



January 2002

## Innovation, Software, and Reverse Engineering

Brian Fitzgerald

Cristina Cifuentes

Anne Fitzgerald

Michael Lehmann

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



Part of the [Law Commons](#)

### Recommended Citation

Brian Fitzgerald, Cristina Cifuentes, Anne Fitzgerald, and Michael Lehmann, *Innovation, Software, and Reverse Engineering*, 18 SANTA CLARA HIGH TECH. L.J. 121 (2001).

Available at: <http://digitalcommons.law.scu.edu/chtlj/vol18/iss1/5>

This Symposium 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](mailto:sculawlibrarian@gmail.com).

---

## SYMPOSIUM REVIEW

---

### INNOVATION, SOFTWARE, AND REVERSE ENGINEERING

**Professor Brian Fitzgerald, Visiting Professor Santa Clara  
University School of Law, Head of Law School, Southern  
Cross University NSW, Australia;**

**Dr. Cristina Cifuentes, Sun Microsystems;**

**Anne Fitzgerald, Technology Lawyer, Australia;**

**Professor Michael Lehmann, Visiting Professor Santa Clara  
University School of Law, University of Munich and Max  
Planck Institute, Germany.<sup>†</sup>**

*This daylong Conference convened by Professor Fitzgerald and Dr. Cifuentes was held at Santa Clara University School of Law, Santa Clara, California, on March 23, 2001. The Conference aimed to bring together leading technologists and lawyers, from across the globe, in order to better define and articulate technological reasons and legal principles for reverse engineering in the area of software.*

While the legality of the reverse engineering of software, within defined limits, has been firmly established in Europe and the United States since the early 1990s, a number of recent developments invite further discussion of the issue. Specifically, three issues are worthy of note: first, the growth of digital or software locks and the enactment of the 1998 Digital Millennium Copyright Act (the “DMCA”) which raises civil and criminal liability for circumvention, and directly addresses circumvention devices in certain instances; second, the growing need to provide quality assurance through reverse engineering in the areas of security and error correction; third,

the massive increase in software patents and the uncertain legal situation of the reverse engineering of patented software.

In addition to these issues, the Conference posited the inherently discursive nature of digital property, in particular software, and the way in which digital property should be facilitated by reverse engineering to augment digital diversity.<sup>1</sup>

## I. REVERSE ENGINEERING: THE TECHNOLOGICAL PERSPECTIVE

### *A. What is Reverse Engineering?*

In opening the seminar, Professor Howard Anawalt of Santa Clara University School of Law highlighted the presence of reverse engineering in our everyday lives: "Have you ever wondered how something works?" he asked, explaining, "that is the genesis of reverse engineering. We do it all the time." However, the fact that reverse engineering is something we inquire about and aspire to do in many aspects of our lives is complicated in the area of software.

For a start, it is difficult to understand the functionality of software from its outward appearance, so that it becomes necessary to take a closer look. Yet, in attempting to understand a software program, some form of copying must take place, which potentially infringes the copyright holder's exclusive right to control reproduction of the work and the patent holder's exclusive right to exercise the invention. Furthermore, once the informative details of the program's operation are discovered, the cost of using that information to make a competing product may be very low. Therefore, in order to facilitate reverse engineering of software, some exceptions must be made to the copyright and patent holder's

---

<sup>†</sup> The convenors acknowledge the enthusiastic support of this Conference by presenters Professors Anawalt, Chisum and Felten; Messrs Chikofsky, Martino, Von Lohman and Aharonian; and by Dean Mack Player, Assistant Dean Elizabeth Powers and Teresa Ochoa of Santa Clara University School of Law. The convenors also thank Lawrence Rosen, Claire Badaracco, Scott Kiel-Chisholm, Bryan Wyman, Karen Spindler and Leda Mouallem for their assistance and Santa Clara University School of Law's High Tech Program, IEEE and REF for their support in running the Conference.

<sup>1</sup> See generally Brian F. Fitzgerald, *Software as Discourse: The Challenge for Information Law*, 2000 EUR. INTELL. PROP. REV. 46; Brian F. Fitzgerald, *Software as Discourse: The Power of Intellectual Property in Digital Architecture*, 18 CARDOZO ARTS & ENT. L.J. 337 (2000); *Associated Press v. United States*, 326 U.S. 1, 20 (1945) (providing an analogy for the notion of diversity in discursive formations); *Turner Broad. Sys., Inc. v. FCC*, 512 U.S. 622, 663-64 (1994) (explaining that the "basic tenet of national communication policy is that 'the widest possible dissemination of information from diverse and antagonistic sources is essential to the welfare of the public.'").

exclusive rights. Nonetheless, these exceptions also need to ensure against subsequent uses of that information which may lead to a situation of free riding.

Elliot Chikofsky of the META Group (META) and the Reverse Engineering Forum (REF), a scholar well known for developing a lexicon on the reverse engineering of software in the early 1990s, addressed the notions of forward engineering (designing and making the product, service, or transaction) leading to reverse engineering (going back to find out how it works). Chikofsky explained the process of going from reverse engineering to re-engineering (going back and then forward again in a process of transformation of the product, service, or transaction) or restructuring the product, service, or transaction).<sup>2</sup> Likewise, in *Kewanee Oil Co. v. Bicron Corp.*, the United States Supreme Court defines reverse engineering as the process of “starting with the known product and working backwards to divine the process which aided in the development of manufacture.”<sup>3</sup>

Chikofsky explained that reverse engineering is not new. Rather, reverse engineering happens whenever software is examined; it is a natural part of software and Internet development, an essential component of R&D, and a necessary element of systems interoperability. Chikofsky summarized some key reasons for the reverse engineering of software: “The programmer is long gone; our business relies on this software program, but we do not know how it works or how it could fail; it did not do what we thought it should have done; we cannot find the documentation; how can it be maintained, adapted, enhanced;” and lastly “we are afraid to touch it.”

More specifically, Chikofsky explained the following as examples where reverse engineering is used: documenting existing code, platform migration, language translation, product evaluation, interactive development, design recovery, integration of components, auditing software assets, and IT integration after mergers. Reverse engineering can start at different levels of abstraction; for example, at the source code level to recover design or documentation, or at the object code level to recover source code.

---

<sup>2</sup> Elliot J. Chikofsky & James H. Cross II, *Reverse Engineering and Design Recovery: A Taxonomy*, 7 SOFTWARE 13, 14–16 (1990).

<sup>3</sup> *Kewanee Oil Co. v. Bicron Corp.*, 416 U.S. 470, 476 (1974).

### *B. The Specific Dimensions of Reverse Engineering of Software*

Dr. Cristina Cifuentes, of Sun Microsystems, presented an outline of the types of reverse engineering processes that might be undertaken in the software context at the object code level.

Dr. Cifuentes explained that software development is a process of design, implementation and testing. Software normally is developed in a high-level language that resembles natural language. This program is then translated to object code (or machine code) by a process known as *compilation* of the program. This process can be viewed as translating high-level language code to assembly code for the particular machine, and then assembling this code to object code (see Figure 1(a)). It is important to note that both assembly code and object code are machine dependent codes. Specifically, both codes use instructions available on the target machine where these instructions vary from one machine to another. In contrast, source code is machine independent.

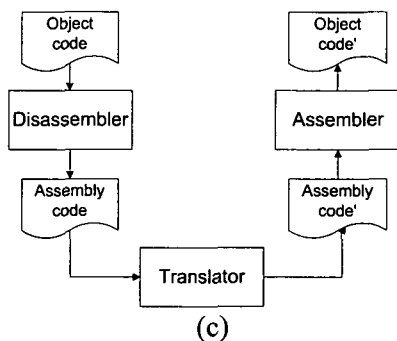
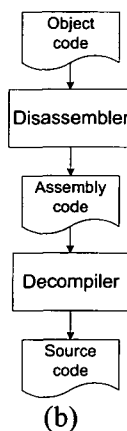
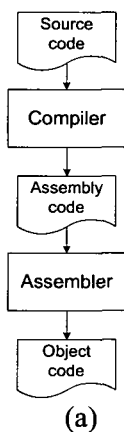


FIGURE 1, preceding page: (a) the Compilation Process, (b) the Decompilation Process, (c) the Emulation Process and Binary Translation Process.

### 1. Disassembly

Disassembly is the process of translating object code to the assembly code of the machine. Disassembly techniques were used in the 1980s and 1990s to aid in the understanding of computer viruses and to treat them. A prominent example is the Internet Worm in 1988, where a worm (virus-like) program infected thousands of computers connected to the Internet, including those at universities, military sites, and medical research facilities.<sup>4</sup> Disassembly techniques were used at the University of California at Berkeley and the Massachusetts Institute of Technology to understand how the worm worked. Disassembly has also been used to determine the software interface to hardware; for example, for interoperability purposes<sup>5</sup> where this information was not made available to the public. Further, disassembly has also been used for the purposes of error correction, for example, in fixing year 2000 problems at the object code level.<sup>6</sup>

### 2. Decompilation

Decompilation is the process of translating object code back to source code; that is, it is the inverse of the compilation process. Figure 1(b) illustrates the decompilation process. In general, a decompiler encompasses the processes of disassembling object code into assembly code, analyzing the assembly code, recovering high-level information from the assembly code, and generating source code. Decompilation is an incomplete process, in general, because of the amount of information lost by the compiler when compiling source code to object code. However, heuristics and human

---

<sup>4</sup> See ZEN AND THE ART OF THE INTERNET 8.1: THE INTERNET WORM (Brendan Kehoe ed.) at [http://sunland.gsfc.nasa.gov/info/guide/The\\_Internet\\_Worm.html](http://sunland.gsfc.nasa.gov/info/guide/The_Internet_Worm.html) (last visited Oct. 23, 2001).

<sup>5</sup> Examples of this usage include the development of Genesis-compatible games by Accolade Inc., and the development of Atari-compatible games by Nintendo. In both cases, the competitors' games were disassembled to obtain access to the functional elements of the program and its interface to the hardware.

<sup>6</sup> The Sydney-based company MFX Research developed MFX2000, a program to fix date-related problems at the object code level. See <http://www.mfxr.com> for details on the company.

intervention can aid in the recovery of source code. The quality of the generated code is normally lesser than that of the original source code—meaningful variable names and comments can be missing. Decompilation techniques were developed in the 1960s when the D-Neliac decompiler<sup>7</sup> was built for the Neliac compiler developed at the Navy Electronic Labs in the United States. This first decompiler was built as a proof of concept; however, in the 1970s, further refinement of this decompilation technology led to commercial uses of the technology as an aid to migrate second generation code to the then new third generation machines.<sup>8</sup> During the 1970s and 1980s, decompilers were used to port programs to newer machines<sup>9</sup> as an aid in the documentation and debugging of programs,<sup>10</sup> and to help recreate lost source code.<sup>11</sup> In the 1990s, decompilers became a reverse engineering tool capable of helping the user with such tasks as checking software for the existence of malicious code,<sup>12</sup> validating compiler-generated code for safety critical systems,<sup>13</sup> recovering lost source to fix the year 2000 bug in operational software,<sup>14</sup> and understanding third party code.<sup>15</sup> After the Java™ language was introduced, decompilers were developed in this language.<sup>16</sup> Java decompilers are more accurate than traditional decompilers for register-based machines because the Java language does not

---

<sup>7</sup> See MAURICE HOWARD HALSTEAD, MACHINE INDEPENDENT COMPUTER PROGRAMMING 143–50 (1962).

