *Zamacois* decision has been criticized strongly, and as a result its value as precedent may be weak.[83] The position taken in *Zamacois* contrasts with that taken by the English Court of Appeal in *Hubbard* v. *Vosper* (per Megaw L.J.):

---

[79] Lahore *et al.*, *supra* note 21 at 9-10. Except for grammatical changes, paragraph 27(2)(a) has remained unchanged since its introduction in the *Copyright Act*, 1921, where it was adopted directly from the Imperial *Copyright Act*, 1911. H.G. Richard *et al.*, *Canadian Copyright Act–Annotated*, vol. 2 (Toronto: Carswell, 1993) at 27-36 suggest the following:

- length of quoted excerpt;
- proportion in relation to the critics' own comments;
- the use made of the work;
- the object of the study.

[80] (1943), 2 C.P.R. 270, 2 D.L.R. 257 (Ex. Ct.) [hereinafter *Zamacois* cited to C.P.R.].

[81] *Ibid.* at 302.

[82] *Ibid.* at 304.

[83] Torno, *supra* note 60 at 32 makes the following criticisms:

> The position that, irrespective of all other considerations, there are no circumstances under which reproduction of a work in its entirety can ever constitute fair dealing:
>
> (a) does not necessarily arise from the language of the statute itself;
> (b) has found no expression elsewhere in Commonwealth case law or commentary; and
> ( establishes inappropriately rigid parameters upon what is universally acknowledged to a flexible rule of reason.

> It is then said that the passages which have been taken from these various works....are so substantial, quantitatively so great in relation to the respective works from which the citations are taken, that they fall outside the scope of "fair dealing". To my mind this question of substantiality is a question of degree. It may well be that it does not prevent the quotation of a work from being within the fair dealing subsection even though the quotation may be of every single word of the work. Let me give an example. Suppose that there is on a tombstone in a churchyard an epitaph consisting of a dozen or of 20 words. A parishioner of the church thinks that this sort of epitaph is out of place on a tombstone. He writes a letter to the parish magazine setting out the words of the epitaph. Could it be suggested that that citation is so substantial, consisting of 100 per cent of the "work" in question, that it must necessarily be outside the scope of the fair dealing provision? To my mind it could not validly be so suggested.[84]

The approach taken in *Hubbard* v. *Vosper* seems to be preferable to that in *Zamacois* in that it allows more flexibility. In the context of reverse engineering of computer programs, an engineer may not necessarily know where to look for the relevant information. In certain circumstances it may be necessary to disassemble an entire program in order to find what is needed.[85]

In order to qualify as fair dealing under the Canadian *Copyright Act* an instance of copying must fall within one of the five enumerated categories. As with the rest of the law on fair dealing, there has been little judicial consideration of the scope of these categories.[86] It would appear, however, that the five categories can be further divided into two subgroups: newspaper summary, review, and criticism forming one subgroup; and private study and research forming another. The first subgroup allows copying of a work for the purpose of appearing in a second work. The second subgroup appears to allow copying if it facilitates the furtherance of the copier's goals (research or private study). The scope of the exception under the second subgroup seems broader. This distinction may be important in supporting a right of reverse engineering under Canadian law, since it would appear to fall in the category of research.

### (b) *Reverse Engineering Under Canadian Law*

The fair dealing exception seems to be wide enough to support reverse engineering both for the development of competitive products, and for interoperability. However,

[84] *Hubbard, supra* note 78 at 98.

[85] This was the case in *Sega Enterprises* (see *supra* note 73 and accompanying text). In most cases, however, it will only be necessary to disassemble part of the work.

[86] E.P. Skone James and W.A. Copinger, *Copinger And Skone James On Copyright*, 11th ed. (London: Sweet & Maxwell, 1971) at 197 deals with this issue in the following manner:

> The meanings of the expressions "research," "criticism" and "review" seem to require no further consideration.

*Hawkes & Son (London), Ltd.* v. *Paramount Film Service, Ltd.* (1934), 1 Ch. 593, 103 L.J. Ch. 281 (C.A.), suggests that the categories are to be strictly interpreted since they represent a derogation from the statutory rights of authors. *University of London Press, supra* note 12, established that private study does not include copying for the purpose of distribution to students. *Hubbard, supra* note 78, held that the scope of criticism is not limited to the expression of a work, but can extend to the ideas or theories contained in the work. *De Garis* v. *Neville Jeffress Pidler Pty. Ltd.* (1990), 95 A.L.R. 625 (Fed.Ct.Aust.), held that the words "study" and "research" ought to be given their dictionary meaning.

since the case law with respect to fair dealing is so limited, it will be necessary to borrow from both Commonwealth and American jurisprudence in order to justify this result.

The first step in establishing reverse engineering as a fair use is to fit it within one of the five enumerated categories. The most likely category for this purpose is research.[87] Webster's Third New International Dictionary defines research as "critical and exhaustive investigation or experimentation having for its aim the discovery of new facts and their correct interpretation." The reason for engaging in a program of reverse engineering is to determine the functional specifications or compatibility requirements needed in order to create an independent program. Reverse engineering can be easily viewed as "critical and exhaustive investigation or experimentation having for its aim the discovery of new facts." In this respect it seems to fall within the category of research under section 27(2)(a).

The next step in determining whether there has been fair dealing is a fact-dependent investigation of the case at bar. In this regard it is not possible to determine whether an individual reverse engineering program will be held to be fair dealing. However, it is possible to examine the factors that would be canvassed by a court, and to speculate on how these factors might be viewed when applied to a typical reverse engineering program.

The first factor suggested by Lahore for determining whether there has been fair dealing is the purpose and character of the use.[88] The ultimate goal of a reverse engineering program is to obtain the information necessary for the independent creation of a competitive product. At first glance it would appear that the purpose and character of the use is commercial exploitation; however, this conclusion arises from a consideration of the end result and not the actual copying itself. For the purpose of copyright law, the focus must be on the copying. The purpose of the copying is to create a disassembled version of the object code which can then be viewed in order to extract information about functionality and compatibility. As the *Sega* court noted, "the purpose of the copying is essentially non-exploitive."[89] The fact that copying occurs is incidental: a reverse engineer is concerned with obtaining functional information. The intent is not to offer the copied portions of the first program in competition with itself.[90]

With respect to the character of the use, a Canadian court might also examine the competing program to ensure that copying was only of elements dictated by functionality or compatibility. For example, in *Sega Enterprises*, an initialization sequence of 20 bytes was taken and used directly in the Accolade product. The Court viewed this direct use as acceptable because compatibility requirements left Accolade with no choice. However, in *Atari Games*, the court found that Atari had copied certain unnecessary instructions.[91] This suggested to the court that the nature of Atari's reverse engineering program was one of copying rather than one of independent creation. This would be a reasonable approach for Canadian courts to take.

The next criterion suggested by Lahore is the nature of the work. The unique nature of computer programs gives rise to the strongest reasons for allowing reverse engineering.

[87] Private study is another possibility, but the use of the word "private" could connote individual activity, as opposed to activity by a commercial entity.

[88] See *supra* note 79 and accompanying text.

[89] *Sega Enterprises, supra* note 68 and accompanying text.

[90] See *Hubbard, supra* note 78 and accompanying text (per Denning M.R.).

[91] *Atari Games, supra* note 5 at 845.

Computer programs distributed in their object code format cannot be studied without being disassembled. The disassembly process requires literal copying of the portion of the program being examined. Unless a certain amount of literal copying is allowed, there is no practical way for the general public to access the unprotected features of a copyrighted computer program. The result of a prohibition on reverse engineering would be equivalent to the granting of a *de facto* monopoly over the functional elements of the protected computer program.

Another unique aspect of computer programs is that they exist in three different forms: source code, assembly code, and machine code. In terms of information conveyed to a programmer, the source code version is the most valuable. The result of disassembling the machine code version of a program is a version of the program in its assembly code format. This version of the copyrighted work has had much of the information that is contained in the source code version removed. In order to extract information about functionality and compatibility, an engineer must use his or her training and experience to reconstruct higher level information from the raw data produced by disassembly. Thus, when reverse engineering is performed, the copying is not of the computer program in its most valuable form.

The next factor to be considered in determining whether there has been fair dealing is the amount of copying in relation to the copyrighted work as a whole. When applied to situations where portions of the copyrighted work appear in a subsequent work; this approach makes sense. However, much of the copying done in a reverse engineering program is done merely to find the needed information. Once located, this information may then be copied for use in a competing work. When applied to the latter type of copying, the amount could help a court in determining whether the particular reverse engineering program is fair dealing.[92] However, as applied to the former type of copying (the "search copying"), the issue of how much copying is done is not useful. For example, would a court punish one engineer over another because one had greater skill and knowledge and was able to find what he or she needed with less copying?[93] It is submitted that the quantity of search copying is not a good indication of whether a particular program of reverse engineering is fair dealing.

The next factor to be considered is the effect of the use upon the potential market or value of the copyrighted work. This factor is the one which weighs most heavily against reverse engineering as fair dealing. If a competitive product is developed as a result of a reverse engineering program, there will be some impact on the potential market or value of the first work. If this view is taken, it virtually negates all the other factors. However, copyright law must concern itself with the copying alone. The result of the copying is an increase in the knowledge that the reverse engineer has of the product that has been copied. This result, in and of itself, does not affect the potential market or

[92] For example, in *Atari Games*, the court decided that more had been copied and used in a competing product than was necessary for the purposes of functionality and compatibility. The amount of copyrighted material used helped the court characterize Atari's "study" as copying rather than legitimate reverse engineering.

[93] Potentially, it may be necessary to disassemble a whole program in order to find the needed information. By the *Zamacois* doctrine such copying could not be fair dealing. In this regard, the approach in *Hubbard* is preferable. While the majority of cases that involve this much copying will not qualify as fair dealing, *Hubbard* acknowledges that there may be occasions where a finding of fair dealing is justified.

value of the copyrighted work. The fact that the knowledge obtained is subsequently used independently to create a competitive product should not change the character of the initial copying.

Another consideration related to the potential impact of reverse engineering on the value of the copied work is that most of the copied portions of the program are not used in competition with the program itself.[94] Copying done for the purpose of reverse engineering is not the same as the copying done in *Zamacois*. When the Zamacois article appeared in print it was essentially in competition with itself. The copying done in a reverse engineering program is done to acquire the information needed in order to create an independent product. In this sense the copied material is not being used directly in competition with itself.

The final consideration suggested by Lahore is the intent to plagiarize. This consideration complements the anti-competitive behaviour protection scheme described later in this paper. By plagiarizing, one gains the advantages of the initial development without having to incur any of the costs associated with that development. However, as illustrated in section II. C. (The Nature of Computer Software – Reverse Engineering), a true program of reverse engineering is a difficult and time-consuming task. The use of "true" reverse engineering does not allow the copier to avoid the costs of development, since the costs of the reverse engineering program will replace the initial development costs. If a court can conclude that there is a sufficient paper trail to evidence an honest and reasonable effort to understand the copied work, a finding of fair dealing should be justifiable on the basis that there is no intent to plagiarize. An honest and reasonable effort to understand the copied work will be evidenced by use of only those parts of the first work which are necessary for the development of a competitive or compatible product. Any use beyond this will be indicative of an intent to plagiarize.

Based on the preceding analysis, it would appear that the Canadian doctrine of fair dealing, as it currently exists, is broad enough to allow reverse engineering both for interoperability and for the development of competitive products. While some of the considerations suggest that reverse engineering should not be allowed as fair dealing, the majority of the traditional considerations suggest that reverse engineering is fair dealing. In particular, the nature of the work and the purpose of the copying strongly support this conclusion. However, given the lack of judicial consideration in this area, any conclusion on the status of reverse engineering in Canada is necessarily speculative.

### (c) *Proposed Treatment of Reverse Engineering in Canada*

There are three possible choices for a Canadian policy on reverse engineering. The first is to disallow completely reverse engineering; the second is to allow limited reverse engineering for compatibility purposes (the European position); and the third is to allow a wide scope for reverse engineering (the American position). Given that the need to reverse engineer is critical to the maintenance of a competitive software industry, it is recommended that Canada adopt the American position and allow a broad scope for reverse engineering. As discussed in the previous section, this position should be

[94] There will be cases, such as *Sega Enterprises*, where exact copying does occur. These cases will merit close scrutiny in order to ensure that such copying is dictated by functional concerns.

supportable under current Canadian law. However, given that reverse engineering is so important to the software industry, and given that this area of copyright law has received so little judicial analysis, consideration should be given to amending the *Copyright Act* to provide legislative approval for reverse engineering. Such an amendment would not be in conflict with Canada's obligations under the *Berne Convention*, since Article 10, under certain circumstances, allows limited copying of portions of copyrighted works.[95]

[95] Article 10 (Rome, 1928):

> As regards the liberty of extracting portions from literary or artistic works for use in publications destined for educational purposes, or having a scientific character, or for chrestomathies, the effect of the legislation of the countries of the Union and of special arrangements existing, or to be concluded, between them is not affected by the present Convention.

