January 1993


Kathleen Gilbert-Macmillan

Follow this and additional works at: http://digitalcommons.law.scu.edu/chtlj

Part of the Law Commons

Recommended Citation
Available at: http://digitalcommons.law.scu.edu/chtlj/vol9/iss1/6

This Comment is brought to you for free and open access by the Journals at Santa Clara Law Digital Commons. It has been accepted for inclusion in Santa Clara High Technology Law Journal by an authorized administrator of Santa Clara Law Digital Commons. For more information, please contact sculawlibrarian@gmail.com.
COMMENTS

INTELLECTUAL PROPERTY LAW FOR REVERSE ENGINEERING COMPUTER PROGRAMS IN THE EUROPEAN COMMUNITY

Kathleen Gilbert-Macmillan*

TABLE OF CONTENTS

I. INTRODUCTION TO THE EC COUNCIL DIRECTIVE ON LEGAL PROTECTION OF COMPUTER SOFTWARE ........ 249
   A. Purpose of the Directive ........................................ 249
   B. The Reverse Engineering Provision in Article 6 .......... 251

II. HISTORY OF THE EC DIRECTIVE DEBATE ......................... 253
   A. Overview of the EC Legislative Process .................... 253
   B. Comparison of Opposing Positions on the EC Directive ........................................ 255

III. INTERPRETING THE REVERSE ENGINEERING PROVISION .................... 257
   A. Terms of Article 6 ............................................. 257
   B. Competing Products ........................................... 259
   C. Error Correction ................................................ 261
   D. Trade Secret Protection ....................................... 261

IV. PROPOSALS TO CLARIFY THE REVERSE ENGINEERING PROVISION ................. 262

V. CONCLUSION .......................................................... 264

Copyright © 1993 by Kathleen Gilbert-Macmillan.

* Candidate, J.D. 1993, Santa Clara University School of Law; Ph.D. (Education), Stanford University; M.S. (Electrical Engineering), State University of New York at Stony Brook; B.A. (Mathematics), University of Colorado. The author gratefully acknowledges the following during the preparation of this Comment: the insightful guidance of Colin Tapper, All Souls Reader of Law and Vice-President of Magdalen College, Oxford University; valuable discussion with Frederick M. Gonzalez, Asst. Corporate Counsel at Amdahl Corp.; and generous encouragement from Howard C. Anawalt, Professor of Law, Santa Clara University.
INTELLECTUAL PROPERTY LAW FOR REVERSE ENGINEERING COMPUTER PROGRAMS IN THE EUROPEAN COMMUNITY

The European Community (EC) Council of Ministers finally adopted the controversial Directive on the Legal Protection of Computer Programs ("the Directive") on May 14, 1991. Sparks flew during the last months of debate before the European Parliament’s approval of the draft Directive. The debate centered on the "decompilation," or "reverse engineering," issue. The reverse engineering of a computer program is a process by which the program’s structure and code may be derived and analyzed. The program is unraveled to learn how it works. The knowledge gained may provide sufficient technical information to connect new software or hardware with the program, or to develop a similar program.

The Directive as adopted authorizes decompilation under limited conditions. As a result, European software producers may have greater access to the inner workings of American computer programs without the risk of facing an injunction aimed at preventing the European product from being marketed. For example, the Directive permits decompiling a computer program where reproducing the code is "indispensable" to figuring out how to connect a compatible product. Exactly what information is indispensable may be left to the interpretation of the courts in years to come when a product created with the help of reverse engineering analysis com-

3. The terms "reverse engineering," "reverse analysis" and "decompilation" are often used interchangeably. It should be noted, however, that the definitions of the terms are not yet firmly established. Sometimes the term, "reverse engineering," is taken to mean a two-step process: reverse analysis plus forward programming. The reverse analysis step includes disassembling the program to analyze how it works. The forward programming step applies the analysis to building a new program. Interview with Frederick M. Gonzalez, Asst. Corp. Counsel & Chief Counsel Operations, Amdahl Corp., in Sunnyvale, CA (Dec. 15, 1992). Also see, Angelika Schnell & Anna M. Freska, SANTA CLARA COMPUTER & HIGH TECH. L.J. 59, 59 n.1 (1990).
4. Reverse engineering has been described as "software archaeology." It requires "extracting the software's functionality (what the software does) and the design (how it does it) by analysing the software's implementation - that is, programming code, data structures, files and databases." Alan Cane, Fin. TIMES, April 23, 1991, at 10, col. 1 (quoting Gilles LaFue).
petes too fiercely with or replaces an established product in the marketplace.