<sup>8</sup> See W.L. Caudle, *On the Inverse of Compiling*, The University of Queensland, at <http://www.program-transformation.org/twiki/bin/view/Transform/OnInverseOfCompiling> (April 26, 1980).

<sup>9</sup> See generally W.A. Sassaman, *A Computer Program to Translate Machine Language Into Fortran*, 28 AFIPS SPRING JOINT COMPUTER CONFERENCE 235-39 (1966); Jim Reuter, *DECOMP*, available at <ftp://ftp.cs.washington.edu/pub/decomp.tar.Z> (1988); Caudle, *supra* note 8.

<sup>10</sup> P. Barbe, *The Piler System of Computer Program Translation* (Sept. 1974) (unpublished manuscript, on file with Dr. Cifuentes).

<sup>11</sup> Gregory Littell Hopwood, *Decompilation* (1978) (unpublished Ph.D. dissertation, University of California, Irvine) (on file with the University of California, Irvine Department of Computer Science).

<sup>12</sup> Cristina Cifuentes & K. John Gough, *Decompilation of Binary Programs*, 25 SOFTWARE – PRAC. & EXPERIENCE 811, 812 (July 1995).

<sup>13</sup> D.J. Pavey & L.A. Winsborrow, *Demonstrating Equivalence of Source Code and PROM Contents*, 36 THE COMPUTER J. 654, 654 (1993).

<sup>14</sup> See, e.g., Thomas Hoffman, *Recovery Firm Hot on Heels of Missing Source Code*, COMPUTERWORLD, March 24, 1997, at 6(1).

<sup>15</sup> See, e.g., AHPAH SOFTWARE, INC., SOURCEAGAIN DECOMPILER, available at <http://www.ahpah.com/product.html> (last visited Sept. 28, 2001).

<sup>16</sup> See, e.g., id.; Hanpeter van Vliet, *The Mocha decompiler*, available at <http://www.brouhaha.com/~eric/computers/mocha-b1.zip> (last visited Nov. 29, 2001).

compile its source code to machine code but instead to an intermediate representation known as bytecodes. Bytecodes resemble the assembly language of a stack-based machine. Java bytecode files are information rich since they need to be interpreted in order to run the program on a Java virtual machine.

### 3. Emulation

Emulation is the process of running a program for a machine (the source machine) on another machine (the target machine) by decoding source machine code instructions and simulating the functionality of such instructions on the target machine. The simulation introduces a time overhead as the program to be emulated needs to be decoded at run-time. This causes the program to appear to the user as if the program is running directly on the target computer. Figure 1(c) illustrates the emulation process. The object code file is disassembled, either one instruction at a time or as a series of instructions at a time, into assembly code. Each assembly instruction is then interpreted by the emulator to perform the equivalent functionality of the instruction on the target machine, creating temporarily (but not storing) object code for the target machine. Naïve software emulators can emulate a program 100 times slower than if the program was run on the source machine. However, the use of caching techniques<sup>17</sup> can bring this factor down to 10 or even 4 times slower.<sup>18</sup>

Emulation is also referred to as interpretation. Technically, emulation is performed at the hardware level and interpretation at the software level; however, in practice, both terms are used indistinctly. In the 1970s, emulation was developed for the purposes of terminal emulation; for example, to emulate a mainframe terminal on a PC. In the 1980s and 1990s, emulation was used to aid in the development of new hardware and operating systems.<sup>19</sup> The 1990s have also seen the

---

<sup>17</sup> Caching is the temporary storage in memory of code that has been translated, so the next time the code needs to be interpreted, it is taken from the cache in memory rather than being translated again.

<sup>18</sup> Cathy May, *Mimic: A Fast System/370 Simulator*, 22 ACM SIGPLAN NOTICES 1, 1 (1987).

<sup>19</sup> For example, SPIM is an R2000/R3000 emulator written by John Hennessy and David Patterson that runs on HP-PA, 80386, MIPS, SPARC and 68000 machines. g88 is a Motorola 88000 simulator that runs on 68000, 88000 and SPARC machines. Mimic is an emulator of an IBM S/370 on an IBM RT PC machine. SoftPC by Insignia Solutions is an 80286 MS-DOS emulator that runs on 68000, 88000, MIPS, HP-PA, SPARC, and Alpha machines. Shade by Sun Microsystems is an instruction-set simulator for SPARC version 8 and 9 machines that runs on a SPARC machine.

use of emulators to run an old program on a new machine, such as arcade games that run on now obsolete machines.<sup>20</sup> The time overhead introduced by the translation process at run-time sometimes is counteracted by the fact that the new target machines are normally several factors faster than the old source machines. Therefore, emulated programs may still perform at a reasonable speed. Recently, manufacturers have developed emulators for games machines currently available in the market; for example, Virtual Game Station, an emulator of Sony's PlayStation, and UltraHLE (Ultra High Level Emulator), an emulator for the Nintendo 64 machine.

#### 4. Binary Translation

Binary translation is the process of automatically translating object code from a source machine to a target machine by emitting native machine instructions for the target machine, instead of emulating the source machine instructions. As can be seen from Figure 1(c), the binary translation process involves the disassembly of the source machine's object code into the source machine's assembly code, translation of the source machine's assembly code to functionally equivalent assembly code for the target machine, and encoding of the target machine's assembly code into the object code format of the target machine. As such, binary translation involves reverse and forward engineering techniques, which borrow from the decompilation and compilation processes. The process of binary translation can be performed in two ways: statically or dynamically. Static binary translation involves the creation of a target object code program that can be run on the target machine, normally with the aid of a run-time interpreter for certain pieces of code, for example, the VEST translator.<sup>21</sup> Dynamic binary translation does not require the generation of a target object code program but instead the dynamic execution of the generated object code while the translation is taking place. The process is similar to emulation but has the advantage of generating native code rather than emulating code; for example, Hewlett Packard's translator Aries.<sup>22</sup>

---

<sup>20</sup> Numerous arcade games emulators, available on the Internet, allow users to run a ROM (Read Only Memory) copy of the arcade game on a PC computer. These games are no longer supported by their developers as their operating hardware has become obsolete.

<sup>21</sup> Richard L. Sites et al., *Binary Translation*, COMM. OF THE ACM, Feb. 1993, at 69, 71-75.

<sup>22</sup> See Cindy Zheng & Carol Thompson, *PA-RISC to IA-64: Transparent Execution, No Recompile*, COMPUTER, Mar. 2000, at 47; Bich C. Le, *An Out-Of-Order Execution*

Binary translation is a relatively new area of research. Binary translation has been used in the late 1980s and 1990s to aid in the porting of code to newer machines (e.g., Moxie,<sup>23</sup> VEST and mx,<sup>24</sup> and FX!32). Binary translation was developed as an alternative to emulation because interpretation was considered too slow for the purposes of running application software on other machines. As reported by Digital, FX!32 runs applications five to twenty times faster than these applications would run under emulation.<sup>25</sup> The relative infancy of this technology implies that the techniques are still under development at research labs (e.g. Hewlett Packard<sup>26</sup> and Compaq<sup>27</sup>) and universities (e.g., The University of Queensland<sup>28</sup>).

### C. Reverse Engineering and Security

Professor Edward Felten of Princeton University, an expert advisor in the cases of *United States v. Microsoft Corp.*,<sup>29</sup> *Universal City Studios, Inc. v. Reimerdes*,<sup>30</sup> and *eBay, Inc. v. Bidder's Edge, Inc.*,<sup>31</sup> gave an overview of the vital need for reverse engineering in the areas of security and error correction. He explained that individuals and corporations needed to understand and test what they had been given. In some cases, software may contain glitches or bugs, and reverse engineering provides an opportunity to find these problems. Reverse engineering is "critical for understanding behavior." From the way Professor Felten spoke, one would easily be convinced that reverse engineering is an important aspect of quality

---

*Technique for Runtime Binary Translators*, 8 PROC. OF THE INT'L CONF. ON ARCHITECTURE SUPPORT FOR PROGRAMMING LANGUAGE AND OPERATING SYS. (ACM) 151 (1998).

<sup>23</sup> Posting of John R. Mashey, mash@mash.engr.sgi.com, to Usenet Newsgroup comp.arch (Apr. 18, 1996), available at <http://groups.google.com> (copy on file with the Santa Clara Computer and High Technology Law Journal).

<sup>24</sup> See Sites, *supra* note 21; COMPAQ COMPUTER CORP., ALPHA MIGRATION TOOLS, at <http://www.support.compaq.com/amt/decmigrate> (last visited Sept. 28, 2001).

<sup>25</sup> R.J. Hookway & M.A. Herdeg, *Digital Fx!32: Combining Emulation and Binary Translation*, 9 DIGITAL TECHNICAL J. 3 (1997) (on file with Digital's successor, Compaq Computer Corporation).

<sup>26</sup> See generally Le, *supra* note 22.

<sup>27</sup> See generally Tom Thompson, *An Alpha in PC Clothing*, BYTE, Feb. 1996, at 195-96; COMPAQ COMPUTER CORP., RELEASE NOTES FOR COMPAQ FX!32 V1.5, at <http://www.support.compaq.com/amt/fx32/fx-relnotes.html> (last modified Aug. 18, 1999).

<sup>28</sup> See generally Cristina Cifuentes & Mike Van Emmerik, *UQBT: Adaptable Binary Translation at Low Cost*, COMPUTER, March 2000, at 60.

<sup>29</sup> *United States v. Microsoft Corp.*, 87 F. Supp. 2d 30 (D.D.C. 2000).

<sup>30</sup> *Universal City Studios, Inc. v. Reimerdes*, 111 F. Supp. 2d 294 (S.D.N.Y. 2000).

<sup>31</sup> *eBay, Inc. v. Bidder's Edge Inc.*, 100 F. Supp. 2d 1058 (N.D. Cal. 2000).

assurance. Reverse engineering is essential to understanding the coded structure of programs and their functionality.<sup>32</sup>

Professor Felten also spoke about an instance where reverse engineering was used to alleviate a problem with Netscape's Java applets. An applet is "a program designed to be executed from within another application. Unlike an application, applets cannot be executed directly from the operating system."<sup>33</sup> With the growing popularity of object linking and embedding (OLE), applets are becoming more prevalent. "A well-designed applet can be invoked from many different applications. Web browsers, which are often equipped with Java virtual machines, can interpret applets from Web servers."<sup>34</sup> Because applets are small in file size, cross-platform compatible, and highly secure (because applets cannot be used to access users' hard drives), they are ideal for small Internet applications accessible from a browser. Professor Felten's team reverse engineered the Netscape Java applet code and identified a linking problem in the way different parts of the program were linked together. A graduate student conducted a search on how linking was meant to be done; as there was no theory available, he developed a theory on linking that was then introduced back into Netscape's, Sun's and others' Java implementations.