The reference to the "right to include excerpts" would seem to prohibit the copying of an entire work (*i.e.* the *Zamacois* position as opposed to the *Hubbard* position). However, as discussed earlier, while a reverse engineering search may involve the copying of an entire work, only a portion of the work may actually be used in a competitive product. Given the unique nature of computer programs, and the fact that it may not be possible to locate the relevant extracts without extensive search copying, it is submitted that reverse engineering falls within the text of the Rome version of the *Berne Convention*.

Article 10 (Paris, 1971):

> (1) It shall be permissible to make quotations from a work which has already been lawfully made available to the public, provided that their making is compatible with fair practice, and their extent does not exceed that justified by the purpose, including quotations from newspaper articles and periodicals in the form of press summaries.
>
> (2) It shall be a matter for legislation in the countries of the Union, and for special agreements existing or to be concluded between them, to permit the utilization, to the extent justified by the purpose, of literary or artistic works by way of illustration in publications, broadcasts or sound or visual recordings for teaching, provided such utilization is compatible with fair practice.
>
> (3) Where use is made of works in accordance with the preceding paragraphs of this Article, mention shall be made of the source, and name of the author if it appears thereon.

Reverse engineering of computer programs would seem to fall easily within the bounds of the Paris version of the *Berne Convention*. In the Paris version, the requirement for making quotations is that "their making is compatible with fair practice, and their extent does not exceed that justified by the purpose." As noted earlier, it is fair practice to examine a copyrighted computer program for the purpose of extracting functional information.

Furthermore, since one does not necessarily know where such information may be found, it is possible that search copying may require the disassembly of an entire program. It is, however, important to note that once the necessary information has been located, any further search copying cannot be justified. (*The Berne Convention for the Protection of Literary and Artistic Works from 1886-1986* (Geneva: International Bureau of Intellectual Property, 1986) at 230 and 233).

## B. *Protection of the Non-Literal Elements of Computer Software*

Both the Canadian and American copyright *Acts* extend protection to the literal aspects of a computer program.[96] However, a computer program consists of more than its literal instructions. The American jurisprudence has referred to these extended features as the "structure, sequence and organization" or "look and feel" of a computer program. The question of whether these elements should be protected, and if so, to what extent, has caused much debate.

### 1. *The American Position on Non-Literal Elements*

American copyright jurisprudence is based on two fundamental principles: the idea/expression dichotomy, and the prohibition against the protection of utilitarian function. The idea/expression dichotomy refers to the principle that copyright does not protect ideas, but only the expression of ideas. This rule was first enunciated in *Baker* v. *Selden*,[97] and has since been embodied in the U.S. *Copyright Act*.[98] The second principle also finds its origin in *Baker* v. *Selden*. Under the *Selden* doctrine, copyright protection cannot be used, directly or indirectly, to prevent the use of utilitarian features. This principle has also been embodied in the U.S. *Copyright Act*.[99]

Under American jurisprudence, copyright protection of literary works has traditionally extended to the non-literal elements of those works.[100] The court in *Whelan* v. *Jaslow*[101] noted that 17 U.S.C. § 102(1)(a) extends copyright protection to "literary works", and that computer programs are classified as literary works for the purpose of copyright. The court then made the following observation:

---

[96] *Copyright Act*, *supra* note 12 at s. 1(3):

> "computer program" means a set of instructions or statements, expressed, fixed, embodied or stored in any manner, that is to be used directly or indirectly in a computer in order to bring about a specific result;

The U.S. *Act* at § 101 (1988):

> A "computer program" is a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result.

[97] 101 U.S. 99 (1879).

[98] The U.S. *Act* at § 102(b) (1988) states:

> In no case does copyright protection for an original work of authorship extend to any idea, procedure, process, system, method of operation, concept, principle, or discovery, regardless of the form in which it is described, explained, illustrated, or embodied in such work.

[99] *Ibid.*

[100] See *e.g.* *Twentieth Century-Fox Film Corp.* v. *MCA, Inc.*, 715 F.2d 1327 (9th Cir. 1983) (alleged plot similarities between *Battlestar Gallactica* and *Star Wars*); *Sid and Marty Krofft Television Productions, Inc.* v. *McDonald's Corp.*, 562 F. 2d 1157 (9th Cir. 1977) (alleged similarities between McDonaldland characters and H.R. Pufnstuf characters); and *Bevan* v. *Columbia Broadcasting System, Inc.*, 329 F. Supp. 601 (S.D. N.Y.1971) (alleged similarities between the play *Stalag* 17 and the television series "Hogan's Heroes").

[101] *Whelan*, *supra* note 5.

By analogy to other literary works, it would thus appear that the copyrights of computer programs can be infringed even absent copying of the literal elements of the program. [Footnotes omitted][102]

While the scope of protection extended by the *Whelan* court has been criticized in both the academic literature[103] and subsequent judicial decisions,[104] the choice to extend some protection to the non-literal elements of computer programs has been accepted. In *Computer Associates*,[105] the following was said:

The syllogism that follows from the foregoing premises is a powerful one: if the non-literal structures of literary works are protected by copyright; and if computer programs are literary works, as we are told by the legislature; then the non-literal structures of computer programs are protected by copyright. See *Whelan*, ("By analogy to other literary works, it would thus appear that the copyrights of computer programs can be infringed even absent copying of the literal elements of the program"). We have no reservation in joining the company of those courts that have already ascribed to this logic. [Citations omitted][106]

Despite the uniformity in the choice to extend protection to non-literal elements, the scope of protection has varied considerably. According to *Whelan*, the scope should be determined through the application of the following test:

Just as *Baker* v. *Selden* focused on the end sought to be achieved by Selden's book, 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 the expression of the idea*....Where there are various means of achieving the desired purpose, then the particular means chosen is not necessary to the purpose; hence, there is expression, not idea. [Citations omitted] [Footnotes omitted] [Emphasis in original][107]

This test has been generally rejected as overly broad. In *Computer Associates*, the Abstraction-Filtration-Comparison test was proposed:

Step 1: Abstraction
As applied to computer programs, the abstractions test will comprise the first step in the examination for substantial similarity. Initially, in a manner that resembles reverse engineering on a theoretical plane, a court should dissect the allegedly copied program's

[102] *Ibid.* at 1234.

[103] Menell, *supra* note 7; S.R. Englund, "Note, Idea, Process or Protected Expression?: Determining the Scope of Copyright Protection of the Structure of Programs" (1990) 88 Mich. L. Rev. 866; M.T. Kretschmer, "Copyright Protection for Software Architecture: Just Say No!" (1988) Colum. Bus. Rev. 823; and P.G. Spivack, "Does Form Follow Function? The Idea/Expression Dichotomy in Copyright Protection of Computer Software" (1988) 35 U.C.L.A. L. Rev. 723.

[104] *Computer Associates, supra* note 5; *Lotus* v. *Borland, supra* note 5.

[105] *Computer Associates, ibid.*

[106] *Ibid.* at 702.

[107] *Whelan, supra* note 5 at 1236.

> structure and isolate each level of abstraction contained within it. This process begins with the code and ends with an articulation of the program's ultimate function.[108]
>
> Step 2: Filtration
> Once the program's abstraction levels have been discovered, the substantial similarity inquiry moves from the conceptual to the concrete. Professor Nimmer suggests, and we endorse, a "successive filtering method" for separating protectable expression from non-protectable material. This process entails examining the structural components at each level of abstraction to determine whether their particular inclusion at that level was "idea" or was dictated by considerations of efficiency, so as to be necessarily incidental to that idea; required by factors external to the program itself; or taken from the public domain and hence is nonprotectable expression. [Citations omitted][109]
>
> Step 3: Comparison
> Once a court has sifted out all elements of the allegedly infringed program which are "ideas" or are dictated by efficiency or external factors, or taken from the public domain, there may remain a core of protectable expression. In terms of a work's copyright value, this is the golden nugget. At this point, the court's substantial similarity inquiry focuses on whether the defendant copied any aspect of this protected expression, as well as an assessment of the copied portion's relative importance with respect to the plaintiff's overall program. [Citations omitted][110]

Since this test was enunciated in *Computer Associates*, it has received approval in a number of other circuits;[111] however, this acceptance has not been uniform.[112]

---

[108] *Computer Associates*, *supra* note 5 at 707.

[109] *Ibid.* at 709-10. The following considerations were proposed for the filtration stage:

- the mechanical specifications of the computer on which a particular program is to run;
- compatibility requirements of other programs with which a program is designed to operate;
- computer manufacturers' design standards;
- demands of the industry being serviced;
- widely accepted programming practices within the industry.

[110] *Ibid.* at 710.

[111] See *Atari Games*, *supra* note 5; *Sega Enterprises*, *supra* note 5.

[112] In the third circuit, *Whelan* remains good authority. *Lotus* v. *Borland*, *supra* note 5 at 216-19, proposed the copyrightability-substantial similarity test:

> FIRST, in making the determination of "copyrightability," the decisionmaker must focus upon alternatives that counsel may suggest, or the court may conceive, along the scale from the most generalized conception to the most particularized, and choose some formulation, some conception of the "idea," "system," "process," "procedure," or "method" – for the purpose of distinguishing between the idea, system, process, procedure, or method and its expression....
>
> SECOND, the decisionmaker must focus upon whether an alleged expression of the idea, system, process, procedure, or method is limited to elements essential to expression of that idea, system, process, procedure, or method (or is one of only a few ways of expressing the idea, system, process, procedure, or method) or instead includes identifiable elements of expression not essential to every expression of that idea, system, process, procedure or method....
>
> THIRD, having identified elements of expression not essential to every expression of the idea, system, process, procedure, or method, the decisionmaker must focus on whether

In applying the Abstraction-Filtration-Comparison test (and other proposed tests), the copyright law limitation on the protection of utilitarian function must also be considered. This limitation was stated in *Apple* v. *Microsoft*:

> Purely functional items or an arrangement of them for functional purposes are wholly beyond the realm of copyright as are other common examples of user interfaces or arrangements of their individual elements – the dials, knobs and remote control devices of a television or VCR, or the buttons and clocks of an oven or stove. Of course, the elements of these everyday user interfaces are seldom conflated into metaphoric images, but that does not mean that the user interface of a computer is less functional.
>
> Under the law of this Circuit, an article which has "*any* intrinsic utilitarian function" can be denied copyright protection "except to the extent that its artistic features can be identified separately and are capable of existing independently as a work of art". [Citations omitted] [Emphasis in original][113]

Thus the American position is based on two fundamental principles: the idea/ expression dichotomy, and the prohibition against copyright protection of utilitarian function. The application of tests based on these principles is problematic for two reasons. The first is that the separation of idea and expression (the abstractions test) is inherently imprecise,[114] and in an industry where the development of a product can cost millions of dollars, such ambiguity is unacceptable. The second reason is the difficulty in separating protectable expression from utilitarian function. It has been suggested that the protection of non-literal elements of a computer program can include items such as "general flow charts as well as the more specific organization of *inter-modular relationships*, parameter lists, and macros."[115] However, as illustrated earlier in this

---

> those expressive elements, taken together, are a substantial part of the allegedly copyrightable "work."

Keeton J. claims that this test is substantively compatible (though not in methodology) with the Abstraction-Filitration-Comparison test (*ibid.* at 215).

[113] *Apple* v. *Microsoft, supra* note 1 at 1023.

[114] The abstractions test was first proposed by Learned Hand J. in *Nichols* v. *Universal Pictures Corp.*, 45 F. 2d 119 at 121 (2nd Cir. 1930):

> 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 "ideas," to which, apart from their expression, his property is never extended. *Nobody has ever been able to fix that boundary, and nobody ever can* (emphasis added) (citations omitted).

Subsequent reflection upon this issue did not change Learned Hand J.'s view as is illustrated in *Peter Pan Fabrics, Inc.* v. *Martin Weiner Corp.*, 274 F.2d 487 at 489 (2nd Cir. 1960):

> Obviously, no principle can be stated as to when an imitator has gone beyond copying the "idea," and has borrowed its "expression." Decisions must therefore inevitably be *ad hoc*.

[115] The Hon. J.M. Walker, Jr., *et al.*, "Copyright Protection: Has Look and Feel Crashed?" (1993) 11 Cardozo Arts & Enter. J. 721 at 730 (emphasis in original) (footnotes omitted).

paper, such features are not created for expressive purposes.[116] A consumer chooses a particular software package based on functionality and the increase in productivity which that product can provide.[117] If a designer adds non-functional expression to that product he or she is adding something that does not enhance the desirability of his or her product in the eyes of a potential consumer. In fact, the addition of such expression might detract from the desirability of the product, since this expression would require the consumption of resources that could be better used to increase the functionality or efficiency of the product.

