Reverse Engineering Computer Programs Under Canadian Copyright Law

Sunuy Hauda*

This article addresses the applicability of copyright law to the reverse engineering of computer programs. Reverse engineering is a method by which programmers may uncover the ideas and processes used within an existing computer program, thereby allowing the construction of compatible computer programs. Recently, both the European Community and the United States have accepted reverse engineering as an exception to copyright infringement. The author argues that Canada should follow suit. It is claimed that such an exception is justified insofar as it is consistent with the underlying policy goals of copyright law within an economic/utilitarian framework. The author posits that reverse engineering fosters the creation of standards which increase societal wealth. The existence of a reverse engineering exception is consistent with the goal of balancing the economic rights of individual authors with societal technological progress, which copyright in general seeks to encourage.

According to the author, an effective exception should be statutorily based. The existing fair dealing exception in the Canadian *Copyright Act* is juridically underdeveloped and too uncertain to provide an effective solution to the reverse engineering problem. A legislative solution would send a clear message to the software industry as well as to the courts, and could prohibit contracting out of the *Copyright Act*, a potential loophole, should a judicial solution be sought. The statutory exception should broadly allow the process of reverse engineering, rather than limiting it to cases where compatibility is sought.

Ce mémoire vise à analyser l'application des droits d'auteurs au processus d'analyse et de recomposition des logiciels, aussi appelé décompilation. Ce terme désigne une méthode utilisée par les programmeurs pour extraire les idées et le processus utilisés dans un logiciel existant afin de produire de nouveaux programmes compatibles avec celui-ci. Récemment, tant les pays de l'Union européenne que les États-Unis ont exclu la décompilation des infractions à la *Loi sur les droits d'auteurs*. Dans ce mémoire, l'auteur soutient que le Canada doit lui aussi adopter une solution qui exclut le processus de décompilation des droits d'auteurs. Il est argué que l'implantation d'une telle exception est justifiée par l'analyse des principaux objectifs visés par la loi concernant les droits d'auteurs dans le contexte d'un encadrement économique/utilitaire. Le processus de décompilation aide à créer des normes qui, selon l'argumentation soumise, contribuent à l'enrichissement de la société. L'exemption de ce processus constitue une démarche logique dans le cadre d'un équilibre entre les droits économiques des auteurs individuels et le progrès technologique de la société que la *Loi sur les droits d'auteurs* cherche à soutenir.

Par ailleurs, l'auteur suggère qu'une exception efficace devrait être confirmée par une loi. Il est soutenu que l'exemption pour l'utilisation équitable contenue dans la *Loi sur les droits d'auteurs* canadienne est juridiquement insuffisante et trop vague pour offrir une solution efficace au problème du processus de décompilation. Une solution législative transmettrait un message clair à l'industrie des logiciels ainsi qu'aux tribunaux et prohiberait l'option d'exclusion volontaire («contracting out») de la *Loi sur les droits d'auteurs*. Cette exclusion serait possiblement accordée si une solution judiciaire était privilégiée. En plus, il est suggéré que l'exemption statutaire devrait englober tout le processus de décompilation plutôt que de le limiter aux cas où une compatibilité est recherchée.

* B. Comm. (McGill), LL.B. (University of Toronto), LL.M., Doctoral Candidate (McGill); Barrister and Solicitor (of the Ontario Bar). Sessional Lecturer, Faculty of Law, McGill University. The author wishes to express his gratitude to Rosemary Shuttleworth, without whose ongoing support this article would have never been written.

Revue de droit de McGill
To be cited as: (1995) 40 McGill L.J. 621
Mode de référence: (1995) 40 R.D. McGill 621

Synopsis

Introduction

> *Originality is nothing but judicious imitation. The most original writers borrowed one from another. The instruction we find in books is like fire. We fetch it from our neighbors, kindle it at home, communicate it to others, and it becomes the property of all.*
>
> Voltaire

> *Nothing can with greater propriety be called a man's property than the fruit of his brains. The property in any article or substance accruing to him by reason of his own mechanical labour is never denied him: the labour of his mind is no less arduous and consequently no less worthy of the protection of the law.*
>
> Copinger and Skone James on Copyright

> *Laws that do not embody public opinion can never be enforced.*
>
> Elbert Hubbard

> *If we desire respect for the law, we must first make the law respectable.*
>
> Louis D. Brandeis

The field of computer law has grown in leaps and bounds in recent years. In fact, the very use of the title "computer law" to denote a separate body of law continues to be regarded with some skepticism by traditionalists. A number of law firms, however, have developed departments or groups that specialize in this fast-growing field, and some practitioners hold themselves out as computer lawyers. What distinguishes computer law from accepted traditional legal categories such as copyright or, even more broadly, intellectual property law, is its interdisciplinary nature. It incorporates many established areas of law including, but not limited to, copyright, patents, trade secrets, semiconductor chip, contract, criminal, and tort law.

In terms of civil and criminal protection against illicit copying of computer programs, copyright law has clearly evolved as the standard form of protection throughout the world. It is generally accepted that international copyright instruments such as the *Berne Convention*[1] and the *Universal Copyright Convention*,[2] while not explicitly referring to computer programs as protectable works, do protect computer programs as literary works. The field of copyright law, not often considered a particularly dynamic area of law, has in the past decade been the subject of increased activity as a result of its application to computer programs. While computer programs generally fit into the underlying framework of copyright law, they do not, by their very nature, lend themselves perfectly to many of the idiosyncratic jurisprudential concepts of copyright law as traditionally understood. Accordingly, copyright law as it applies to computers is rapidly developing its own wealth of jurisprudence.

One of the most topical and difficult decisions facing legislators and jurists in this area is the permissibility of reverse engineering of computer programs under copyright law. Reverse engineering, or decompilation as it is sometimes called, involves working backwards from a finished product in order to gain a better understanding of how the product was produced. The issue of whether software products protected by copyright may be legally reverse engineered without the copyright holder's consent remains unchallenged under Canadian law. Indeed, only a few jurisdictions have yet to deal with the issue. Most notably, the European Community, in its Software *Directive* of May 14, 1991,[3] took a bold step forward in declaring that computer programs may be decompiled for the purposes of achieving interoperability (compatibility) with other computer programs. More recently, several American courts have had to deal with whether, and under what circumstances, reverse engineering of computer programs would be permitted under American copyright laws.[4] To date, the issue of reverse engineering of copyrighted works has not been dealt with in Canada, either legislatively or judicially. A broad reading of the Canadian *Copyright Act* suggests that *prima facie* the reverse engineering of a copyrighted work without the copyright holder's consent is prohibited.

The American copyright legislation, however, while appearing very similar to

---

[1] *Revised Berne Convention*, being Schedule II of the *Copyright Act*, R.S.C. 1985, c. C-42 [hereinafter *Berne Convention*]. The *North American Free Trade Agreement* [hereinafter *NAFTA*], to which Canada is a party and which came into force on January 1, 1994, requires that the contracting member states accede to the (latest) 1971 Paris revision to the *Berne Convention* (*An Act to Implement the North American Free Trade Agreement*, S.C. 1993, c. C-44, s. 5).

[2] 6 September 1952, 216 U.N.T.S. 132.

[3] EC, *Council Directive (EEC) No 91/250 of 14 May 1991 on the legal protection of computer programs*, O.J. Legislation (1991) No L122 at 42 [hereinafter *Directive*].

[4] *E.g. Sega Enterprises Ltd.* v. *Accolade Inc.*, 977 F.2d 1510 (9th Cir. 1992) [hereinafter *Sega*]; *Atari Games Corp.* v. *Nintendo of America Inc.*, 975 F.2d 832 (Fed. Cir. 1992) [hereinafter *Atari*]. For cases involving the reverse engineering of data tables, see *e.g. E.F. Johnson Co.* v. *Uniden Corp. of America*, 623 F.Supp. 1485 (D.C. Minn. 1985) [hereinafter *Uniden*]; *Autodesk Inc.* v. *Dyason* (1992), 22 I.P.R. 162 (Aust. H.C.), rev'ing (1990), 18 I.P.R. 399 (Aust. Fed. Ct.) [hereinafter *Autodesk*].

the Canadian legislation, has been interpreted by American courts as allowing reverse engineering in certain circumstances. Although Canadian courts are not bound by decisions of their American counterparts, much is borrowed from American law in the computer law field.[5] There are also strong public policy arguments in favour of permitting the reverse engineering of computer programs. Reverse engineering allows for the creation of interoperable or compatible computer programs in cases where the copyright holder will not release the program's technical specifications. The balance between public benefit and the protection of an individual's right to claim rewards associated with a work's distribution and use is at the core of intellectual property rights protection. Accordingly, in considering whether to allow reverse engineering, the societal benefits to be gained must be weighed against the potential risks to the copyright holder.

This article will suggest that a broad right of reverse engineering with respect to computer programs should be permitted under Canadian law. This claim will be based on a cost/benefit or economic analysis of the law and of the outcomes associated with various proposed reverse engineering scenarios. Justifications for the use of law and economics as a theoretical basis for allowing reverse engineering, and as an explanation of intellectual property protections more generally, will also be discussed.

I. Concepts

Before proceeding any further, it is necessary to define concepts such as reverse engineering, computer programs and other related terminology for purposes of both clarity and consistency. This is not meant to be a comprehensive review of computer technology, but rather a cursory overview of the technological concepts involved in the reverse engineering debate.

---

[5] In *Compo Co.* v. *Blue Crest Music Inc.* (1979), [1980] 1 S.C.R. 357, 105 D.L.R. (3d) 249 [hereinafter *Compo* cited to S.C.R.], Estey J. stated:

> The United States *Copyright Act*, both in its present and earlier forms, has, of course, many similarities to the Canadian Act, as well as to the pre-existing Imperial *Copyright Act*. However, United States court decisions, even where the factual situations are similar, must be scrutinized very carefully because of fundamental differences in copyright concepts which have been adopted in the legislation of that country. ... That is not to say that we may not find some assistance in examining the experience in the United States (*Compo, ibid.* at 366-67).

O'Leary J. further recognized the value of American jurisprudence in *Delrina Corp.* v. *Triolet Systems Inc.* (1993), 47 C.P.R. (3d) 1 at 28, 9 B.L.R. (2d) 140 (Ont. Ct. (Gen. Div.)) [hereinafter *Delrina* cited to C.P.R.], stating: "The U.S. *Copyright Act* differs somewhat from the Canadian *Copyright Act*, but nevertheless American copyright decisions were heavily relied on by both the plaintiff and defendants in this case and are of great assistance on the issues before me."

A. *Computer Programs*

Section 2 of the Canadian *Copyright Act* defines a "computer program" as "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." This definition is virtually identical to that set out in section 101 of the American *Copyright Act*, which defines "computer program" as "a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result."[6] Subsection 342.1(2) of the Canadian *Criminal Code*[7] defines a computer program as "data representing instructions of statements that, when executed in a computer system causes the computer system to perform a function."

Together, these definitions identify a computer program as an arrangement of instructions that is used in a computer to solve a particular problem. Simple data that does not in itself instruct a computer to perform calculations towards a given end does not qualify as a computer program. Other similar definitions can be found scattered throughout the jurisprudence dealing with the copyrightability of computer programs.

Computer programs can be broadly categorized into two types: operating system programs and application programs. Reed J. articulated a particularly clear definition of each in *Apple Computer Inc.* v. *Mackintosh Computers Ltd.*:

> Application programs are designed for a specific task, such as the playing of a video game, preparation of a tax return, or the writing of a text. Operating system programs are designed primarily to facilitate the operation of application programs and perform tasks common to any application program. Without them each application program would need to duplicate their functions.[8]

B. *Computer Languages*

Also essential to understanding the issues surrounding reverse engineering is a minimal knowledge of how a computer program is built, compiled and executed, as well as an understanding of the related jargon. Computer programs are written in computer languages which vary in their degree of resemblance to "ordinary mathematics and English (or other common languages)."[9] A higher level computer language is said to be closer to "common languages" in its vocabulary than is a lower level language. The level, also referred to as the generation, of the computer

---

[6] 17 U.S.C. § 101 (1994) [hereinafter *Copyright Act (U.S.)*].

[7] R.S.C. 1985, c. C-46.

[8] (1986), [1987] 1 F.C. 173, 10 C.P.R. (3d) 1 at 11 (T.D.) [hereinafter *Apple* cited to C.P.R.], var'd (1987), [1988] 1 F.C. 673, 44 D.L.R. (4th) 74 (C.A.), aff'd [1990] 2 S.C.R. 209, 30 C.P.R. (3d) 257 [hereinafter *Apple* (S.C.C.) cited to C.P.R.].

[9] *Ibid.* at 7.

language depends "upon the ease with which it can be read" by human beings.[10] In order for a computer to process the instructions of any given language, the instructions must first be compiled, or translated, into a language or notation that the computer's processor can understand. This latter notation is known as the lowest, or first level language.

The term "source code" refers to the written form of program that the user physically produces in a given computer language. Programmers today have a wealth of higher languages from which to choose, including PASCAL, COBOL, FORTRAN, BASIC, and ASSEMBLER.[11] Once the program's source code has been written, the programmer will "compile" the source code into machine-readable object code using another computer program, known as a compiler, to perform the conversion. Object code is generally in binary form, a language made up exclusively of "1"s and "0"s, and is used directly by the computer.

C. Computer Memory

Another concept central to the use of computers is that of memory. The Canadian *Copyright Act*'s definition of "computer program" requires that a set of instructions be "expressed, fixed, embodied or stored" in order to qualify as a computer program. In the United States, legislation defines this fixation as existing where a tangible mode of expression is embodied in a form which is sufficiently permanent and stable to be "perceived, reproduced, or otherwise communicated, either directly or with the aid of a machine or device."[12] Traditionally, the fixation of copyrighted literary works was done on paper.[13] This type of storage, while also valid for computer programs, is not the only possible type of fixation. Computer programs stored in a computer's memory also qualify as being stored for the purposes of the *Copyright Act*.[14]

For the purposes of convenience, computer memory devices can be separated into three categories: internal, external and archival.[15] Internal memory consists of

---

[10] *Ibid.*

[11] ASSEMBLER is a second generation, or intermediate, language whereas PASCAL, COBOL, BASIC and FORTRAN are considered to be third generation languages.

[12] *Copyright Act (U.S.), supra* note 6, § 102.

[13] See Part **II.A.1.***d*, below, for a discussion of fixation.

[14] In *Apple*, the Court held that object code stored on a silicon microchip was a reproduction in a material form of copyrightable source code and was therefore protectable as a "computer program" under the *Copyright Act*. According to the Court, it was irrelevant that the object code was not necessarily in human-readable form.

[15] R.R. Panko, *End User Computing* (New York: John Wiley & Sons, 1988) at 315. A simple, but useful, definition of "memory" can be found in *Delrina*:

> [Memory is defined as a]n area of the computer's circuitry that holds applications and any data generated with those applications. Information held in Random Access Memory (RAM) is erased whenever the computer is turned off. Information held in Read Only Memory (ROM) is retained even when the computer is off.

those memory devices, built into the structure of the computer, such as ROM-chips (read-only memory microchips) and RAM-chips (random access memory microchips), that are necessary for the computer to operate at its most basic level. External memory includes devices, such as disks and CD-Roms, that provide additional, non-essential, but more permanent storage at a relatively cheap cost. Archival memory, consisting of devices such as tape-backup machines, does not impact on the reverse engineering debate in any meaningful way and will thus not be discussed in this article.

D. Operating Systems

In order for a computer to execute its programs and interact with the user, an operating system is required. The purpose of the operating system is to set up and manage the computer system's environment and resources. These systems are computer programs the specifications of which are essential to computer application programmers who are writing programs designed for certain computer systems.

E. Reverse Engineering

Reverse engineering, as the name suggests, is the opposite to the above described process of constructing a computer program. Reverse engineering "involves going backwards from a finished product and determining how the product works."[16] Another definition holds that reverse engineering occurs where "one inspects or takes apart a new product ... by translating the unreadable object code of a program into source code that may be studied."[17] The terms "disassembly" and "decompilation", often used as synonyms of "reverse engineering", are actually subsets thereof. Decompilation of a computer program occurs where one "convert[s] the machine code version [of the program] into a high level language,"[18] whereas "[d]isassembly of a computer program is done by translating the machine or object code into humanly-readable assembly language."[19] The only difference between decompilation and disassembly is the product obtained at the end of the process. In the former case it involves converting the machine code into a high level language, whereas in the case of disassembly the final product is in ASSEMBLER, an intermediate level language.

Memory usually refers to the high speed semiconductor storage within a computer that is used to temporarily store data while it is being processed or examined. The term "memory" is also generically extended to refer to data that is stored externally on disks and tapes (*Delrina*, *supra* note 5 at 51-52).

[16] *Sega Enterprises Ltd.* v. *Accolade Inc.*, 785 F. Supp. 1392 at 1394 (N.D. Cal. 1992).

[17] G.R. Ignatin, "Let the Hackers Hack: Allowing the Reverse Engineering of Copyrighted Computer Programs to Achieve Compatibility" (1992) 140 U. Penn. L. Rev. 1999 at 2010.

[18] D.I. Bainbridge, "Computer Programs and Copyright: More Exceptions to Infringement" (1993) 56 Modern L. Rev. 591 at 593.

[19] *Uniden*, *supra* note 4 at 1490.

Practically speaking, most reverse engincering is of the disassembly variety, as the computer programs that are used in performing disassembly are easier to create and more flexible than are decompilers. Furthermore, software engineers and computer programmers involved in reverse engineering are generally comfortable in an ASSEMBLER language environment and do not need to visualize the program in a higher level language.

1. Intermediate Copying

The copying of computer programs as it relates to the reverse engineering process can occur in several ways. The first instance of copying that results from reverse engineering occurs during the deconstruction process. Whether the reverse engineering is conducted through a manual inspection of the program code which is reassembled on paper, or through the more common method of disassembly, the process invariably results in copying.[20] This copying does not result in a completely verbatim copy of the original code, but rather entails translating the original program code several times, each time moving closer to an ASSEMBLER, or higher level, language translation of the original object code.[21] Whether intermediate copies produced during the disassembly process violate copyright rules is not altogether clear.[22]

2. Reverse Engineering and Piracy

Piracy, the most widely publicized form of illicit copying, concerns the direct reproduction of a computer program, usually by a user, without the author's consent. Because computer programs are stored digitally (as "1"s and "O"s), flawless reproductions can be made at little cost to the copier.[23] As a result, computer programs, and more recently digitally-stored audio recordings, have increased intellec-

---

[20] Copying, for the purposes of copyright, is subject to the copies being "fixed" in some form. See Part **II.A.1.*d***, below, for a discussion of fixation.

[21] A disassembler makes several "passes" over the original code, gradually building towards a final translation in ASSEMBLER language.

[22] See text accompanying note 76, below, for a discussion of intermediate copying that remains stored only in RAM. Where the intermediate copies are stored in a more permanent manner, it is more likely that they will violate copyright laws (see Part **II.*B***, below). See also paragraph 3(1)(a) of the *Copyright Act*, which prohibits unauthorized translations of protected works; see also *infra* note 201 and accompanying text for the decision of the United States Court of Appeals (Ninth Circuit) with respect to intermediate copying.

[23] In its final report to Congress concerning the copyrightability of computer programs, the United States National Commission on New Technological Uses of Copyrighted Works (C.O.N.T.U.) stated:

> The cost of developing computer programs is far greater than the cost of their duplication. Consequently, computer programs ... are likely to be disseminated only if ... the creator may spread its costs over multiple copies of the work with some form of protection against unauthorized duplication of the work (National Commission on New Technological Uses of Copyrighted Works, *Final Report* (Washington: Library of Congress, 1979) at 11).

tual property protection needs as the economic incentives to purchase the original product have diminished.[24] Typically, with this form of direct copying, a user will make an unaltered copy of an original computer program[25] or of an existing copy thereof, and will use the copy instead of purchasing the original computer program.[26] Generally, these copies can be easily produced using simple operating system commands, although, in an effort to deter this practice, some computer program manufacturers have attempted to use copy protection schemes.

Copy protection schemes vary in their functioning and, because of their very purpose, cannot be standardized. These schemes, however, can currently be divided into three basic types:[27] (1) the program is stored in such a way that copying programs cannot copy all the necessary parts; (2) the program prompts the user for a code or other piece of information that can only be found in the original packaging; and (3) the program comes with a hardware device that attaches to the computer and sends the program signals or information which it seeks prior to functioning. The first type of protection is often defeated by copying programs developed by third parties, which can copy the required parts. With respect to the latter two schemes, program modifications, known as "cracks", designed to defeat the protection will invariably appear soon after, or sometimes even before, a program's public release.[28]

Program cracking for the purpose of defeating a copy protection scheme often involves some disassembly of the protected computer program. Cracking programs, or enhanced disassemblers with specialized features to assist a cracker, are also generally available as either shareware or freeware.[29] Recently, the reverse engi-

---

[24] W.M. Landes & R.A. Posner, "An Economic Analysis of Copyright Law" (1989) 18 J. Legal Stud. 325 at 327.

[25] In this context, a copy is produced in the absence of authority of the copyright holder, while an original is a copy produced with permission.

[26] "Persons who have not paid for a software copy cannot be excluded from using a program, and use of a program copy by one person does not necessarily diminish the supply of copies available for use by others" (D.A. Rice, "Public Goods, Private Contract and Public Policy: Federal Pre-emption of Software License Prohibitions Against Reverse Engineering (1987) 53 U. Pitt. L. Rev. 543 at 545).