#### *D. Reverse Engineering Tools and Code Obfuscation*

Paul Martino, from Intertrust, gave specialist commentary on the morning session. He also spoke about crafting licenses for decompilation tools and how such licenses had to pay close attention to legal principle.<sup>35</sup> The panel noted that the DMCA, in arguably legislating out of existence the Sony "substantial noninfringing purposes" defense in relation to circumvention devices,<sup>36</sup> had made the offering of decompilation tools a risky business in respect to the circumvention of digital locks. The rise of software patents has also made it difficult to rely on the ability to reverse engineer provided

---

<sup>32</sup> In response to a question from the audience, Professor Felten explained that merely possessing the source code may in some instances not be adequate and that analysis of the running code was necessary.

<sup>33</sup> INT MEDIA GROUP, INC., WEBOPEDIA, at <http://www.webopedia.com/TERM/a/applet.html> (definition of "applet") (last visited Nov. 9, 2001).

<sup>34</sup> *Id.*

<sup>35</sup> See Sony Computer Entm't, Inc. v. Connectix Corp., 203 F.3d 596 (9th Cir. 2000).

<sup>36</sup> See Universal City Studios, Inc. v. Reimerdes, 111 F. Supp. 2d 294, 322-24 (S.D.N.Y. 2000).

under the fair use doctrine in copyright law as enunciated in *Sega Enterprises Ltd. v. Accolade, Inc.*<sup>37</sup>

The issues of digital rights management and anti-reverse engineering structures also were raised, together with the idea of code obfuscation. Code obfuscation consists of a process by which code contains sufficient decoys to obstruct reverse engineering. Such a definition prompted the following question from the audience: whether code obfuscation could be considered a technological protection measure (TPM) for purposes of the DMCA, such that reverse engineering of this code might be seen as circumventing a TPM. The further question then would be: "What about reverse engineering of the object code; is that circumventing a TPM?"

### *E. Forget Copyright and Explore Patent*

Greg Aharonian, the editor of PatNews, was invited to further comment on the session. He argued that a number of reverse engineering problems in the software area would be reduced, if not obliterated, if we abandoned copyright protection of software and more sensibly applied patent law to software. Aharonian explained that the copyright system does not disclose the details of the item seeking copyright protection. Contrarily, a properly functioning patent system will be educative and enabling to software development. This interesting suggestion raised two immediate responses. First, software copyright is embodied in laws all over the world, including in the international Agreement on Trade-Related Aspects of Intellectual Property Rights,<sup>38</sup> and to change things now would be almost impossible. Second, if denying copyright protection for software means denying that software is expressive, that is too much for many people to accept, especially in light of the ongoing DVD litigation.<sup>39</sup>

---

<sup>37</sup> *Sega Enters. Ltd. v. Accolade, Inc.*, 977 F.2d 1510, 1527–28 (9th Cir. 1992) ("[W]here disassembly is the only way to gain access to the ideas and functional elements embodied in a copyrighted computer program and where there is a legitimate reason for seeking such access, disassembly is a fair use of the copyrighted work, as a matter of law.").

<sup>38</sup> Agreement on Trade-Related Aspects of Intellectual Property Rights, Apr. 15, 1994, Marrakesh Agreement Establishing the World Trade Organization, Annex 1C, Part II, § 1, art. 10(1), LEGAL INSTRUMENTS—RESULTS OF THE URUGUAY ROUND vol. 31, 33 I.L.M. 81 (1994) [hereinafter WTO/TRIPs Treaty or TRIPs].

<sup>39</sup> *Universal City Studios*, 111 F. Supp. 2d at 327 (holding that "as computer code—whether source or object—is a means of expressing ideas, the First Amendment must be considered before its dissemination may be prohibited or regulated. In that sense, computer code is covered, or as is sometimes said, 'protected' by the First Amendment. But that

## II. REVERSE ENGINEERING: THE LEGAL PERSPECTIVE

In opening the legal session, Professor Fitzgerald asked the audience to consider the legal issues that would unfold against a tapestry of four theories of intellectual property. The four theories proposed included: the economic or utilitarian theory where intellectual property law is justified in terms of economic efficiency, the Lockean or labor-desert theory where intellectual property rights are natural rights earned by adding labor to the common resource of information, the personhood theory where intellectual property is an emanation of the person and law should facilitate this personal aspect, and the social planning theory where intellectual property law should be designed to enrich culturally democratic society.<sup>40</sup>

It was suggested that this theoretical backdrop could be used as a framework for analyzing the definition of intellectual property rights concerning software, especially in the context of reverse engineering.

### *A. Reverse Engineering and US Copyright Law*

Fred von Lohman of Morrison and Foerster LLP and the Berkeley Center for Law and Technology presented on the issue of the intersection between copyright law and reverse engineering of software. He explained that from the outset there might be a number of legal obstacles to the reverse engineering of software including:

1. traditional copyright law, as intermediate copying (at least) is inherent in the process of looking at running software and therefore most reverse engineering;
2. anti-circumvention law now embodied in section 1201 of the DMCA – where the reverse engineering involves circumvention of a TPM;
3. contract law may involve infringing a license agreement which prohibits reverse engineering, although the validity of such license terms is largely untested (see also UCITA);
4. patent law as once again to look at patented software one normally needs to make a copy, although arguments concerning fair or experimental use are resurfacing;

---

conclusion leaves for determination the level of scrutiny to be applied in determining the constitutionality of regulation of computer code.”).

<sup>40</sup> See generally William Fisher, *Theories of Intellectual Property*, in *NEW ESSAYS IN THE LEGAL AND POLITICAL THEORY OF PROPERTY* 168 (Stephen R. Munzer ed., 2001), available at [www.law.harvard.edu/Academic\\_Affairs/coursepages/tfisher/iptheory.html](http://www.law.harvard.edu/Academic_Affairs/coursepages/tfisher/iptheory.html) (last visited Sept. 4, 2001).

5. trademark law;
6. Computer Fraud and Abuse Act (CFAA);
7. Electronic Communications Privacy Act (ECPA).

Von Lohman proceeded to outline the ambit of fair use privileges to reverse engineer espoused in *Sega Enterprises Ltd. v. Accolade, Inc.* and *Atari Games Corp. v. Nintendo of America, Inc.*, and recently confirmed in *Sony Computer Entertainment, Inc. v. Connectix Corp.*<sup>41</sup>

In *Sony Computer Entertainment, Inc. v. Connectix Corp.*, the Court of Appeals for the Ninth Circuit held firmly in favor of the reverse engineering of software in order to allow different software products to be ported or joined (i.e. to interoperate) with different hardware, firmware or software platforms.<sup>42</sup> In this case, Justice Canby, writing for the court, explained the intermediate copying<sup>43</sup> that occurred in this instance was legitimate because it merely facilitated the copying of the unprotected (i.e., non-copyrighted) function or idea of the software.<sup>44</sup> Interestingly, the existence of a patent would have made things more difficult for Connectix.<sup>45</sup> However, the court clearly stated that intermediate copying was allowable where used to determine function that could facilitate the making of transformative and better or extended products.<sup>46</sup>

In this case, Connectix had developed software that emulated the Sony PlayStation on a personal computer. In other words,

---

<sup>41</sup> *Sega Enters. Ltd. v. Accolade, Inc.*, 977 F.2d 1510 (9th Cir. 1992); *Atari Games Corp. v. Nintendo of America, Inc.*, 975 F.2d 832 (Fed. Cir. 1992); *Sony Computer Entm't, Inc. v. Connectix Corp.*, 203 F.3d 596 (9th Cir. 2000). See also HOWARD C. ANAWALT & ELIZABETH ENAYATI POWERS, *IP STRATEGY, COMPLETE INTELLECTUAL PROPERTY PLANNING, ACCESS AND PROTECTION* (2001).

<sup>42</sup> See generally *Sony Computer Entm't, Inc.*, 203 F.3d at 599–601 (providing a very good overview of reverse engineering).

<sup>43</sup> *Id.* at 599 (defining intermediate copying as that which is undertaken in the process of developing the final product which itself may not be a copy).

<sup>44</sup> [I]n the case of computer programs, the idea/expression distinction poses distinct "unique problems" because computer programs are "in essence, utilitarian articles—articles that accomplish tasks. As such, they contain many logical, structural, and visual display elements that are dictated by the function to be performed, by considerations of efficiency, or by external factors such as compatibility requirements and industry demands." . . . [T]he fair use doctrine preserves public access to the ideas and functional elements embedded in copyrighted computer software programs.

*Id.* at 603 (citation omitted).

<sup>45</sup> After rejection of its copyright action, Sony sued Connectix for patent infringement. *Sony Computer Entm't, Inc. v. Connectix Corp.*, No. 00-520 (N.D. Cal. filed Feb. 14, 2000).

<sup>46</sup> *Sony Computer Entm't, Inc.*, 203 F.3d at 602–03, 606.

Connectix's Virtual Game Station allowed consumers to play Sony games on their personal computer and not just on the Sony PlayStation console. This allowed portability which the court also seemed to support in its discussion of transformative use: "The product creates a new platform, the personal computer, on which consumers can play games designed for the Sony PlayStation. This innovation affords opportunities for game play in new environments . . ."<sup>47</sup> The court was not concerned with the number of intermediate copies that had been made, nor that the end product (which contained different source code but performed the same function) was a competitor of the Sony PlayStation.<sup>48</sup>

Von Lohman then discussed the DMCA and explained that section 1201 could be paraphrased in crude terms as "thou shalt not circumvent," and "do not break into my castle and do not violate my house rules—seen from the perspective of a copyright holder." However, he was quick to note that there are exceptions to this regime as listed in sections 1201(d) through (j); especially 1201(f) concerning reverse engineering, 1201(g) relating to encryption research, and 1201(j) discussing security testing.

Section 1201 provides in part:

§ 1201. Circumvention of Copyright Protection Systems.

(a) Violations Regarding Circumvention of Technological Measures.

(1) . . . .

(2) No person shall manufacture, import, offer to the public, provide, or otherwise traffic in any technology, product, service, device, component, or part thereof, that

(A) is primarily designed or produced for the purpose of circumventing a technological measure that effectively controls access to a work protected under this title;

---

<sup>47</sup> *Id.* at 606.

<sup>48</sup> *Id.* at 604-07. See also Maureen A. O'Rourke, *Towards a Doctrine of Fair Use in Patents Law*, 100 COLUM. L. REV. 1177, 1213, 1223 (2000) (proposing the term "horizontal" interoperability to describe this situation).