The Directive is not effective law by itself; each member country of the EC must enact the terms of the Directive in its national laws. After the Directive is implemented and its effects have been assessed in practice, new provisions may be brought before the Commission to improve European law on the reverse engineering of software.

This comment reviews the purpose of the EC Directive and its reverse engineering provision. A brief review of the EC legislative process and an analysis of the lively debate on the Directive provides a background for the comment's discussion of issues in interpreting the reverse engineering provision. Finally, this comment suggests that the provision is too restrictive and permissible reverse engineering should be interpreted broadly.

I. INTRODUCTION OF THE EC COUNCIL DIRECTIVE ON LEGAL PROTECTION OF COMPUTER SOFTWARE

A. Purpose of the Directive

The goal of a single European market within the EC by 1993 increased the urgency to harmonize legislation among Member States' intellectual property laws. Differences in the laws of the various EC countries to protect computer programs have "direct and negative effects" on the functioning of the common market. Such differences are likely to continue without uniform laws among the Member States as they introduce new computer-related legislation. The inconsistent and, in some cases, absence of legislation protecting computer software across the EC has probably suppressed growth of the software industry in Europe. Commercial software sales in Europe have been substantial, but losses due to piracy of

6. If a directive has any direct effect on the law, it is only minor and depends on the degree to which national courts lean toward the language of the directive in interpreting existing law.

7. Council Directive, supra note 1, at 42. The lack of uniformity of legal protection for software discourages sellers of software to treat the EC as a large, single market. Countries with little or no protection for software are likely to be ignored altogether as good potential markets. The diffusion of high technology is then limited and the European economy as a whole may suffer.

8. Id.

9. Information technology spending in Europe, including hardware, software, services-maintenance systems, integration, and consulting has been estimated at more than $140 billion in 1990. Freiburger, U.S. High Tech Eyes Europe, SAN FRANCISCO EXAMINER, Dec. 8, 1991, at E1, col. 6. In 1985 the Western European software market was estimated at $9.5 billion with the sales of packaged software for personal computers growing at more than 30%
software have also been high.  

The Directive now provides a basis for uniform protection of computer programs in the EC. An analysis of the Directive and the surrounding controversy reveals an effort to find the delicate balance among the interests of large software companies, their smaller competitors, and users. At the center of the balancing act is the extent to which computer programs may be "reverse engineered" to create interoperable products. Large companies such as International Business Machines (IBM) prefer limited provisions for reverse engineering to protect their substantial foothold in the European computer market.  

Smaller competitors, including European computer software companies, want broad rights to use reverse engineering in order to build systems and software to be compatible with the software of the giant computer companies.

Traditionally, computer programs have been characterized as fitting more neatly into the subject matter of copyright than patent. Early programs were usually written in a textual form and appeared more similar to literary works, the subject of copyright law, than to useful inventions, the subject of patent law. In the United States, a Congressional study of the legal protection of software resulted in the proposition that computer programs should be protected under copyright statutes. Copyright seemed to afford the necessary protection with low cost.

Patent protection is expensive and more difficult to obtain. The impetus to restrict, even forbid, patent protection for computer programs came, ironically, from the United States in the late 1960s. Courts in the United States, however, have found it difficult to resolve copyright questions about protection of computer programs and computer companies have been seeking software pat-

10. It has been estimated that software manufacturers lost more than 4.5 billion dollars in 1989 due to piracy. See Software Protection: EEC Adopts Directive, Monthly Report on Eur., (Eur. Info. Serv.), § 3, at 7, (June, 1991). Such figures are highly speculative, however, and are calculated as if each "pirated" program would have been a sale. It is not clear that every copied program is equivalent to a lost sale.


