Reverse engineering

From Wikipedia, the free encyclopedia

Reverse engineering
(RE) is the process of discovering the technological principles of a device, object or system through analysis of its structure, function and operation. It often involves taking something (e.g. a mechanical device, electronic component, or software
program) apart and analyzing its workings in detail, usually to try to make a new device or program that does the same thing without copying anything from the original.

Contents

Motivation

Reasons for reverse-engineering:

- interoperability.
- lost documentation: Reverse engineering often is done because the documentation of a particular device has been lost (or was never written), and the person who built the thing is no longer available. Integrated circuits often seem to have been designed on obsolete, proprietary systems, which means that the only way to incorporate the functionality into new technology is to reverse-engineer the existing chip and then re-design it.
- product analysis. To examine how a product works, what components it consists of, estimate costs, identify potential patent infringement.
- security auditing
- removal of copy protection, circumvention of access restrictions.
- creation of unlicensed/unapproved duplicates
- fraud e.g. revealing secret keys from smart cards

Reverse engineering of mechanical devices

As computer-aided design
has become more popular, reverse engineering has become a viable method to create a 3D virtual model of an existing physical part for use in 3D CAD, CAM, CAE
and other software. The reverse engineering process involves measuring an object and then reconstructing it as a 3D model. The physical object can be measured using 3D scanning technologies like CMMs, laser scanners, structured light digitizers or computed tomography. The measured data alone, usually represented as a point cloud, lacks topological information and is therefore often processed and modeled into a more usable format such as a triangular faced mesh, a set of NURBS surfaces or a CAD model. Applications like Imageware

(http://www.ugs.com/products/nx/imageware/) , PolyWorks (http://www.innovmetric.com) , Rapidform (http://www.Rapidform.com) or Geomagic (http://www.geomagic.com) are used to process the point clouds themselves into formats usable in other applications such as 3D CAD, CAM, CAE or visualization.

Reverse engineering is also used by businesses to bring existing physical geometry into digital product development environments, to make a digital 3D record of their own products or assess competitors' products. It is used to analyze, for instance, how a product works, what it does, what components it consists of, estimate costs, identify potential patent infringement, etc.

Value engineering
is a related activity also used by business. It involves deconstructing and analysing products, but the objective is to find opportunities for cost cutting.

Reverse engineering of software

The term "reverse engineering" as applied to software means different things to different people, prompting Chikofsky and Cross to write a paper researching the various uses and defining a taxonomy. From their paper: Reverse engineering is the process of analyzing a subject system to create representations of the system at a higher level of abstraction.[1] It can also be seen as "going backwards through the development cycle".[2] In this model, the output of the implementation phase (in source code form) is reverse engineered back to the analysis phase, in an inversion of the traditional waterfall model. Reverse engineering is a process of examination only: the software system under consideration is not modified (which would make it reengineering). Software anti-tamper technology is used to deter both reverse engineering and reengineering of proprietary software and software-powered systems. In practice, two main types of reverse engineering emerge. In the first case, source code is already available for the software, but higher level aspects of the program, perhaps poorly documented or documented but no longer valid, are discovered. In the second case, there is no source code available for the software, and any efforts towards discovering one possible source code for the software are regarded as reverse engineering. This second usage of the term is the one most people are familiar with. Reverse engineering of software can make use of the clean room design technique to avoid infringing copyrights. The reverse engineering of software accounts for 95% of all instances of reverse engineering.

On a related note, black box testing in software engineering has a lot in common with reverse-engineering. The tester usually has the API, but their goals are to find bugs and undocumented features by bashing the product from outside.

Other purposes of reverse engineering include security auditing, removal of copy protection ("cracking"), circumvention of access restrictions often present in consumer electronics, customization of embedded systems (such as engine management systems), in-house repairs or retrofits, enabling of additional features on low-cost "crippled" hardware (such as some graphics card chipsets), or even mere satisfaction of curiosity.