(B) has only limited commercially significant purpose or use other than to circumvent a technological measure that effectively controls access to a work protected under this title; or

(C) is marketed by that person or another acting in concert with that person with that person's knowledge for use in circumventing a technological measure that effectively controls access to a work protected under this title.

(b) Additional Violations.

(1) No person shall manufacture, import, offer to the public, provide, or otherwise traffic in any technology, product, service, device, component, or part thereof, that

(A) is primarily designed or produced for the purpose of circumventing protection afforded by a technological measure that effectively protects a right of a copyright owner under this title in a work or a portion thereof;

(B) has only limited commercially significant purpose or use other than to circumvent protection afforded by a technological measure that effectively protects a right of a copyright owner under this title in a work or a portion thereof; or

(C) is marketed by that person or another acting in concert with that person with that person's knowledge for use in circumventing protection afforded by a technological measure that effectively protects a right of a copyright owner under this title in a work or a portion thereof.

Pursuant to section 1201(a)(1), the DMCA, which took effect on October 28, 2000, also prohibits actual circumvention of an access control (as opposed to a copy).<sup>49</sup> The DMCA is subject to exceptions

---

<sup>49</sup> § 1201(a) Violations Regarding Circumvention of Technological Measures.

(1)

(A) No person shall circumvent a technological measure that effectively controls access to a work protected under this title. The prohibition contained in the preceding sentence shall take effect at the end of the 2-year period beginning on the date of the enactment of this chapter.

(B) The prohibition contained in subparagraph (A) shall not apply to persons who are users of a copyrighted work which is in a particular class of works, if such persons are, or are likely to be in the succeeding 3-year period, adversely affected by virtue of such prohibition in their ability to make noninfringing uses of that particular class of works under this title, as determined under subparagraph (C).

(C) During the 2-year period described in subparagraph (A), and during each succeeding 3-year period, the Librarian of Congress, upon the recommendation of the Register of Copyrights, who shall consult with the Assistant Secretary for Communications and Information of the Department of Commerce and report and comment on his or her views in making such recommendation, shall make the determination in a rulemaking proceeding for purposes of subparagraph (B) of whether persons who are users of a copyrighted work are, or are likely to be in the succeeding 3-year period, adversely affected by the prohibition under subparagraph (A) in their ability to make noninfringing uses under this title of a particular class of copyrighted works. In conducting such rulemaking, the Librarian shall examine—

- (i) the availability for use of copyrighted works;
- (ii) the availability for use of works for nonprofit archival, preservation, and educational purposes;
- (iii) the impact that the prohibition on the circumvention of technological measures applied to copyrighted works has on criticism, comment, news reporting, teaching, scholarship, or research;
- (iv) the effect of circumvention of technological measures on the market for or value of copyrighted works; and
- (v) such other factors as the Librarian considers appropriate.

(D) The Librarian shall publish any class of copyrighted works for which the Librarian has determined, pursuant to the rulemaking conducted under subparagraph (C), that noninfringing uses by persons who are users of a copyrighted work are, or are likely to be, adversely affected, and the prohibition contained in subparagraph (A) shall not apply to such users with respect to such class of works for the ensuing 3-year period.

(E) Neither the exception under subparagraph (B) from the applicability of the prohibition contained in subparagraph (A), nor any determination made in a rulemaking conducted under

promulgated by the Library of Congress, under section 1201(a)(1)(C), which relate to two types of works and apply for a three-year period until October 28, 2003. The first type of works consists of compilations made up of lists of websites blocked by filtering software applications and literary works. The second type of works includes computer programs and databases protected by access control mechanisms that fail to permit access because of malfunction, damage or obsolescence.

Under section 1201(a)(2), “[n]o person shall manufacture, import, offer to the public, provide, or otherwise traffic in any technology, product, service, device, component, or part thereof” designed to circumvent a TPM in the form of an access control. Under section 1201(b)(1), “[n]o person shall manufacture, import, offer to the public, provide, or otherwise traffic in any technology, product, service, device, component, or part thereof” designed to circumvent a copy control. Von Lohman emphasized the wording “part thereof,” and explained that the provision could not be avoided by individuals selling or providing different portions of a circumvention device. As noted above, the foregoing provisions are subject to exceptions that permit, in specified circumstances, circumvention for reasons of reverse engineering, for interoperability purposes under section 1201(f), for encryption research under section 1201(g), and for security testing under section 1201(j).

Section 1201(a)(2) of the DMCA was recently at issue in the case of *Universal City Studios, Inc. v. Reimerdes*.<sup>50</sup> This case involved a fifteen-year-old adolescent who allegedly cracked the encryption code (CSS)<sup>51</sup> of the software lock (a TPM)<sup>52</sup> employed (it

---

subparagraph (C), may be used as a defense in any action to enforce any provision of this title other than this paragraph.

Digital Millennium Copyright Act, 17 U.S.C. § 1201 (2000).

<sup>50</sup> *Universal City Studios, Inc. v. Reimerdes*, 82 F. Supp. 2d 211 (S.D.N.Y. 2000).

<sup>51</sup> CSS is a technological measure that effectively controls access to plaintiffs’ copyrighted movies because it requires the application of information or a process, with the authority of the copyright owner, to gain access to those works. Indeed, defendants conceded in their memorandum that one cannot in the ordinary course gain access to the copyrighted works on plaintiffs’ DVDs without a “player key” issued by the DVDCCA that permits unscrambling the contents of the disks. It is undisputed also that DeCSS defeats CSS and decrypts copyrighted works without the authority of the copyright owners. As there is no evidence of any commercially significant purpose of DeCSS other than circumvention of CSS, defendants’ actions likely violated Section 1201(a)(2)(B).

*Id.* at 216–17.

<sup>52</sup> DVDs contain motion pictures in digital form, which presents an enhanced risk of unauthorized reproduction and distribution because digital copies

was argued) to prevent easy copying of the content of Digital Versatile Discs (DVDs).<sup>53</sup> This action was brought under the DMCA's provision<sup>54</sup> prohibiting the offering, providing or otherwise trafficking of a device for circumventing a TPM. In his interim and final judgment,<sup>55</sup> Judge Kaplan held that the reverse engineering exception could not be invoked.<sup>56</sup>

---

made from DVDs do not degrade from generation to generation. Concerned about this risk, motion picture companies, including plaintiffs, insisted upon the development of an access control and copy prevention system to inhibit the unauthorized reproduction and distribution of motion pictures before they released films in the DVD format. The means now in use, Content Scramble System or CSS, is an encryption-based security and authentication system that requires the use of appropriately configured hardware such as a DVD player or a computer DVD drive to decrypt, unscramble and play back, but not copy, motion pictures on DVDs. CSS has been licensed to hundreds of DVD player manufacturers and DVD content distributors in the United States and around the world.

*Id.* at 214.

<sup>53</sup> "DVDs are five-inch wide discs that, in this application, hold full-length motion pictures. They are the latest technology for private home viewing of recorded motion pictures. This technology drastically improves the clarity and overall quality of a motion picture shown on a television or computer screen." *Id.*

<sup>54</sup> Digital Millennium Copyright Act, 17 U.S.C. § 1201 (2000).

<sup>55</sup> *Universal City Studios, Inc. v. Reimerdes*, 111 F. Supp. 2d 294 (S.D.N.Y. 2000) (oral hearing on the merits of this case concluded on July 25, 2000 and final judgment (in large part confirming the reasoning of the interim judgment) was handed down on August 17, 2000).

<sup>56</sup> § 1201(f) Reverse Engineering.

(1) Notwithstanding the provisions of subsection (a)(1)(A), a person who has lawfully obtained the right to use a copy of a computer program may circumvent a technological measure that effectively controls access to a particular portion of that program for the sole purpose of identifying and analyzing those elements of the program that are necessary to achieve interoperability of an independently created computer program with other programs, and that have not previously been readily available to the person engaging in the circumvention, to the extent any such acts of identification and analysis do not constitute infringement under this title.

(2) Notwithstanding the provisions of subsections (a)(2) and (b), a person may develop and employ technological means to circumvent a technological measure, or to circumvent protection afforded by a technological measure, in order to enable the identification and analysis under paragraph (1), or for the purpose of enabling interoperability of an independently created computer program with other programs, if such means are necessary to achieve such interoperability, to the extent that doing so does not constitute infringement under this title.

(3) The information acquired through the acts permitted under paragraph (1), and the means permitted under paragraph (2), may be made available to others if the person referred to in paragraph (1) or (2), as the case may be, provides such information or means solely for the purpose of enabling interoperability of an independently created computer program with other programs, and to the extent that doing so does not constitute infringement under this title or violate applicable law other than this section.

In this case, the TPM was CSS and the circumvention device was DeCSS. In the final instance, the defendant was alleged to have been linking to Web sites that allowed downloading of DeCSS.

The defendants argued that reverse engineering of the software lock was needed in order to play DVDs on other platforms such as Linux, an open code operating system. They argued that DeCSS was necessary to achieve interoperability between computers running on the Linux system and DVDs.<sup>57</sup> Therefore, the interoperability exception in the DMCA was enlivened. The judge explained that he could not accept such an argument for three reasons:

First, defendants have offered no evidence to support this assertion. Second, even assuming that DeCSS runs under Linux, it concededly runs under Windows—a far more widely used operating system—as well. It therefore cannot reasonably be said that DeCSS was developed “for the sole purpose” of achieving interoperability between Linux and DVDs. Finally, and most important, the legislative history makes it abundantly clear that Section 1201(f) permits reverse engineering of copyrighted computer programs only and does not authorize circumvention of technological systems that control access to other copyrighted works, such as movies.<sup>58</sup>

The defendants also argued they engaged in a fair use under Section 107 of the Copyright Act. This argument was also rejected by the judge, who explained:

Section 107 of the Act provides in critical part that certain uses of copyrighted works that otherwise would be wrongful are “not . . . infringement[s] of copyright.” Defendants, however, are not here sued for copyright infringement. They are sued for offering to the public and providing technology primarily designed to circumvent technological measures that control access to copyrighted works

---

(4) For purposes of this subsection, the term “interoperability” means the ability of computer programs to exchange information, and of such programs mutually to use the information which has been exchanged.

Digital Millennium Copyright Act, 17 U.S.C. § 1201 (2000).

<sup>57</sup> But see O’Rourke, *supra* note 48, at 1213, 1219–23 (defining the term “vertical” interoperability, which may describe this situation).

