

An Overview of the Legality and Ethics of Reverse Engineering Software

December 11, 2003

Abstract

This paper reviews the ethical and legal aspects of reverse engineering software. The review includes current U.S. law on the legality of software reverse engineering, including recent laws passed to discourage the practice. The paper also discusses ethical considerations of reverse engineering and its potential benefits and harms to businesses. And finally, there is a review of examples of projects that used reverse engineering to build interoperability with other software.

Resources include current law, periodicals, books, websites, and legal cases and opinions. Several large reverse-engineering projects are also discussed. These resources discuss the issues, views, and trends of reverse engineering software.

Introduction

In the early 1980s, IBM was the leader in the growing personal computer (PC) industry, having almost the entire market share because they owned the copyright to the IBM BIOS. The BIOS links a PC's hardware with the operating system, and whoever controls the BIOS controls the computer. Enter Compaq Corporation, which spent over \$1 million to reverse engineer the IBM BIOS and sell the first IBM-compatible PC clone. Compaq sold 53,000 units in their first year of business, and they had invented a new market. Soon after, they pushed IBM and Packard Bell out of the home PC market altogether (Compaq Computer Corporation). This is just one example of how reverse engineering has changed the PC market over the years.

Reverse engineering is defined as copying the functionality of computer hardware or software without copying the actual piece of hardware or software. Usually this is done without any knowledge of the interior workings of the hardware and software, and reverse engineers have to guess at how the internals are implemented by observation and guesswork. In most cases, copyright law in the United States protects this practice because it is allowable to copyright *how* something does something, not *what* it does. In software terms, programmers can examine what data go in and out of a program, and therefore try to figure out how the data is changed. However, programmers cannot decompile a piece of software and copy the source code into their own programs because that would violate copyright law. Reverse engineering is a discipline that has been around for some time, most notably in the 1970s and 1980s when companies were scrambling to make sure their computers were compatible with their competitors.

To legally reverse engineer the IBM chip in the early 1980s, Compaq had to first “find engineers who had not had anything to do with IBM to write a specification of exactly what the chip should do by examining the chips produced by IBM. Then new engineers were brought in

to develop the chip” (Compaq Computer Corporation). This is also called “clean room” reverse engineering, because it does not break any copyright laws since neither team collaborated.

Reverse engineering is a practice not only used to achieve interoperability but also to update legacy software applications. Legacy software is often written in older languages that may or may not have the source code present, good documentation, or the original developer around. However, “... most product lines start with legacy systems that need to be updated to enable them to interact as well-defined components. Although it is possible to update legacy systems through reverse engineering, these techniques are costly” (Bergey et al., 1999).

The goal of this paper is to provide an overview of the legal and ethical aspects of reverse engineering, as well as well-known stories and examples.

Legal Aspects of Reverse Engineering

In the United States, there has been much talk of the legality of reverse engineering software since the passage of the Digital Millennium Copyright Act of 1998 (DMCA). However, case precedence has ruled in favor of reverse engineering more often than not, for both hardware and software. Even in such strange areas as boat-hull engineering (Samuelson & Scotchmer, 2001), courts have ruled in favor of reverse engineering for the purposes of interoperability and improvement of a competitor's products.

In most reverse-engineering projects, copyright infringement becomes an issue when a programmer decompiles software and makes a copy of the original in the process. However, courts have ruled that in the process of reverse engineering, specifically for interoperability, the copy of the software is ruled under the "fair use" doctrine (Samuelson & Scotchmer, 2001).

One of the defining cases for the legality of software reverse engineering for the purposes of interoperability that addressed the decompilation issue is the 1992 *Sega v. Accolade* case (Samuelson & Scotchmer, 2001). In the case, Sega sued Accolade for copyright infringement for reverse engineering Sega games to make games compatible with Sega's *Genesis* game console. The game compatibility was usually licensed to third-party game makers for a large fee. Accolade defended itself in court with a fair use defense for the disassembly of Sega's games. In their ruling, the Ninth Circuit court ruled in favor of Accolade, stating that the copying had been done "solely in order to discover the functional requirements for compatibility with the Genesis console -- aspects of Sega's programs that are not protected by copyright" (Samuelson & Scotchmer, 2001). The judges also wrote that Accolade's decompilation "led to an increase in the number of independently designed video game programs offered for use with the Genesis console. It is precisely this growth in creative expression...that the Copyright Act was intended

to promote” (Samuelson & Scotchmer, 2001). This ruling has since been followed in almost all subsequent rulings on reverse engineering software (Samuelson & Scotchmer, 2001).