[27] A comprehensive review of more eclectic forms of copy protection is beyond the scope of this article.

[28] A "crack" is often distributed in either printed form (as a set of instructions on how to modify the program to defeat the protection) or as a "patch". A patch is a small computer program, or portion of computer code, that applies itself to the protected program and replaces the required code with instructions that defeat the protection. An example of a simple crack is a set of instructions that tells the program to skip over the code that executes the protection checks. Cracks are commonly distributed on various computer bulletin boards and are easily available on the Internet.

[29] Shareware refers to computer programs which may be used for a trial period without infringing copyright, after which a licence fee is payable to the copyright holder for continued use. Shareware programs are also freely distributable in their unaltered state to other users (hence the "share" in shareware) who may try them out for the trial period without payment. Freeware refers to computer programs where the copyright holder waives his or her rights to any economic return for its use. Waiving of economic returns does not mean a waiver of moral rights which would allow users to modify and to alter the original work. Any such waiver is independent of the free or shareware desig-

neering debate was brought in front of the American courts with respect to video game cartridges which contained computer programs protected with a program check (akin to protection type (3) set out above).[30]

3. Using the Results Obtained Through Reverse Engineering

Another form of illicit copying occurs where the copier, usually a programmer, alters or uses parts of the original program in his or her own work. The amount of modification involved varies greatly in these situations, and may or may not be substantial enough to violate copyright.[31] Copying a work in this manner may not require actual direct copying of the original program code (known as "literal copying").[32] "Non-literal" elements of a computer program are "those aspects that are not reduced to a written code,"[33] and include "components such as general flow eharts as well as the more speeific organization of inter-modular relationships, parameter lists, and macros."[34] Screen displays also fall within the definition of non-literal elements.[35] Copying these non-literal elements may also infringe copyright, and may be accomplished without disseeting the program as previously discussed. For example, copying the layout of a screen may simply involve visual examination of the original program and subsequent replication using entirely new programming. Similarly, copying the order of the keystrokes used in the operation of a computer program, known as "command sequences", may not involve actual literal copying of the original code. The issue of non-literal copying of computer programs has been highly topical in recent years, and is far from being resolved.[36]

Generally, reverse engineering will entail a deconstruction of the original program's literal code to uncover its underlying ideas, rather than non-literal copying. Whether the reverse engineering will be sufficient to uncover these ideas, or whether it will simply stop at the point of uncovering expression will depend on each individual case. This exercise is then followed by an attempt to construct a different program using the results of the reverse engineering. The degree to which the reprogramming will involve copying of the original program code will also vary greatly. This genre of copying may include some forms of non-literal copying, such

---

nation.

[30] *Sega, supra* note 4; *Atari, supra* note 4.

[31] Copyright only protects expressions and not their underlying ideas. See Part II.A.1, below, for a discussion of copyright principles.

[32] Rice, *supra* note 26 at 567.

[33] *Computer Associates International Inc.* v. *Altai Inc.*, 23 U.S.P.Q. (2d) 1241 at 1244 (2d Cir. 1992) [hereinafter *Altai*].

[34] *Ibid.* at 1249.

[35] Screen displays are protected as parts of a computer program "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" (*Delrina, supra* note 5 at 32).

[36] See *Delrina, ibid.*; *Systèmes informatisés Solartronix* v. *Cégep de Jonquière* (1988), 22 C.I.P.R. 101 (Que. Sup. Ct.); *Lotus Development Corporation* v. *Paperback Software International*, 740 F. Supp. 37 (D. Mass. 1990) [hereinafter *Paperback Software*]; *Altai, supra* note 33; *John Richardson Computers Ltd.* v. *Flanders*, [1992] F.S.R. 391 (Ch.) [hereinafter *Richardson*].

as reproducing the layout of the program's subroutines (structure), as well as literal copying, such as copying parts of the original program's code either directly or through a translation into another language.

The results obtained by reverse engineering can be used for a number of purposes including: the programming of cracks to defeat a program's copy protection; the creation of a similar program or of a program that uses the same routines as the original program in order to save time and expense; academic study of the program's underlying ideas and the techniques used in their expression; or the creation of compatible programs. While the first two applications do not usually garner much support as they involve an element of thievery, the last two goals are more readily perceived as morally acceptable justifications of reverse engineering compatible with the underlying spirit of copyright law.[37]

4. Using Reverse Engineering in the Creation of Compatible Programs

Although no statutory definition of "interoperable" or "compatible" program currently exists, the concepts are synonymous and simple to define. Compatibility is a measure of the degree to which one program will function in conjunction with another. In order to create a compatible program, "a programmer must have a complete specification of the other program's 'interface' — a precise description of how the program receives, stores and/or outputs information."[38] Traditionally, compatible programs were written by the same company because their programmers had access to the necessary specifications.

A more modern approach to the design of compatible programs is to create more robust operating systems which can handle a greater number of functions, and to provide application programmers with consistent specifications for all types of data structures (objects) controlled by each operating system. Programmers writing computer programs for use with such operating systems are usually given access to the operating system's specifications by the operating system designers at minimal cost. Objects created by computer programs that follow the standardized specification are then useable in other programs that also follow the specification. The programs are therefore made compatible without the programmers ever having seen or used each other's computer programs because the operating system acts as the standardizing link. Examples of such object-oriented operating systems are Microsoft's Windows™ and Next's NextStep™ operating systems.[39] The move towards

---

[37] Ignatin, *supra* note 17 at 2022. Allowing reverse engineering for the creation of interoperable programs is recognized in Article 6 of the European Community's Software *Directive*, *supra* note 3 at 45, as the sole justification for reverse engineering.

[38] Ignatin, *ibid.* at 2023.

[39] Under the popular Microsoft Windows™ operating system, a system of standardized objects, object linking and embedding (O.L.E.), is used. O.L.E. allows users to share information created in other applications with the application they are using. For example, users of Microsoft's Word™ may either link or embed objects created with Microsoft's Excel™ spreadsheet program inside their

creating standards through operating systems is a sound one; however, the creation of new types of objects not contemplated by the operating system designers may arise. In such cases, the standards are once again protected as the property of the application's designers.[40]

5. Conclusion

As the law currently stands, the newly constructed program or parts thereof, will sometimes be considered a copy and thereby infringe the original program's copyright, whereas in other cases, the new program will not involve a sufficient degree of copying to amount to a copyright violation. This determination is made irrespective of whether the new program is compatible or whether it is constructed to compete with the original as a similar product.[41] A more immediate question, however, is whether the actual reverse engineering of the computer program is an infringement in itself. Both of these issues will be addressed in the following discussion.

II. Intellectual Property Protections for Computer Programs

Computer programs are protected from illicit copying under a number of legal regimes. Copyright, patent, trade secret, and semiconductor chip laws all provide intrinsic measures of protection against the unauthorized copying of a computer program. The term "intrinsic" is used to distinguish the protections granted by these regimes from those contractual provisions fashioned by private parties. The protections provided by each of the regimes listed above are enforceable at law notwith-

---

Word™ document. A large part of the linkage and embedding of these objects is a function of the Windows™ operating system, and not of any specific design created by either application's programmers.

[40] There is also the question whether operating system designers, often members of the same company that designs various applications for use with their operating systems, will release all of the specifications required to make the most effective use of the operating system environment. Clearly, there is motive to withhold some of the technical information so as to provide one's own company with a competitive advantage in the application program market. In 1993, the United States Department of Justice stepped up a Federal Trade Commission anti-trust investigation of Mierosoft Corporation. This investigation was recently coneluded (1995) in favour of Microsoft. Among the charges investigated were claims by competitors in the application program market "that Mierosoft unfairly uses secret features known as 'undoeumented calls' and its advance knowledge of changes to MS-DOS and the related Windows software to place its competitors at a disadvantage" (I.K. Gotts, "Regulators Focusing on Antitrust Issues" *The National Law Journal* (24 January 1994) S12.

[41] *Prima facie* copyright law is currently oblivious to any sueh distinction. A determination of copying under the *Copyright Act* concerns whether the copied code is "substantial" in quality and not in quantity. However, if a newly constructed program is directly competing with an original work from which information was reverse engineered, it is more likely that the parts used will be considered substantial. This occurs because the quality of the parts used in a competing program may appear of greater import than if they had been used in a compatible program which, to a court, seems different in both appearance and structure (*SAS Institute, Inc.* v. *S & H Computer Systems, Inc.*, 605 F. Supp. 816 (N.D. Tenn. 1985) [hereinafter *SAS*]).

standing the absence of specific contractual agreements between parties.

A. *The Law of Copyright*[42]

1. The *Copyright Act*

Following the House of Lords' decision in *Donaldson* v. *Becket*,[43] which effectively abolished the common law of copyright, the British Parliament passed a series of *Copyright Acts*.[44] This body of legislation was eventually the source of the first Canadian *Copyright Act*[45] which came into force on January 1, 1924.[46] The Canadian *Copyright Act* has continued in the tradition of its Imperial forebearers and explicitly states that no copyright or similar right shall exist in Canada other than those recognized under the *Copyright Act*.[47]

Today's copyright laws are no longer limited to the protection of published and unpublished manuscripts. Copyright currently protects dramatic, musical and artistic works, as well as a broad category of literary works including tables, compilations and translations. Section 2 of the *Copyright Act* classifies computer programs as literary works for the purposes of copyright protection. This protection permits the copyright holder to make copies of the work, while prohibiting others from doing so.[48] Copyright holders may, however, freely license or assign their economic rights.[49]

---

[42] Copyright law, also referred to as Anglo-American copyright law or the common law of copyright, is to be differentiated from *droit d'auteur* or continental regimes, which protect similar works but are primarily used in civilian jurisdictions. Copyright, as discussed in this paper, refers to those regimes with common law origins.

[43] (1774), 4 Burr. 2408, 2 Bro. Parl. Cas. 129.

[44] *Copyright Act, 1814* (U.K.), 54 Geo. 3, c. 156; *Copyright Act, 1842* (U.K.), 5 & 6 Vict., c. 45; *Copyright Act, 1911* (U.K.), 1 & 2 Geo. 5, c. 46.

[45] S.C. 1921, c. 24.

[46] The 1927 Canadian *Copyright Act* abrogated all previous *Copyright Acts* enacted by the Imperial Parliament.

[47] *Copyright Act*, *supra* note 1, s. 63. Prior to passage of the 1921 *Act*, a common law of copyright had existed in Canada. This right was replaced by a statutory right under section 42 of the 1921 *Act*. In *Compo*, Estey J. stated:

> [C]opyright is neither tort law nor property law in classification, but is statutory law. It neither cuts across existing rights in property of conduct nor falls between rights and obligations heretofore existing in the common law. Copyright legislation simply creates rights and obligations upon their terms and in the circumstances set out in the statute (*Compo*, *supra* note 5 at 372-73).

[48] See B.B. Sookman, *Computer Law: Acquiring and Protecting Information Technology* (Toronto: Carswell, 1989) at 3-1; H.G. Fox, *The Canadian Law of Copyright and Industrial Designs*, 2d ed. (Toronto: Carswell, 1967) at 1-2.

[49] Copyrighted works consist of two components: economic rights and moral rights. The former refers to the right of the copyright holder to reap economic benefits for authorizing use of the work, whereas the latter refers to the author's (as opposed to the copyright holder's) right to the integrity of the work as well as the right to be associated with the work "in certain circumstances" (*Copyright Act*, *supra* note 1, s. 14.1(1)). "Moral rights may not be assigned but may be waived in whole or in part"

a. The Idea/Expression Dichotomy

Copyright protects the expression of ideas, but does not extend to the ideas themselves.[50] This separation is referred to as the idea/expression dichotomy. Finding the line that separates idea from expression is not an easy task, and the difficulty is even more pronounced when dealing with computer programs which are by their very nature utilitarian works and hence intertwined with the ideas they seek to express.[51] The difficulty inherent in creating a test to distill expression from idea lies in defining the scope of expression. By adopting an overly liberal view of expression, one risks granting monopoly protection to the first authors of certain types of programs:

> [Such a view] would thereby inhibit other creators from developing improved products. [Conversely, d]rawing the line too conservatively would allow programmers' efforts to be copied easily, thus discouraging the creation of all but modest incremental advances.[52]

In order to devise a sound test, American jurisprudence suggests that

> [t]he court must be faithful to the statutory language and mindful of both the ultimate goal of copyright law — the advancement of public welfare — and Congress' chosen method of achieving this goal — private reward to the individual author.[53]

The consequences of an imperfect test are illustrated by the decision in *Whelan Associates, Inc.* v. *Jaslow Dental Laboratory, Inc.*[54] and its short-lived, but highly

---

(*Copyright Act*, *ibid.*, s. 14.1(2)).

[50] In *Moreau* v. *St. Vincent*, [1950] Ex. C.R. 198 [hereinafter *Moreau*], Thorson P. stated:

> [A]n elementary principle of copyright law [is] that an author has no copyright in ideas but only in his expression of them. The law of copyright does not give him any monopoly in the use of the ideas with which he deals or any property in them, even if they are original (*Moreau*, *ibid.* at 203).

[51] See *Delrina*, *supra* note 5 at 33. The doctrine of merger holds that if an expression is necessary to the function or efficiency of a given idea, it is considered necessarily incidental to the idea and is not protectable as an expression. Such a form of expression is said to be purely functional, resulting in the merger of idea and expression. For example, where a program requires the user to type the word "print" followed by the command "full page" in order to direct the output of a program to a printer, the command sequence will not be protected as it is necessarily incidental to the idea of printing.

[52] *Paperback Software*, *supra* note 36 at 53, quoting P.S. Menell, "An Analysis of the Scope of Copyright Protection for Application Programs" (1989) 41 Stanford L. Rev. 1045 at 1047-48.

[53] *Paperback Software*, *ibid.*

[54] 797 F.2d 1222 (3d Cir. 1986) [hereinafter *Whelan*]. A medical software developer, after designing a dental laboratory program for the plaintiff, started up her own company and developed a similar program. The Third Circuit Court of Appeals found that the defendant's program copied the structure, sequence and organization of the plaintiff's program, and that this was enough to constitute an infringement of the plaintiff's copyright. This decision was revolutionary in that it formally extended software copyright protection beyond the literal copying of source code to non-literal elements, in this case the structure, sequence and organization of the program. The Court held that since the plot of a

controversial legacy in the United States. In terms of the idea/expression dichotomy, the *Whelan* Court decided that "[t]he 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."[55] The case was subsequently criticized as having a "somewhat outdated appreciation of computer science" and for ignoring "practical considerations".[56] In *Altai*, decided after *Whelan*, the Second Circuit of the United States Court of Appeals devised a three-part, abstraction-filtration-comparison test that would better distinguish idea from expression in computer programs.[57] According to Walker J. of the *Altai* Court,

> [i]n ascertaining substantial similarity under this approach, a court would first break down the allegedly infringed program into its constituent structural parts [(abstraction)]. Then, by examining each of these parts for such things as incorporated ideas, expression that is necessarily incidental to those ideas, and elements that are taken from the public domain, a court would then be able to sift out all non-protectable material [(filtration)]. Left with a kernel, or possibly kernels, of creative expression after following this process of elimination, the court's last step would be to compare this material with the structure of an allegedly infringing program [(comparison)].[58]

The three-part abstraction-filtration-comparison test has gained widespread acceptance in the United States,[59] and has also been adopted in computer program copyright cases in Canada and the United Kingdom.[60]

b. Formalities, Term and Ownership

In order to avail itself of copyright protection, a work may be either published or unpublished.[61] Under Canadian copyright law there is no registration[62] or mark-

---

story or play is protected by copyright, so should be the sequence and organization of programs. The Court reasoned that there were many possible ways in which to organize the idea of a dental lab program, and therefore the particular manner chosen by the plaintiff was a copyrightable expression of that idea.

[55] *Ibid.* at 1236.

[56] *Altai, supra* note 33 at 1252.

[57] In *CMAX Cleveland, Inc.* v. *UCR, Inc.*, 4 CCH Computer Cases ¶ 46,752 at para. 94 (M.D. Ga. 1992) [hereinafter *CMAX*], Fitzpatrick D.J. stated that the decision in *Whelan* is "conceptually overbroad" and "descriptively inadequate", preferring instead to follow the three-part test outlined in *Altai*.

[58] *Altai, supra* note 33 at 1252-53.

[59] See *CMAX, supra* note 57; *Lotus Development Corporation* v. *Borland International, Inc.*, 788 F.Supp. 78 (D. Mass. 1992) [hereinafter *Borland*].

[60] See the decisions in *Delrina, supra* note 5 and in *Richardson, supra* note 36 with respect to Canada and the United Kingdom respectively.

[61] *Copyright Act, supra* note 1, s. 3.

[62] Although it is not mandatory that one register one's copyright in the work, it is considered prudent to register a work for several reasons. First, registration provides an evidentiary record of the work should a dispute as to its authorship ever arise, and second the copyright holder may have a broader range of remedies available should there be an infringement of the copyright (S. Handa & J. Buchan, "Copyright as It Applies to

ing requirement[63] for a work to be copyrightable; copyright is said to subsist upon the creation of the work.[64] The term for which copyright attaches is the life of the author, or in the case of joint authorship of a work[65] the life of the longest surviving author, plus fifty years.[66] The author of a work is presumed to be the first owner of the copyright therein, except where a work is created under a contract of service whereby the employer is presumed to be the first owner of the work.[67]

c. Originality

Merely expressing oneself in one of the protected forms enumerated above may not in itself be sufficient to obtain copyright protection. The *Copyright Act* requires that protectable works, at a minimum, demonstrate a modicum of originality.[68] Originality in a work refers to the degree of the author's creative or inventive thought, and is a comparatively less rigorous requirement in common law copyright jurisdictions as compared with continental *droit d'auteur* jurisdictions such as Germany or France.[69] In effect, under common law copyright systems, in order to demonstrate originality one need only show that the work originated from the author and was not a copy of an existing work.[70]

---

the Protection of Computer Programs in Canada" (1995) 26 I.I.C. 48 at 51-52).

If a computer program copyright is registered, there is no requirement that the copyright holder file the detailed source code specification with the copyright office. In fact the Canadian Copyright Office will not accept attachments when registering the copyright in a work.

[63] "Canada is a long-standing signatory to the Berne Convention on Copyright (Rome Revision, 1928), which prohibits any requirement that works be registered or that the "©" symbol be used in conjunction with expressions of the work in order for copyright protection to apply" (Handa & Buchan, *ibid.* at note 11, p. 51).

[64] *Copyright Act*, *supra* note 1, s. 5.

[65] Joint authorship exists where the "contribution of one author is not distinct from the contribution of the other author or authors" (*ibid.*, s. 2.9).

[66] *Ibid.*, ss. 6, 9(1).

[67] *Ibid.*, s. 13(3). A "contract of service", which denotes an employment relationship in the traditional sense, is to be differentiated from a "contract for services", which refers to an independent contractor who has arranged to produce a work under a specific contract as opposed to a general employment contract. In either case the presumption created is rebuttable at law (see *Orbitron Software Design Corp.* v. *M.I.C.R. Systems Ltd.* (1990), 48 B.L.R. 147, 32 C.P.R. (3d) 414 (B.C.S.C.); *Positron Inc.* v. *Desroches*, [1988] R.J.Q. 1636 (S.C.)).

[68] *Copyright Act*, *ibid.*, s. 5.

[69] Handa & Buchan, *supra* note 62 at 52.

[70] The word "original" does not in this connection mean that the work must be the expression of a creative 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 in writing. The originality which is required relates to the expression of the thought (*University of London Press Ltd.* v. *University Tutorial Press Ltd.*, [1916] 2 Ch. 601 at 608; see also *Delrina*, *supra* note 5 at 32).

d. Fixation

In order to be worthy of copyright protection, a work must also be fixed or stored in some manner.[71] For computer programs, section 2 of the *Act* explicitly requires that a computer program be "expressed, fixed, embodied or stored in any manner" for copyright protection to apply. The American *Copyright Act* is slightly more specific, in that it requires a work to be expressed in a form which is sufficiently permanent and stable such that it may be "perceived, reproduced, or otherwise communicated, either directly or with the aid of a machine or device."[72]

Issues of fixation arise frequently in relation to computer programs and without further legislative or juridical guidance, a good deal of legal uncertainty will persist. Recent jurisprudence addressing the protection of non-literal elements, such as computer screens, suggests that fixation in volatile memory devices, such as a video interface's RAM,[73] is sufficient to meet the fixation requirement.[74] For example, computer screens are protected because the screens are said to exist under the umbrella of the underlying computer program's copyright.[75] A more difficult question is whether transitory combinations of data, such as the results of a database search conducted at the direction of a user, are sufficiently fixed for the purposes of copyright. These results are often only stored in volatile memory, and cannot be said to be part of the underlying search engine as the user's search criteria are entered only upon use.

In terms of the reverse engineering debate, the issue of fixation is important since in order to reverse engineer a computer program, one must first "copy" that program into a computer's RAM memory to allow for disassembly. As the disassembler performs its passes through the program, it makes increasingly precise translations of the program, which are often stored only in RAM.[76] Once the disas-

---

[71] In *Canadian Admiral Corp.* v. *Rediffusion Inc.*, [1954] Ex. C.R. 382 at 394 [hereinafter *Rediffusion*], the Court ruled 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."

[72] *Copyright Act (U.S.), supra* note 6, § 102.