<sup>58</sup> Universal City Studios, Inc. v. Reimerdes, 82 F. Supp. 2d 211, 218 (S.D.N.Y. 2000), *aff’d*, 111 F. Supp. 2d at 320 (S.D.N.Y. 2000).

and otherwise violating Section 1201(a)(2) of the Act. If Congress had meant the fair use defense to apply to such actions, it would have said so.<sup>59</sup>

A crucial issue arising from such reasoning is whether the DMCA creates an exclusive right in the person or entity setting the TPM to control access to copyrighted works. Such a right would allow a person or entity to protect raw data, embodied in a copyrighted work, or to prevent fair use of copyrighted material by encasing such material in a TPM. Some suggest this is akin to locking a book or a database in a room or bank vault.<sup>60</sup> In *Universal City Studios, Inc. v. Reimerdes*, the court, while acknowledging some limitations, intimated that the DMCA creates a right to control access to copyrighted works in a way that is not unconstitutional under the First Amendment, because such control has only an incidental impact on protected expression which is acceptable in light of the overall objective of Congress to protect copyright in the digital environment.<sup>61</sup> It might be suggested that such an approach serves to eliminate fair use and to create rights in unprotected data, and portrays over-broad regulation.<sup>62</sup> An even more pressing point to appreciate is that fair use rights to reverse engineer software architecture in the name of interoperability, where that software is a

---

<sup>59</sup> *Id.* at 219. See also Digital Millennium Copyright Act, 17 U.S.C. § 1201(c)(1) ("Nothing in this section shall affect rights, remedies, limitations, or defenses to copyright infringement, including fair use, under this title."); Kamiel J. Koelman & Natali Helberger, *Protection of Technological Measures*, in COPYRIGHT AND ELECTRONIC COMMERCE: LEGAL ASPECTS OF ELECTRONIC COPYRIGHT MANAGEMENT 165 (P. Bernt Hugenholtz ed., 2000).

<sup>60</sup> See Pamela Samuelson, *Intellectual Property and the Digital Economy: Why the Anti-Circumvention Regulations Need to be Revised*, 14 BERKELEY TECH. L. J. 519, 539–43 (1999) (providing a critical appraisal of this issue).

<sup>61</sup> *Universal City Studios, Inc.*, 111 F. Supp. 2d at 322, 325–41. See also Council Directive 2001/29 of 22 May 2001 on the harmonisation of certain aspects of copyright and related rights in the information society, art. 6(1), 2001 O.J. (L 167) 10 (prohibiting actual circumvention), available at [http://europa.eu.int/eur-lex/en/lif/dat/2001/en\\_301L0029.html](http://europa.eu.int/eur-lex/en/lif/dat/2001/en_301L0029.html) (last updated July 2, 2001). Cf. Copyright Amendment (Digital Agenda) Act 2000 (Austl.) (expressly prohibiting actual circumvention), available at <http://scaletext.law.gov.au/html/comact/10/6223/top.htm> (last visited Dec. 1, 2001) [hereinafter Digital Agenda Act].

<sup>62</sup> Should such legislation be allowed to protect non-copyrightable data? See generally Marci A. Hamilton, *Database Protection and the Circuitous Route Around the United States Constitution*, in INTERNATIONAL INTELLECTUAL PROPERTY AND THE COMMON LAW WORLD 9 (Charles Rickett & G. Austin, eds., 2000); Yochai Benkler, *Free as the Air to Common Use: First Amendment Constraints on Enclosure of the Public Domain*, 74 N.Y.U. L. REV. 354 (1999); William Patry, *The Enumerated Powers Doctrine and Intellectual Property: An Imminent Constitutional Collision*, 67 GEO. WASH. L. REV. 359 (1999); J.H. Reichman & Pamela Samuelson, *Intellectual Property Rights in Data?*, 50 VAND. L. REV. 51 (1997).

TPM or lock, are severely restricted by the DMCA as interpreted in this case.

Two further points arising in the case and clarifying the meaning of the DMCA must be noted. First, in the process of arguing fair use, the defendants raised the *Sony* defense. The judge responded by saying:

Defendants claim also that the possibility that DeCSS might be used for the purpose of gaining access to copyrighted works in order to make fair use of those works saves them under *Sony Corp. v. Universal City Studios, Inc.* But they are mistaken. *Sony* does not apply to the activities with which defendants here are charged. Even if it did, it would not govern here. *Sony* involved a construction of the Copyright Act that has been overruled by the later enactment of the DMCA to the extent of any inconsistency between *Sony* and the new statute.

*Sony* was a suit for contributory infringement brought against manufacturers of videocassette recorders on the theory that the manufacturers were contributing to infringing home taping of copyrighted television broadcasts. The Supreme Court held that the manufacturers were not liable in view of the substantial numbers of copyright holders who either had authorized or did not object to such taping by viewers. But *Sony* has no application here.

When *Sony* was decided, the only question was whether the manufacturers could be held liable for infringement by those who purchased equipment from them in circumstances in which there were many non-infringing uses for their equipment. But that is not the question now before this Court. The question here is whether the possibility of non-infringing fair use by someone who gains access to a protected copyrighted work through a circumvention technology distributed by the defendants saves the defendants from liability under Section 1201. But nothing in Section 1201 so suggests. By prohibiting the provision of circumvention technology, the DMCA fundamentally altered the landscape. A given

device or piece of technology might have “a substantial noninfringing use, and hence be immune from attack under *Sony*’s construction of the Copyright Act—but nonetheless still be subject to suppression under Section 1201.” Indeed, Congress explicitly noted that Section 1201 does not incorporate *Sony*.

. . . The fact that Congress elected to leave technologically unsophisticated persons who wish to make fair use of encrypted copyrighted works without the technical means of doing so is a matter for Congress unless Congress’ decision contravenes the Constitution, a matter to which the Court turns below.<sup>63</sup>

Second, the judge held that linking was a form of offering, providing or otherwise trafficking in a circumvention device:

To “provide” something, in the sense used in the statute, is to make it available or furnish it. To “offer” is to present or hold it out for consideration. The phrase “or otherwise traffic in” modifies and gives meaning to the words “offer” and “provide.” In consequence, the anti-trafficking provision of the DMCA is implicated where one presents, holds out or makes a circumvention technology or device available, knowing its nature, for the purpose of allowing others to acquire it.

To the extent that defendants have linked to sites that automatically commence the process of downloading DeCSS upon a user being transferred by defendants’ hyperlinks, there can be no serious question. Defendants are engaged in the functional equivalent of transferring the DeCSS code to the user themselves.<sup>64</sup>

In summary, the DMCA provides an exception for circumventing a TPM in order to look at a program to identify and analyze those elements necessary to achieve interoperability in the name of reverse engineering. However, circumvention is allowed only to the extent permitted by copyright law. This situation occurs most commonly where a right to reverse engineer is available under the fair use doctrine. In this context, von Lohman pointed out two

---

<sup>63</sup> *Universal City Studios, Inc.*, 111 F. Supp. 2d at 323–24 (citations omitted).

<sup>64</sup> *Id.* at 325.

crucial issues in determining liability under the DMCA in cases of reverse engineering: whether fair use has occurred and whether a TPM was involved.<sup>65</sup> He listed the following factors as being indicative of a finding of fair use: the code contained unprotected functional elements, disassembly was necessary to get to the unprotected elements, only intermediate copying (as opposed to incorporating code) was involved, indirect competition, the new product does not supplant the original, and the transformative use of the information.

In relation to encryption research and security testing, the DMCA exceptions do not expressly permit copying for reverse engineering purposes. It will be necessary, in both cases, to take a further step and establish a fair use defense in order to be free from liability for reproduction as opposed to circumvention.<sup>66</sup>

### *B. Reverse Engineering in Australia*

Anne Fitzgerald, a technology lawyer in Australia, explained the recent reforms in Australia concerning the reverse engineering of software. The Copyright Amendment (Computer Programs) Act 1999 (the "Computer Programs Act") inserts a new Division 4A, entitled "Acts not constituting infringements of copyright in computer programs," into the Australian Copyright Act of 1968.<sup>67</sup> Further, amendments to Division 4A have been affected by the Copyright Amendment (Digital Agenda) Act 2000 (the "Digital Agenda Act").<sup>68</sup> Division 4A also contains some important provisions of general application which should be noted.

Section 47H expressly provides that an agreement or a provision in an agreement that excludes or limits (or has the effect of excluding or limiting) the operation of the new exceptions relating to back-up copying and reverse engineering<sup>69</sup> has no effect.<sup>70</sup> By nullifying

---

<sup>65</sup> See generally *RealNetworks, Inc. v. Streambox, Inc.*, 2000 WL 127311 (W.D. Wash. 2000); *Sony Computer Entm't Am., Inc. v. Gamemasters*, 87 F. Supp. 2d 976 (N.D. Cal. 1999).

<sup>66</sup> See also Digital Millennium Copyright Act, 17 U.S.C. §§ 1201(g)(2)(D), (j)(2) (2000).

<sup>67</sup> Copyright Amendment (Computer Programs) Act 1999 (Austl.) (codified as amended at Copyright Act 1968 Part III §§ 47B(3) and (4), 47D, 47E, 47F (Austl.)), available at [http://www.austlii.edu.au/au/legis/cth/num\\_act/capa1999415](http://www.austlii.edu.au/au/legis/cth/num_act/capa1999415).

<sup>68</sup> Digital Agenda Act (Austl.), available at <http://scaletext.law.gov.au/html/comact/10/6223/top.htm> (Sep. 4, 2000).

<sup>69</sup> *Id.* § 47H.

<sup>70</sup> *Id.* Compare Council Directive on the Legal Protection of Computer Programs, stating that:

The provisions of this Directive shall be without prejudice to any other legal provisions such as those concerning patent rights, trade-marks,

agreements that purport to exclude the new exceptions, section 47H clarifies that the copyright owner cannot exempt itself from operation of these exceptions by requiring licensees to accept contractual terms that exclude the exceptions.

Where a computer program is reproduced or adapted under any of the Division 4A exceptions, the reproduction or adaptation of the program and any information derived from it are not to be used, sold, or otherwise supplied to others except for the purposes specified in the exceptions. Where the reproduction or adaptation of the computer program is used for extraneous purposes and without the consent of the copyright owner, the exceptions in Division 4A will not apply.<sup>71</sup>

Each of the Division 4A exceptions is subject to the requirement that the relevant act—whether it is running the computer program, or making a reproduction or an adaptation of the computer program—has been “done by, or on behalf of, the owner or licensee of the copy.”<sup>72</sup> It is arguable that this requirement introduces an unwarranted restriction on the Division 4A exceptions, which will limit their application. The requirement certainly causes substantial problems in relation to the specific exception for security testing and severely restricts its operation.<sup>73</sup>

---

unfair competition, trade secrets, protection of semi-conductor products or the law of contract. Any contractual provisions contrary to Article 6 or to the exceptions provided for in Article 5(2) and (3) shall be null and void.

Council Directive 91/250 on the legal protection of computer programs, art. 9(1), 1990 O.J. (L 122) 42, available at [http://europa.eu.int/eur-lex/en/lif/dat/1991/en\\_391L0250.html](http://europa.eu.int/eur-lex/en/lif/dat/1991/en_391L0250.html) [hereinafter EU Software Directive].

<sup>71</sup> Copyright Act 1968, § 47G (Austl.).

<sup>72</sup> *Id.* §§ 47B(1)(b), (3)(b), 47C(1)(a), 47C(2)(a), 47D(1)(a), 47E(1)(a) and 47F(1)(a).