ents in the United States in growing numbers.\(^\text{15}\)

Not foreseeing the difficulties with copyright protection, individual European countries quickly adopted the anti-patent stance promoted by the United States for computer programs.\(^\text{16}\) Article 1 of the EC Directive provides for protecting computer programs by copyright as literary works within the meaning of the Berne Convention.\(^\text{17}\) Copyright is the preferred form of protection in the EC because it is most consistent with the existing laws in the Member States and conforms to the trends among its trading partners.\(^\text{18}\)

**B. Reverse Engineering Issue in Article 6**

Article 6 on “decompilation” (reverse engineering) did not appear in the initial drafts of the Directive. It first appeared formally in the Directive in the common position\(^\text{19}\) adopted by the Council a few months before final adoption after intense lobbying and debate. Decompilation of a computer program under the Directive is permitted without authorization when it is “indispensable to obtain the information necessary to achieve the interoperability of an independently created computer program with other programs” under some limiting conditions. Those conditions include that: (i) the reverse engineering be performed by the licensee; (ii) the information necessary to achieve interoperability has not previously been readily available to the licensee; and (iii) the reverse engineering will be confined to the parts of the program necessary to achieve interoperability.\(^\text{20}\) Reverse engineering is not to be used for goals other than to achieve interoperability of the independently created program. Neither is it permitted for reverse engineering to be used for the development of any computer program “substantially similar in its expression, or for any other act which infringes copyright.”\(^\text{21}\)

There has been controversy over the meaning of “interoper-


\(^{16}\) TAPPER, supra note 14, at 9.

\(^{17}\) The Berne Convention is a series of acts, not a single document. Most EC Member States adhere to the Paris Act of 1971, reprinted in 4 M. NIMMER & D. NIMMER, NIMMER ON COPYRIGHT app. 27 (1988).


\(^{19}\) The common position is the draft form of the proposal which the Council of Ministers is willing to adopt before the draft is returned to the European Parliament for a second reading. See discussion infra part II.A. for an overview of the EC legislative process.

\(^{20}\) Council Directive, supra note 1, art. 6(1).

\(^{21}\) Id., art. 6(2).
ability.” The Directive defines “interoperability” as “the ability to exchange information and mutually to use the information which has been exchanged.”22 Whether or not this means reverse engineering can be allowed for creating replacement products, not merely attaching or “interfacing” products, was much debated. An attempt to clarify the issue was made by a communication to the European Parliament from the Commission: “Decompilation is permitted by Article 6 to the extent necessary to ensure the interoperability of an independently created computer program. Such a program may connect to the program subject to decompilation. Alternatively it may compete with the decompiled program and in such cases will not normally connect to it . . .”23 In other words, decompilation may not be used to reproduce pieces of a program that are unrelated to the interoperability of the original program. However, decompilation may be used to create a competing program as long as the only “reverse engineered” parts of the original program are those that affect the program’s interfaces with other programs and computer systems.24

A goal of Article 6, in its attempt to loosen restrictions on decompilation, is to move the EC in the direction of open systems.25 Open systems, in the broad sense, provide the capability to use the same software on different kinds of computers and to exchange data on a wide variety of computer networks.26 Yet in reality, the narrowness of the Directive’s provisions and the tight circle drawn around permissible reverse engineering only for purposes of interoperability may have only a minor effect on the EC’s movement

22. Id., at 43.
23. SEC 91 final - SYN 183, quoted in Mark Powell, 8 COMPUTER LAW. 13, 16 (1991). Further support was given to the interpretation that reverse engineering may be used both for attaching and competing products during a conference on the Directive in March, 1991. H. C. Overbury, Head of the Merger Task Force, said in a speech, “The Commission believes that where necessary . . . it will be possible for competitors to extract interface information which is not covered by copyright by analysis techniques so as to develop interoperable products. These interoperable products may be attaching products or they may be competing products.” Id., fn. 12 at 15.
26. The definitions of “open systems” vary according to the provider of the definition. A starting point, however, may be the definition from the Institute of Electrical and Electronic Engineers: “[Open systems are] a comprehensive and consistent set of international information technology standards and functional profiles that specify interfaces, services and supporting formats to accomplish interoperability and portability of applications, data and people.” Quoted in IBM SYSTEM USER, Feb., 1992, Vol. 13, No. 2 at 37.
toward becoming a development center and marketplace for open systems. The proposal in this Comment is for a broad interpretation of the Directive's Article 6 provisions to encourage technological developments for the support of open systems.