[73] RAM, also known as volatile or dynamic memory, is a form of internal memory that stores information as long as electrical impulses are being fed through it. Once the power is cut off, the RAM microchips lose all their stored information. RAM is the functional memory that allows a computer to operate, since the programs or parts thereof which the computer processes must be stored in RAM during the operation of the computer (with the exception of ROM programs). If programs are otherwise stored on external memory devices, they must be copied into RAM memory in order to be executed.

[74] See *Delrina, supra* note 5.

[75] Note, however, that a fleeting image broadcast on television has been held not to fulfil the requirements of fixation for the purposes of copyright. If the television program is otherwise fixed, such as on some form of video tape, then the fleeting image described may be protected as part of the underlying copyright of the fixed program (*Rediffusion, supra* note 71 at 396).

[76] Depending on the disassembler and the size of the program code being disassembled, these intermediate translations may be temporarily stored on disk, at which point fixation issues become ir-

sembly is completed, the results are usually stored on more permanent media such as a disk, and in print-out form. It is, however, possible to disassemble a program, or parts thereof, without storing either the intermediate copies or final result in any media other than RAM. Although this may seem impractical since the results will remain within the dynamic RAM of the computer, it may provide a technical way around the difficulties of copyright infringement since without adequate fixation, copyright laws may not consider that a copy has indeed been made. Furthermore, if it is deemed that, even where the final product of disassembly is fixed, the intermediate copies stored in RAM do not constitute infringing copies for want of fixation, the exception to copyright found in paragraph 27(2)(l) of the Canadian *Copyright Act*, discussed below, may apply.[77]

e. Infringement of Copyright

Once copyright subsists in a work, it will be infringed by "any person who, without the consent of the owner of the copyright, does anything that, by th[e *Copyright Act*], only the owner of the copyright has the right to do."[78] It must be stressed that copyright only prevents the copying of a work, or a substantial part thereof;[79] the monopoly rights granted by the *Copyright Act* do not extend to situations where a person independently creates a similar work. Accordingly, the jurisprudence dealing with copyright infringement contends that the onus to demonstrate copying rests on the plaintiff, who must establish both a substantial similarity between his or her work and that of the defendant, as well as a causal connection between the two works.[80] The causal connection requirement can be satisfied by demonstrating that the defendant had access to the original work. Once the plaintiff has discharged the burden, a rebuttable presumption arises whereby the onus of proving independent creation shifts to the defendant. The defendant may also attempt to rely on the exceptions to infringement contained in subsection 27(2) of the *Copyright Act*.

---

relevant.

[77] Paragraph 27(2)(l) allows a *single* reproduction of an authorized copy of a computer program to be made for purposes of compatibility. In order to use that paragraph to exempt disassembly from infringing copyright, the intermediate copies would have to somehow be exempt. If intermediate copies were not qualified as fixed, they could thus be exempt from copyright (see Part II.A.1.*f*.i, below).

[78] *Copyright Act*, *supra* note 1, s. 27(1).

[79] "Substantial" refers not to a strict percentage, but rather to the quality of the part taken (*Breen* v. *Hancock House Publishers Ltd.* (1986), 6 C.I.P.R. 129 at 133, 6 C.P.R. (3d) 433 (F.C.T.D.) [hereinafter *Breen* cited to C.I.P.R.]). In *SAS*, *supra* note 41 at 822, the Court found that 44 examples of copying had occurred out of a total of approximately 186,000 lines of computer source code. It held that this constituted a substantial taking, and that the small number of instances of copying did not render the copying trivial.

[80] *Gondos* v. *Hardy* (1982), 38 O.R. (2d) 555, 64 C.P.R. (2d) 145 (H.C.J.); *Francis Day & Hunter Ltd.* v. *Bron*, [1963] Ch. 587 (C.A.).

f. Exceptions to Infringement

i. Translation — Modification Exception

If the intermediate copies or final results of the disassembly process are indecd considered fixed, the copying of a computer program for use by a computer *may* be exempted by paragraph 27(2)(l) of the *Copyright Act*. The crucial issue is whether such use includes disassembly. This argument is strengthened where the intermediate copies avoid infringement due to lack of fixation, and only the single final disassembled product is fixed. Paragraph 27(2)(l) states that infringement does not occur as a result of

> the making by a person who owns a copy of a computer program, which copy is authorized by the owner of the copyright, of a single reproduction of the copy by adapting, modifying or converting the computer program or translating it into another computer language if the person proves that
>
> (i) the reproduction is essential for the compatibility of the computer program with a particular computer,
> (ii) the reproduction is solely for the person's own use, and
> (iii) the reproduction is destroyed forthwith when the person ceases to be the owner of the copy of the computer program.[81]

In drafting this paragraph, "[t]he House of Commons Sub-Committee on the Revision of Copyright recognized that it is common in the industry for computer programs to be adapted or modificd to meet the particular needs of end users."[82] Accordingly, disassembly was not contemplated as being within the scope of the section.

Nevertheless, this paragraph may, in fact, also be used to support the contention that, in order to function, computer programs must be copied into parts of a computer (usually into RAM) and translated by the central processing unit into microcode.[83] Although it can be argued that paragraph 27(2)(l) protects a computer

---

[81] Computer programs, however, are generally licensed and only rarely sold by the copyright holder. See *infra* note 87 and accompanying text for a discussion of licensing, as contrasted with sale, of software products.

[82] Sookman, *supra* note 48 at 3-203.

[83] A similar exception was placed into the American *Copyright Act* in 1978. The purpose of section 117 was to give authorized users the right to use a computer program without technically infringing the copyright in the program. The section reads:

> Notwithstanding the provisions of section 106, it is not an infringement for the owner of a copy of a computer program to make or authorize the making of another copy or adaptation of that computer program provided:
>
> (1) that such a new copy or adaptation is created as an essential step in the utilization of the computer program in conjunction with a machine and that it is used in no other manner ...

The American exception has, however, been held not to apply to the reverse engineering of computer programs (*Sega*, *supra* note 4 at 1520-21; see also text accompanying note 204, below).

program's use from being declared an infringement, it is more tenuous to argue that this paragraph also applies where a program is copied into RAM and subsequently onto more permanent media for the purposes of dissection by a disassembler.[84] Where the simple use of a program is concerned, however, even if paragraph 27(2)(l) does not apply, it is unlikely that the copying of the program for the purposes of use will be declared an infringement because the conversion of object code into electrical signals may not be a "reproduction of the work in a material form."[85] The fact that the simple intended use of an authorized copy of a computer program could potentially constitute a copyright infringement provides a good illustration of the inappropriateness of copyright as the principal form of intellectual property protection for computer programs.[86]

ii. Making Backup Copies

In addition to the translation/adaptation/modification exception contained in paragraph 27(2)(l), there are two other exceptions to copyright infringement that also apply to computer programs. The more specific of these exceptions is in paragraph 27(2)(m) of the *Copyright Act*, and allows the owner of an authorized copy of a computer program to make a single copy for backup purposes, but requires that this backup be destroyed if the person ceases to be the owner of the authorized copy. Practically speaking, this provision, in addition to paragraph 27(2)(l), has proven to be of limited use because software companies seldom transfer the ownership of their software, preferring instead to license it to users. Licensing effectively allows the software companies to supersede certain provisions of the *Act*, such as this backup exception to infringement, by manipulating terms of their licensing agreement.[87] Most software companies do, nonetheless, permit the making of a

---

[84] The right [under paragraph 27(2)(l)] to convert a computer program or translate it into another computer language will probably give a person who owns an authorized copy of a computer program the right to convert the program from one higher-level language to another. It might also give such a person the right to convert a program from a higher-level language to machine language, *and vice-versa*, but the meaning of the term 'translation' in the Act is still uncertain ... and so the scope of [27(2)(l)] ... is still not known (Sookman, *supra* note 48 at 3-204 [emphasis added]; see also *infra* note 107 and accompanying text).

[85] *Apple*, *supra* note 8 at 30.

[86] Sookman, *supra* note 48 at note 17 and accompanying text, p. 3-3.

[87] As a result of the mass production of off-the-shelf software, it has become impossible for software publishers and vendors to individually negotiate licence terms with each prospective purchaser; "shrink-wrap" licences were therefore developed. These licences consist of a list of licensing terms placed visibly on a product (usually under the cellophane wrapper). Typically, the licence contains a clause that states that if the purchaser does not agree to abide by the terms of the licence he or she should not purchase the product, and, further, that by opening the package the purchaser agrees to abide by its terms. It is unclear whether shrink-wrap licences are enforceable in Canada even though they are an extremely common industry practice. In an oft-cited passage from *Betts* v. *Wilmott* (1871), L.R. 6 Ch. App. 239, the Court held that "[w]hen a man has purchased an article he expects to have the control of it, and there must be some clear and explicit agreement to the contrary to justify the vendor in saying that he has not given the purchaser his licence to sell the article, or to use it wherever

backup copy in their licensing agreement as it avoids the inconvenience of replacing programs if and when the carrying media become defective.[88]

Licensing terms can also be used to prevent the reverse engineering of the computer program by the user. If the user is a licensee rather than the *owner* of a copy of the computer program, then *prima facie* the licence terms will prevail notwithstanding the permissibility of reverse engineering under the *Copyright Act*. If a licence is silent or ambiguous as to its interpretation, the court will fill in the gaps using implied terms reflective of industry practice.[89] Licence terms that run afoul of the *Copyright Act* have not been tested in Canadian courts; however, the experience in the United States suggests that such terms may not be enforceable.[90]

iii. Fair Dealing Under the *Copyright Act*

Another more general copyright exception is the fair dealing exception found in subsection 27(2) of the *Copyright Act*. According to that subsection, "any fair dealing with any work for the purposes of private study [or] research" will not constitute copyright infringement. The *Act* does not further detail the breadth of the fair dealing exception, and very little jurisprudence has elaborated upon its interpretation. Similar exceptions exist in British copyright legislation, where the term "fair dealing" is also used, and more importantly for the purposes of reverse engineering computer programs, in the American copyright legislation where the term "fair use" is used.[91] As a result of the paucity of Canadian jurisprudence on the subject of fair dealing, it is impossible to say how similar our exception will be to those of other jurisdictions. [92] In the context of Canadian cases "related to copyrightability of computer programs, American authorities have been cited and, notwithstanding the differences between the wording of the Acts, have been given qualified approval."[93]

---

he pleases as against himself." In *North American Systemshops Ltd.* v. *King* (1989), 97 A.R. 46, 68 Alta. L.R. (2d) 145 (Q.B.), the Court found unenforceable a shrink-wrap licence agreement not visible to the purchaser at the time of purchase. It remains to be seen whether visible shrink-wrap licences fulfil the requirement of a "clear and explicit agreement".

[88] Unlike in the Canadian and American *Copyright Acts*, a purchaser cannot contract out of similar backup copy provisions contained in the British *Copyright, Designs and Patents Act 1988* (U.K.), 1988, c. 48 (see *infra* note 229 and accompanying text).

[89] As with any new industry, the difficulty with implied licences in the computer software industry is that business practices often are only partially established at the point when a court is forced to decide upon a relationship with implicit terms.

[90] See Part **II.C.**1, below, for a detailed discussion of trade secrets and licensing.

[91] United Kingdom: *Copyright, Designs, and Patent Act 1988*, *supra* note 88, s. 29(1); United States: *Copyright Act*, *supra* note 6, § 107 (see *infra* note 193 for the text of the section).

[92] It is widely believed that "fair dealing" and "fair use" are different concepts. With respect to the home taping issue decided by the United States Supreme Court in *Sony Corporation of America* v. *Universal City Studios, Inc.*, 464 U.S. 417, 78 L. Ed. (2d) 574 (1984) [hereinafter *Sony* cited to U.S.], "there are sufficient differences between the American 'fair use' defence and Canada's 'fair dealing' to conclude that, if an action were brought in Canada, home taping would be found to constitute a copyright infringement" (M. Hébert, *Copyright Act Reform*, rev. ed. (Ottawa: Research Branch, Library of Parliament, 1987) at 11).

[93] Sookman, *supra* note 48 at 3-6.

No court has yet performed a comprehensive comparative analysis of the Canadian "fair dealing" and the American "fair use" exceptions.

The earliest Canadian case to deal extensively with the fair dealing exception was *Zamacoïs* v. *Douville*.[94] In that case, the Exchequer Court laid out the basic principles that govern fair dealing: a verdict of fair dealing must depend on the specific facts of each case; the copying of an entire work cannot qualify as fair dealing; short of copying the entire work, the quantity of the work copied is not solely determinative of fair dealing; and "in considering whether a dealing with a particular work [is] fair, it would have to be considered whether any competition [is] likely to exist between the two works."[95] The other two Canadian cases to deal with the defence of fair dealing both faced the question whether an abridgment or summary of a work could avoid being declared an infringement through a claim of fair dealing.[96] In both cases, the courts held that merely summarizing a work without the addition of some further comment is not fair dealing.

In *Hubbard* v. *Vosper*, the United Kingdom Court of Appeal stated:

> [I]t is impossible to define what is 'fair dealing'. It must be a question of degree. You must consider first the number and extent of the ... 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 matter of impression.[97]

In *Beloff* v. *Pressdram Ltd.*,[98] the United Kingdom Chancery Court expanded the application of the fair dealing defence to unpublished and confidential information. In *Beloff*, the defendant newspaper company published an internal office memorandum written by the plaintiff without obtaining the plaintiff's authorization. The defendant claimed that its purpose in publishing the plaintiff's work was to criticize it and that it was therefore covered by the fair dealing exception under the *Copyright Act*. The Court held that the fact that the memorandum had not been published was not in itself enough to find in favour of the plaintiff; however, it was an aggravating factor to be taken into account in assessing the defendant's conduct. With respect to the confidentiality of the information used, the Court held that

---

[94] (1943), 3 Fox Pat. C. 44, [1943] 2 D.L.R. 257 (Ex. Ct.) [hereinafter *Zamacoïs* cited to Fox Pat. C.].

[95] *Ibid.* at 72-74.

[96] *Breen*, *supra* note 79 at 133; *R.* v. *James Lorimer & Co.*, [1984] 1 F.C. 1065 at 1077-78, 77 C.P.R. (2d) 262 (C.A.) [hereinafter *Lorimer* cited to F.C.].

[97] [1972] 1 All E.R. 1023 at 1027 (C.A.) [hereinafter *Hubbard*].

[98] [1973] 1 All E.R. 241 (Ch.) [hereinafter *Beloff*].

> [t]he vice of the leak of the publication in this case was, to my mind, clearly unjustifiable for the authorised purposes of criticism, review and news, and clearly in my view constituted dealing which was not fair within the statute. ... This ground is ample to defeat the defence of fair dealing.[99]

Each of the aforementioned cases dealt with fair dealing in the context of traditional literary works. No case in Canada or in the United Kingdom has applied the defence of fair dealing in the context of a computer program. The American experience with the fair use exception has been quite different and will be discussed in further detail in Part III.

iv. Public Interest Exception

A final potential exception to copyright infringement in reverse engineering cases is the public interest defence. This defence is judicially created, and does not expressly appear in Canadian copyright legislation.[100] The public interest defence has not yet been raised successfully in Canada with respect to copyright, and has only gained judicial recognition in one small passage in *Lorimer*. In that case, Mahoney J. stated:

> I have no doubt that a defence of public interest as enunciated in the English cases is available in proper circumstances against an assertion of Crown copyright. ... [However t]his is not a "public interest" case in the same sense as the English decisions nor, really, in the sense the defence was advanced here.[101]

The English cases to which Mahoney J. was referring were *Hubbard* and *Beloff*. In both of those cases, the courts recognized that a common law defence of public interest was available notwithstanding its lack of statutory support. In this regard, Ungoed-Thomas J. in *Beloff* stated that the defences of public interest and fair dealing are

> separate defences [which] are governed by separate considerations. Fair dealing is a statutory defence limited to infringement of copyright only. But public interest is a defence outside and independent of statutes, is not limited to copyright cases and is based on a general principle of common law.[102]

The discussion of a public interest defence in the context of a copyright infringement claim was further examined in *Lion Laboratories Ltd.* v. *Evans*.[103] In that case, the United Kingdom Court of Appeal had to decide whether the theft of

---

[99] *Ibid.* at 264.

[100] Subsection 171(3) of the United Kingdom's *Copyright, Designs and Patents Act 1988* expressly recognizes the existence of a public interest defence and states: "Nothing in this Part affects any rule of law preventing or restricting the enforcement of copyright, on grounds of public interest or otherwise."

[101] *Lorimer, supra* note 96 at 1078-79.

[102] *Beloff, supra* note 98 at 259.

[103] [1985] Q.B. 526 (C.A.).

confidential literature that was subsequently published without the copyright holder's permission, could be held to be in the public interest. The literature in question was an internal memorandum suggesting that a breathalyzer device manufactured by the plaintiff was capable of giving false readings which may have led to the conviction of innocent persons. The defendants did not deny that they took the confidential information without the plaintiff's permission, or that its publication did not *prima facie* infringe the plaintiff's copyright; the only defence presented was one of public interest. In deciding how to weigh the public interest against the copyright infringement and breach of confidence perpetrated by the defendants, Stephenson L.J. stated:

> To be allowed to publish confidential information, the defendants must do more than raise a plea of public interest; they must show "a legitimate ground for supposing it is in the public interest for it to be disclosed." ... [W]e "should not restrain it by interlocutory injunction, but should leave the complainant to his remedy in damages."[104]

Griffiths L.J. agreed, and in assessing the applicability of the public interest defence to copyright infringement, stated: "I am quite satisfied that the defence of public interest is now well established in actions for breach of confidence and, although there is less authority on the point, that it also extends to breach of copyright."[105]

Although the jurisprudence dealing with public interest is solidly in place in the United Kingdom, the sole mention in Canada of a public interest copyright defence considered its possible existence only in the context of Crown copyright. It would, however, be nonsensical if it did not also extend to defend against claims by private copyright holders. As it stands, a public interest defence seemingly exists in Canada, independently of any statutory exception to copyright.[106]

It will be shown that both the fair dealing exception and the public interest defence are essential to an argument in support of reverse engineering. With respect to the former, the American fair use exception has been successfully applied in reverse engineering cases, and notwithstanding the differences between the respective exceptions, fair dealing presently remains the most likely candidate to allow the reverse engineering of computer programs under Canadian copyright law. If the fair dealing defence fails to support an argument allowing reverse engineering, then a more tenuous, although nonetheless plausible, claim may be made under the principle of public interest. Copyright, however, may not be the sole hurdle to the reverse engineering of computer programs. Additional protections, such as those

---

[104] *Ibid.* at 538, quoting from *Khashoggi* v. *Smith* (15 January 1980), No. 58 at 15 (C.A.), Sir David Cairns and *Woodward* v. *Hutchins*, [1977] 1 W.L.R. 760 at 764 (C.A.), Lord Denning M.R.

[105] *Ibid.* at 550.

[106] Although section 63 of the *Copyright Act* effectively abolishes common law copyright, it makes no mention of common law defences to copyright infringement. The existence of an implied public interest defence may be rooted in the modern-day general underpinnings of copyright law protection, which seek to balance the public's right to knowledge with the individual's right to be remunerated for work (Sookman, *supra* note 48 at note 2 and accompanying text, p. 3-1).

raised under trade secrets law or by licensing provisions which seek to expand the scope of copyright protection, may also frustrate any attempt to reverse engineer a computer program.

B. Is Reverse Engineering an Infringement Of Copyright Law?

With the fundamental principles in place, we are now prepared to consider whether reverse engineering is indeed an infringement of copyright law, and if so, how. As mentioned above, when reverse engineering a computer program, the disassembler must first load a copy of the program into a computer's memory.[107] This results in the first potentially infringing copy. As the disassembler makes passes over the program, it will continue to produce translations of the work which constitute potentially infringing copies of the program according to paragraph 3(1)(a) of the *Copyright Act*. These copies are referred to as intermediate copies.[108] Once the disassembler has completed its task, it will produce an ASSEMBLER language version of the computer program. This ASSEMBLER source code constitutes yet another potential infringement of the computer program's copyright, although the assembler will not generally reconstitute a program in the exact fashion in which it was written. The resulting source codes will be a translation rather than a reproduction;[109] but either would potentially infringe the original work's copyright.[110] Finally, making a hard, or printed, copy of the work for further examination of the computer program's operating principles will also potentially infringe copyright.[111] The difficulty with reverse engineering a computer program lies in the fact that the program must be put into memory for decompilation. The act of reverse engineering is thus not in itself a violation of copyright, but rather the means which it employs are.[112] This article posits, however, that because of the need to develop standards and achieve program compatibility, reverse engineering can be contemplated within

---

[107] Although the operating system also copies a computer program to execute the program, this action is not regarded as an infringement of copyright (see *supra* notes 84-85 and accompanying text).

[108] *Atari*, *supra* note 4 at 842.

[109] In the case of computer programs, the distinction between a reproduction and a translation for the purposes of copyright law is irrelevant (*Apple* (S.C.C.), *supra* note 8 at 261).

[110] In the United States, the copies of the computer program made by the disassembler in final form would be termed "derivative works". A derivative work is

> a work based upon one or more preexisting works, such as a translation, musical arrangement, dramatization, fictionalization, motion picture version, sound recording, art reproduction, abridgment, condensation, or any other form in which a work may be recast, transformed, or adapted. A work consisting of editorial revisions, annotations, elaborations, or other modifications which, as a whole, represent an original work of authorship, is a "derivative work" (*Copyright Act (U.S.)*, *supra* note 6, § 101).