<sup>73</sup> See EU Software Directive, *supra* note 70, art. 5 (referring to the “lawful acquirer,” which some States have implemented as “lawful user,” i.e. a person having the right to use the program). See also Report from the Commission to the Council, the European Parliament, and the Economic and Social Committee on the Implementation and Effects of Directive 91/250/EEC on the Legal Protection of Computer Programs, COM(2000)199 final at 12, for a statement that the Commission:

shares the view of some commentators that “lawful acquirer” did in fact mean a purchaser, licensee, renter or a person authorized to use the program on behalf of one of the above. . . . In the view of the Commission, what was intended by Article 5(1) and recital 18 was that it should not be possible to prevent by contract a “lawful acquirer,” of a program doing any of the restricted acts that were required for the use of the program in accordance with its intended purpose or for correcting errors. It is, however, possible for a contract to include specific provisions that “control” the restricted acts which may be carried out by the user of the computer program.

The Division 4A exceptions do not apply where the copy of the computer program being reproduced is an infringing copy.<sup>74</sup> The Division 4A exceptions pursuant to sections 116A and 132 of the Copyright Act are “permitted purposes” which can be the basis for exemption from the new civil and criminal prohibitions on dealing in circumvention devices.

## 1. The Reverse Engineering Exceptions

### *a. Reverse engineering for purposes of interoperability*<sup>75</sup>

The new section 47D, inserted by the Computer Programs Act, deals with reverse engineering for purposes of interoperability.<sup>76</sup> It exempts from infringement the reproduction or adaptation of a computer program for the purpose of obtaining information necessary to enable the owner, the licensee, or a person acting on behalf of the owner or licensee of the original program, to independently make a

---

The UK implementation of Article 6 of the Directive refers to “lawful user” rather than “a person authorized on behalf of the licensee or person having a right to use a copy of the program.” EU Software Directive, *supra* note 70, at ¶ 6.

<sup>74</sup> Copyright Act 1968, §§ 47B(2)(a) and (4), 47C(4), 47D(2), 47E(2), 47F(2) (Austl.).

<sup>75</sup> See Anne Fitzgerald & Cristina Cifuentes, *Interoperability and Computer Software Protection in Australia*, 4 COMPUTER & TELECOMM. L. REV. 271 (1998).

<sup>76</sup> (1) Subject to this Division, the copyright in a literary work that is a computer program is not infringed by the making of a reproduction or adaptation of the work if:

- (a) the reproduction or adaptation is made by, or on behalf of, the owner or licensee of the copy of the program (the *original program*) used for making the reproduction or adaptation; and
- (b) the reproduction or adaptation is made for the purpose of obtaining information necessary to enable the owner or licensee, or a person acting on behalf of the owner or licensee, to make independently another program (the *new program*), or an article, to connect to and be used together with, or otherwise to interoperate with, the original program or any other program; and
- (c) the reproduction or adaptation is made only to the extent reasonably necessary to obtain the information referred to in paragraph (b); and
- (d) to the extent that the new program reproduces or adapts the original program, it does so only to the extent necessary to enable the new program to connect to and be used together with, or otherwise to interoperate with, the original program or the other program; and
- (e) the information referred to in paragraph (b) is not readily available to the owner or licensee from another source when the reproduction or adaptation is made.

Copyright Act 1968, § 47D(1) (Austl.).

new program or device to connect to and be used together with, or otherwise to interoperate with, the original program or any other program.<sup>77</sup> The reproduction or adaptation must be made only to the extent reasonably necessary to obtain the required information,<sup>78</sup> and the information must not be readily available to the owner or licensee of the software from any other source at that time.<sup>79</sup> Any new program that reproduces or adapts the original program is to do so only to "the extent necessary to enable the new program to connect to and be used together with, or otherwise interoperate with, the original program or the other program."<sup>80</sup>

*b. Error correction*

Permitting the user of a computer program to reverse engineer it to correct errors is not difficult to justify, particularly where the owner is not able to fix the errors within a reasonable time or at a reasonable price, or that owner has gone out of business. The need for software users to have rights to correct errors has been recognized for some time, as is demonstrated by the inclusion of error correction provisions in the 1991 European Council Directive on the Legal Protection of Computer Programs (the "EU Software Directive").<sup>81</sup> Article 5(1) of the Directive provides that

[i]n the absence of specific contractual provisions, the acts referred to in Article 4(a) and (b) [the exclusive acts of the rightholder] shall not require authorization by the rightholder where they are necessary for the use of the computer program by the lawful acquirer in accordance with its intended purpose, including for error correction.<sup>82</sup>

Under the Computer Programs Act, copyright is not infringed by the reproduction or adaptation of a computer program for the purposes of correcting an error in the original program, which prevents it from operating in conjunction with other software or hardware,<sup>83</sup> as intended by its author, or in accordance with the

---

<sup>77</sup> *Id.* § 47D(1)(b).

<sup>78</sup> *Id.* § 47D(1)(c).

<sup>79</sup> *Id.* § 47D(1)(e).

<sup>80</sup> *Id.* § 47D(1)(d).

<sup>81</sup> EU Software Directive, *supra* note 70.

<sup>82</sup> *Id.* art. 5(1).

<sup>83</sup> (1) Subject to this Division, the copyright in a literary work that is a computer program is not infringed by the making, on or after 23 February 1999, of a reproduction or adaptation of the work if:

specifications or other accompanying documentation supplied with the original copy.<sup>84</sup> According to the Explanatory Memorandum, this provision would not extend to adapting the original software to run on a platform for which it was not designed.<sup>85</sup>

Reproduction or adaptation is permitted only where reasonably necessary to correct the error. Furthermore, a correctly functioning copy must not be available to the owner or licensee within a reasonable time at an ordinary commercial price.<sup>86</sup>

*c. Computer security testing*

Security testing is becoming increasingly important for ensuring the security of computers and computer networks in the global networked environment of the Internet. On a day-to-day basis, security testing organizations are required to examine pirated software, software developed by a recognized software vendor which has been modified by an intruder to fulfill some other purpose (a Trojan horse), and software developed by an intruder or hacker to exploit a vulnerability in a computer program or system (an attack tool, such as a computer virus).

---

(a) the reproduction or adaptation is made by, or on behalf of, the owner or licensee of the copy of the program (the *original copy*) used for making the reproduction or adaptation; and

(b) the reproduction or adaptation is made for the purpose of correcting an error in the original copy that prevents it from operating (including in conjunction with other programs or with hardware):

(i) as intended by its author; or

(ii) in accordance with any specifications or other documentation supplied with the original copy; and

(c) the reproduction or adaptation is made only to the extent reasonably necessary to correct the error referred to in paragraph (b); and

(d) when the reproduction or adaptation is made, another copy of the program that does operate as mentioned in paragraph (b) is not available to the owner or licensee within a reasonable time at an ordinary commercial price.

Copyright Act 1968, § 47E(1) (Austl.).

<sup>84</sup> *Id.* § 47E(1)(b).

<sup>85</sup> Copyright Amendment (Computer Programs) Bill 1999, Explanatory Memorandum, ¶ 17 (Austl.), available at <http://scaleplus.law.gov.au/html/ems/0/1999/0/0642392951.htm> (last visited Dec. 1, 2001).

<sup>86</sup> Copyright Act 1968, § 47E(1)(c)-(d) (Austl.).

Section 47F, inserted into the Copyright Act 1968 by the Computer Programs Act,<sup>87</sup> introduced an exception for security testing. It provides that copyright in a computer program is not infringed by the making of a reproduction or an adaptation of the program for the purpose of: testing, in good faith, the security of the original copy, or a computer system or network of which the original copy is a part, or investigating, or correcting, in good faith, a security flaw in or the vulnerability to unauthorized access of the original copy, or of a computer system or network of which the original copy is a part. The reproduction or adaptation is not to exceed what is reasonably necessary to achieve one of these purposes<sup>88</sup> and is only permitted where the information obtained is not otherwise available to the owner or licensee of the software from another source at that time.<sup>89</sup>

While these provisions restrict the usefulness of the exception created by section 47F, its effect is almost negated by the requirements that the reproduction or adaptation be made by or on behalf of the owner or licensee of the copy of the program used to make the reproduction or adaptation<sup>90</sup> and that it cannot be done from an infringing copy of the computer program.<sup>91</sup> The problem that arises is that in many cases security testing must be carried out on

---

<sup>87</sup>

(1) Subject to this Division, the copyright in a literary work that is a computer program is not infringed by the making of a reproduction or adaptation of the work if:

(a) the reproduction or adaptation is made by, or on behalf of, the owner or licensee of the copy of the program (the *original copy*) used for making the reproduction or adaptation; and

(b) the reproduction or adaptation is made for the purpose of:

(i) testing in good faith the security of the original copy, or of a computer system or network of which the original copy is a part; or

(ii) investigating, or correcting, in good faith a security flaw in, or the vulnerability to unauthorised access of, the original copy, or of a computer system or network of which the original copy is a part; and

(c) the reproduction or adaptation is made only to the extent reasonably necessary to achieve a purpose referred to in paragraph (b); and

(d) the information resulting from the making of the reproduction or adaptation is not readily available to the owner or licensee from another source when the reproduction or adaptation is made.

*Id.* § 47(F)(1).

<sup>88</sup> *Id.* § 47F(1)(c).

<sup>89</sup> *Id.* § 47F(d).

<sup>90</sup> *Id.* § 47F(1)(a).

<sup>91</sup> *Id.* § 47F(2).

infringing copies of programs, but it is often not known whether a program is an infringing copy until after the event. In a typical situation, it is not possible for a security testing organization to know what kind of software it is looking at until it embarks upon the analysis. Where a security testing organization is examining a Trojan horse or an attack tool, it may have to reproduce or adapt the computer program contrary to the provisions of sections 47F(1)(a) and (2). In the case of an attack tool or a Trojan horse developed or adapted by an intruder, its author is invariably difficult or impossible to identify, making compliance with section 47F(1)(a) a practical impossibility.

Section 47F and, by incorporation, sections 116A and 132 are based on a flawed understanding of what is involved in computer security testing. Section 47F is deficient in that it creates a security testing exception which comes into operation only where the computer program is copied or adapted by or on behalf of the owner or licensee of the original copy,<sup>92</sup> and where the copy or adaptation is not made from an infringing copy of the computer program.<sup>93</sup> No protection is available under section 47F (and therefore items 116A(3), 116(4), 132(5G) and 132(5H)) if security testing is conducted on an infringing copy of a program.<sup>94</sup> Section 47F will not cover many of the activities of security testing organizations, and the range of security testing activities which it does cover will be very restricted.

In view of the fact that the Digital Agenda Act creates new civil and criminal copyright infringement provisions, the scope of the permitted exceptions must be appropriately delineated and clearly described. The penalties for criminal infringement of the anti-circumvention provisions will be severe: imprisonment for up to five years, and fines of up to \$60,500 for an individual and \$302,500 for a corporation.<sup>95</sup> The severity of these penalties will cause security testing organizations to curtail their activities unless the area within which they are permitted to operate is clearly delineated.

