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 to be used in maintenance, or to try to make a new device or program that does the same thing without copying anything from the original.

Reverse engineering has its origins in the analysis of hardware for commercial or military advantage [1]. The purpose is to deduce design decisions from end products with little or no additional knowledge about the procedures involved in the original production. The same techniques are currently being researched for application to legacy software systems, not for industrial or defense ends, but rather to replace incorrect, incomplete, or otherwise unavailable documentation[2].

Contents

- 1 Motivation
- 2 Reverse engineering of mechanical devices
- 3 Reverse engineering of software
  - 3.1 Binary software
    - 3.1.1 Binary software techniques
- 4 Source code
- 5 Reverse engineering of integrated circuits/smart cards
- 6 Reverse engineering for military applications
- 7 Legality
- 8 See also
- 9 Further reading
- 10 References
- 11 External links

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 it 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, and identify potential patent infringement.

- Security auditing.
- Military or commercial espionage. Learning about an enemy's or competitor's latest research by stealing or capturing a prototype and pulling it apart.
- Removal of copy protection, circumvention of access restrictions.
- Creation of unlicensed/unapproved duplicates.
- Academic/learning purposes.
- Curiosity
- Competitive technical intelligence (understand what your competitor is actually doing versus what they say they are doing)
- Learning: Learn from others mistakes. Do not make the same mistakes that others have already made and subsequently corrected

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 [computer-aided design], CAM, CAE and other software[3]. 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.

The point clouds produced by 3D scanners are usually not used directly since they are very large unwieldy data sets, although for simple visualization and measurement in the architecture and construction world, points may suffice. Most applications instead use polygonal 3D models, NURBS surface models, or editable feature-based CAD models (aka Solid models). The process of converting a point cloud into a usable 3D model in any of the forms described above is called "modeling"'.

- POLYGON MESH MODELS: In a polygonal representation of a shape, a curved surface is modeled as many small faceted flat surfaces (think of a sphere modeled as a disco ball). Polygon models -- also called Mesh models, are useful for visualization, for some CAM (i.e., machining), but are generally "heavy" (i.e., very large data sets), and are relatively un-editable in this form. Reconstruction to polygonal model involves finding and connecting adjacent points with straight lines in order to create a continuous surface. Many applications are available for this purpose (eg. kubit PointCloud for AutoCAD, photomodeler, imagemodel, PolyWorks, Rapidform, Geomagic, Imageware, Rhino, etc.).

- SURFACE MODELS: The next level of sophistication in modeling involves using a quilt of curved surface patches to model our shape. These might be NURBS, TSplines or other representations of curved topology using higher ordered polynomials (i.e, curved, not straight). Using NURBS, our sphere is a true mathematical sphere. Some applications offer patch layout by hand but the best in class offer both automated patch layout and manual layout. These patches have the advantage of being lighter and more manipulable when exported to CAD. Surface models are somewhat editable, but only in a sculptural sense of pushing and pulling to deform the surface. This representation lends itself well to modeling organic and artistic shapes. Providers of surface modelers include NX, Imageware, Rapidform, Geomagic, Rhino, Maya, T Splines etc.

- SOLID CAD MODELS: From an engineering/manufacturing perspective, the ultimate representation of a digitized shape is the editable, parametric CAD model. After all, CAD is the common "language" of industry to describe, edit and maintain the shape of the enterprise's assets. In CAD, our sphere is described by parametric features which are easily edited by changing a value(e.g., centerpoint and radius).

These CAD models describe not simply the envelope or shape of the object, but CAD models also embody the "design intent" (i.e., critical features and their relationship to other features). An example of design intent not evident in the shape alone might be a brake drum's lug bolts, which must be concentric with the hole in the center of the drum. This knowledge would drive the sequence and method of creating the CAD model; a designer with an awareness of this relationship would not design the lug bolts referenced to the outside diameter, but instead, to the center. A modeler creating a CAD model will want to include both Shape and design intent in the complete CAD model.

Vendors offer different approaches to getting to the parametric CAD model. Some export the NURBS surfaces and leave it to the CAD designer to complete the model in CAD(e.g., Geomagic, Imageware, Rhino). Others use the scan data to create an editable and verifiable feature based model that is imported into CAD with full feature tree intact, yielding a complete, native CAD model, capturing both shape and design intent (e.g. Rapidform). Still other CAD applications are robust enough to manipulate limited points or polygon models within the CAD environment(e.g., Catia).