[111] Ignatin, *supra* note 17 at 2011-12.

[112] Reverse engineering of other products can be accomplished without any copying, simply by obtaining the necessary examples, taking them apart, studying and testing them (*ibid.* at note 50, p. 2011). The idiosyncratic application of copyright law to computer program products makes the inspection of these products a potential infringement. Once again, it is apparent that copyright may not provide the optimal manner by which to protect these products.

the scope of the fair dealing provision in the *Copyright Act*.[113] Furthermore, it will be argued that intermediate copies produced during reverse engineering should be exempted from copyright through a statutory exception, since the prohibition of reverse engineering falls beyond the scope of copyright protection. The rationale for these arguments will be more fully discussed in Part IV.

C. *Reverse Engineering Under Other Legal Regimes*

1. Trade Secrets

The law governing trade secrets is intertwined with the law of copyright insofar as it applies to reverse engineering. Trade secret law is judge-made law that protects commercial confidences.[114] It can coexist with other intellectual property protections, and in the case of literary, dramatic, artistic or musical matter, may even protect the underlying ideas which are not in themselves copyrightable.[115] Trade secret protection covers a broader range of informational elements than do other forms of intellectual property proteetion, but does so in a more limited manner.

Canadian and British courts have applied three general requirements for a successful trade secret claim: (1) the information must have the necessary quality of confidence; (2) the information must have been imparted in circumstances importing an obligation of confidence (*i.e.* a "special relationship" must exist between the parties); and (3) there must be an unauthorized subsequent use of that information to the detriment of the party communicating it.[116] The burden of proving the secrecy of the information falls on the plaintiff.[117] However, once the information loses its quality of secrecy through authorized disclosure, legitimate independent research or any other honest means, its purveyor may no longer avail him or herself of trade secret protection.

---

[113] In 1984-85, the Canadian government did consider implementing in the *Copyright Act* what effectively amounted to a statutory exception which would have allowed reverse engineering of computer programs. This proposal was rejected (see *infra* note 257).

[114] According to the *Restatement of Torts* § 757b (1939), "[a] trade secret may consist of any formula, pattern, device or compilation of information which is used in one's business, and which gives him an opportunity to obtain an advantage over competitors who do not know or use it." This definition was accepted in Canada in *R.L. Crain Ltd.* v. *Ashton* (1949), O.R. 303 at 308, [1949] O.W.N. 246 (H.C.J.), aff'd [1950] O.R. 62, [1950] 1 D.L.R. 601 (C.A.).

[115] See *Q-Co Industries, Inc.* v. *Hoffman*, 625 F.Supp. 608 (S.D. N.Y. 1985).

[116] See *Lac Minerals Ltd.* v. *International Corona Resources*, [1989] 2 S.C.R. 574, 69 O.R. (2d) 287 [hereinafter *Lac Minerals* cited to S.C.R.]; *Software Solutions Associates Inc.* v. *Depow* (1989), 99 N.B.R. (2d) 110, 250 A.P.R. 110 (Q.B.); *Coco* v. *A.N. Clark Ltd.*, [1969] R.P.C. 41 (Ch.); *Ridgewood Resources Ltd.* v. *Henuset* (1982), 35 A.R. 493, 18 Alta. L.R. (2d) 68 (C.A.).

[117] The possessor of the secret neither has to go to unreasonable lengths to maintain secrecy (*Creditel of Canada Ltd.* v. *Faultless* (1977), 18 O.R. (2d) 95, 81 D.L.R. (3d) 567 (H.C.J.)), nor does he or she have to guard against unanticipated, undetectable or unpreventable methods of discovery (*International Corona Resources Ltd.* v. *Lac Minerals Ltd.* (1986), 53 O.R. (2d) 737, 25 D.L.R. (4th) 504 (H.C.J.)). See also Sookman, *supra* note 48 at 4-29.

Trade secret protection is often considered to be based on the laws of property, contracts and/or trusts. In Canada, Sopinka J. stated that the legal basis for trade secret actions is "*sui generis* relying of each of these areas to enforce the policy of the law that confidences be respected."[118] This law of confidentiality arises from an obligation of good faith in commercial settings or from a fiduciary relationship. Where no fiduciary relationship exists, the courts will look for a fiduciary quality to the relationship which would require an element of confidence. For example, the relationships between manufacturers and designers, licensors and licensees, joint venturers intending to do business with one another, and employees may all fall within the ambit of trade secret laws.

Section 63 of the *Copyright Act* states that the *Act* is to be the sole source of copyright, but the section also contemplates the existence of trade secret laws that may operate independently of the copyright legislation.[119] With respect to the reverse engineering debate, the applicability of trade secret laws is generally ensured through licensing agreements or contracts between the parties.

Although section 63 most certainly allows trade secret laws to supplement copyright principles, it is not clear whether trade secret laws would be paramount in a situation of conflict between the two laws.[120] It has been suggested that in the case of reverse engineering computer programs, the true aim of trade secret restrictions "is not preservation of confidentiality or security against disclosure to third parties; it is foreclosure of competition."[121]

Clearly, an agreement purported to restrain competition, rather than to protect against a breach of trust or confidence, is not contemplated by section 63 and will infringe both copyright and competition laws.[122] Practically speaking, the intent of most contracts that restrain reverse engineering supports both objectives. In such a case, it is not clear whether a court would rely solely on the primary purpose of the agreement, or whether an incidental objective to restrain a breach of confidence or trust would suffice to support the paramountcy of the terms of the agreement over copyright law.

---

[118] *Lac Minerals*, *supra* note 116 at 615. Aside from the common law basis for dealing with trade secrets, several American states have enacted trade secret statutes (see M.D. Stein, "The Importance of a Trade Secret as a Supplement to Copyright Protection of Computer Software" (1993) 12:1 I.P.L. Newsletter 28 at 29).

[119] The text of section 63 reads:

> No person is entitled to copyright or any similar right in any literary, dramatic, musical, or artistic work otherwise than under and in accordance with this Act, or of any other statutory enactment for the time being in force, but nothing in this section shall be construed as abrogating any right or jurisdiction to restrain a breach of trust or confidence.

[120] See *infra* note 154 and accompanying text.

[121] Rice, *supra* note 26 at 623.

[122] See Part II.C.2.*a*, below.

When computer programs are mass produced, however, the situation is different because no negotiated agreements are involved. If an object code copy of computer software is released to the public without a licensing agreement, there is no "special relationship" between the parties, and for the purposes of trade sccrects law, members of the public are free to reverse engineer the product to determine the source code. In the absence of a copyright prohibition on reverse engineering (imparting an obligation of confidence), anyone not in a "special relationship" with the computer program owner is legally free to attempt to reverse engineer a lawful copy of the computer program.[123] Generally, however, mass-marketed computer programs use shrink-wrap license agreements to enhance their copyright.[124]

2. Enhancing Copyright Protection Through Licensing/Contract Law

Difficulties immediately arise with the use of shrink-wrap licences to enhance copyright protection. First, with mass-marketed software it is unclear whether shrink-wrap licensing is valid. For the purposes of a trade secrets argument, it seems unlikely that a shrink-wrap agreement would sufficiently constitute the "special relationship" required between the software developer and the purchaser for trade secret protection. This type of relationship is more likely to exist where the licensing terms have been negotiated between the vendor and purchaser. Second, if reverse engineering is deemed to be in the public interest under copyright legislation, trade secret protection, as a creature of the common law, might not be extended to cover the reverse engineering of computer programs. This would be especially likely if the purpose of the licence agreement were found to be a restraint of competition. The following three subsections set out defences that may be used where a copyright holder attempts to prevent a user from reverse engineering a computer program through licensing or contract provisions, assuming that such reverse engineering is permitted under the *Copyright Act*.

a. Copyright Misuse Doctrine

The general question whether contract/licensing law can indecd supersede copyright law provisions in a situation of conflict is yet another issue that has still to be comprehensively addressed by the courts. Specifically, if copyright law supports a limited revcrse engineering exception, it is debatable whether protcction

[123] In *GEAC Canada Ltd.* v. *Prologic Computer Corp.* (1989), 35 B.C.L.R. (2d) 136, 25 C.P.R. (3d) 91 (S.C.), the plaintiff alleged that one of the defendants, who was in a software licensing relationship with the plaintiff, had violated "its common-law duty of confidentiality" to the plaintiff by allowing another of the defendants to reverse engineer the lieensed software, and to use the results in the creation of compatible and even competing products. The plaintiff's claim was rejected; however, the Court based its ruling solely on the fact that the plaintiff was guilty of laches and acquiescence. The Court did not express an opinion on the issue of reverse engineering as it relates to trade secrcts *per se*, although it did state that, broadly speaking, the claims raised "serious issues to be tried." No claim concerning a potential infringement of copyright was made by the plaintiffs, and, accordingly, the Court in no way addressed any issue related to copyright.

[124] See *supra* note 87 for a discussion of shrink-wrap licensing.

obtained through licensing can be used to override the copyright exception. Although Canadian courts have not yet faced this problem, American courts have developed a "copyright misuse doctrine" that may potentially be used to pre-empt the "enforcement of software license terms that prohibit reverse engineering."[125] An attempt to widen copyright law beyond its limited statutory scope through contraet would, under the misuse doctrine, render any attempt to enforce one's copyright invalid.[126]

The copyright misuse doctrine was recently applied by the United States Federal Court of Appeals in *Lasercomb America, Inc.* v. *Reynolds*,[127] where the Court held that the misuse doctrine extends to render "a copyright unenforceable against any person regardless of whether they entered into a contract containing the offending term."[128] Under the ruling in *Lasercomb*, merely *attempting* to widen the scope of copyright protection beyond its accepted limits is a bar to its use. Although the United States Supreme Court has not applied the copyright misuse doctrine, it did acknowledge its existence in *United States* v. *Loew's Inc.*[129] The fact that the misuse defence is grounded in equity means that claimants must show clean hands in order to use it,[130] and that the defence is potentially available to Canadian litigants.[131]

In fact, the existence of a copyright misuse defence has been implicitly acknowledged by the Supreme Court of Canada in *Massie & Renwick Ltd.* v. *Underwriters' Survey Bureau Ltd.*[132] In that case, the defendants, aceused of copyright infringement, alleged that the plaintiffs' withholding of their plans and insurance rating schedules constituted "a combine and conspiraey" under both the *Combines Investigation Act*[133] and the *Criminal Code*.[134] The trial judge stated that he did not have to rule on the misuse defence because the defendants failed to demonstrate that the plaintiffs were guilty of anti-competitive behaviour. In *obiter dictum*, the trial judge held:

---

[125] Rice, *supra* note 26 at 551.

[126] *Ibid.*

[127] 911 F.2d 970 (4th Cir. 1990) [hereinafter *Lasercomb*].

[128] Rice, *supra* note 26 at note 24, p. 551.

[129] 371 U.S. 38, 9 L. Ed. (2d) 11 (1962). See also *Atari*, *supra* note 4 at 846.

[130] See *Atari*, *ibid.*

[131] Principles of both common law and equity apply to actions concerning federal laws in Canada (*Aldrich* v. *One Stop Video Ltd.* (1987), 13 B.C.L.R. (2d) 106, 17 C.P.R. (3d) 27 (S.C.)).

[132] [1940] S.C.R. 218, [1940] 1 D.L.R. 625 [hereinafter *Massie (1940)* cited to S.C.R.], var'g [1938] Ex. C.R. 103, [1938] 2 D.L.R. 31 [hereinafter *Massie (1938)* cited to Ex. C.R.]. A defence based on misuse has also been raised in several Canadian patent cases (see *e.g. Philco Products Ltd.* v. *Thermionics Ltd.*, [1939] Ex. C.R. 147, [1939] 3 D.L.R. 133, aff'd [1940] S.C.R. 501; *Thermionics Ltd.* v. *Philco Products Ltd.*, [1941] Ex. C.R. 209, 1 Fox Pat. C. 166, var'd [1943] S.C.R. 396, 3 Fox Pat. C. 92; *Eli Lilly & Co.* v. *Marzone Chemicals Ltd.* (1976), 29 C.P.R. (2d) 253 (F.C.T.D.), aff'd [1977] 2 F.C. 104, 29 C.P.R. (2d) 255 (C.A.)).

[133] R.S.C. 1927, c. 26.

[134] R.S.C. 1927, c. 36, s. 498.

> Even if the wrongs imputed against the plaintiffs were established in fact, I do not think that would deprive them of their right to protect their copyright; their copyrights would not perish because they had offended against another statute.[135]

On appeal, the Supreme Court of Canada upheld the trial judge's ruling, but with respect to the statement as to misuse, the Court added:

> [I]f the plaintiffs in an action for the infringement of copyright are obliged, for the purpose of establishing the existence of, and their title to, the copyright to rely upon an agreement and that agreement constitutes a criminal conspiracy, and their title rests upon such agreement and upon acts which are criminal acts by reason of their connection with such an agreement, then it would be difficult, on general principles to understand how sueh an action could succeed.[136]

The Supreme Court's ruling in *Massie (1940)* suggests that some form of copyright misuse defence may be available in Canada. This can be reconciled with the Federal Court of Appeal's decision in *Bell Canada* v. *Intra Canada Telecommunications Ltd.*[137] which seemed to reject a defence of misuse by recognizing that the former case dealt with a criminal conspiracy while the latter dealt solcly with anti-competitive behaviour under the *Combines Investigation Act*.[138] It has often been thought that the misuse defence, though grounded in equity, has its origins in principles which encourage competition. It is not clear from these cases, however, whether a mere finding of anti-competitive behaviour will suffice to invoke the misuse defence, or whether some other violation, such as a criminal act, is required. The current *Competition Act*, which has replaced the *Combines Investigation Act* as the guardian of competitive behaviour, has its own remedies for intellectual property misuse.[139]

b. Pre-emption of Conflicting Laws

The United States Court of Appeals' decision in *Lasercomb*, while clearly acknowledging the existence of a copyright misuse defence, did not deal with reverse engineering *per se*. The only American case to deal with the conflict between copyright and contract in the context of reverse engineering was *Vault Corp.* v. *Quaid Software Ltd.*[140] In that case, Heebe J. held that contract terms which widened the scope of rights granted by the *Copyright Act* and by the legislation that sanctioned the use of these terms, were pre-empted by the federal copyright legisla-

---

[135] *Massie (1938)*, *supra* note 132 at 119.

[136] *Massie (1940)*, *supra* note 132 at 244.

[137] (1982), 70 C.P.R. (2d) 252 (F.C.A.), rev'g (1982), 62 C.P.R. (2d) 21 (F.C.T.D.).

[138] Pratte J., facing a claim that anti-competitive behaviour by the plaintiff, in contravention of the *Combines Investigation Act*, R.S.C. 1970, c. C-23, serves as a defence in an action for copyright infringement, stated, *ibid.* at 253: "We cntertain serious doubts that they constitute a valid defence to the action."

[139] *Competition Act*, R.S.C. 1985, c. C-34, s. 32. See Part II.C.3, below.

[140] 655 F.Supp. 750 (E.D. La. 1987), aff'd 847 F.2d 255 (5th Cir. 1988) [hereinafter *Vault*].

tion.[141] The case concerned the validity of the Louisiana *Software License Enforcement Act*,[142] a state statute that allowed for the enforceability of shrink-wrap licences. Heebe J. ruled that the state legislation widened the protections granted by section 106 of the Ameriean *Copyright Act*, and allowed contractual terms to impede the archival copy privilege conferred on authorized users by section 117.[143] The Court of Appeals (Fifth Circuit) upheld the District Court's decision, stating that the state legislation was pre-empted "because it touched upon federal copyright in a manner that set fcderal policy at naught."[144]

In a Canadian context, one would argue that any provineial law that expressly permits restrictions on rights and privileges guaranteed by the *Copyright Act* is *ultra vires* because copyright is an exelusively federal power by virtue of subsection 91(23) of the *Constitution Act, 1867*.[145] This, however, does not guarantee that provinces will not create legislation in their own areas of competence which impacts indirectly on copyrighted works. The law of contract, as well as the power to create and administer property rights, falls within the property and eivil rights powers given to the provinces.[146] Accordingly, "the publication, distribution and sale of ... [many] forms of literature may be regulated by the province within which the publication, distribution or sale oceurs. These are matters within property and civil rights in the province."[147] As a result, much of the licensing of computer programs may be said to fall generally within the competence of the provinces although the intellectual property aspect remains within federal competence. The potential for conflict, as evidenced by the litigation arising from the Louisiana licensing statute, also exists in Canada.

Where such a conflict exists in a federal system, the doctrine of federal paramountcy is used to resolve the dispute. Under this principle, "where there are inconsistent (or conflicting) federal and provincial laws, it is the federal law which prevails. ... The doctrine of paramountcy applies where there is a federal law and a provincial law which are (1) each valid, and (2) inconsistent."[148] Therefore provincial legislation and common law principles impacting upon copyright may not enhance

---

[141] Rice, *supra* note 26 at 612-13 .

[142] 27B La. Rev. Stat. Ann. §§ 51: 1962ff (West 1987).

[143] The relevant portion of §117, that could potentially be upset by the state legislation, reads:

> Notwithstanding the provisions of section 106, it is not an infringement for the owner of a copy of a computer program to make or authorize the making of another copy or adaptation of that computer program provided: ...
>
> (2) that such new copy or adaptation is for archival purposes only and that all archival copies are destroyed in the event that continued possession of the computer program should cease to be rightful.

[144] Rice, *supra* note 26 at 612-13.

[145] (U.K.), 30 & 31 Vict., c. 3. Any residual powers, not specifically enumerated by the *Constitution Act, 1867* remain with the federal government unlike the case in the United States where they are left with the states (P.W. Hogg, *Constitutional Law of Canada*, 3d ed. (Toronto: Carswell, 1992) at 108).

[146] *Constitution Act, 1867, ibid.*, s. 92(13).

[147] Hogg, *supra* note 145 at 601.

[148] *Ibid.* at 418-19.

or restrict rights and privileges granted under federal copyright law. This argument exists independently of any argument made by virtue of the wording of section 63 of the *Copyright Act*, which expressly prohibits the granting of any rights similar to copyright other than under the *Act*.

c. *Statutory Paramountcy*

Arguably, the linchpin that most effectively secures the pre-emption of licensing provisions that conflict with federal legislation is section 301 of the American *Copyright Act*. It is interesting to note that this section was not directly used in the *Vault* decision. Section 301 states that the *Act* is the sole grantor of copyright; any similar rights granted by the common law or by state legislation are pre-empted by the *Act*.[149] By virtue of paragraph 301(b),[150] trade secret provisions continue to apply, but only insofar as they are consistent with the provisions of the *Act*.[151] According to one commentator,

> [t]he far-reaching public policy Section 301 implements clearly requires pre-emption of contract-based protection of expression *as* expression where the effect is to secure rights in that expression which are greater than, equal to, or supplemental of those which Section 106 secures. ... The inescapable conclusion is that contractual reverse engineering prohibitions cannot survive a Section 301(a) challenge.[152]

While section 301 of the American *Copyright Act* is markedly similar to section 63 of the Canadian *Copyright Act*, its construction is slightly different with respect to the pre-emption of trade secret laws. The American *Act* bases itself on the proposition that once copyright exists, nothing may exist that is inconsistent with it.[153] Section 63 of the Canadian *Copyright Act*, however, states that nothing in that *Act* "shall be construed as abrogating any right or jurisdiction to restrain a breach of trust or confidence." It might therefore be argued that trade secrets are paramount

---

[149] Where other laws compete with copyright by creating similar rights or restricting copyright, there are provisions in the Canadian, American and British copyright legislation which expressly state that no other law may create a copyright or similar rights (Canada: *Copyright Act*, *supra* note 1, s. 63; United States: *Copyright Act*, *supra* note 6, § 301; United Kingdom: *Copyright, Designs and Patents Act 1988*, *supra* note 88, s. 171).

[150] The text of the paragraph reads:

> Nothing in this title [§§101 *et seq.*] annuls or limits any rights or remedies under the common law or statutes of any State with respect to —
>
> (1) subject matter that does not come within the subject matter of copyright as specified by sections 102 and 103, including works of authorship not fixed in any tangible medium of expression; or ...
>
> (3) activities violating legal or equitable rights that are not equivalent to any of the exclusive rights within the general scope of copyright as specified by section 106.

[151] Rice, *supra* note 26 at 605-606.

[152] *Ibid.* at 614-16.

[153] See *supra* note 149.

under Canadian copyright law even where a conflict exists. If we apply this line of reasoning to reverse engineering, the determination that a protectable trade secret exists could lead a Canadian court to ignore the *Copyright Act* and its possible sanctioning of such an exercise.[154] Conversely, under the American *Act*, if the exercise of a trade secret claim were to conflict with the rights and privileges granted under the *Act*, the claim would fail.

While reverse engineering may be legally prohibitable under trade secret law, the case for claiming copyright paramountcy is much stronger where no trade secret exists. Unlike trade secret terms, simple contractual terms are not expressly exempted from copyright principles by section 63 of the *Copyright Act*. Furthermore, the application of section 63 is not restricted to other statutory instruments, and is presumably wide enough to encompass contractual rights as well. Accordingly, a public interest argument that stresses both paramountcy and copyright misuse is likely to prevail where non-trade-secret contractual terms attempt to bar rights that are otherwise allowed under the *Copyright Act*.