II. HISTORY OF THE EC DIRECTIVE DEBATE

A. Overview of the EC Legislative Process

Twelve Western European countries are Member States of the European Economic Community. The Community is governed by five institutions: the European Commission, Council of Ministers, European Council, European Parliament, and the European Court of Justice. The Commission proposes legislation to the Council of Ministers to implement as treaties. The Commission is also to ensure proper implementation of the Directives adopted by the Council of Ministers. If a Member State fails to implement a Directive correctly or in time with its own national legislation, the Member State may be called before the European Court of Justice for treaty violations.

The Commission formally initiates the legislative process and submits an initial proposal to the Council of Ministers whose members are appointed by their respective national governments. After the Council comes to an agreement on the proposal, it is reviewed, debated, possibly amended, and written as draft legislation. The European Parliament may recommend to adopt the draft and it is returned to the Commission. The Commission then presents a modified proposal to the Council. The Council works to reach a common position, the draft form of the proposal the Council is willing to adopt. This draft is returned to the Parliament for a second reading. Parliament returns its final recommendations for adopting, rejecting, or amending the common position so that the Council can officially adopt or reject the proposal. If the Parliament rejects the common position, a unanimous vote by the Council is required to pass the legislation.

In June, 1988 the Commission published the 237-page Green

---

27. The European Economic Community was established under the Treaty of Rome, Mar. 25, 1957, art. 2, 298 U.N.T.S. 11, 15. The current Member States are Belgium, Denmark, France, Germany, Greece, Ireland, Italy, Luxembourg, the Netherlands, Portugal, Spain and the United Kingdom.

Chapter 5 of the Green Paper is concerned with the protection of copyright in computer programs. It emphasizes the economic importance of computer software, the present dominance of the U.S. firms in the world software market, and the need for appropriate legal protection for software to encourage investment and innovation by Community firms, permitting the Community industry to catch up with its competitors.30

The Green Paper called for, as a matter of urgency, a proposal for a directive for the protection of computer programs. After a public hearing and replies to its questionnaire, the Commission submitted its first proposal in December, 1988.31 The Economic and Social Committee, with few comments, gave overall approval to the draft.32 Although the reverse engineering issue was heavily debated at this time, the draft was devoid of specific language on the topic. It was at least arguable, however, that reverse engineering would be prohibited.33 A milestone in consideration of the Directive was achieved in July, 1990 when the European Parliament completed its first reading and adopted numerous amendments to the Directive.34

A revised draft of the proposal was submitted by the Commission in October, 199035 and a common position was adopted by the Council in that same year. Language was included that more clearly would permit limited decompilation. The common position was the form in which the Council was prepared to adopt the legislation. In April, 1991, the proposed Directive received its second reading by the European Parliament. Eleven amendments to the common position were proposed. The amendments were intended to broaden the scope of the Directive for research and analysis, as well as to clarify the status of interfaces under copyright law. The leading advocate promoting adoption of the amendments was the European Committee for Interoperable Systems (ECIS), a group of

30. Id.
computer companies in favor of authorizing extensive reverse engineering of software. 36 However, the amendments failed to get sufficient votes in the European Parliament and the Council formally adopted the common position in May, 1991. EC Member States are required to enact legislation in compliance with the Directive before January 1, 1993. 37

B. Comparison of Opposing Positions on the EC Directive

The controversy during the passage of the Directive centered on two related provisions of the first draft: protection of interfaces and the prohibition of reverse engineering.