Based on the discussion above, it is reasonable to conclude that the presence of any non-literal copyrightable expression is *de minimus*. This conclusion leads to a question: if this type of protectable expression is not present in a computer program, how effective can a test that is designed to search for such expression be? Furthermore, even if such non-functional expression is present, who would copy it? The answer is, nobody would copy this expression unless he or she had literally copied the code, or was engaging in anti-competitive behaviour.[118] Thus, the current law mandates an extensive search for that which is at best minimally present, and which, if found, is not worth copying. The result is the nonsensical proposition that if it is hard to find and not worth copying, it is worth protecting.[119] The primary difficulty with the American approach is that when applied to computer software, the combination of the idea/expression dichotomy and the prohibition against the protection of utilitarian function results in an intractable problem. In some instances courts have been able to use this approach to reach the correct result;[120] however, it is submitted that these results could have been reached in a more straightforward manner.

## 2. *Protection of Non-Literal Elements in Canada*

Canadian law on the protection of the non-literal elements of computer software is still largely undeveloped. However, this issue has arisen in a number of cases. The following section will give a brief overview of the Canadian case law on the protection of non-literal elements. This overview will be followed by an analysis of how the protection of the non-literal elements could be treated based on the *Copyright Act* and traditional Anglo-Canadian jurisprudence.

---

[116] See *supra* note 55 regarding design of a computer program.

[117] This general rule does not apply to video games, because their primary purpose is to entertain. Video games should be treated as separate copyrightable works.

[118] As previously discussed, anti-competitive behaviour would not include:

- a legitimate program of reverse engineering;
- the taking of elements which are not subject to literal copyright protection (for example, code which is part of the product, but which is also in the public domain).

[119] Traditional copyright doctrine has held that if something is worth copying, it is worth protecting. As discussed earlier, artistic expression in a computer program is not worth copying. This leads to the question, why does the law protect such expression?

[120] See *Computer Associates*, *supra* note 5. In light of the prohibition against the protection of utilitarian features, it is difficult to see how any non-literal elements can survive the filtration stage. *Apple* v. *Microsoft*, *supra* note 1, shows how fine the filter can be.

### (a) *Canadian Case Law*

Canadian copyright jurisprudence differs from American jurisprudence in a number of fundamental ways. The most notable difference is with regard to the idea/expression dichotomy. In Canada, copyright protection has been extended to works in order to protect the skill-and-labour expended in their creation.[121] This kind of approach allows some protection of ideas, and contrasts with the strong application of the idea/expression dichotomy in the United States.[122] A further difference arises with respect to the treatment of useful articles. In cases such as *Doral Boats*[123] and *Spiro-Flex Industries*[124] the full scope of copyright protection has been extended to works despite the fact that they embodied utilitarian features. The Canadian position is currently uncertain because the protection of utilitarian works is now covered by section 64.1 of the *Copyright Act*,[125] and this section has yet to receive extensive judicial consideration.

The question of the scope of protection to be granted to non-literal elements of computer programs has arisen directly in only a few Canadian cases. In the majority of these cases, the issue was considered in the context of interlocutory injunctions, and, as a result, the substantive analysis was limited. The remainder of this section will examine the conclusions that have been reached in these cases.

In *Gemologists International*,[126] the court found that the defendants, former employees of the plaintiff, had taken a computer program from the plaintiff. The program was designed to perform appraisals that were relevant to the running of a jewelry business. Once in possession of the program, the defendants used it to develop their own appraisal program. An interlocutory injunction was granted for copyright infringement of:

> [P]rograms and program manuals by copying the overall logical structure of the programs and substantially copying the program sequence of menus and menu options and by use of the applicant's flow chart.[127]

This case, which arose in the context of an interlocutory injunction, is of little value as precedent since the basis on which protection was extended does not seem to flow from the *Copyright Act* or prior case law. The following passage gives some indication of the kind of analysis that the court engaged in:

> Theft of intellectual property is as serious as theft of any other property. Conversion of Gemologists' software package is just as culpable as parking a truck into a loading dock and stealing its computer. Creating a new software package with stolen parts is akin to

---

[121] *Ladbroke (Football) Ltd.* v. *William Hill (Football) Ltd.*, [1964] 1 All E.R. 465, 1 W.L.R. 273 (H.L.); *Ascot Jockey Club Ltd.* v. *Simons* (1968), 56 C.P.R. 122, 39 Fox Pat. C. 52 (B.C.S.C.) [hereinafter *Ascot Jockey Club* cited to C.P.R.]; *British Columbia Jockey Club* v. *Standen (Winbar Publications)* (1983), 73 C.P.R. (2d) 164, 146 D.L.R. (3d) 693 (B.C.S.C.), aff'd (1985), 8 C.P.R. (3d) 283, 22 D.L.R. (4th) 467 (B.C.C.A.).

[122] See *supra* note 5.

[123] *Doral Boats, supra* note 23.

[124] *Spiro-Flex Industries, supra* note 23.

[125] See *infra* note 161.

[126] *Gemologists International, supra* note 11.

[127] *Ibid.* at 257.

removing components from an employer's warehouse in a lunch pail and attempting to sell the assembled product.[128]

A more substantial case involving the protection of non-literal elements of a computer program is *Delrina*.[129] This case again involved a former employee. While employed by the plaintiff, the employee developed a performance monitoring tool (Sysview). After leaving the plaintiff, the employee developed a competitive product (Assess) which was marketed through the corporate defendant. Much of the case was argued on the basis that there was literal copying.[130] However, the court took the opportunity to comment on the copyrightability of non-literal elements of computer programs. The defendants took the position that the statutory definition of "computer program" as a set of instructions embodied or stored in any manner implies that only the literal instructions are subject to copyright protection.[131] The court responded with the following comments:

> I conclude that under U.S. copyright law and based on statutory provisions equivalent to those in the Canadian *Copyright Act*, copyright is extended to all parts of computer programs, except in the case of programs whose very purpose is to produce screen displays for use in playing of games or for some artistic or other like purpose.
>
> The attitude of the U.S. Copyright Office and that of the U.S. courts in *Computer Associates* and in *Lotus* strengthen my conclusion that both the literal and non-literal portions of the Sysview program could be protected by copyright under the *Copyright Act*.[132]

While acknowledging that there are differences between the U.S. and Canadian *Copyright Acts*, the court adopts much of the U.S. jurisprudence. An approach such as this is not without its dangers.[133] In addition to the differences between the respective *Acts*, there are also differences in judicial approach to statutory interpretation. American judges, as compared to their Canadian counterparts, are much more inclined to indulge in considerations of policy and legislative history.[134] In this regard, policy choices made by American courts (which forward American goals) may not forward Canadian goals.[135]

---

[128] *Ibid.*

[129] *Delrina*, *supra* note 11.

[130] The claim of literal copying was based on the following allegations:

- Assess could not have been created in so little time unless there was a copying of Sysview;
- the similarities between Assess and Sysview are so great that the former must have been copied into the latter; and
- there is intrinsic evidence of copying in that both programs contain the same unusual phrasing and mistakes (bugs).

[131] *Supra* note 11 at 27.

[132] *Ibid.* at 32.

[133] See Estey J.'s comments, *supra* note 19.

[134] In this area, the U.S. courts have given significant consideration to the National Commission on New Technological Uses of Copyrighted Works (CONTU). Strictly speaking, CONTU is not legislative history, though it has been accorded similar status by U.S. courts.

[135] See Vaver, *supra* note 20.

*Delrina*, in addition to deciding that there should be protection of the non-literal elements of computer programs, also adopted a number of other important principles from U.S. jurisprudence. The first of these was the idea/expression dichotomy. While the idea/expression dichotomy is a part of Anglo-Canadian jurisprudence, the court seems to be endorsing the classic American position. The second principle adopted by the *Delrina* court relates to the treatment of utilitarian elements:

> Even if the expression originated with the author, the expression of the idea is not copyrightable if the expression does no more than embody elements of the idea that are functional in the utilitarian sense. [Citations omitted][136]

In the following section it will be suggested that the adoption of these principles may not have been open to the court in light of the Canadian *Act* and existing Anglo-Canadian jurisprudence.[137]

### (b) *Possible Protection of Non-Literal Elements Under Canadian Law*

The extension of copyright protection to the non-literal elements of computer programs in the United States was based on an analogy to the protection of non-literal elements of other works. There is Canadian[138] and Commonwealth[139] authority supporting the protection of the structure, organization, and framework of copyrighted works, including the selection of ideas, patterns of incidents, and compilation of information therein. Based on this authority it is open to Canadian courts to adopt this analogy and extend copyright protection to the non-literal elements of computer programs. This was the approach taken in *Delrina*.[140] It is worth noting that when *Whelan* adopted this approach it was following well-established American precedent.[141] In *Delrina* the justification for this approach was taken from U.S. sources since this area of copyright law is relatively undeveloped in Canada.[142] This suggests that should this issue arise before other Canadian courts, consideration should be given to whether Canadian copyright protection should be extended in this manner.

---

[136] *Delrina, supra* note 11 at 41.

[137] While the adoption of these principles may not have been open to the court, it is not being suggested that Canadian legislators should not examine the American position on copyright protection of computer programs (particularly the position on non-protection of utilitarian, functional programs).

[138] *Preston* v. *20th Century Fox Canada Ltd.*, 33 C.P.R. (3d) 242, 38 F.T.R. 183 (F.C.T.D.) [hereinafter *Preston*] alleged similarities between Space Pets and Revenge of the Jedi. *Hutton* v. *Canadian Broadcasting Corporation* (1989), 29 C.P.R. (3d) 398, 102 A.R. 6 (Q.B.), aff'd (1992), 41 C.P.R. (3d) 45, 120 A.R. 291 (C.A.) alleged similarities between Star Tracks and Good Rockin' Tonight.

[139] B.B. Sookman, *Computer Law: Acquiring and Protecting Information Technology* (Toronto: Carswell, 1989) at 3-160. The following cases are cited for this proposition: *Ainsworth Nominees Pty. Ltd. & Anor* v. *Andclar Pty. Ltd.* (1988), A.I.P.C. 90,550 (Fed. Ct. Aust.); *Green* v. *Broadcasting Corp. of New Zealand*, [1989] R.P.C. 469 (N.Z.C.A.), aff'd [1989] R.P.C. 700 (P.C.); *Ravenscroft* v. *Herbert and New English Library Ltd.*, [1980] R.P.C. 193 (Ch. D.); *Zeccola* v. *Universal City Studios*, [1982] A.I.P.C. 90,019 (Fed. Ct. Aust.); *AGL Sydney Ltd.* v. *Shortland County Council* (1990), A.I.P.C. 90,166 (Fed. Ct. Aust.); *Galago Publishers (Pty.) Ltd.* v. *Erasmus*, [1989] 1 S.A. 276.

[140] The court quoted from *Computer Associates* with respect to the analogy proposed by *Whelan*. See *Delrina, supra* note 11.

[141] *Whelan, supra* note 5.

[142] *Preston, supra* note 138.

A fundamental principle of Anglo-Canadian copyright law has been the protection of skill-and-labour. This principle can at times conflict with the idea/expression dichotomy. While the idea/expression dichotomy has been recognized in Anglo-Canadian jurisprudence,[143] it has not been as strongly applied in Canada and the Commonwealth[144] as it has been in the United States.[145] The strong application of the idea/expression dichotomy in the United States has led to a rejection of copyright based on the protection of skill-and-labour.[146] This has not been the case in Canadian law;[147] however, no Canadian court has analyzed the protection of non-literal elements of a computer program in terms of the protection of skill-and-labour. A detailed analysis of possible protection under this doctrine will be given in the next section.

Another area where Canadian and American copyright law differs is in the protection accorded useful articles. The Canadian position has been to treat utilitarian works no differently than non-utilitarian works. American copyright law has drawn a strong distinction between utilitarian works and non-utilitarian works. In application, this principle has been used to deny copyright protection to any item that has utilitarian function except to the extent that its artistic features can be identified separately.[148] The early Canadian position is illustrated in *Cuisenaire* v. *South West Imports Ltd.*:[149]

> Although there are some old decisions, such as *Baker v. Selden* referred to by Pape J. in *Cuisenaire v. Reed* which refuse the protection of copyright to objects which have a functional use or which could form the subject of a patent of invention, there is nothing that I can see in the *Copyright Act* to support the argument that intended use or use in industry of an article or its patentability otherwise eligible for copyright bars or invalidates registration or protection and I cannot read such a limitation into the *Copyright Act.* [Citations omitted][150]

---

[143] *Hollinrake* v. *Truswell*, [1894] 3 L. R. Ch. 420 (C.A.) at 428:

> No doubt one may have copyright in the description of an art: but, having described it, you give it to the public for their use; and there is a clear distinction between the book which describes it, and the art or mechanical device which is described.