3. Competition Law

Even without the copyright misuse doctrine, or a statutory pre-emption of legislative or contractual terms in favour of copyright, it is arguable that an attempt to restrict reverse engineering through contractual terms may run afoul of Canadian competition law. Grants of intellectual property protections such as patents and copyrights are exempted from competition laws because of their statutory basis.[155] Any misuse of these rights that results in the lessening of market competition may, however, be prohibited by the courts. Section 32 of the *Competition Act*, deals with the misuse of intellectual property rights, stating:

> (1) In any case where use has been made of the exclusive rights and privileges conferred by ... a copyright ... so as to
> (a) limit unduly the facilities for ... producing, manufacturing, supplying ... or dealing in any article or commodity that may be a subject of trade or commerce,
> (b) restrain or injure, unduly, trade or commerce in relation to any such article or commodity,
> (c) prevent, limit, or lessen, unduly, the manufacture or production of any such article or commodity ... , or
> (d) prevent or lessen, unduly, competition in the ... sale ... or supply of any such article or commodity ...
> (2) The Federal Court, on an information exhibited by the Attorney General of Canada, may, ... make one or more of the following orders,
> (a) declaring void, in whole or in part, any agreement, arrangement or licence relating to that use;

---

[154] Subject to the arguments previously discussed (see text accompanying notes 120-21, above).

[155] Subsection 79(5) of the *Competition Act*, subject to section 32 of that *Act*, expressly exempts intellectual property rights obtained under the *Copyright Act* from being deemed "anti-competitive" and thereby subject to a prohibitive order by the Competition Tribunal.

> (b) restraining any person from carrying out or exercising any or all of the terms or provisions of the agreement, arrangement or licence;
> (c) directing the grant of licences under any such ... copyright ... to such persons and on such terms and conditions as the court may deem proper ... ; ...
> (e) directing that such other acts be done or omitted as the Court may deem necessary to prevent any such use.

This section gives the Federal Court broad powers to restrict the misuse of any of the intellectual property rights set out therein. If reverse engineering is indeed permitted as an exception under the copyright regime, then any licence term that would restrict reverse engineering would arguably be a use of the privileges of copyright to "unduly limit or lessen" the production of other computer programs. To date, there have yet to be any copyright disputes under the *Competition Act*, so the breadth of section 32 remains unclear.

In summary, although it is a common practice in the computer software industry to prohibit reverse engineering through licensing agreements, these prohibitions cannot be supported by trade secret laws where there is no fiduciary or similar relationship between the parties. If copyright law is found to be broad enough to allow reverse engineering, this type of licensing agreement provision will become unenforceable for a host of reasons, including copyright misuse, federal paramountcy, statutory paramountcy and competition laws. If a valid trade secret is found to exist, the argument to prohibit reverse engineering remains strong, threatened only by the potential paramountcy of competition law principles.

4. Patents

Patent law is another intellectual property regime that is increasingly important to computer program protection. Patents are monopoly rights granted by the federal government to inventors.[156] Obtaining a patent does not entitle one to specific sums of money or to any other positive act, but rather creates a right to exclude others from making, using or selling the invention for the duration of the patent. Patents are distinct from other forms of intellectual property protection because they are limited to functional products or processes that create a tangible product. Patents are granted not for the underlying ideas of these products or processes, but only for the physical manifestations of the ideas. Traditionally, it was thought that patents were not applicable to computer programs; however, in recent years the Canadian and American Patent Offices and the courts have allowed software patents for certain products.

To obtain a patent, the invention must possess novelty, utility and some measure of inventive step (also known as "non-obviousness").[157] Once a patent is

---

[156] Patents are governed in Canada by the *Patent Act*, R.S.C. 1985, c. P-4.

[157] The inventive step does not require that the invention be a revolutionary development: rather, it may improve upon already existing technology. The rule of thumb for this test is that the invention

granted, its object is protected in Canada for twenty years from the date of filing the patent application, after which the invention falls into the public domain.

Although it is widely believed that software patents are *prima facie* prohibited under patent legislation, both the Canadian and American Patent Offices grant patents for computer programs the operation of which results in a real world manifestation. That is, the computer program must be characterizable as "something more than a mere algorithm ... [and can neither be] merely directed to making calculations nor to the presentation of an algorithm and its solution."[158] This view was articulated by the Federal Court of Appeal in *Schlumberger Ltd.* v. *Canada (Patent Commissioner)*,[159] which now stands as the authoritative Canadian decision in the field of software patents. In that case, the Court held that the involvement of a computer could not render patentable that which was unpatentable, and since mathematical formulae are not patentable under the *Patent Act*,[160] they could not become patentable merely because they took the form of a computer program.[161] The Court did not, however, state that a computer program was, as a result of its form, unpatentable. Presumably, if a computer program embodied otherwise patentable subject matter, the program would be patentable. The Canadian Patent Office, rejecting its earlier blanket ban on computer program patents, seized upon this interpretation of the *Schlumberger* decision.

The debate with respect to reverse engineering in the realm of patents is virtually non-existent. Any use of a patented invention, within the scope of the patent claims filed, will require the payment of a royalty fee to the patent holder. If a patented invention is reverse engineered, any use of the material discovered through the reverse engineering process will still potentially be covered by the claims in the patent document.[162] Accordingly, royalties would still be payable to the patent holder if the use were to fall within the patent claims.

Many commentators continue to believe that the functionality of computer programs makes patent, and not copyright, law best equipped to provide intellectual property protection.[163] Providing copyright protection to computer programs over-

---

must elicit some reaction of marvel or amazement (*i.e.* "why didn't I think of that?") on the part of others in the industry. If the patent is issued for an improvement to an existing technology, any production of the new invention must be authorized by the patent holders of the existing technology. This permission is usually in the form of licensing agreements.

[158] R. Trudeau, "Software Patents" (1992) 9 C.I.P.R. 234 at 238-39.

[159] (1981), [1982] 1 F.C. 845, 56 C.P.R. (2d) 204 (C.A.) [hereinafter *Schlumberger*].

[160] No patent may be granted for mere scientific principles or abstract theorems (*Patent Act*, *supra* note 156, s. 27(3)).

[161] Sookman, *supra* note 48 at 6-21.

[162] The patent granting process requires full disclosure concerning the manufacture of the invention. Reverse engincering is therefore unnecessary since anyone may obtain a copy of this disclosure and the specifications contained therein.

[163] See Z. Hau, "Securing Patent Protection for Computer Program-Related Inventions" (March 1993) *Patent World* 34; S.A. Becker, "Drafting Patent Applications On Computer-Implemented Inventions" (1991) 4 Harv. J. L. & Tech. 237; Trudeau, *supra* note 158; Sookman, *supra* note 48 at 6-2.

protects them because it improperly treats them as literary works and not solely as functional works of technology.[164] The limitations of copyright to adequately reflect the policy goals associated with technological works have become clearer with the advent of computer programs, and have forced governments to disregard copyright when implementing protections for other technological works, such as semiconductor chips.

5. Semiconductor Chip Protection

In the past decade, both Canada and the United States have enacted *sui generis* legislation to protect semiconductor chips.[165] An examination of these statutes informs our discussion of copyright and reverse engineering as it provides some insight into how intellectual property legislation can be customized to address the needs of a particular technology. The recently enacted semiconductor legislation may be contrasted with more general catch-all intellectual property legislation, such as the copyright and patent statutes. Moreover, semiconductor legislation is also relevant because it contains specific reverse engineering provisions.[166]

The protection afforded by both the American *Semiconductor Chip Protection Act*[167] and the Canadian *Integrated Circuit Topography Act*[168] extends to the "mask work fixed in a semiconductor chip product", or "topography", as it is referred to under the Canadian *Act*. Mask works, or integrated circuit topographies, are generally defined as a series of images that represent in their totality a three-dimensional rendering of the chip product.[169] Each image represents a layer of a chip that is conceptually peeled

---

[164] Ignatin, *supra* note 17 at 2021.

[165] Semiconductor or microchips which contain computer programs, such as ROM chips, may receive two-tiered protection as the computer programs are further protected by copyright legislation (see *Apple*, *supra* note 8).

[166] The over-protection of software, as compared to semiconductor chips, is particularly relevant because of the great similarities between the two technologies, both in the way they are developed and the way they operate. ... [In fact,] the dividing line between hardware (such as semiconductor chips) and software is extremely fuzzy (Ignatin, *supra* note 17 at 2020 and at note 83).

[167] 17 U.S.C. §§ 901ff (1994) [hereinafter S.C.P.A.].

[168] S.C. 1990, c. 37 [hereinafter I.C.T.A.].

[169] Section 2 of the I.C.T.A. defines topography as:

> the design, however expressed, of the disposition of,
>
> (a) the interconnections, if any, and the elements for the making of an integrated circuit product, or
>
> (b) the elements, if any, and the interconnections for the making of a customization layer or layers to be added to an integrated circuit product in an intermediate form.

Section 901(a)(2) of the S.C.P.A. defines mask work as:

> a series of related images, however fixed or encoded —
>
> (A) having or representing the predetermined, three-dimensional pattern of metallic, insulating, or semiconductor material present or removed from the layers of a semiconductor chip product; and

away from the chip so as to be represented in two dimensions. These images are protected by the respective *Acts*.

Both *Acts* protect the topographies of original works for ten years.[170] In keeping with the American and Canadian *Copyright Acts*, the protection afforded by the S.C.P.A. and I.C.T.A. extends only to the expression of the topography and not to any underlying "idea, procedure, process, system [or] method of operation" embodied in the work.[171] The S.C.P.A. requires that the mask work be fixed in order to receive protection,[172] but the I.C.T.A. has no such requirement. Under both *Acts*, the required standard of originality falls somewhere between the standard of originality under the *Copyright Act*[173] and the standard of novelty under the *Patent Act*.[174] Originality for the purposes of semiconductor chip protection under the I.C.T.A. requires that in addition to the work not being a "mere reproduction of another topography," it must also be the "result of an intellectual effort and ... not ... commonplace among creators of topographies or manufacturers of integrated circuit products."[175]

The reverse engineering provisions, appearing in similar fashion in both *Acts*, distinguish these *Acts* from the copyright provisions that apply to computer programs. Paragraph 6(2)(a) of the I.C.T.A. allows a person, to undertake any act "in relation to that registered topography for the sole purpose of analysis or evaluation or of research," that does not "commercially exploit the topography or any substantial part thereof." Thus, any act that does not copy the whole topography or a substantial part thereof and that is commercially exploitive is allowed under the I.C.T.A. This level of similarity should be much more loosely construed than are similar enquiries as to copying under the *Copyright Act*. According to the proposal that resulted in the implementation of the I.C.T.A.,

> [t]he reverse-engineered chip and the protected chip might legitimately be identical in electronic function and external fit. But, reverse engineering would produce a chip with a three-dimensional layout neither identical nor virtually identical to the topography embodied in the protected chip. The proposed reverse-engineering measure would legitimate the creation of a substantially similar topography for a fully compatible chip, potentially offering for example:
>
> - an improved signal/noise ratio;
> - fewer fabrication steps;
> - greater thermal stability;
> - decreased die size;

(B) in which series the relation of the images to one another is that each image has the pattern of the surface of one form of the semiconductor chip product.

[170] S.C.P.A., *supra* note 167, § 904 (a) - (b); I.C.T.A., *supra* note 168, paras. 5 (a) - (b).

[171] S.C.P.A., *ibid.*, § 902(c); I.C.T.A., *ibid.*, s. 3(3).

[172] S.C.P.A., *ibid.*, § 902(a)(1).

[173] See Part **II.A.** 1.c, above, for a discussion of this issue.

[174] See text accompanying note 157, above.

[175] I.C.T.A., *supra* note 168, s. 4(2). Similar protection is afforded by S.P.C.A., *supra* note 167, § 902(b).

- faster performance; and
- lower manufacturing cost.[176]

Infringement of the rule would occur only where the reverse engineering resulted in the "production and marketing: of an exact copy of a protected topography; or of a chip embodying a topography virtually identical to the protected topography or to a substantial portion thereof."[177] A *sui generis* regime was favoured over simply extending copyright protection to semiconductor chips because the underlying policy recognized that this genre of "legislation must ... suppress chip piracy without creating unnecessary obstacles to a free market in semiconductor chips and to the spread of chip technology."[178] The use of

> pure copyright principles to prevent the unauthorized copying of a chip topography would not meet the needs of the semiconductor industry ... who wish [*sic*] to make an unauthorized copy of all of their competitor's topography for analysis; and to manufacture a suhstantially similar chip derived from their competitor's topography.[179]

The test for valid reverse engineering under both the I.C.T.A. and the S.C.P.A. requires a "two-step inquiry":

> First, if it is determined that a competitor has substantially studied and analyzed a protected mask work to produce its own chip, i.e. valid reverse engineering, that chip does not infringe even if it is substantially similar to the mask owner's. However, if the competitor's design incorporates identical parts of the protected design, infringement may yet be found.[180]

As with the S.C.P.A., the drafters of the I.C.T.A. expected that the work involved in reverse engineering a chip would be substantial and would justify reverse engineering as an alternative to continually "re-inventing the wheel".[181] Both the Canadian and American *Acts* rely on the concept of a "paper trail" as proof that analysis has been performed on a chip, which is an activity allowed as a fair use under subsection 6(2) of the I.C.T.A.[182] Proof that "sweat of the brow" has been expended may involve the use

---

[176] A.Z. Hertz, *Semiconductor Chip Protection in Canada: Proposals for Legislation* (Ottawa: Department of Consumer and Corporate Affairs, 1987) at 46. This report specifically recommended, *ibid.* at 49: "Canada's chip-protection law should contain a reverse-engineering exception allowing the unauthorized copying of a protected topography in a process of analysis and redesign leading to the creation of a substantially similar chip topography."

[177] *Ibid.* at 46.

[178] *Ibid.* at 45.

[179] *Ibid.* at 43-44.

[180] S.P. Kasch, "The Semiconductor Chip Proteetion Act: Past, Present and Future" (1992) 7 High Tech. L.J. 72 at 77.

[181] It is estimated that reverse engineering a semiconductor chip "requires thousands of person-hours and about a quarter of the money needed to create the original chip" (Hertz, *supra* note 175 at 47).

[182] The end product of the reverse engineering process is not an infringement, and itself qualifies for protection under the [S.C.P.A.], if it is an original mask work, as contrasted with a substantial copy. If the resulting semiconductor chip product is not sub-

of "ordinary business documents and technical materials [such as] invoices, employment and payroll records, logic and circuit diagrams, trial layouts and computer simulations of the chip."[183]

D. Conclusion

Without further legislative guidance as to the extent of the fair dealing exception to copyright infringement, copyright law prohibits the reverse engineering of computer programs. Aside from copyright, no other intellectual property regime disallows the reverse engineering of computer programs in a blanket fashion. In fact, semiconductor chip and patent legislation either expressly allow reverse engineering or provide adequate disclosure of the ideas and processes underlying the protected work. Trade secret laws, although prohibiting reverse engineering by those who have a "special relationship" with the work's owner, allow persons not in such a relationship to reverse engineer a computer program. If it is determined that the reverse engineering of computer programs falls within the exception of fair dealing, attempts to enhance protection, other than through trade secret laws, may be considered anti-competitive and may result in the suspension of copyright enforcement privileges.

III. Existing Approaches to the Reverse Engineering Problem

Although the legal problems associated with reverse engineering have not yet appeared in a Canadian context, they have received attention in the United States and in the European Union (E.U.). The American experience indicates that court challenges based on reverse engineering claims are imminent in Canada and that a legislative response should precede such court challenges, thus avoiding the uncertainty of a judicial decision based on statutory language that was not enacted in contemplation of computer technology. The European Community's legislative response to reverse engineering serves as a good example of a potentially workable regulatory scheme.

---

stantially identical to the original, and its design involved significant toil and investment so that it is not a mere plagiarism, it does not infringe the original chip, even if the layout of the two chips is, in substantial part, similar. ... [T]he courts are not likely, as a practical matter, to find it unduly difficult to draw the line between reverse engineering and infringement, because the additional work required to come within the privilege established by s. 906(a) will ordinarily leave a "paper trail" (United States House and Senate Explanatory Memorandum, 130 Cong. Rec. 28,960 (daily ed. 3 October 1984) ("Explanatory Memorandum — Mathias-Leahy Amendment to s. 1201").

[183] Hertz, *supra* note 176 at 47. The idea is that a *bona fide* reverse engineer will leave a trail as compared with a pirate who "is not able to produce a long paper trail because he has not done genuine design-development work."

A. *United States*

The issue of reverse engineering computer programs under the American *Copyright Act* was recently considered in two United States Court of Appeals cases which mark the first time any court has dealt with the issue under copyright laws.[184] The judgments in these cases were released a mere month apart, and together present a consistent, albeit surprising, policy-oriented approach to reverse engineering. Prior to the release of these cases the United States District Court examined the issue with respect to the reverse engineering of data tables.[185] Although the more recent appellate cases present more sophisticated analyses of the copyright issues involved, the lower court case applies a rudimentary approach to reverse enginecring that, in the opinion of this author, arrives at an appropriate solution.[186] The reasoning in this case reflects an elegant simplicity which speaks well to the issue of reverse engineering.

1. *E.F. Johnson Co.* v. *Uniden Corp. of America*

In 1985, the United States District Court addressed a claim involving the reverse engineering of data tables contained in a microchip, and its permissibility under the *Copyright Act*. In *Uniden*, the plaintiff moved for a preliminary injunction restraining the defendant from publishing, selling, marketing or in any way distributing software that would allow users to access a range of frequencies in the defendant's two-way radio product. The radios in question were controlled by software provided to users by the plaintiff along with the radios. The defendants reverse engineered the plaintiff's computer program, uncovering data tables that listed activation codes for the radios, and then produced their own program using identical data tables. The defendants' tables, however, also contained unnecessary elements, such as errors and duplications made in the plaintiff's table. The Court granted an injunction based on the substantial likelihood of success of the plaintiff's copyright claim on its merits. It was decided that alternative means to copying the plaintiff's software were available, by which the defendant could have placed the required radio codes into their own program. It was further held that the defendants had acted unfairly and were guilty of copying copyrighted expression.

The Court did not discuss whether the reverse engineering activities of the defendant were infringements of copyright. The Court chose rather to look at the final product and examine it for traces of infringing expression. It remains unclear whether the Court's silence indicates conscious implicit acceptance of reverse engineering as a permissible act under copyright legislation. It is more likely that, because this was an early case involving computer programs under American copyright law, the *Uniden* Court simply missed the issue. One could, however, discern acceptance of reverse engineering. This interpretation is supported by several pas-

---

[184] *Sega, supra* note 4; *Atari, supra* note 4.

[185] *Uniden, supra* note 4.

[186] See Part IV.C, below, for recommendations designed to deal with the problem of reverse engineering.

sages in the judgment stating that there would have been no infringement had the defendants only used functional parts of the data tables which could not be produced through some alternate means.[187] The Court's decision presupposes that the defendants were permitted to uncover these codes, which could only be done through reverse engineering the chip containing the data tables. As no statement was made to the contrary, presumably the intermediate copies of the table produced at this stage would not constitute copyright infringements.

2. *Atari Games Corp.* v. *Nintendo of America Inc.*

The first of the two decisions to be released concerning the reverse engineering of computer programs was that of the Federal Circuit Court of Appeals in *Atari Games Corp.* v. *Nintendo of America Inc.* In that case, Nintendo sued Atari for copyright infringement of its 10NES lock-out program.[188] 10NES is a program embedded in chips contained in the popular Nintendo NES game console, which intercepts the flow of information from game cartridges.[189] Nintendo allowed other software developers to become licensees and write software for the NES. The licence agreements were highly restrictive, but allowed the licensees to produce software which would be bundled with Nintendo's secret unlocking message.[190]

In 1986, Atari had attempted to crack the 10NES program by both monitoring the data flow between the console and the cartridges, and by chemically peeling microchip layers from the 10NES chips and microscopically examining the data paths.[191] Unable to crack the 10NES program, Atari decided to become a Nintendo licensee in 1987, and was barred under the terms of that agreement from gaining access to the 10NES code and from uncovering its operation. In 1988, a lawyer representing Atari approached the American Copyright Office and applied for a copy of the 10NES program, which had been filed for copyright registration purposes. The lawyer stated that the code was required for litigation that had been commenced against Atari. The code was provided under the condition that "the requested copy [would] be used only

---

[187] *Uniden, supra* note 4 at 1504.

[188] Nintendo had originally sued Atari for unfair competition, patent infringement, copyright infringement and trade secret violations. Atari countersued for unfair competition, *Sherman Act* violations, and patent infringement. The cases were consolidated and Nintendo obtained a preliminary injunction from the District Court for the Northern District of California based on Atari's unauthorized use of Nintendo's copyrighted expression. Atari appealed this injunction to the United States Court of Appeals, Federal Circuit.

[189] 10NES waits for the cartridge to send a coded message to the console. The coded message is contained in a microchip and is referred to as the 10NES "slave" chip. The portion of the 10NES program contained in the console is also in microchip form and is referred to as the 10NES master chip. Once the coded message is detected, 10NES allows the console to operate the cartridge. If the appropriate message is not detected, 10NES will not "unlock" the console, and will not allow the cartridge to operate.

[190] In fact, under the licensing agreements, software developers would provide Nintendo with the computer programs. Nintendo would then package the games into cartridge form with the 10NES slave chip and resell them to the software developer.