1. Interfaces

The Directive defines interfaces as the parts of the program which provide for the interconnection and interaction between elements of software and hardware. 38 A goal of the Commission, stated in the Green Paper, is to encourage interoperability within and among computer systems. A prerequisite to interoperability is open interfaces, that is, published, freely available specifications or documentation containing the information required to be able to connect to or interact with the computer systems. The first draft of the Directive gingerly gave access to interfaces by making interoperability an exception to general copyright rules, but the language was still cloudy. 39 The final Directive substituted new language which effectively removed some doubt, but pressure through groups like the Business Software Association (BSA), a group of business software producers and SAGE, a group of primarily American hardware manufacturers, resulted in a narrower scope of allowable access to interfaces. 40 Groups such as BSA and SAGE who wanted to prohibit analysis of interfaces argued that opening the access to interfaces and exempting these parts of computer programs from copyright protection would harm the software industry. 41 They claimed that an exception for interfaces could not be clearly drafted and so would

36. Cane, supra note 4.
40. See, e.g., Colombe & Meyer, supra note 34.
41. See, e.g., William T. Lake, John H. Harwood II, and Thomas P. Olson, Seeking Compatibility or Avoiding Development Costs? A Reply on Software Copyright in the EC, 12 EUR. INTELL. PROP. REV. 431 (1989).
result in permission to copy the detailed expression of a successful
program. They argued further that the real issue raised by the ex-
ceptions for interfaces, as well as reverse engineering, was whether
easy cloning should be allowed under EC copyright law. In their
view, the present system works well without exempting interfaces
from copyright protection: the software industry is flourishing and
access to interfaces, while often available by industry choice, is not
a legislated exemption under the law of any country.42

Others, primarily represented by ECIS and smaller European
computer firms, argued that the Directive should state that the
specification of interfaces to computer programs are exempt from
protection under copyright law. “The majority of people in the in-
dustry, as well as computer users throughout Europe, will be best
served by clear language in the Directive that authorises use of spec-
fications underlying program interfaces and permits reverse analy-
sis of existing computer products . . .”43 The final Directive has
clarified that ideas underlying an interface are exempt from copy-
right protection. However, the formulation contained in the Direc-
tive is general and producing a clone without infringing the
copyright in the original may still be difficult.44

2. Reverse Engineering

Reverse engineering is the second major issue that dominated
debate on the Directive. It is also the issue that continues to be a
primary source of uncertainty concerning future application of the
new legislation. Under the first draft of the Directive, reverse en-
engineering was essentially prohibited. Those in favor of providing an
exemption for decompilation included the ECIS led by Fujitsu, the
Japanese computer manufacturer. They argued that without such
an exemption, it would be impossible for competitors to develop
competing software products since it is necessary to understand
how a program functions before one can develop a competitive pro-
gram, or software that will interact with the original program.45

Another argument in favor of permitting decompilation is that
decomping computer programs is merely the discovery of rules es-
tential in the process of original programming.

42. Id. at 431, 432.
43. Colombe & Meyer, Seeking Interoperability: An Industry Response, 3 EUR. INTELL.
PROF. REV. 79, 82-3 (1990).
44. Chris Reed, Reverse Engineering Computer Programs Without Infringing Copyright,
45. Small, supra note 33, at 19.
[Decompilation] is not a procedure for picking apart a complex object into its elements, so that each may be imitated and the whole copied in an exact or closely similar imitation. Nor is it even an identification of the elements of a product so that they may be adapted in some improved way to a new end. It is simply the discovery of the rules which have to be complied with when the independent producer constructs his own program.46

Arguments against this position, supported by BSA and SAGE, include the following: (i) permitting reverse engineering would be a dramatic change from existing law; (ii) reverse engineering is unnecessary to develop interoperable products because manuals and other documentation can be used; (iii) imitation programs could be reproduced at much lower costs than the original; and (iv) legalizing reverse engineering would dramatically reduce the lead time that motivates investment in new software.47

In adopting the final Directive, the Council voted in favor of reverse engineering provided that: (i) it is performed by the licensee; (ii) the information necessary to achieve interoperability has not previously been readily available to the licensee; and (iii) the reverse engineering will be confined solely to the parts of the program necessary to achieve interoperability.48 The Directive prohibits any information acquired from permissible reverse engineering to be used for any goal other than to achieve interoperability. Perhaps reflecting the lobbying efforts of groups such as SAGE, reverse engineering cannot be used in the development of any competing product.