Legal issues with reverse engineering encompasses patent law, copyright law, trade secret law, and contract law. For a defense against patent infringement, reverse engineering could be in trouble if some part of the patented invention is found in the code that was released to be compatible. For trade secret law, reverse engineering probably could mount a successful defense because reverse engineering does not constitute misappropriation of that secret. For copyright law, reverse engineering is allowed if it is the only way to extract the ideas underlying the software to make a compatible product. As for contract law, there is no indication that a shrink-wrap or click-wrap license is legal to ban this practice, although most of them expressly forbid it (Lemly, 2001).

DMCA

Any discussion of the legality of reverse engineering today needs to discuss the Digital Millennium Copyright Act (DMCA) of 1998, and other similar laws being considered in Europe and elsewhere. The DMCA was one of the first laws of its kind to expressly make illegal certain types of reverse engineering of software, mainly for the purposes of circumventing anti-copying protection on digital movies and music. Up until this point, Congress and courts had in most cases affirmed the right to reverse engineer for the purpose of learning the underlying technology and for interoperability.

The most controversial section of the DMCA has been the section on reverse engineering:

(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.

(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. (DMCA, Section 1201, f (1)-(4)).

Recently, there have been a number of cases that have ruled on violating the DMCA in reverse engineering cases, sometimes brought by companies looking to remove a competitor who reverse engineered their product. In a recent case, Chamberlain, a garage door company, sued a universal garage door opener manufacturer claiming that their reverse engineered door opener working with Chamberlain's garage doors violated the DMCA. However, the U.S. District Court disagreed, saying "Under Chamberlain's theory, any customer who loses his or her Chamberlain transmitter, but manages to operate the opener either with a non-Chamberlain transmitter or by some other means of circumventing the rolling code, has violated the DMCA. In this court's view, the statute does not require such a conclusion" (Dean, 2003).

Another ongoing and notable DMCA cases is the *MPAA v. 2600: The Hacker Quarterly*, in which the Motion Picture Association of America (MPAA) sought to restrict access to the DeCSS program for decrypting the CSS code used to protect DVDs (Electronic Freedom Foundation). Eventually, the courts decided that software in this case could be considered speech and was therefore protected under the First Amendment of the U.S. Constitution.

This is in holding with the accepted constitutionality of reverse engineering for the purpose of “the promotion of ‘the Progress of Science....’” (U.S. Const., Article I, §8, cl. 8). The court in *Atari Games Corp. v. Nintendo* (Fed. Cir. 1992) found that the legality of reverse engineering could constitute fair use, in that “the Copyright Act permits an individual in rightful possession of a copy of a work to undertake necessary efforts to understand the work’s ideas, processes, and methods of operation” (Lemly, 2001).

In fact, the Library of Congress and the U.S. Copyright Office recently reviewed and granted exceptions to the DMCA because of the controversy surrounding the law (Dean, 2003). The four classes of work that were exempted from the anti-circumvention rule allow programmers to “... bypass a digital lock to access lists of websites blocked by commercial filtering companies, circumvent obsolete dongles to access computer programs, access computer programs and video games in obsolete formats, and access e-books where the text-to-speech function has been disabled” (Dean, 2003). This is a start in what could be a long public debate of the more subtle reverse engineering restrictions contained in the DMCA.

Court rulings from the past few decades have shown that what is important in determining the legality of reverse engineering software is the intent. If the intent is interoperability between software vendors, then the courts seem to rule in favor of reverse engineering, as in the *Sega v. Accolade* case (Samuelson & Scotchmer, 2001). However, if the intent is outright copying of code, the courts have ruled in favor of the copyright holders, as in

the DVD CSS case (Electronic Frontier Foundation). Therefore, the legality of reverse engineering has a direct influence on the ethical aspects of this practice.

Ethical Aspects of Reverse Engineering

The ethics of reverse engineering is still a topic of debate among software developers and IT managers. On the one hand, reverse engineering software could be seen as a useful educational tool. On the other hand, reverse engineering could be seen as infringing on the copyrights of the code if the programmer does not have rights to that code. There are many programmers who actively practice the discipline of reverse engineering, and there is even a Working Conference on Reverse Engineering (WCRE) that has the support of the IEEE Computer Society's Technical Council on Software Development (TCSE), which endorses the legality of reverse engineering as a practice.