[191] *Atari, supra* note 4 at 836.

in connection with the specified litigation."[192] No litigation actually existed between the parties at that time.

Atari used the code provided by the Copyright Office to develop its own unlocking program entitled "Rabbit". Rabbit was a program that, although written in a different computer language, replicated the data stream provided by the 10NES slave chips. Nintendo sued Atari, and was granted a preliminary injunction for copyright infringement.

Atari's defence to the copyright infringement claim was primarily based on the fair use exception.[193] Section 107 of the American *Copyright Act* sets out four factors to consider in a fair use claim:

> 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.

The *Atari* Court did not perform a detailed analysis of these factors in its decision. Instead, the Court looked to copyright to balance the monetary self-interest of individual authors with society's interest in promoting the free flow of ideas. With respect to reverse engineering, the Court found irrelevant the fact that Nintendo's code was in object code form and stored on a microchip. It held that it is a fair use to reverse engineer a computer program in order to gain a better understanding of the program's underlying ideas:

> An author cannot acquire patent-like protection by putting an idea, process, or method of operation in an unintelligible format and asserting copyright infringement against those who try to understand that idea, process, or method of operation. ... The Copyright Act permits an individual in rightful possession of a copy of a work to undertake necessary efforts to understand the work's ideas, processes, and methods of operation.
>
> This permission appears in the fair use exception to copyright exclusivity.[194]

---

[192] *Ibid.*

[193] Fair use, like its Canadian fair dealing counterpart, is an exception to copyright infringement, set out in the American *Copyright Act* at § 107:

> [T]he fair use of a copyrighted work, including such use by reproduction in copies or phonorecords or by any other means specified by [§§ 106, 106A], for purposes such as criticism, comment, news reporting, teaching (including multiple copies for classroom use), scholarship, or research, is not an infringement of copyright.

[194] *Atari, supra* note 4 at 842.

Fair use protects the intermediate stages of reverse engineering, along with the resulting related copies from being deemed infringing. According to the Court, fair use does not extend to reverse engineering undertaken for the purpose of profiting by replicating protected expression, but rather only applies where reverse engineering is necessary to discern unprotectable ideas.

In the case of Nintendo's 10NES program, the Court determined that the underlying unprotectable ideas were the codes required to unlock the NES console. Anything in excess of that was presumed to consist of protectable expression.[195] Atari's copy of various parts of the 10NES program included errors and deletions made by Nintendo in their program. This copying of unnecessary portions of the Nintendo table resulted in a decision against Atari for copyright infringement since, as the Court stated, "these unnecessary instructions strongly suggest that the Rabbit program is substantially similar to the 10NES program ... [which] Nintendo is likely to show ... contains protectable expression."[196] Furthermore, Atari's unfair appropriation of the 10NES source code by misrepresenting its intentions to the Copyright Office pre-empted any use of the fair use defence which only applies to authorized copies of a work.[197] Atari's dishonest behaviour in obtaining the 10NES code also precluded Atari from using a copyright misuse defence based on Nintendo's extensive licensing agreement.[198] The Court failed to recognize the possibility that Atari copied elements unnecessary to unlock the Nintendo console in order to ensure against "the possibility that Nintendo could alter its console in the future to utilize currently unused portions of the compatibility code."[199]

Atari lost its appeal to lift the injunction imposed by the District Court, and the Court of Appeals also upheld the District Court's substantive ruling concerning the fair use exception.

3. *Sega Enterprises Ltd.* v. *Accolade Inc.*

Immediately following the Federal Circuit Court of Appeals' decision in *Atari*, the Ninth Circuit Court of Appeals released its decision in *Sega*. In that case, Accolade had sought to reverse engineer cartridges containing video game programs with the intention of uncovering how the software interacted with Sega's Genesis game consoles. Using the results of the reverse engineering, Accolade created a manual for use by its programmers in developing games that would operate on the Sega console. The information used by Accolade at this early stage was purely functional and did not involve any literal copying of Sega program codes. During

---

[195] *Ibid.* at 843-44.

[196] *Ibid.* at 845.

[197] *Ibid.* at 843; *Harper & Row Publishers, Inc.* v. *Nation Enterprises*, 471 U.S. 539 at 562-63, 85 L. Ed. (2d) 588 (1985).

[198] *Atari, ibid.* at 846. See also text accompanying note 130, above.

[199] H.C. Moore, "Atari v. Nintendo: Super Mario Uses 'Expressive' Security Feature to 'Lock' Out the Competition" (1992) 18 Rutgers Comp. & Tech. L.J. 919 at 933. This possibility manifested itself in *Sega*.

the period that Accolade was beginning to release its own compatible software, Sega, in order to combat high degrees of piracy of its games, decided to license a copy protection system for use in its newly developed Genesis III consoles. The trademark security system (T.M.S.S.) would be placed in each Sega console and, when a cartridge was plugged in, would search for the letters "SEGA" contained in the program code of each cartridge. Upon finding this initialization code, the Genesis III console would produce a display of "PRODUCED BY OR UNDER LICENCE FROM SEGA ENTERPRISES LTD.", and would unlock the console for game play, much like Nintendo's 10NES system.

Upon finding that its game cartridges would not work with the Genesis III system, Accolade examined the results of its reverse engineering efforts and found that each licensed Sega cartridge contained a small unused portion of code in each game's "power-up" sequence, which was now being used by Sega in its T.M.S.S. lockout system.[200] In response, Accolade created a small header file (twenty to twenty-five bytes of data) containing the SEGA initialization code, to be used in all future game development. This information was also incorporated into its development manual.

Sega brought an action against Accolade for copyright infringement, trademark infringement and false designation of origin. Accolade countered with claims of false designation of origin, unfair competition, false or misleading statements and intentional interference with prospective economic advantage. Sega was granted a preliminary injunction by the United States District Court based on Sega's copyright and trademark infringement claims. Accolade appealed to the United States Court of Appeals (Ninth Circuit), whereupon the injunction was lifted and the matter was remanded to the District Court.

The decision of the Court of Appeals to lift the preliminary injunction and allow reverse engineering based on the fair use exception provides a detailed analysis of the reverse engineering issue as it applies to computer programs. The *Sega* Court enunciated a very useful step-by-step analysis of the four fair use factors, and discussed three related reverse engineering arguments: intermediate copying; the idea/expression dichotomy as a possible justification for reverse engineering; and a section 117 defence. Although Accolade ultimately succeeded in its claim that the reverse engineering at issue constituted a fair use, the Court rejected Accolade's other three arguments.

With respect to intermediate copying, the Court held that the fact that the prod-

---

[200] The use of apparently useless code in an upgrade of the system highlights a difficulty inherent in the idea/expression analysis performed by courts in order to determine what may fairly be reverse engineered and used in a competing program. If computer code, and/or errors, serve no purpose, then, as was the case in the *Atari* and *Uniden* decisions, they may not be legally copied. However, the original program developers may subsequently use these pieces of computer code in a lockout system which would render all other manufacturers' products useless. This possibility was ignored by the Court in the *Atari* case (see *supra* note 198 and accompanying text).

uct is intermediate, as opposed to final in nature in no way exempts it from the *Copyright Act*. Whether intermediate copying properly constitutes an infringement of copyright must be examined, notwithstanding whether the final product itself constitutes an infringement.[201] According to the Court's reasoning,

> the computer file generated by the disassembly program, the printouts of the disassembled code, and the computer files containing Accolade's modifications of the code that were generated during the reverse engineering process all ... [fall] squarely within the category of acts that are prohibited by the statute.[202]

Concerning Accolade's claim that the idea/expression dichotomy provides a justification for reverse engineering, the Court ruled that the fact that ideas may be contained within protected expression does not mean that the expression accompanying those ideas may be ignored. Accolade had argued that the nature of computer programs distinguishes them from other, more traditional copyrightable works, the ideas of which may readily be perceived and understood by human beings. Consequently, reverse engineering was perceived as simply a crutch to put computer programs on an equal footing with these other works.

The final argument disposed of by the Court in Sega's favour concerned Accolade's claim that under section 117 of the *Copyright Act* the production of intermediate copies does not constitute a copyright infringement.[203] The Court reviewed the C.O.N.T.U. report responsible for the implementation of section 117, and ruled:

> Accolade's use went far beyond that contemplated by CONTU and authorized by section 117. Section 117 does not purport to protect a user who disassembles object code, converts it from assembly into source code, and makes printouts and photocopies of the refined source code version.[204]

The Court did, however, accept Accolade's fair use argument:

> Because, in the case before us, disassembly is the only means of gaining access to those unprotected aspects of the program, and because Accolade has a legitimate interest in gaining such access (in order to determine how to make its cartridges compatible with the Genesis console), we agree with Accolade. Where there is good reason for studying or examining the unprotected aspects of a copyrighted computer program, disassembly for purposes of such study or examination constitutes a fair use.[205]

Of the four fair use factors mentioned in section 107 of the American *Copyright Act*,[206] the Court found in favour of Accolade on the first, second and fourth factors, and in favour of Sega on the third factor.

---

[201] *Sega*, *supra* note 4 at 1518.

[202] *Ibid.*

[203] See *supra* note 83 for the text of section 117.

[204] *Ibid.* at 1520.

[205] *Ibid.*

[206] For a list of the factors, see text following note 193, above.

The first factor concerns the purpose and character of the use, including whether such use is commercial in nature or for nonprofit educational purposes. The Court ruled that "[t]he commercial nature of a use is a matter of degree, not an absolute," and in the case of Accolade, the direct purpose of reverse engineering the Sega code was to ensure the compatibility of their cartridges with the Sega console.[207] The commercial nature of cartridges was secondary and, at best, indirect. In considering this factor, the Court also seemed to hint at a "sweat of the brow" rationale combined with some required measure of creativity.[208] The Court held that "there is no evidence in the record that Accolade sought to avoid performing its own creative work ... [as] it wrote its own procedures. ... [T]hese facts indicate that ... its direct use of the copyrighted material ... was simply to study the functional requirements for Genesis compatibility."[209] In support, the Court also noted that the public interest in promoting compatibility and competition reduced the strength of Sega's claim that Accolade stood to gain commercially.

With respect to the fourth factor listed in section 107, the effect of the use upon the potential market for or value of the copyrighted work, the Court held that Accolade's development will not directly cause Sega to lose customers because the games developed by Accolade are not copies of those developed by Sega.[210] Although the Court recognized that Accolade's entrance into the market will "undoubtedly affect" the sale of Genesis games, the public policy rationale underlying copyright legislation is to promote creative expression and not to stifle competition in a specific market.

The final factor decided in Accolade's favour dealt with the nature of the copyrighted work. The basis of the Court's analysis here was that computer programs are essentially utilitarian in nature and contain many elements the expression of which is often dictated by notions of efficiency, or external factors such as compatibility.[211] These elements are not protectable under copyright legislation. Accordingly, if Accolade were unsuccessful in its fair use claim Sega would unrightfully

---

[207] *Sega, supra* note 4 at 1522.

[208] The application of "sweat of the brow" to copyright protection has recently gained much notoriety as a result of the United States Supreme Court's decision in *Feist Publications, Inc.* v. *Rural Telephone Service Company*, 499 U.S. 340, 113 L. Ed. (2d) 358 (1991). This case raised the issue of whether or not copyright could exist in a telephone book. The Court held that mere "sweat of the brow" was not sufficient in itself to attract copyright protection. Creative effort, including particular selection or arrangement, applied to the data was also necessary for copyright to exist. Although the context was decidedly different in *Sega*, the interplay between "sweat of the brow" and originality/creativity is gradually being redefined with respect to utilitarian works such as databases and computer programs

[209] *Sega, supra* note 4 at 1522.

[210] The *Sega* Court's decision seems to indicate that in order to successfully argue this factor as a bar to fair use, it must be shown that "the new work ... supplant[s] the direct market for the particular copied work" (D.L. Hayes, "The Legality of Disassembly of Computer Programs" (1993) 12 Comp. L.J. 1 at 8).

[211] *Sega, supra* note 4 at 1524.

obtain a monopoly over the functional ideas underlying its work. Sega argued that there were methods other than disassembly by which its initialization code could be uncovered. The Court disagreed and found that "disassembly of the object code in Sega's video game cartridges was necessary in order to understand the functional requirements for Genesis compatibility."[212] In effect, the Court recognized that while Accolade's disassembly necessarily involved potentially infringing intermediate copying, to allow Sega to succeed on this claim would expand copyright beyond its intended scope and stifle competition. The Court therefore chose the lesser of two evils and, based on public policy reasons, found that Accolade's actions constituted a fair use of Sega's copyrighted material. The Court did not clarify whether this balance would have tipped in Sega's favour had methods other than disassembly been available to Accolade.

The Court found in favour of Sega, however, with respect to the third factor mentioned in section 107, the amount and substantiality of the portion used in relation to the copyrighted work as a whole. Accolade disassembled entire programs written by Sega in order to uncover the initialization sequence. The *Sega* Court, however, citing the decision in *Sony*, ruled that "the fact that an entire work was copied does not ... preclude a finding of fair use."[213] The Court here stated that the fact that the entire work was copied was "of very little weight".[214] The Court's ruling on this point runs counter to that of the Canadian Exchequer Court in *Zamacoïs*, which held that the fair dealing defence under the Canadian *Copyright Act* cannot be invoked where an entire work is copied.[215] *Zamacoïs* was decided in an era when copyright did not contemplate the protection of computer programs the functional nature of which may dictate a revision of existing copyright principles. This "special functional nature" of computer programs was, however, clearly recognized by the *Sega* Court in its analysis of the idea/expression dichotomy.

B. The European Union

The area of intellectual property, and more specifically copyright, is not addressed in the constitutive treaties of the European Union, and it has proved difficult to harmonize. According to one commentator, "[t]here will not be a unified European copyright law in the near future. The national copyright laws of member states will continue to apply."[216] The European Commission has, however, recog-

---

[212] *Ibid.*

[213] *Ibid.* at 1526. See also *Sony*, *supra* note 92 at 449-50; *Hustler Magazine, Inc.* v. *Moral Majority, Inc.*, 796 F.2d 1148 at 1155 (9th Cir. 1986): "[T]he copying of an entire work does not preclude fair use *per se*."

[214] *Sega*, *ibid.* at 1527.

[215] See *supra* note 94 and accompanying text.

[216] S. von Lewinski, "Copyright in the European Communities: The Proposed Harmonization Measures" (1992) 18 Brooklyn J. Int'l Law 703. In 1990 the European Commission adopted a proposal that would require all member states to accede to various international copyright and related rights treaties including: the *Berne Convention for the Protection of Literary and Artistic Works* (as amended), 30 June 1972, 828 U.N.T.S. 221; *International Convention for the Protection of Perform-*

nized that specific industries, including the software industry, cannot wait for a general harmonized European copyright law. Due to their relative economic importance and to the disparity of treatment among various member-states, these industries require that common legal rules be applicable throughout member-states without delay.

1. The E.E.C. *Directive* on Computer Programs

Accordingly, in order to harmonize existing protections and establish common principles within the Community with respect to the intellectual property of computer programs, in 1991 the European Council passed the *Directive on the Legal Protection of Computer Programs*. The objectives of the *Directive* were two-fold:

> [T]o prevent the unlawful copying of computer software, or "computer piracy", within the Community by ensuring an adequate level of protection for those who create computer software; and
>
> to promote the free circulation of computer software within the Community and allow industry to take advantage of the single market by harmonising the national laws of the Member States relating to the use and reproduction of computer software.[217]

The European Commission, charged with designing the *Directive*, opted for copyright as the basis for protection because of its interpretative flexibility, its ability to balance monopoly rights with societal interests, its limited scope of protection and most importantly because, to date, copyright had been the intellectual property protection of choice of a number of Community member-states.[218] The Commission's choice of copyright was hardly revolutionary or controversial. However, the decompilation exception to infringement was hotly debated prior to its inclusion in the *Directive*.[219]

Early in its conception, the Commission's Proposal for the *Directive* (the 1989 Proposal) contained two sections which could be interpreted as allowing the reverse engineering of protected computer programs. Article 1.3 of the 1989 Proposal stated that only a computer program's expression would be afforded legal protection, and not

> the ideas, principles, logic, algorithms or programming languages underlying the program. Where the specification of interfaces constitutes ideas and principles which underlie the program, those ideas and principles are not copyrightable subject matter.[220]

---

ers, Producers of Phonograms and Broadcasting Organizations, 26 October 1961, 496 U.N.T.S. 43.

[217] Clifford Chance, *The European Software Directive* (London: Clifford Chance, 1991) at 5.

[218] *Ibid.* at 5-6.

[219] The *Directive* employs the term "decompilation" in its broad sense as equivalent to the term "reverse engineering", as opposed to its literal meaning. "Decompilation" as used in the *Directive* therefore includes disassembly as well (see *supra* notes 18-19 and accompanying text).

[220] EC, *Proposal for a Council Directive on the Legal Protection of Computer Programs*, O.J. In-

Article 5.1 of the Proposal allowed the reproduction and adaptation of a computer program, potentially including the reverse engineering thereof, where such manipulations were necessary for the use of the computer program. This adaptation right would exist notwithstanding the copyright-holder's refusal. Although these sections would implicitly allow reverse engineering, the 1989 Proposal did not contain express wording to that effect. The European Parliament accepted the Commission's 1989 Proposal, subject to certain amendments. The Commission responded with its amended proposal in 1990, which contained express wording that would allow reverse engineering in certain circumstances. Article 5bis of the amended proposal was eventually adopted almost verbatim into the *Directive* (as Article 6), and is unique in that it expressly allows reverse engineering in the context of copyright protection of computer programs.[221]

Subsection 6(1) states that decompilation may be used where it is "indispensable to obtain the information necessary to achieve the interoperability of an independently created computer program with other programs." It goes on to list three conditions which limit the scope of the decompilation, all of which must be met in order for the copyright exemption to apply. First, individuals decompiling software must be licensees or titularies of some other right to use the software; second, the information being sought through decompilation must not be readily available; and third, the decompilation must be carried out only on those parts of the software necessary to access the information being sought.

In order to further clarify and strengthen the meaning of subsection 6(1), subsection 6(2) emphasizes those purposes which do not justify the decompilation of a computer program. Any information retrieved through the use of the decompiling exception cannot be used to achieve something other than interoperability, or to undertake any other task that violates copyright, such as developing software similar in expression.[222] Clearly, in its request that the Commission include an express re-

---

formation (1989) No 89/C91/05 at 13. See J.D. Byrne, "Computer Programs and Reverse Engineering: Recent European Developments" (1991) 8 Can. Computer L.R. 45 at 47.

[221] The Commission's 1990 proposal stated as background:

> [T]he Commission has been persuaded that the original proposal, which left the matter of "reverse engineering" not explicitly regulated, lacks sufficient clarity. It is therefore proposed that an additional Article 5bis dealing with a derogation allowing "reverse engineering" of programs for the purpose of interoperability of the program should be added. Nothing in this *Directive* should prevent however the "reverse engineering" of a program, whether incorporated into hardware or not, under the conditions of Article 5bis for the purpose of independently creating an interoperable program, wherever it may be incorporated (Byrne, *ibid.* at 48).

[222] The enacted *Directive* contains a definition of interoperability in its preamble:

> Whereas the parts of the program which provide for such interconnection and interaction between elements of software and hardware are generally known as "interfaces";
>
> Whereas this functional interconneetion and interaction is generally known as "interoperability"; whereas such interoperability can be defined as the ability to exchange information and mutually to use the information which has been exchanged (*Directive, supra* note 3 at 43).

verse engineering section, the European Council felt it necessary to make its intentions, as expressed in articles 1.3 and 5 of the Commission's 1989 Proposal, as unambiguous as possible in light of the Council's revolutionary and controversial stance. The Council wished to ensure that any decompilation would be both limited in scope and subject to strong barriers to abuse.

2. Implementation of the *Directive*'s Provisions in the United Kingdom

The provisions of directives are not enforceable until they are implemented by national legislation; however, directives are binding, and must be adopted by member-states within the prescribed time.[223] To implement directives, member-states are not required to adopt the exact wording used in the directive, although they must formulate legislation with the same effect. The deadline for implementing the software *Directive* was January 1, 1993, which was met by only a handful of member-states, including Denmark, Italy and the United Kingdom.[224]

The required changes to British legislation were implemented by the *Copyright (Computer Programs) Regulations 1992*, which came into force on January 1, 1993.[225] The *Directive*'s reverse engineering provisions have been incorporated into section 50B of the British *Copyright, Designs, and Patent Act 1988*, but they do not in any way supplant the *Act*'s fair dealing provisions, which may also potentially be used to permit reverse engineering.[226] Section 50B allows the reverse engineering of a computer program in order:

Critics of the *Directive* were quick to point out that the original proposals failed to adequately define the term "interoperability". One interest group that had pushed hard on this point was the Software Action Group for Europe (S.A.G.E.). S.A.G.E. vehemently argued that without an adequate definition of "interoperability", decompilation could be used for purposes other than those intended by the *Directive*. That is, without the establishment of strict definitions of the level at which a program may validly interface with another, software developers could decompile in a manner that violates the author's protected expression.

On the other hand, the European Committee for Interoperable Systems (E.C.I.S.), and other similar groups support the *Directive*'s decompilation provisions and favour a more standardized computer environment. Although all groups, including S.A.G.E. and other decompilation opponents, favour standardization in the computer industry, the two sides prioritize their concerns differently. Groups like S.A.G.E. feel that on balance, there is more to lose than there is to gain through allowing decompilation in this fashion, while groups like the E.C.I.S. charge that although their opponents' concerns are not altogether without foundation, groups like S.A.G.E. are overly alarmist and unrealistic.