III. INTERPRETING THE REVERSE ENGINEERING PROVISION

A. Terms of Article 6

Article 6 of the Directive is under the heading "Decompilation" and states its terms as follows:

46. Cornish, supra note 39, at 391.
47. Lake et al., supra note 41. It has been suggested that authorizing reverse engineering would discourage software developers from investing in the creation of new programs. The argument is that "the prospect of almost immediate competition from an unconsent adaptation of his own work - which could be sold cheaply because the imitator bore little development expense" might be sufficient to discourage "[especially] newer and smaller developers." Id. at 434. Perhaps, however, prohibiting reverse engineering is more likely to prevent development of new products by smaller developers because they are forbidden to do research and analysis to create products that would otherwise have been interoperable. In reality, manuals and other written materials that fully and accurately document interface information are rarely available.
1. The authorization of the rightholder shall not be required where reproduction of the code and translation of its form . . . are indispensable to obtain the information necessary to achieve the interoperability of an independently created computer program with other programs, provided that the following conditions are met:

(a) these acts are performed by the licensee or by another person having a right to use a copy of a program, or on their behalf by a person authorized to do so;
(b) the information necessary to achieve interoperability has not previously been readily available to the persons referred to in subparagraph (a); and
(c) these acts are confined to the parts of the original program which are necessary to achieve interoperability.

2. The provisions of paragraph 1 shall not permit the information obtained through its application:

(a) to be used for goals other than to achieve the interoperability of the independently created computer program;
(b) to be given to others, except when necessary for the interoperability of the independently created computer program; or
(c) to be used for the development, production or marketing of a computer program substantially similar in its expression, or for any other act which infringes copyright.49

Although the terms of Article 6 authorize some reverse engineering, the range of permissible purposes is narrow. Reverse engineering may be used to extract information necessary to interface one program to another, but not for developing any computer program that would be so similar as to result in a copyright infringement. "In practical terms, . . . it would be permissible to reverse engineer Microsoft MS-DOS to produce a properly engineered IBM PC compatible . . . application, but not in order to produce an IBM compatible . . . operating system (although it would be possible to reverse engineer the IBM BIOS [basic input-out system] to do that)."50

Although some people fear that any degree of permissible reverse engineering will harm the owners of the original program, the Directive provides those owners substantial protection. The owners of the copyright in the original program are protected against the reverse engineering of that program by anyone who has not bought or licensed it. Furthermore, the Directive permits reverse engineer-

49. Id.
50. Small, supra note 33, at 20.
The original developer can prevent reverse engineering of the program by making the information necessary to achieve interoperability readily available to the buyer or licensee. An underlying problem, however, is the determination of the factual question concerning what and how much information is necessary for a software developer to build an interoperable program. It is not clear whether or not the standard should be different for a small independent software producer as compared to a large multinational computer company with sophisticated technology. Also unclear is who decides whether the interface information is sufficient, accurate and up-to-date. These unanswered questions point to a potentially significant weakness of the Directive.

The Directive emphatically protects the rightholder from misappropriation of software by disallowing reverse engineering to be used for any goal other than to achieve interoperability of the independently created computer programs. As if to underscore that provision, two paragraphs later the Directive states that reverse engineering shall not be used for developing, producing or marketing any program substantially similar to the original program.

B. Competing Products

One of the chief concerns about Article 6 is that the strict control of reverse engineering could result in limiting the supply of competitive products and seriously harm computer users' ability to maintain and integrate systems. Small software developers are likely to be the most reluctant to undertake the reverse engineering necessary for them to develop similar competing products. They can least afford the risk of protracted litigation to clarify whether their reverse engineering was, for example, "confined to the parts of the original program which [were] necessary to achieve interoperability" or "used for goals other than to achieve the interoperability of the independently created computer program." Although it appears that Article 6 allows reverse engineering in order to develop noninfringing competing products, the reverse engineering is limited to the parts of the original program related to the interface.