[144] *Apple* v. *Mackintosh*, *supra* note 11:

> There is another branch of the merger doctrine which has prevailed in the United States and which it is necessary to consider. It would appear to have originated with the decision in *Baker v. Selden*, a case referred to in several "commonwealth" decisions, but without reliance on the full scope of the decision given therein (citations omitted) (footnotes omitted).

[145] *Delrina* adopted the idea/expression dichotomy without reference to this difference. Given the strong reliance on American cases, this difference should have been highlighted.

[146] *Feist Publications, Inc.* v. *Rural Telephone Service Company, Inc.*, 111 S.Ct. 1282 at 1289-90 (1991):

> It may seem unfair that much of the fruit of the compiler's labor may be used by others without compensation. As Justice Brennan has correctly observed, however, this is not "some unforeseen byproduct of a statutory scheme" (citations omitted).

See also *Miller* v. *Universal City Studios, Inc.*, 650 F.2d 1365 (5th Cir. 1981), for criticisms of the "sweat of the brow" approach.

[147] *Supra* note 121.

[148] See *supra* note 113 and accompanying text.

[149] (1967), 54 C.P.R. 1, 37 Fox Pat. C. 93 (Ex. Ct.).

[150] *Ibid.* at 14.

The British case of *Dorling* v. *Honnor Marine Ltd.*[151] gave rise to the possibility that copyright in artistic works might be infringed by copying the useful articles illustrated in those drawings.[152] This trend eventually surfaced in a number of Canadian decisions. In *Doral Boats Inc.* v. *Bayliner Marine Corp.*,[153] Walsh J. relied on *King Features Syndicate, Inc.* v. *O. & M. Kleemann, Ltd.*[154] and *L.B. (Plastics) Ltd.* v. *Swish Products Ltd.*[155] to support the following conclusion:

> [I]t would appear that the drawings are clearly subject to copyright as literary works and s.3 of the *Copyright Act* gives protection to the work or any substantial part thereof "in any material form whatever" and the weight of jurisprudence seems to support the proposition that copying an object made from a drawing, even if the drawing itself is not used, constitutes an infringement.[156]

*Doral Boats* was reversed by the Federal Court of Appeal on other grounds.[157] However, this reasoning was subsequently adopted by the British Columbia Supreme Court in *Spiro-Flex Industries Ltd.* v. *Progressive Sealing.*[158] Gibbs J. discussed *Doral Boats* in light of the then-recently released *British Leyland* v. *Armstrong*[159] decision and concluded:

> [T]he argument is that in *British Leyland Motor Corp. Ltd. v. Armstrong Patents Co. Ltd.*, handed down after the judgement of Walsh J. in *Bayliner*, but before the Court of Appeal judgement in *Bayliner*, by implication the House of Lords confined the *King Features* copying aspect to indirect reproduction or copying of drawings which have artistic merit, as opposed to those which depict purely functional objects devoid of eye appeal.
>
> On the issue before the court here, the *British Leyland* case is not very helpful. The result in that case turned upon an interpretation of the *Copyright Act*, 1956 (U.K.) which is so vastly different from the Canadian Act that English copyright decisions after 1956 must be applied with extreme caution. Furthermore, with the greatest respect to their noble Lordships, it is doubtful if remarks made in *obiter* which appear to consider objects of artistic merit more deserving of copyright protection than those of purely functional merit would find widespread acceptance in Canada. [Citations omitted][160]

---

[151] [1964] 1 All E.R. 241, [1964] R.P.C. 610 (C.A.).

[152] The reasoning adopted by the British courts subsequently spread to other jurisdictions: see *Wham-O Manufacturing Co.* v. *Lincoln Industries Ltd.*, [1985] R.P.C. 127 (N.Z.C.A.); *Allibert S.A.* v. *O'Connor*, [1981] F.S.R. 613, [1982] F.S.R. 317 (H.C.).

[153] *Doral Boats*, *supra* note 23.

[154] [1941] 2 All E.R. 403, 58 R.P.C. 207 (H.L.).

[155] [1979] R.P.C. 551, [1979] F.S.R. (H.L.).

[156] *Doral Boats*, *supra* note 23 at 305-06.

[157] *Ibid.*

[158] *Spiro-Flex Industries*, *supra* note 23.

[159] [1986] 1 All E.R. 830, [1986] F.S.R. 221 (H.L.).

[160] *Spiro-Flex Industries*, *supra* note 23 at 210-11.

In 1988, section 64.1 (then s. 46.1)[161] was introduced to allow copying of the solely utilitarian features of an article without infringement of copyright or moral rights. According to William Hayhurst,[162] paragraph 64.1(1)(a) specifically overrules the "troublesome" *Spiro-Flex* decision. The wording, however, also seems wide enough to include the utilitarian aspects of computer programs. According to Webster's Third New International Dictionary, a "thing" is "an entity that can be apprehended or known as having existence in space or time as distinguished from what is purely an object of thought." A computer program is capable of "being apprehended or known as having existence". Thus, it would seem that a computer program could qualify as an article for the purposes of section 64.1, being "a thing made by hand, tool or machine". There is certainly no doubt that computer programs perform utilitarian functions. Therefore, a computer program would seem to qualify as a useful article, with the result that it should not be an infringement of copyright to reproduce features which are "dictated solely by a utilitarian function of the article".[163]

In *Delrina* the court cited *Lotus* v. *Paperback*[164] for the proposition that:

> Even if the expression originated with the author, the expression of the idea is not copyrightable if the expression does no more than embody the elements of the idea that are functional in the utilitarian sense.[165]

However, based on the analysis above, it would seem that if the court was to grant an exemption from copyright protection with respect to utilitarian features it should have

---

[161] *Copyright Act*, S.C. 1988, c. 10, s. 11:

> s. 64 (1) In this section and section 64.2,
> "article" means any thing that is made by hand, tool or machine;
> "design" means features of shape, configuration, pattern or ornament and any combination of those features that, in a finished article, appeal to and are judged solely by the eye;
> "useful article" means an article that has a utilitarian function and includes a model of any such article;
> "utilitarian function", in respect of an article, means a function other than merely serving as a substrate or carrier for artistic or literary matter.
>
> s. 64.1
> (1) The following acts do not constitute an infringement of the copyright or moral rights in a work:
> (a) applying to a useful article features that are dictated solely by a utilitarian function of the article;
> (b) by reference solely to a useful article, making a drawing or other reproduction in any material form of any features of the article that are dictated solely by a utilitarian function of the article;
> (c) doing with a useful article having only features described in paragraph (a), or with a drawing or reproduction made as described in paragraph (b), anything that the owner of the copyright has the sole right to do with the work; and
> (d) using any method or principle of manufacture or construction.

[162] W.L. Hayhurst, "Intellectual Property Protection in Canada for Designs of Useful Articles: Sections 46 and 46.1 of the Copyright Act" (1989) 4 I.P.J. 381.

[163] Sookman, *supra* note 139 at 3-139 also takes the position that computer programs may fall within the scope of s. 64.1.

[164] *Supra* note 5.

[165] *Delrina, supra* note 11 at 41.

done so through section 64.1. Since the court did not turn its attention to section 64.1, two questions remain: first, does the section 64.1 exemption include the utilitarian features of computer programs? And second, if the exemption does apply, what is the scope of this exemption?

It is submitted that the statutory language of section 64.1 does include computer programs. The question of scope is more difficult. The inclusion of the word "solely" indicates that the scope of the exemption may be limited. The U.S. experience has shown that there is a reluctance on the part of the judiciary to accept that the design of computer programs is based solely on functional-utilitarian considerations.[166] Since there have been so few cases on this issue in Canada, it is likely that the Canadian judiciary will, initially, also be reluctant to adopt this view.

### (c) *Skill-and-Labour Based Protection of Non-Literal Elements*

As mentioned in the previous section, the strong application of the idea/expression dichotomy in the United States has resulted in a rejection of the "sweat of the brow" doctrine.[167] However, in Anglo-Canadian jurisprudence the idea/expression dichotomy has been applied more weakly, and, as a result, copyright protection can be granted for the protection of skill-and-labour.[168]

In *Ladbroke (Football)* v. *William Hill (Football)*,[169] the House of Lords held that the preparatory work done by the plaintiff bookmakers in compiling lists for betting could not be excluded in determining whether copyright existed. Under this approach the court inquired whether the work produced by the plaintiffs through the expenditure of their skill-and-labour was worthy of protection. In this regard reference is often made to the *dicta* of Peterson J. in *University of London Press*[170] that "there remains the rough practical test that what is worth copying is prima facie worth protecting."

Compared to protection based on the idea/expression dichotomy, protection of skill-and-labour seems to have some conceptual advantages. The primary flaw in the idea/expression approach is that it requires a search for something that is not generally present in a computer program – expression. However, a great deal of skill-and-labour will have been expended in the creation of a fully debugged computer program. Thus, a skill-and-labour based approach at least protects something that is present and identifiable in a computer program.

The type of protection provided under a skill-and-labour based scheme would be similar to that which is currently provided for compilations. The Canadian law with respect to compilations is outlined by Fox in *The Canadian Law of Copyright and Industrial Designs*:[171]

[166] This has been particularly true in the case of user interfaces. However, recent decisions indicate that American judges are developing a more sophisticated understanding of the computer industry. This change is probably attributable to the large number of cases that have been litigated and the vigorous debate that has taken place in the academic journals.

[167] *Supra* note 146.

[168] *Supra* note 121.

[169] *Ibid.*

[170] *Supra* note 12 at 610.

[171] *Fox, supra* note 21.

> [M]ere similarity between the two works is insufficient to found a case of infringement. Where an author shows that his work was independently compiled from original but common sources, no amount of similarity, in the absence of any evidence of copying, will be held to be an infringement. [Footnotes omitted][172]

> In the preparation of any compilation a second compiler must prepare his work by his own independent labour and research. He may go to common sources for materials, make his own calculations and prepare his own work. But he must do this himself. It is not sufficient for him to say or to show that common sources of information existed. He must go to those sources and obtain his information from them. Provided the second work originated from his own resources and through his own labour, the author thereof may produce a work in the same general form as an earlier work on the same subject and in so doing he may (a) use all common sources of information; (b) use the earlier work as a guide to those common sources; and (c) use the earlier work to test the completeness of his own. But it is a piratical use of the earlier work if the second author takes therefrom the matter borrowed from common sources, in order to save his own labour and expense of consulting the original work. [Footnotes omitted][173]

Proof of similarity would be *prima facie* evidence of infringement. This would result in a reversal of the burden of proof, thus requiring the defendant to show that the similarity was not a result of copying. The presumption of infringement could be rebutted by bringing forward evidence of independent creation.[174] As modified for computer programs, the scheme outlined above would prohibit copying from an initial developer's source code.[175] A subsequent developer would be obliged either to go to the original materials from which the first program was developed, or to reverse engineer the first program from its object code. This type of protection would be stricter than the current regime in the United States. According to current American jurisprudence a subsequent developer can directly copy literal source code in certain instances, for example, when the source code is in the public domain, or if it is necessary for mechanical compatibility with the computer.[176] Later in this paper it will be suggested that a skill-and-labour based scheme is overly protective; however, it is not so overly protective as to be unworkable, and, in fact, this approach may have some advantages over schemes based on the idea/expression dichotomy.

While a skill-and-labour based approach is protectionist, the fact that software is almost always distributed in object code format reduces the impact on subsequent developers.[177] A purchaser of a computer program in object code format cannot see the source code version of that program. As a result, a subsequent developer is not forced to view an initial developer's implementation. This leaves the possibility of independent development. Through the use of reverse engineering techniques a subsequent developer

---

172 *Ibid.* at 355.

173 *Ibid.* at 357.

174 *Caron* v. *Association des pompiers de Montréal Inc.* (1992), 42 C.P.R. (3d) 292 at 296, 54 F.T.R. 161 at 165 (F.C.T.D.).

175 Such a scheme, taken to its logical extreme, could potentially cover the extraction of information from object code through reverse engineering techniques. However, it is more reasonable to limit the protection of skill-and-labour to the source code level. When the work is converted to its object code format, the amount of effort required to extract useful information is roughly equivalent to the effort of independent creation.

176 See *supra* note 109 which sets out relevant considerations from *Computer Associates*.

177 A prohibition on reverse engineering would have a much greater impact.

can extract the information that is needed for competitiveness or compatibility. Since reverse engineering is labour intensive, the extraction of this information should not be considered an appropriation of the skill-and-labour used in the creation of the initial work. In effect, the effort required to reverse engineer a program is a substitute for the skill-and-labour expended by the initial developer.