Section 47F of the Copyright Act 1968 should be amended to remove the requirement for security testing to be done by or on behalf of an owner or licensee of a program. The legitimacy of security testing should not depend upon whether the program being tested is being copied or adapted by or with the authorization of the copyright

---

<sup>92</sup> Copyright Act 1968, § 47F(1)(a) (Austl.).

<sup>93</sup> *Id.* § 47F(2).

<sup>94</sup> *Id.*

<sup>95</sup> *Id.* § 132(6A).

owner or licensee. Rather, the question of whether the security testing is legitimate should be determined on the basis of a non-exhaustive list of factors to be considered by the courts.

These problems with section 47F, in the form in which it was enacted in the Computer Programs Act, were acknowledged by the House of Representatives' Standing Committee on Legal and Constitutional Affairs in its Advisory Report on the Copyright Amendment (Digital Agenda) Bill 1999.<sup>96</sup> The Committee agreed that section 47F is too narrowly drafted and, in its present form, prohibits activities that security testing organizations need to perform.<sup>97</sup> The Committee concluded that section 47F should be amended to permit security testing to be done without the permission of the owner or licensee, and on infringing copies of programs.<sup>98</sup>

### *C. Reverse Engineering and Patented Software*

Professor Donald Chisum of Santa Clara University School of Law and author of the leading treatise on patent law,<sup>99</sup> offered some brief yet very perceptive comments in response to recent academic writings arguing for clearer reverse engineering rights in the context of patented software.<sup>100</sup> Professor Chisum opened by reasserting his

---

<sup>96</sup> Copyright Amendment (Digital Agenda) Bill 1999 Advisory Report (Austl.), available at <http://www.aph.gov.au/house/committee/laca/digitalagenda/contents.htm> (Dec. 6, 1999).

<sup>97</sup> *Id.* ¶ 4.65.

<sup>98</sup> *Id.* ¶ 4.66.

<sup>99</sup> See DONALD S. CHISUM, CHISUM ON PATENTS: A TREATISE ON THE LAW OF PATENTABILITY, VALIDITY AND INFRINGEMENT (LEXIS Publishing 2000) (1978).

<sup>100</sup> See O'Rourke, *supra* note 48; Julie E. Cohen & Mark A. Lemley, *Patent Scope and Innovation in the Software Industry*, 89 CALIF. L. REV. 1 (2000). See also Commission of the European Communities, Directorate General for the Internal Market, *The Patentability of Computer-Implemented Inventions: Consultation Paper by the Services of the Directorate General for the Internal Market* (October 19, 2000), at 10,

The issue of interoperability, in particular, appears to be sufficiently dealt with by general patent law. In fact, the requirement to adequately disclose the computer-implemented invention in the patent application and the experimental use exception should enable a person skilled in the art to adapt a program to another, pre-existing program created on the basis of the patented invention. To adapt the program, the person must, in accordance with general patent law, secure a license from the patent holder. The situation that a license needs to be obtained to get access to one or several patent rights is common in complex industries, and ways have been found by the business community to use licensing and cross-licensing in order to satisfy the needs.

at [http://europa.eu.int/comm/internal\\_market/en/indprop/soften.pdf](http://europa.eu.int/comm/internal_market/en/indprop/soften.pdf).

support for an experimental use exception in patent law, something he said was explicit in other patent laws throughout the world.<sup>101</sup>

For example, when a person purchases a television set, that person has the right to take it apart or reverse engineer it, which generally does not create a case of patent infringement. Professor Chisum noted that in some instances contract law has been used to control activities beyond the first sale of the patented item. Nonetheless, when a person reverse engineers software due to the nature of the product, that person engages in the act of reproduction which is technically "making" for the purposes of patent law. A clear and full disclosure of the source code in the patent specifications would help alleviate this problem; although in many instances, there would be a need to look at running code, which would entail a reproduction.

Professor Chisum also explained that the temporary copying involved in an experimental use should be allowed, and to this extent the reverse engineering of software should not be an issue. In general, he declared that the enabling and disclosure aspects of the patent system, at least potentially, make it much more conducive to innovation in software than is copyright law, as patent law places an obligation to disclose to the public the relevant information. The adverb "potentially" is stressed as it is widely argued that the necessary information is not being disclosed in the patent specifications.

Cohen and Lemley have argued for a limited right to reverse engineer patented software to permit study of the program and copying of the unprotected elements.<sup>102</sup> They argue this would preserve competition and compatibility between products, reinforce the enabling and disclosure function of the patent system, and set appropriate limits to the scope of the patent.<sup>103</sup> In a similar way, O'Rourke has argued for a qualified application of fair use principles to balance patent rights in software.<sup>104</sup>

#### *D. European Union Law on Reverse Engineering and Software*

Professor Michael Lehmann of the Max Planck Institute and the University of Munich addressed the question of reverse engineering

---

<sup>101</sup> Donald S. Chisum, *The Patentability of Algorithms*, 47 U. PITT. L. REV. 959, 1017-19 (1986). But see Janice M. Mueller, *No "Dilettante Affair": Rethinking the Experimental Use Exception to Patent Infringement for Biomedical Research Tools*, 76 WASHINGTON L. REV. 1 (2001).

<sup>102</sup> Cohen & Lemley, *supra* note 100, at 6.

<sup>103</sup> *Id.* at 21, 25.

<sup>104</sup> O'Rourke, *supra* note 48, at 117.

under European Union (EU) law. In his overview of the 1991 EU Software Directive,<sup>105</sup> he made the very important point that "every property right needs to have limitations which may be endogenous or exogenous; for example, in patent law, the experimental clause and the doctrine of patent misuse are endogenous, whereas antitrust provisions, or the control of general terms and conditions in license contracts, are exogenous limitations by law."

Professor Lehmann continued: "The reverse engineering provision in Article 6 of the EU Software Directive,<sup>106</sup> can be

---

<sup>105</sup> See EU Software Directive, *supra* note 100.

<sup>106</sup> Article 5 - Exceptions to the restricted acts

1. In the absence of specific contractual provisions, the acts referred to in Article 4(a) and (b) shall not require authorization by the rightholder where they are necessary for the use of the computer program by the lawful acquirer in accordance with its intended purpose, including for error correction.

2. The making of a back-up copy by a person having a right to use the computer program may not be prevented by contract insofar as it is necessary for that use.

3. The person having a right to use a copy of a computer program shall be entitled, without the authorization of the rightholder, 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.

#### Article 6 - Decompilation

1. The authorization of the rightholder shall not be required where reproduction of the code and translation of its form<sup>106</sup> within the meaning of Article 4(a) and (b) 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

characterized as a specific limitation on the copyright protection of software. Furthermore, one should take into consideration that property regimes are governed by the principle of territoriality and this has been reinforced by the WTO/TRIPs Treaty, which generated a so-called 'Berne plus' system for more than 150 states worldwide. Starting from internationally respected principles of jurisdiction and conflicts of laws, in instances of alleged infringement, there will be difficult issues if reverse engineering is done off shore, or, say, in the Bahamas where no copyright law exists."

Professor Lehmann asked: "Why should one limit the reverse engineering of software when the reverse engineering of any literature has always been permitted—as my colleagues in the Humanities will know best." Seen from a perspective of law and economics, there exists a problem of sunken costs with respect to the object code, because free access to the source code (which is the reverse engineered object code) makes it easy to copy the software at a very low production cost, and thereby provides the copyist with a free ride on the investment in the software made by the copyright holder. Furthermore, the burden of proving infringement in the digital world is very difficult. Therefore, the EU Software Directive—the first statutory provision on reverse engineering of software worldwide—allows reverse engineering, but only under very narrow conditions such as to facilitate interoperability and error correction.

According to Professor Lehmann: "Every competitor must first ask for information about the interfaces from the copyright holder or use information gained pursuant to antitrust provisions, as in the IBM cases, wherein IBM was required to give details of its interface configuration to avoid antitrust proceedings under Articles 85 and 86

---

(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.

3. In accordance with the provisions of the Berne Convention for the protection of Literary and Artistic Works, the provisions of this Article may not be interpreted in such a way as to allow its application to be used in a manner which unreasonably prejudices the right holder's legitimate interests or conflicts with a normal exploitation of the computer program.

Article 9 - Continued application of other legal provisions

1. The provisions of this Directive shall be without prejudice to 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. Any contractual provisions contrary to Article 6 or to the exceptions provided for in Article 5 (2) and (3) shall be null and void.

*Id.* arts. 5, 6, 9.

(now Articles 81 and 82) of the EC Treaty.”<sup>107</sup> As explained in Article 6(1)(b), the information, sought by reverse engineering, must be necessary to achieve interoperability and not previously readily available to the person who wants to create the interoperability. Therefore, only where information is not given in due time is reverse engineering permitted. It should also be noted that the permission for every lawful acquirer or licensee to bring about error correction does not allow reverse engineering of the whole software but only where the error lies.<sup>108</sup> Professor Lehmann explained:

Further, these European provisions have become mandatory limits on the exclusive right of the copyright holder and are to be applied even if the principles of free choice of law would point to the application of a non-European copyright statute. For example, if a European corporation has made a license agreement with a U.S. copyright holder, while that agreement may be governed by U.S. copyright law, the minimum user rights of the EU Software Directive are guaranteed to every European licensee. One could regard this as a kind of European copyright *ordre public* in favor of every European buyer or licensee of software. This European reverse engineering provision tries to balance out the rights of the copyright holder with some generally vested rights of every European software user.

Underpinning the EU Software Directive is the notion that ideas and principles underlying interfaces are not eligible for copyright protection, the general principle for which is now explained in TRIPs Article 9(2). The reverse engineering exceptions bring about limitations of the exclusive rights of the creator so as to allow adequate access to these ideas and principles. More precisely, one of the recitals of this Directive reads: “[A]n objective of this exception [Article 6] is to make it possible to connect all components of a computer system, including those of different manufacturers, so that they can work together.”

---

<sup>107</sup> See A HANDBOOK OF EUROPEAN SOFTWARE LAW, Part I, 178 (M. Lehmann & C.F. Tapper, eds., 1993).

<sup>108</sup> EU Software Directive, *supra* note 70, art. 5(3).

Similarly, the new EU Directive on copyright in the information society (the "Copyright Directive"),<sup>109</sup> which aims to transform the WIPO Copyright Treaty (WCT)<sup>110</sup> into European law and to harmonize all copyright limitations in Europe, explicitly does not want to alter or modify this balance provided by the EU Software Directive. Article 1(2)(a), stating the scope of the Copyright Directive, provides that this new Directive shall leave intact and in no way affect existing Community provisions relating to "(a) the legal protection of computer programs." According to Professor Lehmann: "Any potential conflict between Article 6 of the Copyright Directive and Article 6 of the Software Directive has to be solved in favor of the permitted possibilities for reverse engineering under the Software Directive. Although the new Copyright Directive generates obligations, in line with the WCT, to prohibit the circumvention of 'technological measures,' they do not modify Article 6 of the Software Directive, i.e., permitted instances of reverse engineering."