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 to assess competitors' products. It is used to analyze, for instance, how a product works, what it does, and what components it consists of, estimate costs, and identify potential patent infringement, etc.

Value engineering is a related activity also used by businesses. 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, they state, "Reverse engineering is the process of analyzing a subject system to create representations of the system at a higher level of abstraction."[4] It can also be seen as "going backwards through the development cycle".[5] 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.

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.[6] 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 (DMCA) 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.[7] Aside from restrictions on circumvention, reverse engineering of software is protected in the U.S. by the fair use exception in copyright law.[8] 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[9][10]. In some cases, it is even possible to attach a probe to measure voltages while the smart card is still operational. Engineers employ sensors to detect and prevent this attack. [11] It takes very high effort to break a smart card used for payment, 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 militaries in order to copy other nations' technologies, devices or information that 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.
- China has reversed many examples of US and Russian hardware, from fighter aircraft to missiles and HMMWV cars.

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."[12]

See also

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

Further reading

- Eilam, Eldad (2005). *Reversing: Secrets of Reverse Engineering*. Wiley Publishing. pp. 595. ISBN 0764574817.
- James, Dick (January 19, 2006). "Reverse Engineering Delivers Product Knowledge; Aids Technology Spread". *Electronic Design*. Penton Media, Inc. http://electronicdesign.com/Articles/Index.cfm?AD=1&ArticleID=11966. Retrieved on 2009-02-03.
- Raja, Vinesh; Fernandes, Kiran J. (2008). *Reverse Engineering - An Industrial Perspective*. Springer. pp. 242. ISBN 978-1-84628-855-5.
- Thumm, Mike (2007). "Talking Tactics". *IEEE 2007 Custom Integrated Circuits Conference (CICC)*. IEEE, Inc. http://ewh.ieee.org/r5/denver/sscs/References/2007_09_Torrance.pdf. Retrieved on 2009-02-03.

References

1. ^ E. J. Chikofsky and J. H. Cross, II, “Reverse Engineering and Design Recovery: A Taxonomy,” IEEE Software, vol. 7, no. 1, pp. 13-17, January 1990.
2. ^ A Survey of Reverse Engineering and Program Comprehension. Michael L. Nelson, April 19, 1996, ODU CS 551 - Software Engineering Survey
3. ^ T. Varady, R. R. Martin, J. Cox, Reverse Engineering of Geometric Models—An Introduction, Computer Aided Design 29 (4), 255-268, 1997.
4. ^ Chikofsky, E.J.; J.H. Cross II (January 1990). "Reverse Engineering and Design Recovery: A Taxonomy in IEEE Software". *IEEE Computer Society*: 13–17.
5. ^ Warden, R. (1992). *Software Reuse and Reverse Engineering in Practice*. London, England: Chapman & Hall. pp. 283–305.
6. ^ Chuvakin, Anton; Cyrus Peikari (January 2004). *Security Warrior* (1st ed.). O'Reilly.
7. ^ "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.

8. ^ See Pamela Samuelson and Suzanne Scotchmer, "The Law and Economics of Reverse Engineering", 111 *Yale Law Journal* 1575-1663 (May 2002).
9. ^ Wolfgang Rankl, Wolfgang Effing, Smart Card Handbook (2004)
10. ^ T. Welz: Smart cards as methods for payment (2008), Seminar ITS-Security Ruhr-Universität Bochum, "http://www.crypto.rub.de/its_seminar_ws0708.html"
11. ^ David C. Musker: Protecting & Exploiting Intellectual Property in Electronics, IBC Conferences, 10 June 1998
12. ^ Sega v. Accolade, 203 F.3d 596 (9th Cir. 1993)

External links

- Article on legal considerations (http://www.jenkins-ip.com/serv/serv_6.htm) by David C. Musker

Retrieved from "http://en.wikipedia.org/wiki/Reverse_engineering"
Categories: Engineering | Production and manufacturing | Computer security | Patent law
Hidden categories: All articles with unsourced statements | Articles with unsourced statements since April 2009 | Articles with unsourced statements since April 2007 | Law articles needing expert attention | Articles needing expert attention since April 2009