A copyright scheme based on skill-and-labour has another advantage over a scheme based on the idea/expression dichotomy. Distinguishing idea from expression is inherently difficult: in a given case even those with legal training can disagree on where this line should be drawn. If experts have difficulty making this distinction, it will be even more difficult for the lay person. A law must be able to provide the public with a standard by which it can judge, and if necessary, modify its behaviour. The ambiguity of the idea/expression dichotomy coupled with the inherent functionality of computer software does not provide this standard. The resulting confusion will in all likelihood lead to an increase in the amount of litigation.[178] The protection of skill-and-labour does not provide a bright-line test, but it does establish a standard that is more comprehensible to the software industry. Management can estimate the amount of time and effort that should be expended in the development of a product. If a product is developed in much less time, this should serve as a warning.[179] The use of a scheme that is more comprehensible to the industry will allow it to govern its behaviour more easily. The result should be a reduction in the amount of litigation.

Protection based on skill-and-labour would also make it easier, at trial, to determine if infringement has occurred. Under the current scheme the focus is on metaphors and analogies;[180] a skill-and-labour based approach would instead focus on the ability of the alleged infringer to produce tangible evidence of a program of legitimate reverse engineering. The lack of such a paper trail would be an indication of infringing behaviour.

While the scheme outlined above has certain advantages over the current regime, it also has certain disadvantages. The primary problem with this scheme is one that is shared with schemes based on the idea/expression dichotomy: overly broad protection is extended to functional (non-literal) features. It has already been suggested that the extension of protection to functional features is dangerous. The major flaw in a skill-and-labour based approach is that it does not exclude functional features. In this respect, protection of skill-and-labour may now be in conflict with section 64.1 of the *Copyright Act.*

A skill-and-labour based approach would also reduce access to source code.[181] Under such a scheme, subsequent developers would be obligated to access original

---

[178] See Slind-Flor, *supra* note 3.

[179] The situation in *Computer Associates, supra* note 5, is a good example. An employee of Computer Associates joined Altai and took copies of certain Computer Associates' programs with him. The management of Altai was not aware that the source code had been taken, or that it was being used in the development of Altai's competing product. By using the stolen source code, the new employee was able to develop a competing product in a short time. If copyright had been based on the protection of skill-and-labour, questions would have been raised regarding the short development period. Under this kind of scrutiny, it is likely that the appropriation of Computer Associates' source code would have been discovered.

[180] See *Apple* v. *Microsoft, supra* note 1.

[181] Assuming that *Computer Associates* is a correct statement of the current American position.

source materials. This rule is inefficient and could reduce access to new ideas and techniques. Such a reduction would affect the industry's ability to innovate. A better approach would be to allow copying of certain literal source code elements – for example, elements that are required for compatibility, efficiency, or standardization.[182]

A final concern is that this doctrine, if taken to its protectionist extreme, could regard reverse engineering as an appropriation of skill-and-labour. Although this paper has taken the position that reverse engineering should not be regarded as such, it is still possible that the courts might adopt the opposite view. In such a case, the protection granted under a skill-and-labour based scheme would be excessive.

### (d) *Proposed Solution: Prohibition of Anti-Competitive Behaviour*

Despite what might be concluded from all the high profile software copyright litigation, the major threat to intellectual property rights in computer software still comes from piracy. It was estimated that the software industry lost $1.2 billion in 1991 due to unauthorized copying.[183] Thus, the primary focus of a protective scheme should be on the prevention of piracy. Copyright protection of literal source and object code provides this type of protection. Once the piracy issue has been addressed, the next question is how much protection should be granted beyond this "base" level. It is submitted that any "higher level" protection should only be concerned with anti-competitive behaviour. The primary object should be to ensure that the level of copyright protection does not prevent the industry from advancing the state of the art.[184] In order to achieve this goal it will be necessary to balance the return given to the author with the right of public access to the work. Because of the functional nature of computer software, and the incremental and cumulative fashion in which the state of the art advances, protection beyond the literal code should be set at the lowest level that allows an equitable return to the author. At this level of protection the public benefit should be maximized. In determining what

[182] See *supra* note 109. The *Computer Associates* filtration stage effectively allows copying of these elements.

[183] C. McManus, "Software Piracy Still Costs Vendors Billions" *Computer Shopper* vol. 13 no. 7 (July 1993) 648. B.A. Nilsson, "Cracking Down On Computer Counterfeiters" *PC-Computing* vol. 5 no. 7 (July 1992) 188, describes a computer bulletin board service which provides access to the executable versions of the latest application programs. The article also lists the "Top 10 Pirate BBS Downloads":

1. Windows 3.1 (Microsoft)
2. Excel 4.0 (Microsoft)
3. Norton Utilities 6.0 (Symantec)
4. WordPerfect for Windows 5.1 (WordPerfect)
5. Stacker 2.0 (Stac Electronics)
6. AutoMap (AutoMap)
7. Procomm Plus 2.0 (Datastorm Technologies)
8. PC Tools Deluxe 7.1 (Central Point Software)
9. QEMM-386 6.0 (Quarterdeck Office Systems)
10. WordPerfect 5.1 (WordPerfect)

This list was very similar to the top 10 list for programs being sold through legitimate sources.

[184] Copyright is granted for the purpose of advancing learning. The reward to the author is the means by which this goal is achieved. See *supra* note 21.

this level should be, it will be instructive to examine the treatment of other technological works.

The traditional treatment of technological works has been to grant patent protection to those advances which meet the requirements of novelty and non-obviousness. All other features of a technological work can be reverse engineered for the purpose of developing a competitive product. A manufacturer who develops a product which has non-patentable features, while unable to obtain monopoly protection for these features, still retains the advantage of being first to market. Competitors who want to provide the same features have to reverse engineer the product, tool up, and then market their competing version. The problem with software is that this competitive advantage can be eliminated through literal copying. The way to combat this problem is to ensure that the traditional advantage is maintained. Thus, a protective scheme should seek to prevent competitors from taking "short cuts". In this respect, the type of protection to be provided will be similar to, but weaker than, the protection provided under a skill-and-labour approach. Under this scheme the main focus will be on whether there has been appropriation without sufficient analysis.[185] The first stage of the investigation consists of an examination designed to identify "surprising similarities".[186] The presence of such similarities would give rise to a presumption that their presence was due to anti-competitive behaviour. The alleged infringer could rebut this presumption by demonstrating a sufficient program of study and analysis (reverse engineering). If such a paper trail can be properly established, there has not been anti-competitive behaviour, and thus there is no infringement.

A fundamental difference between this type of protection and protection based on skill-and-labour lies in the fact that a subsequent developer can use an initial developer's work. As long as it can be demonstrated that there was sufficient analysis of the first work,[187] use of the features of that work in a competitive work will not be considered an infringement. A further difference between this approach and the traditional skill-and-labour approach is that the focus is on the skill-and-labour of the subsequent developer. If sufficient skill-and-labour is used by a subsequent developer there is no anti-competitive behaviour with respect to the initial developer.

A scheme based on the prevention of anti-competitive behaviour retains the advantages that the skill-and-labour based approach had over the idea/expression approach: by examining skill-and-labour instead of idea/expression, the investigation focuses on something that is clearly present in a computer program. A further advantage of this scheme is that it allows the industry to govern itself more easily. The average software engineer will know what constitutes a legitimate program of reverse engineering; the average software engineer will not be able to distinguish between idea and expression.

[185] Karjala, *supra* note 7 first proposed this type of scheme.

[186] *Ibid.* at 87. Some examples of such similarities are:

- non-functional code;
- inefficient code;
- traps.

[187] It would also be permissible to take that which is in the public domain.

The prohibition against anti-competitive behaviour does not suffer from some of the problems associated with a skill-and-labour based approach. Because subsequent developers are not prevented from examining and learning from prior works, the spread of new ideas and techniques should not be adversely affected.[188] This type of protection does not concern itself with the protection of non-literal elements, thus removing the possibility that access to functionality will be limited. In this regard there is no conflict between this approach and the principles embodied in section 64.1 of the *Copyright Act*.

A concern which may arise with respect to this proposal is that it leaves computer software under protected. However, as compared to other technologies, this does not seem to be the case. It must be remembered that software is distributed in object code format: any competitor who wants to access the functional aspects of a program must reverse engineer it. The cost of this effort will ensure that there is no anti-competitive advantage.

In addition to the protection provided by the copyright scheme proposed above, there are other types of protection which the law has traditionally extended to technological works. If a manufacturer's product meets the required criteria it can be the subject of patent protection;[189] if the ideas and techniques used are not generally known, they can be protected as confidential information (trade secrets).[190] Finally, a responsible manufacturer can protect his or her source code by ensuring that it does not fall into a competitor's hands. The use of non-disclosure contracts, employee non-disclosure agreements, and secure source code management techniques can increase source code confidentiality.

The implementation of this scheme would not seem to conflict with Canada's international obligations under the *Berne Convention*. The *Convention* requirement is stated in Article 2(3): "The countries of the Union shall be bound to make provision for the protection of the above mentioned works." Since the proposed solution provides for copyright protection of the literal source and object code, the *Convention* obligation should be met. The only possible point of contention is with respect to the failure to extend protection to the non-literal elements of computer programs. However, as discussed above, there is good reason to believe that because of the nature of computer software there is no need to extend protection to non-literal elements in order to provide

---

[188] It could be argued that under this scheme developers would be more reluctant to release their source code. However, since source code is not usually provided with a product, it does not seem likely that this would significantly affect the dissemination of ideas and information.

[189] Sookman, *supra* note 139 at 6-23:

> Although nothing in the way of comprehensive guidelines are available for determining the patentability of software-related inventions, decisions after *Schlumberger* have recognized that certain types of applications, such as control systems, data manipulation or information enhancement systems, and certain software systems, may present proper subject matter for patentability.

See also W.C. Kent & E. Cheung, "Patent Protection in Canada for Computer Related Technology" (1987) 3 C.I.P.R. 249.

[190] *Software Solutions Associates Inc.* v. *Depow* (1989), 25 C.P.R. (3d) 129, 99 N.B.R. (2d) 110 (Q.B.).

adequate protection. Under Article 4(2),[191] there would also seem to be some flexibility with respect to the scope of protection under national law.

Since there are no explicit stipulations about copyright protection extending to the non-literal elements of works, it would seem that the choice not to extend such protection would not be a violation of the *Convention*. Thus, the implementation of an anti-competitive behaviour protection scheme should not be precluded by Canada's obligations under the *Berne Convention*.

## C. *Protection of Computer User Interfaces*

The user interface is an extremely important part of a software package. Since the user interface is the part of the package with which the user becomes most familiar, it will have a large influence on the success or failure of a given application. Because of this importance, large amounts of money and effort have been invested in the development of more effective user interfaces. A consequence of this effort has been the development of the discipline known as human factors analysis. Human factors analysis is used to identify user needs. Once these needs are known, specific techniques are used to facilitate efficient computer-human interaction.[192] The emergence of the user interface as an important economic asset has led developers to seek legal protection for their proprietary interfaces. The question of how to protect computer user interfaces has been a particularly difficult one, and until recently, most courts have had difficulty accepting the proposition that user interfaces are functional and not expressive.[193] The result of this judicial reluctance has been a high level of copyright protection for user interfaces.

### 1. *The American Position on Computer User Interfaces*

The current position in the United States is unclear. While all of the cases have found that copyright protection extends to user interfaces, the scope of the protection has varied considerably. At one extreme, the *Lotus* decisions[194] favour a position that accentuates the expressive view of user interfaces. At the other extreme, the *Apple* v. *Microsoft* decisions[195] favour a view that accentuates the functional nature of user interfaces. These

[191] Article 4(2) states:

> [A]part from the express stipulations of the present Convention, the extent of protection, as well as the means of redress secured to the author to safeguard his rights, shall be governed exclusively by the laws of the country where protection is claimed.

[192] Menell, *supra* note 7, identifies the following five human factors goals:

> (1) minimization of learning time;
> (2) maximization of speed of performance;
> (3) minimization of the rate of user error;
> (4) maximization of user satisfaction;
> (5) maximization of user retention of knowledge over time.

[193] See *supra* note 36 and subsequent text for a description of the functional considerations related to the creation of a user interface.

[194] *Lotus* v. *Paperback, supra* note 5; *Lotus* v. *Borland, supra* note 5.

[195] *Apple Computer, Inc.* v. *Microsoft Corp.*, 799 F.Supp. 1006 (N.D. Cal. 1992); *Apple* v. *Microsoft, supra* note 1.

views are not compatible, and ultimately it seems that this contradiction will have to be resolved by the United States Supreme Court.