51. Council Directive, supra note 1, art. 6(1)b.
52. Council Directive, supra note 1, art. 6(2)a.
53. Council Directive, supra note 1, art. 6(2)c.
55. Council Directive, supra note 1, art. 6(1)b.
The interfaces of computer programs are rarely neatly defined. Paradoxically, it may be difficult or impossible to determine which parts of a computer program are technically necessary to achieve interoperability, hence permissible for reverse analysis, without a full analysis of the entire program.

Competition can encourage innovation in technology. Restricting reverse engineering to only the purpose of obtaining interface information will limit the diffusion of ideas and principles underlying the original program. The Directive allows users of computer programs to "observe, study or test the functioning of the program in order to determine the ideas and principles which underlie any element of the program if he does so while performing any of the acts of loading, displaying, running, transmitting or storing the program which he is entitled to do." Denying the user the right to engage in reverse engineering to study the ideas and principles underlying the program as a whole may reduce innovation and inhibit competition.

Although reverse engineering is permitted for developing independently created software under the constraints of Article 6, there is no explicit extension to applying the same reverse engineering to the development of hardware. The recitals preceding Article 1 of the Directive expressly refer to hardware, however, in defining interoperability:

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 interconnection 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;

It is therefore sometimes argued that independently created hardware may be produced as a result of information learned through reverse engineering authorized under Article 6. On the other hand, it is argued that the permitted reverse engineering pertains only to the creation of the interconnecting software and not to the creation of new hardware. Ultimately, then, the issue of whether or not new hardware may be built by utilizing information acquired

56. For example, the rapid development of IBM personal computer products is largely attributable to the widespread activity of IBM clone-makers.
58. Id. at 43.
60. CZARNOTA & HART, supra note 25.
through decompilation of software may be left to either the courts or to the standards and customs of the industry.

C. Error Correction

Translating, adapting, or otherwise altering a computer program to correct errors in the program where necessary for the use of the program is permitted by the Directive without authorization of the rightholder in the absence of contractual provisions to the contrary.\(^6\) It appears then that the decompilation restrictions of Article 6 do not apply to error correction. What constitutes an error under the Directive, however, is unclear. The user may identify a behavior of a program as an error, while the owner may define the behavior as an intended feature. The effect may be to diminish the limitations of the decompilation provisions as long as the user can show that reverse engineering was "necessary" for the intended use of the program.

D. Trade Secret Protection

Article 6 does not override "any other legal provisions such as those concerning patent rights, trade-marks, unfair competition, trade secrets, protection of semi-conductor products or the law of contract."\(^6\) A software company might then claim that it could protect a given program against reverse engineering through a contract with the licensee preventing the licensee from disclosing or using any trade secret of the licensor. While the licensee could derive information about the program under Article 6, in this example, the software company's interpretation of the Directive implies that the licensee could not use that information. Such a result would be contrary to a stated objective in the Directive: "to make it possible to connect all components of a computer system, including those of different manufacturers, so that they can work together."\(^6\) Article 9(1) of the Directive provides that any contract contrary to the provisions of Article 6 shall be deemed null and void under the Directive.\(^6\) The Directive must then be interpreted to mean that contractual restrictions based on trade-secret protection cannot be used to retain exclusive rights to the interface information which

---

62. Id., art. 9(1).
64. Id.
may legitimately be obtained without infringement.65

IV. PROPOSALS TO CLARIFY THE REVERSE ENGINEERING PROVISION

It is usually difficult to prove that reverse engineering has been performed on a computer program unless an infringing copy is produced. In that case, the law of copyright suffices and one may question the need for Article 6 of the Directive. Typically intellectual property law does not prohibit the study and analysis of the ideas and principles of the underlying product, even for those who intend to create competing products.66 The strongest protection is given by patent law where all the underlying ideas are disclosed to the public after a strict review to verify that the invention is novel and non-obvious.67 The law of trade secrets clearly allows the analysis of ideas once the secrets have been learned by another without a breach of confidence. Reverse engineering is a likely way to gain that knowledge. Neither does the law of copyright, as applied to works other than computer programs, prevent the study of the work’s underlying ideas and principles. A person may read and analyze all the ideas in a book for the purpose of creating a competing work as long as she does not infringe the expression in the original book. Computer programs under the EC Directive appear to be the sole exception to these general intellectual property law principles.68