Binary software

This process is sometimes termed *Reverse Code Engineering* or RCE.[3] As an example, decompilation of binaries for the Java platform can be accomplished using Jad. One famous case of reverse engineering was the first non-IBM implementation of the PC BIOS which launched the historic IBM PC compatible industry that has been the overwhelmingly dominant computer hardware platform for many years. An example of a group that reverse engineers software for enjoyment is CORE which stands for "Challenge Of Reverse Engineering". In the United States, the Digital Millennium Copyright Act
exempts from the circumvention ban some acts of reverse engineering aimed at interoperability of file formats and protocols, but judges in key cases have ignored this law, since it is acceptable to circumvent restrictions for use, but not for access.[4]
Aside from restrictions on circumvention, reverse engineering of software is protected in the U.S. by the fair use exception in copyright law.[5] The Samba software, which allows systems that are not running Microsoft Windows systems to share files with systems that are, is a classic example of software reverse engineering, since the Samba project had to reverse-engineer unpublished information about how Windows file sharing worked, so that non-Windows computers could emulate it. The Wine project does the same thing for the Windows API, and OpenOffice.org is one party doing this for the Microsoft Office file formats. The ReactOS project is even more ambitious in its goals as it strives to provide binary (ABI and API) compatibility with the current Windows OSes of

the NT branch, allowing software and drivers written for Windows to run on a clean room reverse engineered GPL open source counterpart.

Binary software techniques

Reverse engineering of software can be accomplished by various methods. The three main groups of software reverse engineering are

1. Analysis through observation of information exchange, most prevalent in protocol reverse engineering, which involves using bus analyzers and packet sniffers, for example, for accessing a computer bus or computer network
   connection and revealing the traffic data thereon. Bus or network behavior can then be analyzed to produce a stand-alone implementation that mimics that behavior. This is especially useful for reverse engineering device drivers. Sometimes reverse-engineering on embedded systems is greatly assisted by tools deliberately introduced by the manufacturer, such as JTAG ports or other debugging means. In Microsoft Windows, low-level debuggers such as SoftICE are popular.
2. Disassembly using a disassembler, meaning the raw machine language of the program is read and understood in its own terms, only with the aid of machine language mnemonics. This works on any computer program but can take quite some time, especially for someone not used to machine code. The Interactive Disassembler is a particularly popular tool.
3. Decompilation using a decompiler, a process that tries, with varying results, to recreate the source code in some high level language for a program only available in machine code or bytecode.

Source code

A number of UML
tools refer to the process of importing source code in order to generate UML diagrams, as "reverse engineering". See List of UML tools.

Reverse-engineering of integrated circuits/smart cards

Reverse Engineering is an invasive and destructive form of analyzing a smart card. The attacker grinds away layer by layer of the smart card and takes pictures with an electron-microscope. With this technique it is possible to reveal the complete hardware and software part of the smart card. The major problem for the attacker is to bring everything into the right order to find out how everything works. Engineers try to hide keys and operations by mixing up memory positions, for example busscrambling[6][7]. In some cases it is even possible to attach a probe to direct measure voltages while the smart card is still operational. Engineers employ sensors to detect and prevent this attack. [8] It takes very high effort to break a smart card used for payment e.g., and the technical equipment is only available to large chip-producers. Additionally the gain is low due to other security mechanisms like shadow accounts.

Reverse-engineering for military applications

Reverse engineering is often used by military in order to copy other nations' technology, devices or information, or parts of which, have been obtained by regular troops in the fields or by intelligence operations. It was often used during the Second World War and the Cold War. Well-known examples from WWII and later include:

- Jerry can: British and American forces noticed that the Germans had gasoline cans with an excellent design. They reverse engineered copies of those cans. The cans were popularly known as *Jerry cans*.
- Tupolev Tu-4: Three American B-29 bombers on missions over Japan were forced to land in the USSR. The Soviets, who did not have a similar strategic bomber, decided to copy the B-29. Within a few years they had developed the Tu-4, a near perfect copy.
- V2 Rocket: Technical documents for the V2 and related technologies were captured by the Western Allies at the end of the war. Soviet and captured German engineers had to reproduce technical documents and plans, working from captured hardware, in order to make their clone of the rocket, the R-1, which began the postwar Soviet rocket program that led to the R-7 and the beginning of the space race.
- K-13/R-3S missile (NATO reporting name **AA-2 'Atoll**), a Soviet reverse-engineered copy of the AIM-9 Sidewinder, made possible after a Taiwanese AIM-9B hit a Chinese MiG-17 without exploding; amazingly, the

missile became lodged within the airframe, the pilot returning to base with what Russian scientists would describe as a university course in missile development.
- BGM-71_TOW
Missile: In May 1975 negotiations between Iran and Hughes Missile Systems on co-production of the TOW and Maverick missiles are stalled over disagreements in the pricing structure. The subsequent 1979 revolution ended all plans for such co-production. Iran was successful in reverse engineering the missile, and are currently producing their own copy: the Toophan.

Legality

In the United States and many other countries, even if an artifact or process is protected by trade secrets, reverse-engineering the artifact or process is often lawful as long as it is obtained legitimately. Patents, on the other hand, need a public disclosure of an invention, and therefore patented items do not necessarily have to be reverse engineered to be studied. One common motivation of reverse engineers is to determine whether a competitor's product contains patent infringements or copyright infringements.