Both the *Lotus* and *Apple* cases used tests that were based on the idea/expression dichotomy.[196] The courts, however, disagreed on the degree to which functionality constrained expression in the respective interfaces. In *Lotus* v. *Borland* the defendants suggested that copying of the precise menu commands and menu structures was necessary for functional compatibility. The court rejected this submission and proceeded to analyze the 1-2-3 user interface in order to identify protectable expression. The following passages give an indication of the approach taken by Keeton J.:

> A very satisfactory spreadsheet menu tree can be constructed using different commands and a different command structure from those of Lotus 1-2-3. In fact, Borland has constructed just such an alternative tree for use in Quattro Pro's native mode. Even if one holds the arrangement of menu commands constant, it is possible to generate literally millions of satisfactory menu trees by varying the menu commands employed.[197]
>
> I conclude that it cannot be genuinely disputed that a large part of the structure and arrangement of the menu commands is not driven entirely by functional considerations. There are sufficient non-functional aspects that at least hundreds and perhaps thousands of different expressions of the function were possible when Lotus chose the particular structure of menu commands incorporated into Lotus 1-2-3.[198]

The problem with the position adopted by the *Lotus* court is that it grants a *de facto* monopoly to any user interface developer whose product gains a dominant market

[196] The *Lotus* court used the "Substantial Similarity and Copyrightability" test: see *supra* note 5. The 1992 *Apple* court used the "Extrinsic-Intrinsic" test:

> The Ninth Circuit employs a two-part test to determine whether a work infringes the copyright in another work. First, the "ideas" of the works in suit are compared for substantial similarity, using an "extrinsic test" or "objective analysis of expression." Analytic dissection, employing a list of criteria of comparison informed by expert testimony, is a part of this exercise, which makes this well-suited for determination as a matter of law.
>
> If the ideas are substantially similar, then an "intrinsic test" or "subjective analysis of expression" is used....Moreover, the "intrinsic test" entails a comparison of the portions of a work that can be the subject of copyright protection....If the similarity of the works in suit stems solely from unprotectable features, then the plaintiff's case is missing an essential element of infringement (citations omitted).

At 1021, the court states:

> Merger means there is practically only one way to express an idea. But if technical or conceptual constraints limit the available ways to express an idea, even though there is more than one avenue of expression available, copyright law will abhor only a virtually-identical copy of the original (citations omitted).

[197] *Lotus* v. *Borland*, *supra* note 5 at 217.

[198] *Ibid.* at 218. It is apparent that the *Lotus* court has adopted the expressive view as opposed to the functional view of user interfaces. Under the expressive view, it is assumed that any one of the thousands of different versions of the user interface is as good as any of the others. However, under the functional view, using human factors analysis, some versions of the user interface will be measurably superior to others.

position. Once users become accustomed to a particular system they will not want to incur the cost of re-education in order to switch to another product. An equally good user interface cannot be independently developed, because, in the context of user interfaces, "equally good" means exact functional duplication.[199] The effect of this type of copyright protection is to grant a patent-like monopoly.

The *Apple* court applied the "extrinsic-intrinsic" test,[200] which gives consideration to the need to standardize functionality:

> Under the law of this Circuit, an article which has "*any* intrinsic utilitarian function" can be denied copyright protection "except to the extent that its artistic features can be identified separately and are capable of existing independently as a work of art." The very purpose of a computer user interface "lies in its ability to help people prepare and analyze their work quickly and flexibly." Copyright protection can attach only to such a product's separate artistic features or can afford only such limited protection as appropriate when its features are the product of a compilation.
>
> The similarity of such functional elements of a user interface or their arrangement in products of like kind does not suggest unlawful copying, but standardization across competing products for functional considerations. Standardization of the visual features in a computer's interface helps to achieve its purpose, a point which Apple learned early on when it insisted on interface uniformity for Macintosh applications and which has also been implicitly recognized by this court. [Emphasis in original] [Citations omitted][201]

The reasoning in the *Apple* decision indicates that the court recognizes and accepts the need for standardization of the functional features of computer interfaces.[202] The test enunciated by the court provides subsequent developers with much more of an opportunity to develop a standardized competitive product. The extrinsic test allows the

---

199 See *supra* note 45 and accompanying text.

200 *Supra* note 196.

201 *Apple* v. *Microsoft*, *supra* note 195 at 1023 (1992 decision).

202 Contrast the *Apple* position with that taken in *Lotus* v. *Paperback*, *supra* note 5 at 78:

> Defendants have argued that 1-2-3, and specifically, 1-2-3's menu structure and macro command facility, has set a *de facto* industry standard for all electronic spreadsheets. Thus, defendants had no choice, they argue, but to copy these expressive elements from 1-2-3. Had they not copied these elements (including the macro facility), users, who had been trained in 1-2-3 and had written elaborate macros to run on 1-2-3 spreadsheets, would be unwilling to switch to VP-Planner. VP-Planner would be a commercial failure. Neither the factual nor the legal predicate of the argument is supportable.

And again at 79:

> Defendants' standardization argument is flawed for another reason as well. As explained above, one object of copyright law is to protect expression in order to encourage innovation. It follows, then, that the more innovative the expression of the idea is, the more important is copyright protection for that expression. By arguing that 1-2-3 was so innovative that it occupied the field and set a *de facto* industry standard, and that, therefore, defendants were free to copy plaintiff's expression, defendants have flipped copyright on its head. Copyright protection would be perverse if it only protected mundane increments while leaving unprotected as part of the public domain those advancements that are more strikingly innovative.

inclusion of all of the functionality of the prior product. The intrinsic test, which is applied to the "artistic features", is strictly circumscribed, requiring virtual identity for a finding of infringement.[203]

The courts in *Lotus* and *Apple* v. *Microsoft* clearly differ on the question of whether user interfaces are expressive or functional. As discussed earlier, an understanding of the goals and techniques underlying user interface design lead to the conclusion that user interfaces are functional. On this basis, the approach taken in *Apple* v. *Microsoft* would seem to be preferable.

### 2. *Protection of User Interfaces in Canada*

As with the Canadian law on the protection of non-literal elements of computer software, the law on the protection of computer user interfaces is still largely undefined. The cases that dealt with non-literal elements have generally also dealt with user interfaces; unfortunately, the treatment to date has been cursory.

#### (a) *Canadian Case Law*

The issue of the protection of computer user interfaces first arose in the context of an interlocutory injunction in *F & I Retail Systems Ltd.* v. *Thermo-Guard Automotive Products of Canada Ltd.*[204] Although user interfaces were not mentioned in the actual judgment, both source code and screen displays were given copyright protection.[205] Since the issue arose in the context of an interlocutory injunction, there is very little analysis of the substantive law, and as a result this case is of little value as precedent. The issue of the protection of user interfaces next arose in *Gemologists International*.[206] Once again, in the context of an interlocutory injunction, it was held that there was infringement because of the copying of "the program sequence of menus and menu options".[207] As discussed with respect to the protection of non-literal elements, the reasoning in this case is very limited, and makes no reference to the *Copyright Act* or any previous copyright case law. The value of this case in terms of precedent is also very limited.

A more substantial analysis of the protection of user interfaces is given in *Delrina*.[208] In this case the Ontario Court (General Division) extended protection to user interfaces based on section 3 of the *Copyright Act*:

> In my view, the words in s. 3(1) of the Canadian Act which define "copyright" as "the sole right to produce or reproduce the work....in any material form whatever" are equivalent to the words in the American Act which say that copyright protection subsists "in....works....fixed in any tangible medium of expression....from which they can be perceived or reproduced....either directly or with the aid of a machine or device"....

[203] *Supra* note 195 at 1041-42 (1992 decision).

[204] *Supra* note 11.

[205] R. Highley, "Copyright Law and Computer Screen Displays" (1990) 48 U.T. Fac. L. Rev. 48 at 81.

[206] *Supra* note 11. For a discussion of the facts of this case, see section IV. B. 2(a) (Protection of Non-Literal Elements of Computer Software – Canadian Case Law).

[207] *Ibid.* at 257.

[208] *Supra* note 11. For a discussion of the facts of this case, see section IV. B. 2(a) (Protection of Non-Literal Elements of Computer Software – Canadian Case Law).

> I conclude that under U.S. copyright law and based on statutory provisions equivalent to those in the Canadian *Copyright Act*, copyright is extended to all parts of computer programs, except in the case of programs whose very purpose is to produce screen displays for use in playing of games or for some artistic or other like purpose.[209]

Although the analysis is unclear, the court seems to be suggesting that a user interface can be protected as a reproduction of the underlying computer program in another material form, or, alternatively, as a non-literal element of the underlying program. As discussed previously, protection of non-literal elements is probably supportable under current Anglo-Canadian jurisprudence. However, reproduction of the underlying program in another material form does not appear to have been previously considered.

As a reproduction of a literary work in another material form, copyright could arguably be extended to a user interface. This kind of approach has been approved for other types of works in prior Anglo-Canadian jurisprudence.[210] The extension of this doctrine to include computer interfaces may be problematic. The Canadian position with respect to fixation of a work is stricter than that in the United States. On this basis it is possible that a computer user interface cannot stand alone as a copyrightable work.[211] A further problem arises from the fact that there are many different source code implementations which can generate the same user interface. It may not be correct to say that the user interface is a reproduction of each of these underlying versions.[212]

After the *Delrina* court concluded that copyright extended to computer user interfaces, it went on to consider the scope of that protection. In establishing the scope of copyright protection to be given to non-literal elements and user interfaces, the court considered *Lotus* v. *Borland*, *Computer Associates* v. *Altai*, and *Apple* v. *Microsoft*. Based on the passages quoted in the decision, it seems that a more restrictive scope of protection is favoured:

> Whether a Canadian court should adopt the abstraction-filtration-comparison method in deciding an action for copyright infringement or some other similar method, it seems

---

[209] *Ibid.* at 32.

[210] See *King Features Syndicate Inc.* v. *O. & M. Kleemann, Ltd.*, [1941] 2 All E.R. 403 at 414, 58 R.P.C. 207 at 218 (H.L.) [hereinafter *King Features* cited to All E.R.]:

> In the present case, on a careful comparison of the brooches and the sketch, I find substantial similarity between the different editions of the brooch and sketch No. 3, sufficient to raise a *prima facie* case of actual copying. The respondents have called no evidence to rebut that *prima facie* case. I agree, accordingly, with Simonds J., in finding that copying is proved in respect of the brooches, equally with the dolls and toys.

[211] In the Australian case of *Autodesk Inc.* v. *Dyasan* (1990), 18 I.P.R. (Fed. Ct. Aust.), rev'd (1992), A.I.P.C. 90,855 (Aust. H.C.), Sheppard J. of the Australian Federal Court concluded that the display of images on a screen did not constitute the making of a reproduction.

[212] The decision in *King Features*, *supra* note 210, was not made on the basis of one sketch. The plaintiffs provided 55 examples of the Popeye character. While all of the Popeyes look more or less the same, computer programs which implement the same user interface can be very different. For example, they need not use the same high-level language, processor, operating system, or programming paradigm.

clear that before a computer program or some part of it can be held to be copyrightable, some method must be found to weed out or remove from copyright protection those portions which, for the various reasons already mentioned, cannot be protected by copyright. After the portions that are not copyrightable have been filtered out, there may not be any kernels or golden nuggets left to which copyright can attach.[213]

The following comments were made with respect to the user interface at issue:

As already mentioned, Carolian spent considerable time at the trial attempting to establish that Duncombe copied the look and feel of the Sysview user interface (the display screens and keyboard commands). That proposition presupposes that Sysview's user interface or substantial portions of it are copyrightable. After considering all the evidence, I am unable to conclude that any substantial portion of the Sysview interface is copyrightable.

The screen displays simply record for the user of the program the results of the search being made as to how the various capabilities of the HP3000 are then being used. While some variations can take place as to the layout and content of any screen, those variations are limited. The screen must display the result of the search and must express that result in percentable [*sic*] of total capacity of the computer, absolute numbers or some combination of both. Similarly, the screen display can only be efficiently and conveniently arranged in a limited number of ways when one considers that we are trained to read from left to right and top to bottom.[214]

The approach taken in *Delrina* seems to favour the view of a user interface as a functional work. In this respect, the analysis in *Delrina* is closer to that in *Apple* v. *Microsoft* than that in the *Lotus* decisions. While this paper has taken the view that this approach is preferable, it is not clear whether the *Apple* treatment of the copying of functional elements is sustainable under the Canadian *Copyright Act*. In this respect the court should have given consideration to section 64.1. The discussion of Canadian copyright protection of functional works given previously is also relevant to the protection of computer user interfaces.[215]

### (b) *Possible Protection of User Interfaces Under Canadian Law*

The first question to be addressed with respect to the current Canadian law is whether computer user interfaces are covered by copyright at all. The definition of "computer program"[216] contained in the *Copyright Act* does not explicitly include user interfaces. It has been suggested in the academic literature that the lack of a separate category for audiovisual works in the Canadian *Act* makes the protection of screen

[213] *Delrina, supra* note 11 at 37.