[223] Chance, *supra* note 217 at 9.

[224] "Software Directive" (1993) 9 Clifford Chance Computer & Communications Bulletin 3.

[225] Bainbridge, *supra* note 18 at 591.

[226] It has been pointed out that the existence of an arguably broader fair dealing exception makes a separate reverse engineering provision unnecessary. Fair dealing may, in fact, allow reverse engineering for a host of reasons not contemplated by the *Directive*, such as commercial research (*ibid.* at 595-96).

> (a) to convert it into a version expressed in a higher level language, or
> (b) incidentally, in the course of so converting the program, to copy it.

Subsection 50B(2) restricts the use of reverse engineered material, as required by subsection 6(2) of the *Directive*, to situations where:

> (a) it is necessary to decompile the program to obtain the information necessary to create an independent program which can be operated with the program decompiled or with another program ('the permitted objective'); and
> (b) the information so obtained is not used for any purpose other than the permitted objective.

The United Kingdom's implementation of the *Directive* has been criticized for only exempting the reverse engineering of lower level language code into a higher language, whereas the "*Directive* is more generous, allowing translation, adaptation, arrangement or alteration."[227] Another potential troublespot in the British legislation is paragraph 50B(2)(b), which potentially restricts a software developer who has reverse engineered a computer program to create a compatible program, from later deciding to use the information retrieved to create another, potentially competing program.[228] The use of the "permitted objective" mechanism seems to dictate a sense of immediacy —, such that the reverse engineering must be conducted with a given objective in mind, and the results only used in conjunction with that objective. A strict interpretation of the *Act*'s wording may require the developer to rework the reverse engineering should he or she want to create other compatible computer programs at some later date. Another notable point, especially for North American observers, is that under regulation 12(2) of the British *Act*, the ability to reverse engineer a computer program cannot be pre-empted by any term in a licensing agreement; any attempt to do so would be void.[229]

Notwithstanding various technical differences, the British *Act* has captured the general spirit of the *Directive*'s reverse engineering exception, although arguably its choice of wording serves to indicate a certain discomfort with using legislative exceptions to relax copyright restrictions. The British legislature seems to prefer to defer the task of fine tuning copyright protection to the courts through the more general fair dealing and public interest exceptions.[230]

---

[227] *Ibid.* at 594. Implicitly, the British *Act* seems to prohibit the translation of a work to an equivalent level language such as from binary code to hexadecimal notation.

[228] *Ibid.* at 595.

[229] *Ibid.* This applies only to licensing agreements executed after the implementation of the provisions on January 1, 1993. This differs from the right found in section 50C of the *Act* to copy or adapt a program as required for its use. The legislation makes no mention of the inability to contract out of this provision.

[230] Whether this discomfort is peculiar to Anglo-American systems of copyright, as opposed to continental *droit d'auteur* regimes, is an interesting question which unfortunately lies beyond the scope of this article.

C. *Australia*

If Canada were to look around the world for an analogous jurisdiction in terms of facing computer copyright issues, Australia would be the natural choice. Since 1984, Australian copyright law has expressly placed computer programs within the scope of copyright protection, but has left the proposition rather vague as did the 1988 Canadian amendments.[231] In terms of jurisprudential evolution, Canada and Australia are also similarly situated.[232] With respect to reverse engineering, however, Australian courts are a step ahead of their Canadian counterparts.

1. *Autodesk Inc.* v. *Dyason*

In 1992, the Australian High Court rendered its decision in *Autodesk*, a case that concerned the reverse engineering of data tables found within a computer program. Once again, the subject matter of the reverse engineering process concerned a security lockout scheme designed to protect against software piracy. The difference between *Autodesk* and the abovementioned American cases, however, is the manner in which the reverse engineering was conducted. In this case, the defendants used an oscilloscope to map out the signals being sent between a hardware key, referred to as the "AutoCAD lock", and the computer program, known as "Widget C". In order to run the AutoCAD program, the AutoCAD lock must be introduced into a computer port. When the AutoCAD program is executed, it runs the Widget C program, which generates a stream of computer signals based on a data look-up table. These signals are sent to the AutoCAD lock, manipulated, and then sent back. Only where the returning signals match the profile contained in Widget C will the AutoCAD program be permitted to proceed. The defendants used an oscilloscope to detect the signals flowing both to and from the AutoCAD lock, and figured out the "transitions" being performed by the hardware device. The defendants then constructed their own "Auto Key lock" and marketed it as a substitute for the AutoCAD lock. Presumably, a large number of the intended purchasers of the defendant's product would be persons with pirated copies of the AutoCAD program who lacked the AutoCAD lock necessary to make the program run.

Since the disassembly was performed with an oscilloscope and yielded a non-conventional translation of the original data tables, the Court ignored the issue of

---

[231] S. McGregor, "Look & Feel — Australia" (1993) 1 Mealey's Litigation Rep. 44.

[232] Whereas a Canadian court recently rendered the first trial decision dealing with the "look and feel" issue and the protection of non-literal elements in a computer program, Australian courts have not yet done so (see *Delrina*, *supra* note 5).

> "Look and feel" properly refers to the manner in which a computer program interacts with its user (the "user interface"). "Look and feel" includes the manner in which the graphics, sounds, commands, output and design presentation exist in a given program, although much of the case law to date has focused solely on the screen display element of these programs. "Look and feel", it must be stressed, does not apply to elements of program construction that are unrelated to the user interface (Handa & Buchan, *supra* note 62 at 50-51).

intermediate copying. The Court concentrated instead on the final use of the uncovered expression and ruled that the defendants had reproduced a substantial part of the Widget C program in their Auto Key lock and had therefore infringed the plaintiff's copyright. The Court adopted a very strict interpretation of the language of the *Act*, which states that a computer program may be in any material form and in any notation.[233] It also noted that the stream of digits was not random and was therefore worthy of copyright protection.[234] It ignored the fact that the data reproduced was unique and purely functional in purpose, and that there was no other way of creating a sequence that would unlock Widget C.

This case did not raise issues related to the creation of independent interoperable computer programs to work in conjunction with AutoCAD. Whether the Court, in such cases, would be willing to override its finding of copyright infringement in favour of a fair dealing exception remains unclear. The Court's decision, however, foreclosed upon at least one compatible product, the Auto Key lock. While it is tenable that fair dealing cannot apply to the creation of directly competing products, this argument, and consequently the scope of a fair dealing and public interest defence, was never expressly addressed in the case. We are left with the troublesome impression that this decision was perhaps made without a thorough appreciation of the compatibility concerns that are potentially at issue in reverse engineering cases.

Notwithstanding this reservation, *Autodesk* is sufficiently narrow and non-specific with respect to the process of reverse engineering that future Australian courts may allow reverse engineering through a fair dealing exception without offending principles of judicial consistency. As with the decision of the United States District Court in *Uniden*, the *Autodesk* Court did not oppose the process of reverse engineering. The Court simply concentrated upon the defendant's final product and (rightly or wrongly) came to a factual conclusion that it did contain protectable expression copied from the plaintiff's program. Whether ignoring the reverse engineering question implies tacit permissibility remains unclear. The decision in *Uniden*, while similarly overlooking the reverse engineering issue, is logically structured in such a manner that a tacit acceptance of reverse engineering can more readily be inferred than in the instant case.[235]

D. Conclusion

Each of the judicial decisions discussed above has either avoided finding reverse engineering and intermediate copying to be infringements of copyright, or has found fair use provisions sufficient to allow the process. None of the decisions thus far has opposed reverse engineering as an intermediate process. The decisions

---

[233] *Autodesk, supra* note 4 at 174.

[234] *Ibid.* at 169.

[235] For a discussion of this case, see Part **III.A.1**, above.

all properly concentrate on and apply the infringement tests to the final product. Considering that European Union member-states are presently bound to implement reverse engineering provisions in their national legislation, there is a clear trend towards allowing reverse engineering, notwithstanding that it *prima facie* constitutes an infringing act.

In the jurisprudence to date, if the use has been to copy protected expression, a finding of infringement has been made, subject to considerations of compatibility. The determination as to whether the expression is indeed protected by copyright has been based on standard principles of copyright law, such as whether a work is more properly defined as an idea or expression, or whether it is only expressible in a singular manner (doctrine of merger).[236] If the work is protectable, as it was in *Sega*, the court may rely on issues of compatibility to hold that a defence of fair use applies. The following section attempts to further reconcile these different approaches and to provide an analysis of reverse engineering in light of the goals of copyright law. Several solutions designed to deal with the problem of reverse engineering in a Canadian context are also recommended.

IV. Jnstifying the Reverse Engineering of Computer Programs

Conceptually, reverse engineering moves from a finished product to its underlying ideas. Yet even where the goal of the reverse engineering is to obtain the unprotectable ideas, the process typically involves creating intermediate copies, and will be considered an infringing act. This unique and novel conundrum raises the question whether copyright law should be modified to create an exception addressing the problem of intermediate copying, and when such an exception should be applied. Clearly, these questions must be viewed in the context of copyright policy and its intended goals.

A. The Economics of Intellectual Property Rights

The object of intellectual property protection is often described in the language of economic theory as "public goods". Public goods are those goods characterized by non-exclusivity and non-rivalry. Non-exclusivity requires that a good, once produced, be equally available to all members of a group irrespective of their contribution to its production.[237] Non-rivalry occurs where the use of a good by one person will not affect its use by others.[238]

The basic model for setting the optimal, or efficient, price of a non-public good in a competitive market is to pinpoint the price at the point where it is equal to the

[236] See *supra* note 51 for a discussion of the doctrine of merger.
[237] D. Schmidtz, "Contracts and Public Goods" (1987) 10 Harv. J. L. & Public Pol. 475.
[238] *Ibid.*

cost of producing the last unit (marginal cost).[239] This model does not work for a public good because its marginal cost of production is theoretically zero or very close to it. Accordingly, it is expected that with a public good such as a computer program, competitors will flood the market with copies and eventually force the price towards zero.[240] If the price goes below the author's cost of producing the work, given that he or she is a rational economic actor, the work will not be produced.[241] In order to avoid this result, some form of legalized monopoly protection is required to combat the effects of non-rivalry and non-exclusivity, thereby giving the work a number of characteristics naturally attributable to non-public goods.[242]

Certain preconditions are necessary to promote efficiency with tangible property:

(i) *Universality*: all scarce resources should be owned by someone.
(ii) *Exclusivity*: property rights should be exclusive rights.
(iii) *Transferability*: this is necessary to ensure that resources will be transferred from low-valued uses to high-valued uses.[243]

This list forms the basis for granting property rights protection in Western society, and may consist of additional factors depending on a given economic perspecitve.[244] Absent any legal rules to the contrary, public goods do not generally display any of these characteristics because of the principles of non-exclusivity and non-rivalry.

If a legal regime were to impart these characteristics to computer programs, the other economic concepts that generally underlie modern government policy towards markets would be more readily applicable to these goods.[245] Unfortunately, the story does not end there, since by granting owners of "public good" works these characteristics, the law is guaranteeing the owners extremely high returns for their

---

[239] See R. Lipsey *et al.*, *Economics*, 5th ed. (New York: Harper & Row, 1982) at 200.

[240] Landes & Posner, *supra* note 24 at 328.

[241] The cost of producing a copyrightable work can best be thought of as comprising two components: the cost to the author of producing the work, and the cost of copying and distributing the work. In order for a work to be created, the difference between expected revenues and the cost of copying and distribution must be greater than the cost to the author of creating the work (*ibid.* at 326-27).

[242] Other examples of public goods include: national defence, police protection, road construction and environmental protection (Schmidtz, *supra* note 237 at 475-76).

[243] F.H. Stephen, *The Economics of the Law* (Ames: Iowa State University Press, 1988) at 14.

[244] Other features identified might include: durability — that property rights must be granted for substantial periods and cannot be merely transitory; that inaccessible and unique resources are not made the object of property rights protections; and that generally individuals, as opposed to groups, should be given property rights (E. Mackaay, "Informational Goods: Property of a Mirage" (1985) 1 Computer L. & Practice 193 at 195).

[245] According to Mackaay,

> Information[al goods are] a peculiar commodity. Traditional commodities are captured in law as physical goods. But information does not coincide with its physical support. ... This poses particular problems to lawyers: the law traditionally attaches itself to material forms; yet the content, the information, which is immaterial, eludes it (*ibid.* at 194).

efforts. These owners may continue to duplicate and distribute their works forever at virtually no cost. Therefore, in order to regulate the rate of return, the law provides a time limit on intellectual property works after which the works fall into the public domain and assume the characteristics of public goods.[246]

B. The Economics of Reverse Engineering

Framed in the language of the instant debate, the economic analysis of whether to permit reverse engineering under the law of copyright must consider the benefits of this activity in light of the goals of copyright law.[247]

1. Software Compatibility

Reverse engineering benefits the public by creating compatible programs and a standardized environment, which in turn "offers rewards by making programs easier to use, providing greater productivity, and offering greater networking capabilities."[248] In the context of technological progress, societal wealth is maximized by facilitating the creation of a greater number of computer programs. This maximization will depend on authors being adequately protected, and hence remunerated for their efforts. A blanket prohibition of reverse engineering would increase the remuneration payable to certain authors and thus stifle the creation and dissemination of

---

[246] Copyright is protected in Canada for the life of the author plus 50 years (*Copyright Act*, *supra* note 1, s. 6). There has been much debate as to whether the length of time should be decreased for computer programs since, given the rate of technological progress, after 50 years computer programs will be obsolete. Furthermore, as a result of the long term of protection and the purchasing characteristics associated with computer programs, copyright holders stand to make supernormal returns. In 1984, facing a revision that eventually expressly placed computer programs under the jurisdiction of the *Copyright Act* in Canada, the Ministry of Consumer and Corporate Affairs presented a proposal to the Canadian government that would have limited the length of protection of computer programs under the *Act* to five years (Consumer and Corporate Affairs Canada, *From Gutenberg to Telidon: A White Paper on Copyright* (Ottawa: Department of Communications, 1984) at 79ff [hereinafter *From Gutenberg to Telidon*]). This proposal was not acted upon and currently the term of protection for computer programs under the *Copyright Act* is the same as for any other literary work (Subcommittee on the Revision of Copyright, *Second Report* (Ottawa: House of Commons, 1985) at 46 [hereinafter *Second Report*]). This lengthy term of protection is consistent with the approach in other jurisdictions.

[247] Copyright is not primarily concerned with individual rights *per se*. Rights attributed to individuals are merely incidental to the broader purpose of maximizing societal wealth. This is evidenced by the wording of Article I, Section 8, of the United States Constitution which states that Congress has the power "[t]o promote the Progress of Science and useful Arts by securing for limited Times to Authors and Inventors the exclusive Right to their respective Writings and Discoveries." Clearly, the protection of individual rights is recognized as necessary to societal progress. If copying were freely allowed, the incentive for authors to create works would diminish to the detriment of society at large. In such a situation, the cost of obtaining works would also increase, since "[t]here would be a shift toward the production of works that are difficult to copy; authors would be more likely to circulate their works privately rather than widely to lessen the risk of copying; and contractual restrictions on copying would multiply" (Landes & Posner, *supra* note 24 at 332).

[248] Ignatin, *supra* note 17 at 2030.

compatible programs.[249]

A typical countervailing argument posits that banning reverse engineering would only redistribute wealth in favour of the original program developer rather than leading to an increase in societal wealth; however, this argument seems to ring false. If a computer programmer has to pay a licensing fee to link his or her program to another program, he or she may decide to omit the link because the fee could fatally impact on the programmer's cost/benefit analysis. A reduction in the number of compatible programs would negatively impact on societal wealth for several reasons. Without the ability to transfer data between programs, users would be more likely to stay with one program, whether or not it is best suited to their tasks. The cost of switching the data would be so high that the use of inefficiently written programs would become pervasive.

2. The Costs and Benefits of a Standardized Computing Environment

Whether the creation of a standardized computing environment should be encouraged is best examined from an economic perspective. Arguably the most significant benefit to creating standards is that scarce resources are preserved by allowing programmers to use their creative energies to build upon an existing standardized base. Although many software developers have their own internal standards for use by "in-house" programmers, clearly this wealth-maximizing effect would be even more pronounced if industry-wide standards were to exist. Furthermore, scarce resources would be redirected to further technological progress, which seems to be viewed as a normative good — although this is debatable.[250]

Moreover, as development costs would decrease, more programmers would be able to afford to enter the market and thereby increase competition, which is consistent with the spirit of our society's economic structure. Even if these new market players did not possess an absolute advantage over existing firms, they could possess a comparative advantage through specialization and thus allow a more efficient allocation of resources leading to higher levels of output and lower prices.[251] A grant of monopoly protection to any one market player would defeat this effect by creating barriers to entry and increasing trading friction amongst market partici-

---

[249] Encouraging "[c]ompatibility allows innovative new products to enter a monopolized market and lowers development costs by allowing a programmer to attach his single innovative component to a preexisting complete system," in effect allowing the programmer to avoid reinventing the wheel (*ibid.*).

[250] Western economies seem to thrive on and encourage technological progress, whether or not it is a normative good. Technological progress is generally regarded as a societal benefit (Landes & Posner, *supra* note 24). When we talk of societal wealth maximization in the context of reverse engineering we are really talking about reducing inefficiencies and the waste of scarce and valuable resources. Whether these resources are indeed better used in creating a greater number of products and fueling technological progress is a purely subjective question.

[251] Lipsey *et al.*, *supra* note 239 at 350-51.

pants through licensing costs.

From the consumer's point of view, a standardized computing environment seems beneficial because it lowers prices. Software consumers also stand to benefit from savings generated because less time will be required to learn new software packages.

Notwithstanding the benefits mentioned above, the creation of a standardized environment would not be without difficulties. One significant disadvantage would be the risk of "going down the wrong path" of technology, and of only finding out that it is wrong after having expended a great deal of resources. In such a case, the transaction costs required to "get on the right track" could be so great that the correct path might never be chosen.[252] Unfortunately, this risk is not quantifiable and therefore difficult to compare to the associated advantage of creating a standardized environment.[253]

Related to the "down the wrong path" detraction is the possibility that where a standard becomes pervasive, technological myopia will afflict the industry. Once again, however, the disadvantages would not always exist in reality. The existence of better solutions based on the available technology is as unknown as is the degree to which this myopia deters research into other potentially valuable areas.

Notwithstanding whether the development of a standardized environment is a benefit or a hindrance, the development of standards in the computer industry appears to be a naturally occurring phenomenon. One reason for this situation is that users often need to share computer data files generated by various application programs. An example might be an accounting firm that does the books for various businesses. This firm will recommend that its clients use specific software for in-

---

[252] Ignatin, *supra* note 17 at 2028.

[253] Two recent examples of the negative effects associated with the adoption of a standard are informative: the rejection of the Betamax™ video tape standard in favour of the VHS standard, and, more related to the instant discussion, the development of the Personal Computer (PC) architecture by IBM in the late 1970s.

With respect to the latter, computer engineers did not foresee the explosion in computer processing and memory technologies which occurred as a result of rapid tcchnological progress and the falling prices of computer chips. The architects of the original IBM PC and its DOS operating system, which now pervade our society through the distribution of PC "clones", limited its memory acccssing abilities to 1024 kilobytes (KB) of which 640 KB were accessible to application programs. This restriction is now widely referred to in the computer industry as the "640KB barrier". Generations of PC technology computers and software that followed on the heels of the hugely successful PC were also forced to incorporate the 640KB barrier in order to retain compatibility with other generations of personal computers. To many manufacturers and users, retaining compatibility with the PC standard was a priority even though alternate, more effcctive, technologies were readily available soon after the PC's advent. The reasons for programmers' loyalty to this inferior standard were based on cost, marketing, and availability of compatible software. Anyone in the computer industry is aware that the limitations brought about by the 640KB barrier have hampered software development over the years, and resulted in huge inefficiencies reflected as opportunity costs (cconomic losses) in the computer software industry and in society as a whole.

house recordkeeping that is compatible with the firm's own accounting software. Once these relationships develop, an industry standard surfaces, since users want to retain the greatest level of compatibility with other users to ensure reduced costs (through greater competition).

Software developers are themselves often the greatest proponents of a standardized environment. One of the most effective ways of marketing a computer program is to make it compatible with as many existing programs as possible. The newer program will allow users of existing programs to switch easily to it by presumably incorporating new technological advantages. Once the software developer has captured the market it seeks and developed a standard of its own, however, it often changes its tune; it will then seek to protect its standards so that it may exercise monopoly power over its users and obtain greater profits.

Another example of the industry's tendency towards adopting standards is the increased use of object-oriented programming. Object-oriented programming uses standardized routines, known as "objects", and ties them together with other objects and with one's own code to create a finished product. Objects are generally fine-tuned to accomplish a specific task and are typically superior to similar program procedures that are ereated each time a program is written. An object-oriented approach allows more-effeetive programs to be written more efficiently.