According to Reynolds (2003), "Most information technology managers would consider [reverse engineering commercial software] as unethical as the software user does not actually own the right to the software. In addition, a number of intellectual property issues would be raised depending on whether the software were licensed, copyrighted, or patented."

There are several computing groups that put out recommendations on the ethical aspects of software development, although there was no specific reference to reverse engineering. In the most common and widely adopted recommendation, the IEEE *Code of Ethics* (1999) states that a computer professional should "2.02. Not knowingly use software that is obtained or retained either illegally or unethically" and "2.06. Identify, document, collect evidence and report to the client or the employer promptly if, in their opinion, a project is likely to ... violate intellectual property law, or otherwise to be problematic." These statements seem to rule out reverse engineering software specifically because of intellectual property (IP) law issues. However, the validity of an IP prosecution in reverse-engineering systems has not been tried and resolved in the courts as of this date.

Ethical uses for reverse engineering software could include interoperability, fixing proprietary code when the source is lost or not available, and for the purpose of learning.

Unethical uses for reverse engineering software could include circumventing security measures for the express purpose of gaining illegal access to systems and decompiling for the purpose of copying and stealing code.

The ethical considerations of reverse engineering still hinge on the legal aspects, which seem to be driven by the intent of the reverse engineering and the means to access the information. Therefore, applying an ethical analysis to determine whether reverse engineering of software constitutes an ethical dilemma is appropriate. Frameworks such as Laudon and Laudon's (2002) offer the following procedures:

1. Identify and describe clearly the facts.
2. Define the conflict or dilemma and identify the higher-order values involved.
3. Identify the stakeholders.
4. Identify the options that you can reasonably take.
5. Identify the potential consequences of your options (Laudon and Laudon, 2002).

This framework could be a good fit for determining the ethics of reverse engineering because it allows a review of whether the process and final product would violate any laws and also allows the determination of whether there are other ways to accomplish the same task.

Because there exists a variety of opinions on the ethics of reverse engineering, IT projects should consider the legal aspects of the project first and determine whether any copyright laws could be broken, and then consider the ethical aspects of the project using the framework above. Because the legal aspects of reverse engineering have ruled that interoperability is acceptable, one should be careful that their project does not cross that boundary.

Examples of Reverse Engineering

Reverse engineering software for the purpose of interoperability is a well-proven practice in software development. There are numerous applications that have been developed to provide interoperability with competitor's software, protocols, and proprietary libraries.

One of the best examples is the Samba server application (www.samba.org), which runs on Linux and uses the Server Message Block (SMB) protocol to connect to Microsoft's proprietary file server. Samba allows Linux users to share documents and files with Microsoft Windows operating systems. To achieve this, the Samba team had to reverse engineer the proprietary SMB protocol to figure out how to interoperate with Windows machines.

Another good example is Open Office (www.openoffice.org), a free, open-source alternative to Microsoft Office. Open Office allows users on Windows, Mac, and Linux to open and save native Microsoft Office document formats, such as Word (.doc), Excel (.xls), and PowerPoint (.ppt) files. To be able to read and write these proprietary formats, Open Office software engineers had to reverse engineer the Microsoft Office file formats to be able to read and write them.

A final example is Gaim (gaim.sourceforge.net), a free, open-source instant-messaging client, like America Online's (AOL) Instant Messenger client. Gaim developers had to not only reverse engineer the AOL Instant Messenger client and protocol, but it also had to do so for Microsoft Messenger and Yahoo! Messenger so that it could speak to each of these proprietary instant-messaging protocols.

Conclusion

Reverse engineering software is a topic that is debated among software developers worldwide. It is a very difficult process that requires many hours of work, patience, and a large body of knowledge. Unfortunately, the issue of whether reverse engineering software violates intellectual property law has not been established in court rulings because most cases are resolved out of court. Once courts do decide on the legality, there will be precedence set for future cases on which to base their decisions.

The legal and ethical aspects of reverse engineering still has many unanswered questions. What if a product is reverse engineered in a country like China and sold in the U.S.? Is it against U.S. law to purchase that product? What if the end product that reverse engineering produces was done in another country under contract with a U.S. company? Is it unethical to reverse engineer software for the purposes of research and learning? If not, what about publishing information about that research or learning? I believe that in the future reverse engineering will probably continue to raise as many questions as it answers.

In my opinion, courts should rule in favor of reverse engineering for the purpose of interoperability. Any theft of code or hardware designs that are used in a competitor's product is already covered by existing copyright law, and it would therefore be redundant to legislate against reverse engineering. However, until a body of law and court rulings interpreting these laws have been established in courts around the world, the debate over the legality and ethics of reverse engineering software will continue.