[214] *Ibid.* at 44. The interface in this case is similar to the one at issue in *Softklone, supra* note 5. The *Softklone* court found that this type of interface was copyrightable as a compilation.

[215] See generally section IV. B. 2(b) (Possible Protection of Non-Literal Elements Under Canadian Law).

[216] *Copyright Act, supra* note 9 at s. 1(3).

"computer program" means a set of instructions or statements, expressed, fixed, embodied or stored in any manner, that is to be used directly or indirectly in a computer in order to bring about a specific result;

displays produced by any sort of computer program problematic.[217] However, in light of U.S. Copyright Office practice[218] and Canadian[219] and American[220] case law, it seems likely that the lack of an audiovisual category will only affect video game displays.[221] If Canadian courts choose to extend protection to computer user interfaces, there are a number of possible approaches. The key question is whether any of these approaches will result in a level of protection that advances the stated goals of copyright law.

Although the basis on which protection was extended to user interfaces is not clearly stated, two possible approaches appear to arise out of *Delrina*.[222] The court seemed to intend to extend protection on the basis that a user interface is a non-literal element of the underlying computer program; however, other statements seem to give rise to the possibility of protection of user interfaces as separate works. Both of these approaches will be considered in light of existing Canadian law.

---

[217] Highley, *supra* note 205 at 83.

[218] Sookman, *supra* note 139 at 3-72 (note 3):

> The United States Copyright Office has taken the position that computer program code and screen displays are integrally related and ordinarily form a single work. Under its present practice, the office does not register separately textual screen displays, reasoning that there is no authorship in ideas, or the format, layout, or arrangement of text on a screen and that any literary authorship in the screen display would presumably be covered by the underlying computer program – itself a literary work. As the program and screen displays is considered a single work, the U.S. Copyright Office maintains it should be registered on a single application form, as one work. In order to clarify copyright claims in computer screen displays, applicants may, at their option, deposit visual or audio-visual reproductions of computer screens along with identifying material for the computer code. Where a work contains different forms of authorship, the registration class will be determined on the basis of which authorship predominates.

[219] *Delrina, supra* note 11.

[220] *Lotus* v. *Paperback, supra* note 5; *Lotus* v. *Borland, supra* note 5: protection was extended to a computer user interface as a non-literal element of the underlying computer program. *Softklone, supra* note 5: protection was extended to a user interface as a compilation.

[221] Consumer and Corporate Affairs, *From Gutenberg To Telidon: A White Paper On Copyright* (Ottawa: Supply and Services, 1984) at 11 suggests that the *Copyright Act* should be amended to include audiovisual works as a separate category.

[222] At some points the court seems to be trying to justify copyright in a user interface as a stand-alone work; at other points the court seems to be trying to justify copyright in a user interface as a non-literal element of the underlying program. An example of the court's "separate work" approach is given in *Delrina, supra* note 11 at 32:

> Keeton J. was not relying on the program's user interface, and screen displays in particular, as having been copyrighted as audiovisual works; rather, he found the user interface including the screen displays copyrightable as being literary works.

The court's synopsis of Keeton J.'s finding is incorrect. Keeton J. found the user interface to be a non-literal element of a literary work (the computer program). The user interface itself was not the literary work.

An example of the court's non-literal element approach also appears in *Delrina, ibid.* note 11 at 32:

> I conclude that under U.S. copyright law and based on statutory provisions equivalent to those in the Canadian *Copyright Act*, copyright is extended to all parts of computer programs, except in the case of programs whose very purpose is to produce screen displays for use in playing of games or for some artistic or other like purpose.

The *Delrina* court quotes extensively from American case law, but seems to imply that protection may be available to computer user interfaces as separate works.[223] O'Leary J. quoted the following passage from *Computer Associates*:

> As a caveat, we note that our decision here does not control infringement actions regarding categorically distinct works, such as certain types of screen displays. These items represent products of computer programs, rather than the programs themselves, and fall under the copyright rubric of audiovisual works. If a computer audiovisual display is copyrighted separately as an audiovisual work, apart from the literary work that generates it (i.e., the program), the display may be protectable regardless of the underlying program's copyright status.

This statement by Walker J. distinguishes game displays (audiovisual works) from normal computer program displays, which must be protected as non-literal elements of the underlying literary work. O'Leary states that he does not interpret the *obiter* statements of Walker as ruling out copyright protection for "all screen displays in all computer programs".[224] This is true; however, Walker is saying that protection will only be granted to non-audiovisual screen displays as a non-literal element of an underlying computer program. But O'Leary seems to indicate that protection can arise in some other manner, presumably as a separate work.

In *Lotus* v. *Borland*, the court clearly extended protection to the computer user interface on the basis that it was a non-literal element of the underlying program. It was also explicitly stated that the underlying program is the copyrightable work.[225] The American cases cited in *Delrina* do not stand for the proposition that the user interface is a separate copyrightable work;[226] they stand for the proposition that the user interface is protectable as a non-literal element of the underlying computer program.

The *Delrina* court's stated reliance on section 3(1) of the *Copyright Act* leads to the conclusion that protection can be granted to user interfaces, as separate works, on the

---

> The attitude of the U.S. Copyright Office and of the U.S. courts in *Computer Associates* and in the *Lotus* cases strengthens my conclusion that both literal and non-literal portions of the Sysview program could be protected by copyright under the *Copyright Act.*

[223] *Ibid.* at 31.

[224] *Ibid.*

[225] The following comments were made in *Lotus* v. *Borland*, *supra* note 5:

> The reasoning underlying the Second Circuit's "Abstraction-Filtration-Comparison" test extends beyond program code to non-literal expression. That is, although the issue in that case concerned program code and its structural aspects, the Second Circuit based its conclusions on reasons of broader application. In fact, the court explicitly approved determinations with respect to the copyright of certain nonliteral, noncode (nonstructural) aspects of the 1-2-3 spreadsheet in *Paperback.* Thus, from statements made in the Second Circuit's opinion, albeit *obiter dicta,* I draw support for my conclusion that certain expressive elements of the 1-2-3 user interface may be protected by copyright.

[226] *Lotus* v. *Borland, supra* note 5 at 208, discussed the U.S. requirement for fixation. However, this discussion is not in the context of the user interface as a separate work. Borland contended that the menu command hierarchy was a "set of functional relationships" that were nowhere displayed in the 1-2-3 user interface. Since the user interface is protected as a non-literal element of the underlying computer program, the interface's fixation requirement is satisfied through the underlying program:

> All that is required in this regard is that the expression be embodied in a copy "by or under the authority of the author" in a form "sufficiently stable to permit it to be perceived,

basis that they are a material reproduction, in another form, of the underlying computer program.[227] The *Delrina* reasoning also relies heavily on U.S. authorities such as *Lotus*, *Computer Associates*, and *Apple* v. *Microsoft*. The question of whether the user interface is a material reproduction of the underlying computer program is not part of the American approach. If the *Delrina* court was in fact following the American approach, its analysis should have been directed towards the issue of whether the allegedly infringing computer program had copied, as one of the non-literal elements, the user interface of the copyrighted computer program. While it may have been open to the *Delrina* court to analyze this as a case of reproduction in another material form, this approach is not supported by the cases and passages which were cited in the decision.

The following analysis will attempt to show that the reproduction-in-another-material-form approach is not sustainable under current Canadian copyright law. If the underlying computer program is the initial work, then the user interface as displayed on the screen must be the reproduction in another material form. However, it is submitted that a computer user interface is not a work to which copyright can attach. According to *Canadian Admiral Corp. Ltd.* v. *Rediffusion Inc.* (per Cameron J.):

> I have given careful consideration to the terms of the *Copyright Act* and more particularly to the provisions of ss. 2 and 3, and the conclusion seems inescapable – at least to me – that for copyright to subsist in a "work" it must be expressed to some extent at least in some *material form*, capable of identification and having a more or less *permanent endurance*. [Emphasis added][228]

By these criteria, a user interface displayed on a computer screen should not be sufficiently fixed in that it lacks "permanent endurance" in a "material form". If the user interface is not fixed, it will not attract the protection of copyright law, and thus cannot be a material reproduction in another form for the purposes of copyright law.

The question of fixation of active computer memory and user interfaces is an area of some debate. Laddie has taken the following position on fixation:

> Although the following represent our considered views, it must be emphasised that the matter is not free from controversy and it is for the courts to provide the authoritative solution. It is submitted, first, that a literary, dramatic or musical work is susceptible of protection if it is fixed in any kind of material form. This could be a manuscript, tape recording, a computer memory, a film, and so on.[229]

---

reproduced, or otherwise communicated for period of more than transitory duration." The output of a computer program, at least insofar as it is typical of the program, predictable from it, and directed by the operation of the program, satisfies these requirements (citations omitted).

[227] If s. 3 of the *Copyright Act* is being used to extend protection to a user interface, the reasoning must be that the user interface is a material reproduction in another form of the underlying computer program. A user interface is not a literary work, and since there is no audiovisual category, there is no copyright category into which a user interface can fall. The only possibility left under s. 3 is that of a material reproduction in another form.

[228] (1954), 20 C.P.R. 75 at 86, [1954] Ex. C.R. 382 (Ex. Ct.) at 394.

[229] H. Laddie, P. Prescott & M. Vitoria, *The Modern Law of Copyright* (London: Butterworths, 1980) at 14.

Laddie suggests that a computer memory is a material form for the purposes of fixation. However, a computer user interface is not solely contained in memory. It is, in fact, a dynamic entity that is a composite of electrical impulses, cpu states, and memory states. It would seem that even by the Laddie criteria, a computer user interface does not meet the standard for fixation. If this view is correct, a computer user interface should not be protected by copyright. As noted by Laddie, this area is not without controversy, and it is certainly possible that the courts will reach the opposite conclusion.[230] However, in Canada, adoption of this position would seem to imply a modification of the *Canadian Admiral* doctrine.

A further difficulty with the material reproduction approach is that many different programs can produce the same user interface. Because of this many-to-one mapping, the *Delrina* reasoning results in one user interface being a material reproduction of many different computer programs. The material reproduction doctrine as enunciated by the *Copyright Act*, and interpreted by *King Features*, may not support this result. In any event, if a court decides to extend this type of protection, it should consider the possible consequences. Based on this difficulty and those cited previously, it is submitted that copyright protection cannot be extended to computer user interfaces using the material reproduction approach.

The *Delrina* decision also seemed to suggest that copyright protection could be extended to computer user interfaces as a non-literal element of the underlying computer program. This is the current U.S. approach. As was indicated earlier in this paper, current Canadian copyright law would probably support this approach.[231] However, copyright will only subsist in those elements of functional, utilitarian works that are not solely functional. The *Delrina* court engaged in a functional analysis; however, this analysis was based on the American jurisprudence. As was suggested earlier, the court's functional analysis should have been based on section 64.1 of the *Copyright Act.*

The protection of a user interface as a non-literal element of a computer program is subject to the same criticisms made with respect to the protection of other non-literal elements.[232] This paper has suggested that this choice is not in the best interests of the general public.[233] For the reasons given previously, it is recommended that this type of protection not be extended to computer user interfaces.

Some American courts have granted protection to computer user interfaces as compilations.[234] Compilations are protected by copyright through their inclusion in the definition of a "literary work".[235] As applied to computer user interfaces, this type of protection would probably be limited to the type of interfaces that were at issue in cases

[230] This is the position that has been adopted in the United States.

[231] See generally section IV. B. 2(b) (Possible Protection of Non-Literal Elements Under Canadian Law).

[232] See generally section II (The Nature of Computer Software).

[233] See generally section IV. B. 2(d) (Proposed Solution: Prohibition of Anti-Competitive Behaviour).

[234] *Supra* note 214 and reference to *Softklone*, *supra* note 5.

[235] *Copyright Act*, *supra* note 12 at s. 1(17). There have been a number of Canadian cases dealing with compilations. See *e.g. Underwriters' Survey Bureau Ltd.* v. *Massie & Renwick Ltd.*, [1940] S.C.R. 218, [1940] 1 D.L.R. 625; *Ascot Jockey Club*, *supra* note 121; *Collins* v. *Rosenthal* (1974), 14 C.P.R. (2d) 143, [1974] 1 F.C. D-747 (T.D.).

such as *Delrina* and *Softklone*.[236] These interfaces are primarily textual, and are thus more analogous to traditional compilations. This type of approach would probably have questionable effectiveness when applied to sophisticated graphical user interfaces such as those at issue in *Lotus* and *Apple* v. *Microsoft*. The question of whether a computer user interface qualifies as a compilation has not been considered in Canada. However, it is suggested that the analogy between a user interface (which is a system of prompts, inputs and response) and a traditional literary compilation is very tenuous. Even if it was found that a user interface qualified as a compilation, the scope of protection would be limited by section 64.1 of the *Copyright Act*. Any grant of protection under this approach would still be subject to the issue of fixation.