In summary, a standardized computing environment has associated economic benefits and costs, although the costs tend to be less quantifiable. Given this state of affairs, it would seem preferable to pursue a standardized computing environment. Furthermore, since the phenomenon is naturally occurring, pursuing standardization involves little or no legal intervention. Conversely, preventing the development of standards would apparently require legislation and would defeat the benefits associated with a standardized environment. The justification for pursuing this latter course of action should involve something more than unquantifiable probabilities of negative consequences. Mere uncertainties are not sufficient to prompt intervention into this naturally occurring economic order. The proverb that "a bird in hand is better than two in the bush" would seem to apply to this thinking. The pursuit of a standardized computing environment based on available information should be considered an economic good, and therefore merits a *laissez-faire* approach.

Because granting monopoly protection over standardized computer environments would reduce their potential efficiencies, a strong reason should be required for such a grant. An analysis of copyright law, the fundamental goal of which is to maximize the goal of societal progress in the fields of arts and technology by protecting authors, does not yield any clear signals as to the possible effect on societal progress of a prohibition of reverse engineering. At worst, the signals are mixed, and at best, they lean in favour of allowing reverse engineering because it leads to the development of a standardized computing environment. If reverse engineering is to be prohibited, more convincing data and a clearer indication of the prohibition's beneficial effects must be demonstrated.

C. Developing a Solution to the Problem

Once it is accepted that reverse engineering should be encouraged, a solution to the potential copyright infringement problems must be developed. The two possible models are the creation of a statutory exception (the European approach), and expansion of the fair dealing exception to cover reverse engineering (the American approach).

1. Creating a Statutory Exception

A statutory exception can be most effectively used to clearly delineate the desired scope of a given legal exemption. Furthermore, the use of a statutory exception would allow the legislature to preclude overrides of copyright terms, which occur through trade secret licensing, as is the case with the British legislation.[254]

Because the use of trade secrets in cases involving reverse engineering is not altogether consistent with the breach of confidence pre-emption clause in section 63 of the Canadian *Copyright Act*, a statutory section drafted with the purpose of allowing reverse engineering should correct this inconsistency.[255] Accordingly, such a section should expressly provide that, in the case of reverse engineering computer programs, the breach of confidence exception outlined in section 63 of the *Copyright Act* shall not apply.[256] Furthermore, since we cannot foresee future challenges to current thinking, the statutory section should be sufficiently flexible and avoid, to the extent possible, any mention of specific technologies. A broad reverse engineering right to create an authorized (legitimate) copy of a computer program should result. The section should expressly provide that it creates an exception to intermediate copying only for the purposes of reverse engineering, and not for use of the materials once the reverse engineering is complete. If at the end of the reverse engineering process, those reverse engineering the product wish to use the fruits of their labour, they will still have to respect the limits of the idea/expression dichotomy. That is, they will be able to appropriate the uncovered ideas but not the protected expression.

The implementation of a statutory exception to reverse engineering is not an altogether novel idea in Canada. A recommendation which implicitly favoured the enactment of such an exception and expressly placed computer programs within the

---

[254] For an analysis of the British legislation, see Part **III.B.**2, above.

[255] A strong argument can be made that the fundamental purpose of advancing a trade secrets argument to prohibit reverse engineering is really the prevention of competition (see text accompanying note 121, above).

[256] Implementation of such a section will undoubtedly require redrafting section 63 so that exclusive jurisdiction of the *Copyright Act* and the exception for breach of confidence laws can be delineated into sub-parts, such as paragraphs 63(a) and 63(b). If, for example, the breach of confidence exception was placed in paragraph 63(b), the reverse engineering exception would then read "section 63(b) shall not apply to the operation of this section."

jurisdiction of the *Copyright Act* was made to Parliament in 1985.[257] Although the recommendation suggested that the government further study issues concerning the shared use of sub-programs, the government decided against doing so, and no such statutory exemption was ever enacted.[258] Since then, both technology and its legal protection have changed. The Canadian government's unwillingness to implement the "shared sub-program" exception in the mid-1980s was not accompanied by any statement that this route would be forever closed. Its refusal should not be taken as indicating any more than a cautious approach to what were uncharted waters. Furthermore, the "shared sub-program" proposal was far more revolutionary than any reverse engineering exception would ever be, since it proposed an exception to the general copyright principle that prohibits the substantial copying of a work. The reverse engineering exception was merely necessarily incidental to the ultimate purpose mandated by that proposal. A true reverse engineering exception, as proposed in this article, would only challenge traditional copyright principles insofar as it would allow intermediate copying; it would not exempt the use of what was uncovered by the process of reverse engineering from traditional copyright principles.

Such an enactment would no longer be revolutionary, given that the nations of the European Union will each soon have statutory exceptions permitting reverse engineering and that recent American decisions allow reverse engineering as a fair use. In fact, Canada's failure to explicitly allow the reverse engineering of computer programs may soon place it in the minority amongst industrialized nations. Clearly, the time has come to reconsider the adoption of such a statutory exception.

2. Relying on the Fair Dealing Exception

Another approach to the problem of reverse engineering would be judicial or legislative expansion of the fair dealing defence.[259] Legislative changes to fair

---

[257] *Second Report, supra* note 246 at 46. The Sub-Committee on the Revision of Copyright recommended that the government "study the possibility of providing an exception to permit the reproduction of a substantial part of a pre-existing program as a non-substantial part of another program." This would have created an exception similar to the reverse engineering exception contained in the I.C.T.A. The Sub-Committee felt that the innovation of computer programs would be accelerated through an exception which allowed the shared use of modular constructs. The Sub-Committee also found that this was a "normal and healthy" practice in the computer industry, and that the law should not impose costs on the industry by preventing the sharing of program code. Although no mention was made of reverse engineering in the copyright revision proposals, it is clear that in order to use sub-programs interchangeably with one's own computer programs, one would need the specifications and parameters. Reverse engineering would be the only means by which to obtain these specifications and parameters, if they were not provided by the manufacturer of the first program.

[258] B. Romaniuk, "Are Computer Software and Integrated Circuitry Legally Vulnerable to Reverse Engineering — Part One" (1986) 3 Can. Computer L.R. 177 at 178-79.

[259] The recommendation discussed in this section applies equally to a public interest defence. However, since this defence is not statutory, and has rarely received judicial notice in the field of copyright, it has relatively little chance of being applied when compared with fair dealing (see Part II.A.1.*f*.iv, above).

dealing have in the past been recommended[260] and subsequently rejected.[261] Legislative widening of fair dealing for the sole purpose of providing an exception to reverse engineering would presumably elicit similar hostility. Because fair dealing is supposed to apply generally to works protected under the *Act*, it would make far better sense to enact a *sui generis* exception, as set out above.

A judicial widening of fair dealing, on the other hand, would avoid these difficulties by putting the problem entirely in the hands of judges. Moreover, this solution provides greater flexibility than would a legislative exception. Further, judicial intervention would not run the risk of being tied to statutory language which might not fully contemplate future challenges to our potentially limited current knowledge. Given our American neighbours' application of their fair use doctrine to reverse engineering cases, it would certainly be no surprise to see Canadian courts similarly expand fair dealing. After all, as copyright is used more frequently in situations which were not contemplated by the drafters of the *Act*, the fair dealing exception, designed to provide courts with flexibility, should not remain stagnant. In accordance with the language of section 27 of the *Copyright Act*, in order to allow reverse engineering under fair dealing, a court would be forced to qualify the reverse engineering as occurring for the purposes of either "research" or "private study". Any intermediate copying required in the reverse engineering process would be conceived as fair dealing since the ultimate goal of the process is to uncover underlying ideas or unprotectable expression for the purposes of achieving compatibility and standardization.[262]

The other side of the argument in favour of judicial intervention is, of course, the uncertainty of waiting for a court to apply fair dealing to any new subject matter. Given the juridical history of fair dealing, it is not clear that the courts will attempt to delineate the concept any time soon. Furthermore, a court must be presented with an infringement claim concerning reverse engineering and a defence based on fair dealing before any judicial widening is even conceivable. Until such a case is presented, the uncertainty associated with the permissibility of reverse engineering under copyright law will create an environment that could potentially dis-

---

[260] In *From Gutenberg to Telidon*, *supra* note 246 at 39, it was recommended that fair dealing, because of its "lack of statutory definition", be replaced with a fair use section containing a "prioritized list of factors to be considered in determining whether a particular use of a work is a fair use." Presumably this list would be similar to that provided in the American *Act*.

[261] *Second Report*, *supra* note 246 at 65-66. The Sub-Committee on the Revision of Copyright recommended, *ibid.* at 64-65, that fair dealing be retained and that a prioritized list of factors not be enacted as "the flexibility so essential to fair dealing would be destroyed by the fact that they would be mandatory and exhaustive." The Sub-Committee, however, was not against the further use of illustrative, non-mandatory factors.

[262] A recommendation that the "research" objective, as outlined in the fair dealing section (*Copyright Act*, *supra* note 1, s. 27), be revised to read "private research" was rejected by the Canadian Parliament (*Second Report*, *ibid.* at 66). The intent of this proposal was to preclude commercial organizations from making use of the fair dealing defence. The rejection of this proposal suggests that commercial organizations are indeed allowed to use the fair dealing defence to justify research and development activities.

suade reverse engineering, and result in the loss of potential efficiencies and societal wealth. Should reverse engineering have been expressly permitted, it is unknown what products, technologies and standards would have developed. Finally, as already mentioned, using fair dealing to permit reverse engineering would not extend to cases of trade secret protection by virtue of section 63 of the *Copyright Act*.[263] If, however, it is determined that existing trade secret protection constitutes a valid limit on the scope of allowable reverse engineering, section 63 would not pose any difficulty.[264]

As a result of the apparent limitations on a solution framed in the context of fair dealing, a clear legislative statement would probably be more successful in effectively resolving the problems associated with reverse engineering computer programs.

3. Alternatives to Copyright

A third, more revolutionary, method of dealing with reverse engineering is one which many critics have been calling for: the redrafting of intellectual property protections as they apply to computer programs. Ever since the protection of computer programs was recognized as a problem, there have been calls for the enactment of *sui generis* legislation designed to deal with it. As a clearer understanding of the economic importance of computers has emerged, coupled with the law's relative inability to effectively deal with the subject matter through existing legislation and judicial decision making, the calls for computer program protection other than copyright have been received less skeptically. Unfortunately, global acceptance of copyright protection of computer programs has now largely precluded a complete shift away from copyright. The resulting inertia is similar to the "going down the wrong path" difficulties, mentioned previously with respect to standardization.[265]

As the differences between computer programs and more traditional copyrightable works become increasingly apparent, and as more legislative exceptions become warranted, it would seem logical to eventually enact legislation to supplement the *Copyright Act*, and cover those aspects for which copyright protection is not appropriate. For example, there have been many difficulties in relation to the pro-

[263] It is likely that under the present copyright regime, the use of a valid trade secrets argument would be sufficient to supplant an argument of fair dealing under the *Copyright Act*. As previously mentioned, however, it is unlikely that the mere use of shrink-wrap licensing will constitute a sufficient relationship between the parties so as to allow the trade secrets argument to be successfully invoked.

[264] There is a remote possibility, however, that the courts could broadly construe compiled object code as confidential and thus be unwilling to apply fair dealing as a result of *Beloff*. This argument, if successful, would grant computer programs special additional protection because of their form. While this type of confidentiality protection exists within the realm of trade secrets, it is beyond the intended scope of copyright. For an analysis of this case, see text accompanying note 98, above.

[265] See text accompanying note 252, above.

tection of computer screens. The Canadian government has indicated that it would not afford such protection under its copyright legislation,[266] however, under certain circumstances, the courts feel that screens are indeed the proper subject matter of copyright.[267] The legal reasoning of the decisions, largely influenced by American case law, is tenuous and does not offer regulatory certainty to those in the industry. This is not the fault of the judiciary, who have been given poor guidance from the copyright legislation and are thus ill-equipped to couch their decisions in clear and simple terms. A more sound approach would be to supplement the copyright legislation with laws expressly designed to handle those issues unique to computer programs. An exception to reverse engineering may be more at home within such legislation, as reverse engineering is similarly unique to computer programs. While it is beyond the scope of this article to present such a *sui generis* regime of computer program protection, one should nonetheless be aware of the possibility of supplementing copyright with other legislation.

Related to that possibility is the more realistic notion that patent protection will become increasingly important as a form of intellectual property protection for computer programs. Patent legislation is more and more frequently being used to protect computer programs. Computer programs protected under the patent regime must fully disclose the manner in which they operate. The trade-off for meeting the more rigorous requirements of the patent system is the greater monopoly protection. Unlike under the copyright regime, even if a patented product is reproduced completely independently of the original product, a royalty must still be paid to the patent holder. Although the term for patent protection is considerably less than that for copyright, it is likely that the term will be more than adequate for computer programs. With patent protection, there is no need for a reverse engineering right because full disclosure is mandatory prior to obtaining the patent, and the relevant information is subsequently made publicly available.

D. The Scope of a Reverse Engineering Right

If an exception to copyright is to be applied, it should not be restricted to the discovery of ideas, as was held in *Atari*. Similarly, permission to reverse engineer a computer program should not be restricted to those parts of a computer program essential to understand the underlying ideas, but should rather be construed broadly. Attempting to limit the process to only those parts of a computer program that capture the underlying ideas is conceptually difficult and sometimes impossible. Furthermore, a specific enquiry that attempts to uncover the various ideas within a program runs the risk of ignoring the interaction of those ideas. This interaction is

---

[266] In their report to the House of Commons, the Sub-Committee on the Revision of Copyright recommended that "there should be no right of display [with respect to user interface screens] in the revised law" (*Second Report, supra* note 246 at 41). At the time, the government responded by stating that it would study this recommendation in further detail although the first round of amendments to the *Act* did not address these recommendations.

[267] See *Delrina, supra* note 5; *Gemologists International Inc.* v. *Gem Scan International Inc.* (1986), 9 C.P.R. (3d) 255, 7 C.I.P.R. 225 (Ont. H.C.J.).

so conceptually abstract that it too falls within the realm of ideas. Restricting reverse engineering in this manner is wholly consistent with copyright law and its intended purpose of protecting expression and not ideas.

Similarly, reverse engineering which seeks to uncover expression should also not be restricted. However, idea and expression are so intertwined that those seeking to reverse engineer a computer program will always claim that they are attempting to discover an underlying unprotectable idea. The increased potential for erroneous judicial decision-making coupled with the waste of resources in bringing such actions to court makes them undesirable. Furthermore, at the end of the day, any decision to allow or prohibit reverse engineering deters from the more appropriate enquiry as to whether protectable expression was incorporated into the newly developed computer program. It is extremely difficult for a court to determine *a priori* whether reverse enginecring was meant to uncover idea or expression, but a court could much more easily decide whether protectable expression had been incorporated into a new computer program. Furthermore, there is a convincing argument to be made that any ideas, however small, underlying a computer program are beyond the mandate of copyright protection. If one is to allow reverse engineering to uncover "large" ideas, whatever that may mean, then surely such reasoning must extend to all ideas. The fact that computer programs exist in a naturally encoded state should not cause copyright law to prevent the uncovering of ideas, whatever their relative importance.

If protectable expression is uncovered during reverse engineering and is illicitly used in the construction of a new computer program, this use would violate copyright independently of the act of reverse engineering.[268] The holder of the first program's copyright would simply bring an action for infringement against the creator of the infringing work. The fact that reverse engineering was performed would not be in issue. Given the potential liability for infringement if copied expression is used in the construction of another program, the only justification for such an action would be that the computer program being reverse engineered contains expression which somehow impedes access to or compatibility with it, as in the

[268] This is akin to the first argument made by Accolade in *Sega*, *supra* note 4 at 1517, where Accolade "maintain[ed] that intermediate copying does not infringe the exclusive rights granted to copyright owners in section 106 of the Copyright Act unless the end product of the copying is substantially similar to the copyrighted work." Accolade lost on this argument because the *Sega* Court felt bound by its decision in *Walker* v. *University Books, Inc.*, 602 F.2d 859 (9th Cir. 1979) [hereinafter *Walker*], which concerned the intermediate copying of fortune-telling cards. The *Walker* decision was framed in broad language, which the *Sega* Court felt must be applied to computer programs. The Court was therefore unwilling to provide a less restrictive interpretation of the fair use doctrine, preferring instead to allow reverse engineering only where the purpose of the process was to gain an understanding of ideas and purely functional concepts, not proteeted by copyright, which are embodied in a computer program. Had the Court deeided otherwise, it might have distinguished the *Walker* decision based on the uniquely functional nature of computer programs. That is, intermediate copying for more traditional works should not be equated to intermediate copying for computer programs since the nature of the copying is for altogether different purposes. Oddly enough, this argument is not a far cry from the "purposeful" analysis engaged in by the *Sega* Court.

aforementioned lock-out cases.[269] Although it is entirely within the right of a programmer to create such mechanisms, it is not within the intended purpose of copyright law to provide additional legal protection to such devices. If the expression used is found to be protectable independently of the reverse engineering issue, then an action for infringement will be successful in any event.

Conclusion

Canadian courts have not yet faced a case of reverse engineering, but it is nonetheless prudent for our legislators to consider the issue prior to such a court challenge, where the results are bound to be unpredictable. Thus far, at least thirteen other Western nations have confronted the problem and have accepted that the reverse engineering of computer programs, under certain circumstances, does constitute a valid exception to copyright infringement. Given the increasing reliance of the Canadian economy on the software industry, it is only a matter of time before the question of reverse engineering is raised in Canada. Furthermore, by confronting this issue earlier rather than later, the legislature has an opportunity to accelerate technological progress in the computer software field. This reasoning underlay the formulation of the decompilation provisions in the European Community's Software *Directive*, and was similarly recognized by the Canadian Parliament in the reverse engineering provisions of the *Integrated Circuit Topography Act*.

The case for creating some form of reverse engineering exception arises from the fact that, practically speaking, the only manner in which a computer program may be disassembled constitutes an infringement of the copyright protections granted to computer programs under the *Copyright Act*. Economically, the advantages to allowing reverse engineering far outweigh the disadvantages and remain consistent with the goals of the copyright legislation. An economic perspective was chosen as the appropriate lens through which to examine the issue because, in an Anglo-American context, the origins and basis of copyright protection are fundamentally rooted in economic issues. By applying the copyright legislation as it currently stands, thereby *prima facie* prohibiting reverse engineering, the serious threat arises of a monopoly over functional standards. Copyright legislation was designed well before the existence of computer technology, and its application to this field is awkward at best because it was not designed to protect ideas or purely utilitarian works. By placing computer programs under the umbrella of copyright, the copyright monopoly may now stifle the development of standards and slow the pace of technological advancement. Such a result is in direct contrast to the stated purpose of copyright law. Preventing the disclosure of a computer's functional processes to the public at large does not occur in any form of intellectual property protection

[269] According to Ignatin, *supra* note 17 at 2022-23, "[t]he inadvertant protection of ideas under copyright may suggest that reverse engineering should be permitted in all instances [although it] is particularly justified [in] making a new program compatible with existing copyrighted software."

other than copyright as it applies to computer programs.

Although the American legislature has thus far remained silent on the matter, higher American courts have faced reverse engineering issues in several cases and have begun to carve out a reverse engineering exception for intermediate copying, based on the defence of fair use found in American copyright legislation. The reasoning of the Ninth Circuit Court of Appeals in *Sega* is essentially correct. The Court's purposive analysis, however, which requires that reverse engineering be limited to only those elements that are not protected by copyright, falls just short of the mark. Unfortunately, the Court felt constrained by the language of an earlier decision[270] concerning the intermediate copying of traditional literary works, and did not recognize the special and unique nature of computer programs.

Although the scope of the reverse engineering right under fair use remains relatively restricted as compared with the European Community's legislative provisions, the extent to which American courts will allow reverse engineering has not been fully settled. The Canadian *Copyright Act*'s fair dealing defence, although not as evolved as the fair use defence, presents Canadian jurists with the flexibility to allow the intermediate copying of computer programs necessary for reverse engineering, notwithstanding the fact that such copying is *prima facie* an infringement. Although fair dealing would allow courts to circumvent the intermediate copying problem, it would be more suitable for Parliament to implement an express exception for reverse engineering that construes the process broadly, rather than restrictively.

Broadly construing a statutory exception is wholly consistent with the purpose of copyright protection, and would continue to protect the expression of a work from being copied. This position is, however, revolutionary in light of those exceptions that have thus far been passed. There is a general consensus that reverse engineering should be limited to cases where compatibility with a computer program is sought; however, the difficulty and subjectivity of such a limitation renders it effectively meaningless and highly uncertain in its application. It is extremely difficult to determine if one who reverse engineers a computer program truly does so for purposes of compatibility. Furthermore, in cases where reverse engineering occurs, it is logical to assume that no specifications have been provided by the original program designers. It is therefore very difficult for the individual performing the reverse engineering to determine *ex ante* exactly what she or he is looking for, since effectively interfacing with a computer program requires an understanding of its general structure as well as specifics about its operational characteristics. A court is thus in a poor position to decide at what point the reverse engineering process no longer concerns compatibility. Even if such a determination could be made, it would serve little purpose in the context of the reverse engineering process. Instead, the determination of whether compatibility is being achieved should be made when examining the allegedly infringing computer program. At this stage, if it is deter-

[270] See *supra* note 268.

mined that expression that does not relate to compatibility has been used, a decision of infringement can be rendered. If a program is reverse engineered and is not otherwise copied, there is no loss to the original owners, save the exposure of underlying ideas, expression and processes to the reverse engineer. To keep the entire underlying structure of a computer program secret because of its binary form properly falls under the law of trade secrets and is well beyond the mandate of copyright.