A better policy might be to revise Article 6 to permit reverse analysis to study the underlying ideas and principles of a computer program so long as an infringing product is not produced. Such a provision could be similar to the reverse engineering allowed in the semiconductor field by the Semiconductor Chip Act of 1984.69 This legislation legitimized “the general industry practice of ‘reverse engineering’ whereby existing chips were improved upon enough to constitute original new designs.”70

The Directive is unclear as to whether new, independently created hardware may be developed from information acquired

66. See, e.g., SEGA Enterprises v. Accolade, Inc., No. 92-15655, 1993 U.S. App. LEXIS 78, at 27 (9th Cir. Jan. 6, 1993). The court stated that “[w]here there is a 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.”
68. Colombe & Meyer, supra note 34, at 327-8.
70. TAPPER, supra note 14, at 43.
through the reverse engineering of software.\textsuperscript{71} Contrary to arguments that it is inappropriate to permit software reverse engineering in order to create new hardware,\textsuperscript{72} this Comment proposes adding specific language to the Directive in support of allowing reverse engineering for the purpose of building new, interconnecting hardware. Often the technological line between software and hardware is blurry, if not invisible. The addition of language permitting hardware development based on software interface decompilation would simply remove concerns about that blurry line. Such language would also expand the possibilities for applying existing information about particular software interfaces to new technological developments in hardware.

Software developers who choose not to make interface information available under the terms of the Directive face the risk of their products being reverse engineered by competitors. But the Directive does not make it clear whether or not the information must be made \textit{freely} available or if the developer can charge the licensee specifically for the interface specifications. It has been suggested that if the licensee refused to pay for the information, the original developer could protect its rights in the program against reverse engineering.\textsuperscript{73} This interpretation appears to run counter to the intention of the Directive to encourage interoperable systems. If original developers are permitted to charge for interface information, the exemption for reverse engineering in Article 6 is potentially vacuous. A better interpretation would be that if original developers do not freely make interface information available, they run the risk of their products being decompiled by others. The alternative is for the developer to make enough information available so that another programmer can write interface software to be fully interoperable with the original program. The right of the developer to charge money for the interface information should be specifically and vehemently denied in the Directive and in the implementing legislation in order to protect the fundamental intent of Article 6 in favor of interoperable systems.

However, Article 6, as finally adopted in the Directive, has been the subject of vigorous long-term debate and compromise. It is unlikely that the Directive will be rewritten in the near future. Instead, as Member States write their own legislation to implement

\textsuperscript{71} See supra, part III.B.
\textsuperscript{72} CZARNOTA \& HART, supra note 25.
the Directive as it stands, they should clarify some of the general and uncertain sections with more specific language. At the same time, wide latitude should be given for study and analysis of the underlying ideas and principles of computer programs. For example, the condition of "necessity" required in Article 6 to conduct reverse engineering should be taken to mean "reasonable necessity." As written, the provision requires reverse engineering to be "confined to the parts of the original program which are necessary [emphasis added] to achieve interoperability."74 "Necessary" should be rewritten, or interpreted, to mean "reasonably necessary." Otherwise, in those cases where it is discovered later that a particular well-intended reverse analysis was not necessary to achieve interoperability, the court may lack flexibility to apply an appropriate standard.

Finally, as the Directive is implemented and interpreted, the technology of computer programs will continue to change and introduce new complexities. Each Member State, its legislature and courts, should strive to balance their national goals with both the competition and protection required for technological progress in the EC and globally.

V. CONCLUSION

The Directive has been praised as finding a balance among the interests and needs of the market leaders in the computer industry, those who depend on information about the interfaces of the market leaders' products, and legitimate program users.75 Despite some difficulties with general wording and uncertain terms, the expected result is that soon there will be uniform protection of computer programs throughout the EC. European software producers may be better able to compete with the market leaders as a result of the opening of interfaces to reverse engineering. The price the market leaders will have to pay appears slight and they will receive the benefits of widespread copyright protection for their software throughout Europe.

74. Council Directive, supra note 1, art. 6(1)c.
75. Dreier, supra note 65, at 319.