If any of the approaches suggested above are accepted and copyright protection is extended to computer user interfaces, the question of the scope of copyright protection must be considered. Because of conflicting judicial decisions, the American position with respect to the scope of protection is currently unclear.[237] The Canadian position is also unclear, however, due to the lack of case law. The only full decision at trial, *Delrina* v. *Triolet*, seems to have adopted the more restricted scope of protection, as exemplified in the United States by *Apple* v. *Microsoft*. If copyright protection does apply to user interfaces, then, of the two dominant American positions, this is the better choice. Computer user interfaces are functional, utilitarian works, and, as such, should not be accorded wide copyright protection, since this could potentially result in patent-like protection of the functional aspects of the user interface. The major problem with the approach taken in *Delrina* is that it is based primarily on American case law and has failed to consider the Canadian case law and Canadian statutory provisions. In particular, consideration should have been given to the Canadian jurisprudence dealing with the protection of utilitarian works, since these principles will also apply to computer user interfaces.[238]

### (c) *Skill-and-Labour Based Protection of User Interfaces*

As with the protection of non-literal elements, copyright protection in Canada could be extended to computer user interfaces in order to protect skill-and-labour.[239] While this approach could result in a workable, though protectionist, scheme for other non-literal elements, it breaks down completely when applied to computer user interfaces. Unlike the situation with other non-literal elements of computer programs, a skill-and-labour based scheme for user interfaces does not provide the opportunity for independent creation. The difficulty arises out of the fact that user interfaces are, by their very nature, visible. Once a subsequent developer has seen a competing user interface, he or she has seen the protected features and is no longer capable of creating an independent product. With other non-literal elements, the subsequent developer has the option of not viewing

[236] These interfaces were textual in nature, thus giving rise to a strong analogy to traditional literary compilations.

[237] See generally section IV. C. 1. (The American Position on Computer User Interfaces).

[238] See generally section IV. B. 2(b) (Possible Protection of Non-Literal Elements Under Canadian Law).

[239] See generally section IV. B. 2(c) (Skill-and-Labour Based Protection of Non-Literal Elements). See *supra* note 34 and subsequent text for a discussion of the science of human factors analysis and the development process used in the creation of user interfaces.

the protected features, thus retaining the opportunity to create an independent competitive product. The use of reverse engineering does not compromise this opportunity, since only a limited version of the program can be viewed through disassembly.[240] In the case of screen displays this choice is not present, since all of the information necessary to create a competitive product is derived directly from the copyrighted work. As Menell noted, an equally good user interface cannot be developed independently, because "equally good" in the context of standardization means exact duplication.[241] A skill-and-labour based approach requires a subsequent designer to go to the original materials in order to develop a competing product. This is not possible in the case of a user interface. The result is an intractable problem which leads to the conclusion that protection of skill-and-labour is not appropriate for computer user interfaces.

### (d) *Proposed Solution: Protection Under* Sui Generis *Legislation*

The question of whether computer user interfaces are protected by copyright is still open; however, there are strong arguments supporting the view that user interfaces are not covered by copyright law. If this is the case, the possibility of protection under a *sui generis* legislative scheme becomes possible.

There is no question that there is something of value in a well-designed computer user interface, and, on this basis, there should be some way of ensuring that the developer receives a proper return for his or her intellectual effort. The granting of a monopoly is one way to allow a developer to get a return on his or her efforts. Used carefully, a monopoly can achieve the dual purpose of encouraging creation and allowing access. The principal difficulty with the copyright monopoly is the length of the protection period: the life of the author plus 50 years is simply too long. Further difficulties arise out of the scope of the protection. As the law has developed in the United States, there is uncertainty as to what is, and what is not, protected. The average user interface designer cannot reliably distinguish between idea and expression. If *sui generis* legislation were adopted, it would be possible to address both of these concerns. The keys in developing a successful *sui generis* scheme is two-fold; first, the proper balancing of the rights of creators with the general public good, and, second, international agreement on the type of protection to be granted.

Under a *sui generis* scheme it will be important to allow creators to receive enough of a return to stimulate the creation of new works; on the other hand, it is also important to allow access to these new works, so that they can provide a basis for further development. It is submitted that these goals can be achieved by granting a short period of monopoly for newly created user interfaces. The proposed system would be similar to that used for industrial designs[242] in that the grant of the monopoly would be predicated on registration. Registration would be granted, upon inspection, if the user interface was not identical or very similar to a previously registered user interface. During the period of the monopoly there would be an absolute prohibition against the use of that particular user interface by anyone other than the registered owner (or licensee).[243] After expiration

[240] Reverse engineering does, however, allow access to the functional aspects of the work.
[241] Menell, *supra* note 7.
[242] *Industrial Design Act*, R.S.C. 1985, c. I-9.
[243] This prohibition would also cover colourable imitations of the interface.

of the protection period there would be complete freedom to use the formerly protected user interface.[244]

The success of such a scheme would depend on careful selection of the protection period. Given the rapid rate of change in the computer industry, a likely range for this period would be 2 to 5 years.[245] By forcing competitors to remain out of the market for a period of time, the initial developer would be able to take advantage of being first into the market with a new product. This should be enough of an advantage to allow a reasonable return on the development effort. Once the monopoly period has lapsed, competitors would be free to standardize their products with the formerly protected interface, thus eliminating potential problems caused by consumer unwillingness to learn a new interface. A further advantage of this scheme is that it provides developers with a definitive standard by which they can govern their behaviour; depending on registration, they either can or cannot use the interface.[246]

In determining whether Canada can implement a *sui generis* scheme, consideration must be given to the *Berne Convention*. Article 1 of the *Convention* provides that contracting states must provide for "the protection of the rights of authors over their literary and artistic works." Article 2(1) defines the term "literary and artistic works" as:

> The term "literary and artistic works" shall include every production in the literary, scientific and artistic domain, whatever may be the mode or form of its expression, such as books, pamphlets and other writings; lectures, addresses, sermons and other works of the same nature; dramatic or dramatico-musical works, choreographic works and entertainments in dumb show, the acting form of which is fixed in writing or otherwise; musical compositions with or without words; works of drawing, painting, architecture, sculpture, engraving and lithography; illustrations, geographical charts, plans, sketches, and plastic works relative to geography, topography, architecture or science.

The broadly-worded phrase "every production in the literary, scientific and artistic domain, whatever may be the mode or form of its expression" could potentially include

[244] T.S. Teter, "Merger and the Machines: An Analysis of the Pro-Compatibility Trend in Computer Software Copyright Cases" (1993) 45 Stan. L. Rev. 1061, discusses the suggestion that copyright protection should not be given to user interfaces which have become *de facto* standards. The use of a registration scheme like that used for industrial designs is a natural extension of this kind of approach in that the initial developer is given a period of monopoly during which time he or she can receive compensation for his or her efforts. Once the initial developer has been adequately compensated, the interface is then made available to the general public. This kind of registration scheme seems superior in that it provides more certainty as to which interfaces are in the public domain. It also ensures that all interfaces will eventually pass into the public domain.

[245] Consumer and Corporate Affairs, *supra* note 221. With respect to computer programs, it was suggested that:

> The term of protection for an unpublished machine-readable program will be five years from the date of creation.

While this recommendation (with respect to computer programs) is reasonable in terms of technological considerations, it fails to take into account Canada's international obligations under the *Berne Convention*. Since computer user interfaces arguably do not fall within the ambit of the *Berne Convention*, the choice of a shorter protection period should not be a problem.

[246] If there is concern about whether a particular interface is infringing, it would be possible to determine whether this is so by attempting to register it. If it can be registered, then there is no infringement.

computer user interfaces as an independent work. However, it is submitted that in order to qualify as a work under the *Berne Convention* there must be sufficient fixation.[247] On this basis, computer user interfaces, as an independent work, should not be covered because, as previously discussed, they are not sufficiently fixed.

If a user interface does not qualify as an independent work it is still possible that *Convention* protection could arise as a result of protection of the underlying computer program.[248] Under this approach the user interface would have to qualify as a non-literal element of the computer program. However, as discussed in relation to other non-literal elements of computer programs, it is unlikely that the minimum protection provisions of the *Convention* would require the extension of copyright protection to the non-literal elements of computer programs. This result leaves user interfaces outside the ambit of the *Convention*. It therefore seems that the implementation of a *sui generis* scheme for the protection for computer user interfaces would not be a violation of any of Canada's international copyright obligations.

## V. Conclusion

Currently there is a great deal of uncertainty in the computer industry with respect to the scope of copyright protection. There is also a great deal of concern that the scope of protection that is eventually adopted will provide overly broad protection, resulting

[247] In the original version of the *Berne Convention*, Article 2 read as follows:

> The expression "literary and artistic works" shall include any production in the literary, scientific or artistic domain, whatever may be the mode or form of its reproduction, such as books, pamphlets, and other writings, dramatic or dramatico-musical works, choreographic works and entertainments in dumb show, the acting form of which is fixed in writing or otherwise; musical compositions with or without words; works of drawing, painting, architecture, sculpture, engraving and lithography; illustrations, geographical charts; plans, sketches, and plastic works relative to geography, topography, architecture or science.

Under this phrasing, the definition of "literary and artistic works" could have been read *ejusdem generis* to the effect that fixation was required. The listed works are either tangible and obviously fixed (for example, books and pamphlets) or, when more ephemeral, required to have fixation before protection is extended (for example, "entertainments in dumb show, the acting form of which is *fixed* in writing or otherwise"). The phrasing of Article 2 in the Rome revision of the *Convention*, to which Canada subscribes, is, however, not susceptible to this interpretation, since it extends protection to "lectures, addresses, sermons and other works of the same nature." There is no mention of a fixation requirement for these works. In the Stockholm version of the *Convention*, Article 2(2) provides that:

> It shall, however, be a matter for legislation in the countries of the Union to prescribe that works in general or any specified categories of works shall not be protected unless they have been fixed in some material form.

It is submitted that the apparent lack of a fixation requirement in the Rome version was inadvertent. The fact that the requirement was initially present and then was made explicit in a subsequent revision supports this conclusion. The need for this reasoning may now be moot, in light of the fact that Canada has recently agreed under *NAFTA* to give effect to the substantive provisions of the Paris revision of the *Berne Convention* (which gives national flexibility with respect to fixation).

[248] Computer programs would certainly fall within the ambit of Article 2, as they are a production in the "literary, scientific or artistic domain" which is sufficiently fixed.

in limited access to key technologies. Unfortunately, these difficulties are likely to continue for some time. Once again the industry is awaiting the results, this time on appeal, of the *Apple* v. *Microsoft* litigation. Ultimately, the American position may not be conclusively established until the Supreme Court speaks on these issues.

From the Canadian perspective it is important not to become overly dependent on the U.S. law. This paper has illustrated that a consideration of existing Canadian copyright jurisprudence can lead to results which differ from those under U.S. law. While many of the issues discussed in this paper have not come before Canadian courts, there is evidence to suggest that American copyright law is being adopted without consideration of the Canadian case law and the Canadian *Copyright Act*. It is important that Canadian courts have proper regard for the Canadian law. Furthermore, if Canadian courts choose to adopt the American position, they should do so only after careful consideration of the merits of that position, and not simply because many of these issues have already been addressed in the United States.

From an economic perspective, the policy choices made by American legislatures and courts may not be appropriate for Canada. As a net importer of copyrighted works, Canada must be careful to choose laws and policies that suit its needs. These choices may well involve a less protectionist copyright scheme than that adopted by the United States. This paper has examined some ways of implementing a copyright regime which protects, but does not over-protect, computer software. The key to the approach suggested in this paper is the acceptance of reverse engineering as a legitimate activity. Without reverse engineering the industry will lose the principal means by which new ideas and techniques are disseminated. The result can only be a decrease in innovation, and consequently a reduction in the number of new works made available to the public. Once the legality of reverse engineering is accepted, the limitation of copyright protection of computer software to the prohibition of anti-competitive behaviour flows naturally. The focus of such a scheme would be to ensure that any copying of works is only done through legitimate reverse engineering. The result of such an approach would be to grant software technology a level of protection equivalent to that given to other technological works.

The United States is currently the dominant player in the computer software industry; because of its position, there is a danger that the American protectionist approach will be adopted by default. Parliament must ensure that this does not happen. If care is not taken now, the result may well be a Canadian copyright law that does not serve Canadian needs and interests.