Bibliography

- Bergey, J., Smith, D., Weideman, N., & Woods, S. (1999). *Options Analysis for Reengineering (OAR): Issues and Conceptual Approach*. from <http://wuarchive.wustl.edu/languages/ada/sei/documents/99.reports/pdf/99tn014.pdf>
- Compaq Computer Corporation. (n.d.). Retrieved December 10, 2003, from Wikipedia: <http://en2.wikipedia.org/wiki/Compaq>
- Dean, K. (2003). New Ways to Skirt DMCA ... Legally!, *Wired*. from <http://www.wired.com/news/politics/0,1283,60996,00.html>
- Dean, K. (2003). Opening Doors With the DMCA, *Wired*. from <http://www.wired.com/news/business/0,1367,61232,00.html>
- Digital Millennium Copyright Act (DMCA), HR2281, from http://www.eff.org/IP/DMCA/hr2281_dmca_law_19981020_pl105-304.html
- Electronic Frontier Foundation, *Free "Jon Johansen!" Campaign*, from http://www.eff.org/IP/Video/Johansen_DeCSS_case/
- Gotterbarn, D. (1999, November/December), How the New Software Engineering Code of Ethics Affects You. *IEEE Software*, November/December 1999, 58-64.
- IEEE-CS/ACM Joint Task Force on Software Engineering Ethics and Professional Practices. (1999). *Software Engineering Code Of Ethics And Professional Practice*.
- Laudon, K. & Laudon, J. (2002). *Management Information Systems: Managing the Digital Firm*. 7th ed. Upper Saddle River, New Jersey: Prentice Hall.
- Lemly, M. (2001). *Brief Of Amice Curiae In Support Of Petition For Panel Rehearing And Rehearing En Banc Of Defendant – Appellant Baystate Technologies, Inc.*, from http://www.eff.org/IP/Emulation/20020918_baystate-amicus.pdf.

Reynolds, G. (2003). *Ethics in Information Technology*. Thomson Course Technology, Boston.

Samuelson, P. & Scotchmer, S. (2001). *The Law and Economics of Reverse Engineering*, *Yale Law Journal*, May 2002, 1575-1663.

United States Constitution, from <http://www.house.gov/Constitution/Constitution.html>

References

- Ethics and Reverse Engineering: <http://onlineethics.org/eng/problems/rev-ind.html>
- EFF Reverse Engineering Papers: <http://www.eff.org/IP/Emulation/>
- Law++, Reverse Engineering: <http://www.lawplusplus.com/column10.htm>
- Questions and Answers with DOJ IP Lawyers:
<http://interviews.slashdot.org/article.pl?sid=03/07/24/1326224>
- Intro to Reverse Engineering: <http://www.acm.uiuc.edu/sigmil/RevEng/index.html>
- Legal Protection of Digital Information: <http://www.digital-law-online.com/lpdi1.0/treatise48.html>
- Chilling Effects: <http://www.chillingeffects.org/reverse/>
- WCRE - Working Conf. On Reverse Engineering: <http://reengineer.org/wcre2001/>
- I, Cringley: <http://www.pbs.org/cringely/pulpit/pulpit19990930.html>

Open Source Engineering Projects

- GAIM: <http://gaim.sourceforge.net> – An open source Instant Messaging application that can talk to AOL Instant Messenger, Yahoo! Messenger, and MSN Messenger, among others. Each of the IM clients that access these different implementations of IM had to be reverse engineered.
- Open Office: www.openoffice.org – An open source alternative to Microsoft Office, it reads and writes MS Word, PowerPoint, and Excel documents. Programmers had to reverse engineer these file formats to write compatible files.
- LinuxDrivers.com: <http://linuxdrivers.foundries.sourceforge.net/> -- Many of the drivers to run graphics cards, peripherals, and other devices are not written for Linux, so many developers have to reverse engineer the MS Windows or Macintosh drivers and rewrite them to work with Linux. You can find drivers for most any device for Linux because of this.
- Mono: <http://www.go-mono.com/> -- Mono is a project that aims to be compatible with Microsoft's .NET code. It allows developers to write applications in .NET and port them to any computer that can run Mono.
- Ximian: <http://www.ximian.com/> -- A Personal Information Management application that mimics Microsoft Outlook with e-mail, tasks, calendar, and a to do list. Programmers had to reverse engineer the Microsoft Exchange Server messages in order to be compatible.