The European anti-circumvention provision in Article 6 of the Copyright Directive does not provide for the same far-reaching criminal sanctions as the DMCA in the United States.<sup>111</sup> Professor Lehmann explained his concerns about the criminal liability imposed by the DMCA, saying: "As parts of the DMCA were premised on whether fair use existed, and this doctrine in itself was a case-by-case notion, the penal provisions appeared to lack certainty and, as such, if enacted in the same way in Europe, would violate constitutional principles."

In conclusion, Professor Lehmann alerted the audience to the existence of the new EU Charter of Fundamental Rights,<sup>112</sup> signed by all member states in Nice, France on December 7, 2000, which states in Article 17(2): "Intellectual property shall be protected." This provision is certain to add an interesting dimension to the debate over where to draw the boundaries of the digital estate.

---

<sup>109</sup> Council Directive 2001/29 of 22 May 2001 on the harmonisation of certain aspects of copyright and related rights in the information society, 2001 O.J. (L 167) 10, available at [http://europa.eu.int/eur-lex/en/lif/dat/2001/en\\_301L0029.html](http://europa.eu.int/eur-lex/en/lif/dat/2001/en_301L0029.html) (last updated July 2, 2001).

<sup>110</sup> WIPO Copyright Treaty, Dec. 20, 1996, 1996 O.J. (L 089) 8, available at <http://www.wipo.int/treaties/ip/copyright> (last visited Dec. 1, 2001).

<sup>111</sup> See generally Howard C. Anawalt, *Using Digital Locks in Invention Development*, 15 SANTA CLARA COMPUTER & HIGH TECH. L.J. 363 (1999).

<sup>112</sup> Charter of Fundamental Rights of the European Union, 2000 O.J. (C 364) 1, available at [http://europa.eu.int/comm/justice\\_home/unit/charte/index\\_en.html](http://europa.eu.int/comm/justice_home/unit/charte/index_en.html) (last visited Dec. 1, 2001).

### *E. Reverse Engineering and Digital Diversity*

Professor Fitzgerald emphasized throughout the day that, while innovation (particularly in an economic sense as opposed to a social sense) is the "buzzword," we need to appreciate that the reverse engineering of software is also vital to ensuring digital diversity. Diversity is an ethic we pursue in real space, a cultural and constitutional value embodied in the First Amendment, and there is no reason why we should not aspire to it in digital space, especially in light of Professor Lessig's claim that technology is a primary determinant of regulation in the digital environment.<sup>113</sup> If our actions are constrained, if not somewhat defined, by digital architecture including software, then we must be open to the issue of the structure of the digital space.

According to Professor Fitzgerald, software is discourse in the information society;<sup>114</sup> that is, software acts like language does in real space to construct, facilitate, mediate, and communicate meaning or knowledge. As a consequence of the discursive aspect and capacity of software, at least two issues must be closely considered. First, if we continue to grant intellectual proprietary rights in software through copyright and patent, we must keep firmly in mind the democratic value of open and free discourse. For the more we bestow proprietary rights on private entities (and remember the *United States*

---

<sup>113</sup> The "nature" we inhabit in the digital world is that constructed through technology and technologists. In Lessig's theory there are four modalities of regulation: customary norms, the market, law, and architecture. If I want to stop someone speeding, I can employ the four modalities of regulation: by encouraging a customary norm that speeding is bad through means such as advertising, by raising the price of petrol (market), by enacting a law to say speeding is an offense, and by building a restraining architecture such as a mechanical limit in the car or speed bumps. It is as simple as speed bumps. Just as architecture in real space can constrain our action, Lessig explains architecture in the digital world (code) can regulate what we do. See LAWRENCE LESSIG, *CODE AND OTHER LAWS OF CYBERSPACE* (1999).

<sup>114</sup> See generally Fitzgerald, *supra* note 1 (The basic idea is that software is a discourse, it is a medium for constructing knowledge. Like language, like air, like water, software now is an important piece of architecture in our daily lives. This leads me to the thesis that software is a form of discourse; and if we are going to allow proprietary rights in software, we are allowing proprietary rights in discourse, which is a fundamental issue that we must appreciate. What is a discourse? An interesting definition is the one given by the famous German linguistic philosopher, Martin Heidegger. In 1928, Heidegger wrote a book called *BEING AND TIME: A TRANSLATION OF SEIN AND ZEIT*, in which he defined discourse as "something that allows something to be seen or to be made manifest." And, as the linguistic philosophers told us, language is something that allows things to be seen or made manifest, as are many other things in our social lives. Software then allows things to be seen or made manifest. It is a medium that constructs the communication between myself and others I want to communicate with in digital space. If software is discourse, we need to scrutinize the proprietary rights we allow in it, just as we would question proprietary rights in real space language.).

*v. Microsoft Corp.*<sup>115</sup> case was fundamentally about the power of copyright in software) in relation to software (which in the information society is integral to the construction of knowledge and more generally communication), we bestow the power on the private entity to construct discursive frameworks. In essence, we are giving a private entity ownership of digital language. In particular, the patenting of software, in relation to business methods, is where this issue is most obvious and acute.<sup>116</sup> This is not to say all information should be free; but rather that in commodifying informational products, such as software, through intellectual property rights, we need to assert the value or politics of open discourse.

Second, if software acts to construct identity, we must ensure that there is diversity in the discourses that we employ. If one private entity has a monopoly over the software that is used to browse or search websites, send e-mail, or stream audio or video, then that entity has tremendous power over the way we live our lives. In the case of *United States v. Microsoft Corp.*, antitrust and competition law were invoked as a way of ensuring greater diversity in software products, i.e., in discourses. This means that an individual's digital existence and communication is not mediated by just one type of software and discourse, but by a diversity of software products. To ensure the diversity of digital identity, we must argue for diversity in software products. Fundamental to software diversity is the notion of reverse engineering, especially for interoperability purposes.

### III. CONCLUSION

The Symposium raised many interesting and difficult issues in relation to reverse engineering. It highlighted through the excellent and spirited discussion that reverse engineering is an extremely important issue in this the digital age. While many may have thought this area to be well settled, new and important questions of law continue to be raised.

---

<sup>115</sup> *United States v. Microsoft Corp.*, 87 F. Supp. 2d 30 (D.D.C. 2000). See also *In re Indep. Serv. Orgs. Antitrust Litig.*, 203 F.3d 1322 (Fed. Cir. 2000).

<sup>116</sup> See *State St. Bank & Trust Co. v. Signature Fin. Group Inc.*, 149 F.3d 1368 (Fed. Cir. 1998). See also THE LEAGUE FOR PROGRAMMING FREEDOM, SOFTWARE PATENTS, at <http://lpf.ai.mit.edu/Patents/patents.html> (last visited Sept. 10, 2001) for a list of recent e-commerce software patent claims.

APPENDIX - PROGRAM

Santa Clara University School of Law  
in conjunction with  
Reengineering Forum Industry Association  
and  
IEEE Technical Council on Software Engineering, Reverse Engineering Committee

PRESENTS A ROUNDTABLE DISCUSSION ON:

***INNOVATION, SOFTWARE, AND REVERSE ENGINEERING:***

**TECHNOLOGICAL AND LEGAL ISSUES**

Chaired by:  
Professor Brian Fitzgerald  
Dr. Cristina Cifuentes

Friday, March 23, 2001      8:00a.m.-4:30p.m.      Brass Rail, Santa Clara University

**AGENDA**

8 – 8:45 a.m.

Registration/Continental Breakfast

8:45 – 9:15 a.m.

Dean's Welcome/Opening Comments

9:15 a.m. – 12:15 p.m.

**Technology Panel**

Chair:

Professor Brian Fitzgerald

Panelists:

Elliot Chikofsky (META Group and REF)

*"Reverse Engineering: A Technological Definition"*

Dr. Cristina Cifuentes (Sun Microsystems and IEEE)

*"Reverse Engineering for Interoperability: The Road to Innovation"*

Professor Edward W. Felten (Princeton University)

*"Reverse Engineering and Information Security Assurance"*

**Specialist Commentator: Paul Martino (Intertrust)**

12:15 – 1:30 p.m.

Lunch

1:30 – 4:30 p.m.

**Legal Panel**

Chair:

Professor Brian Fitzgerald

Panelists:

Fred von Lohman (Morrison & Foerster)

*"Reverse Engineering and Copyright Law"*

Dr. Anne Fitzgerald (Technology Lawyer)

*"Comparing U.S. and Australian Experience"*

Professor Donald Chisum (Santa Clara University Law School)

*"Reverse Engineering and Patent Law"*

Professor Michael Lehmann (University of Munich, Max Planck Institute)

*"Reverse Engineering and European Union Law"*

**Specialist Commentator: Professor Howard C. Anawalt** (Santa Clara University School of Law)

**Professor Brian Fitzgerald:** BA, LL.B. BCL (Oxon), LL.M. (Harvard). Head of Law School, Southern Cross University, NSW, Australia. Visiting professor at Santa Clara University School of Law teaching Digital Property (<http://www.scu.edu/law/FacWebPage/Fitzgerald>).

**Dr. Cristina Cifuentes:** B.Sc. (Comp), Ph.D. (Queensland University of Technology). Principal Investigator, Walkabout project, Sun Microsystems Laboratories, California. Currently on leave from The University of Queensland (<http://www.csee.uq.edu.au/~cristina>).

**Elliot Chikofsky:** Director in META Group's enterprise architecture and technology strategies consulting practice; Secretary and Past-Chair of the IEEE Technical Council on Software Engineering.

**Edward W. Felten:** B.S. (California Institute of Technology), M.S. and Ph.D. (University of Washington). Professor of Computer Science, Princeton University.

**Paul Martino:** M.S (Princeton) Senior Director of Acquisitions at Intertrust Technologies

**Professor Howard C. Anawalt:** JD (Boalt Hall). Professor of Law at Santa Clara University School of Law, author of several publications and books on intellectual property, and adviser to the Santa Clara Computer and High Technology Law Journal.

**Professor Donald Chisum:** AB JD (Stanford) Phil and Bobbie Sanfilippo Professor of Law at Santa Clara University School of Law and author of a fifteen volume treatise on patent law: *Patents: A Treatise on the Law of Patentability, Validity and Infringement*.

**Anne Fitzgerald:** LL.M. (Columbia University Law School); LL.M. (University of London). Former appointee to the Australia Federal Government Advisory Council on Intellectual Property, and the Expert Group assisting the Copyright Law Review committee.

**Professor Michael Lehmann:** Dr. jur. (Munich) Professor University of Munich and Max Planck Institute for Industrial and Intellectual Property Law - Visiting Professor at Santa Clara University Law School.

**Fred von Lohman, Esq.:** JD (Stanford). Visiting Researcher with the Berkeley Center for Law & Technology, and Of Counsel with Morrison & Foerster, LLP.