Reverse engineering software or hardware systems which is done for the purposes of interoperability (for example, to support undocumented file formats or undocumented hardware peripherals), is mostly believed to be legal, though patent owners often contest this and attempt to stifle any reverse engineering of their products for any reason.

"...[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."[9]

See also

- Antikythera mechanism
- Benchmarking
- Bus analyzer
- Chinese copy method
- Code morphing
- Clean room design
- Connectix Virtual Game Station
- Decompiler
- Digital Millennium Copyright Act
- Forensic engineering
- Interactive Disassembler
- J. Brant Arseneau
- Knowledge Discovery Metamodel
- List of production topics
- Logic analyzer
- Paycheck (film)
- Value engineering

References

1. ^
Chikofsky, E.J.; J.H. Cross II (January 1990). "Reverse Engineering and Design Recovery: A Taxonomy in IEEE Software". *IEEE Computer Society*: 13–17.
2. ^ Warden, R. (1992). *Software Reuse and Reverse Engineering in Practice*. London, England: Chapman & Hall, 283–305.
3. ^ Chuvakin, Anton; Cyrus Peikari (January 2004). *Security Warrior*, 1st ed., O'Reilly. Retrieved on 2006-05-25.
4. ^ US Code: Title 17,1201. Circumvention of copyright protection systems (http://www4.law.cornell.edu/uscode/html/uscode17/usc_sec_17_00001201----000-.html) . Retrieved on 2006-05-25.
5. ^ See Pamela Samuelson and Suzanne Scotchmer, "The Law and Economics of Reverse Engineering", 111 *Yale Law Journal* 1575-1663 (May 2002).
6. ^ Wolfgang Rankl, Wolfgang Effing, Smart Card Handbook (2004)
7. ^ T. Welz: Smart cards as methods for payment (2008), Seminar ITS-Security Ruhr-Universität Bochum, "http://www.crypto.rub.de/its_seminar_ws0708.html"
8. ^ David C. Musker: Protecting & Exploiting Intellectual Property in Electronics, IBC Conferences, 10 June 1998

9. ^ Sega v. Accolade, 203 F.3d 596 (9th Cir. 1993)

External links

- Reverse Code Engineering (http://www.reverse-engineering.net/) , as entry point for Reverse Code Engineering
- IITAC.org: Reverse Code Engineering certification (http://www.iitac.org) according to ISO 17024
- Crackmes.de The longest running and most complete Crackmes web page on the internet. Free Reverse Code Engineering training material (http://www.crackmes.de)
- Reverse Code Engineering: The complete resource for RE of software. Archives of the most important Reverse Code Engineering sites (http://www.woodmann.com)
- OpenRCE: Reverse Engineering Portal (http://www.openrce.org)
- MoDisco: (Model Discovery), an Eclipse Project on Model Driven Reverse Engineering (http://www.eclipse.org/gmt/modisco/)
- ERESI: The ERESI Reverse Engineering Software Interface (http://www.eresi-project.org) for RCE on UNIX
- Radare: Advanced CmdLine Hexadecimal Editor, Disassembler and Debugger (http://radare.nopcode.org) multi-platform full-featured RCE tool
- Program transformation wiki on Reverse Engineering (http://www.program-transformation.org/Transform/ReverseEngineering)
- Introduction to Reverse Engineering Software (http://www.acm.uiuc.edu/sigmil/RevEng/) , preprint of a book by Mike Perry and Nasko Oskov.
- Reverse Engineering Shapes (http://www.ercim.org/publication/Ercim_News/enw44/varady.html) , article by Tamás Várady
- Online Resource for Reverse Engineering Software (http://www.reteam.org/)
- Article on legal considerations (http://www.jenkins-ip.com/serv/serv_6.htm) by David C. Musker
- CNN: How Soviets copied America's best bomber during WWII (http://archives.cnn.com/2001/US/01/25/smithsonian.cold.war/)
- Very good RE definitions from Software Engineering (http://w3.umh.ac.be/genlog/SE/SE-contents.html#ReverseEngineering)
- Reverse engineering the vertebrate brain (http://www.abrg.group.shef.ac.uk/reverb/public/)
- Architecture-Driven Modernization group at OMG (http://adm.omg.org)
- Reverse Engineering Delivers Product Knowledge Aids Technology Spread (http://www.elecdesign.com/Articles/Index.cfm?ArticleID=11966) - Article in Electronic Design by Dick James, Senior Technology Adviser, Chipworks