RE- ENGINEERING

The **reengineering** of software was described by Chikofsky and Cross in their 1990 paper, as **"The examination and alteration of a system to reconstitute it in a new form"**. Less formally, reengineering is the modification of a software system that takes place after it has been reverse engineered, generally to add new functionality, or to correct errors.

This entire process is often erroneously referred to as reverse engineering; however, it is more accurate to say that reverse engineering is the initial examination of the system, and reengineering is the subsequent modification.

Re-engineering is mostly used in the context where a legacy system is involved. Software systems are evolving on high rate because there more research to make the better so therefore software system in most cases, legacy software needs to operate on a new computing platform. '**Re-engineering'** is a set of activities that are carried out to re-structure a legacy system to a new system with better functionalities and conform to the hardware and software quality constraint.

FORWARD ENGINEERING

Forward engineering is the opposite of reverse engineering. In forward engineering, one takes a set of primitives of interest, builds them into a working system, and then observes what the system can and cannot do.

Forward engineering is the foundation of synthetic psychology (Braitenberg, 1984; Dawson, 2004; Pfeifer & Scheier, 1999). Braitenberg has argued that forward engineering is likely to produce simpler theories than reverse engineering because the latter tends to attribute behavioural complexities to the internal mechanisms of the agent. Braitenberg calls this the law of uphill analysis and downhill synthesis. There is a long history of forward engineering showing that simple agents can produce surprisingly complicated behaviours when they are embedded in an interesting environment

REVERSE ENGINEERING

(Reverse Engineering is Reverse Forward Engineering)

It is a key component of software RE-Engineering, which is any activity that improves understanding of software, prepare or improves software, usually for increased Maintainability, reusability or evolvabilty

Reverse engineering is traditionally been defined by **ARNOLD as a "two step process "**of 1)information extraction followed by 2)information abstraction.

A new approach to reverse engg.was **advocated by Scott Tilley** in 1995 that refines this TRADITIONAL two step approach of information extraction an d abstraction as:

1) MODEL: Construct domain specific models of the application using Conceptual Modeling techniques.

2) EXTRACT: Gather the raw data from the subject system using appropriate Extraction mechanism.

3) ABSTRACT: create abstraction that facilitates program understanding and permits the navigation analysis and presentation of resultant information structures.

GOALS OF REVERSE ENGINEERING:

1) Facilitating software reuse.

2) Coping with complexity.

3) Generating alternate views.

4) Recovering lost information.

5) Detecting side effects.

6) Synthesizing Higher level abstraction.

GEN. Defi: Reverse engg. Is a part of Re-Engg process to develop better understandability of a system.

WHEN, WHAT IS DONE???

System spec----------→new system(FORWARD ENGG)

Source code ----------→system design or spec(REVERSE ENGG)

Existing s/w system----------→ more maintainable sys.(RE ENGINEERING).

Reasons for Reverse Engg.:

1) There is in adequate document of the original design.

2) The original design has been lost or never existed.

3) Some bad features of a product need to be designed out.

4) Strengthen the good features of s/w product.

5) To analyze the good and bad features of competitors product.

6) To explore new avenues to improve s/w product performance and features.

TYPES OF REVERSE ENGG:

1) RE –DOCUMENTATION.

2) STRUCTURAL REDOC.

3) DESIGN RECOVERY.

Explanation of types of reverse engg.:

1)Re-documentation:

It is one of the oldest form of reverse engg.

One way of producing accurate documentation for an existing s/w is through Re-documentation.In lack of proper doc only reliable and objective information is the source code itself. Re doc is the process of retro activity providing doc for an existing s/w system. One can think of it as a transformation from source code to pseocode and /or process specification. The latter of which is considered to be at higher abstraction level than the former.

2) Structural Redocumentation:

For large legacy system, an understanding of the structural aspect of system architects is more important than any single algorithmic component. Program understanding is especially problematic for software engineer and technical managers. The doc that exist usually describes the isolated parts of the system,it does not describe the overall architecture. Moreover the doc

is often scattered throughout the system and on different media. Manually creating just one such architectural document is always difficult, creating the necessary documents that describes the architecture from multiple points of view is often impossible. Structural Redocumentation does not involve physically restructuring the code.

3) Design Recovery:

Design recovery is a sub-area of reverse engg. That uses domain knowledge, external and /or informal information to aid program understanding. Its aggressive aim is to reproduce all the info needed for someone to fully understand the subject system. Teleological maintenance is related to design recovery which is attempted to recover info from subject system based on specific user model.Example of reverse engg.

1) Antivirus.

2) Studying exe of any system to obtain it's all documentation or specification.

RE Engineering	
subject	**fact**
reengineering	**has definition** A type of maintenance performed to improve the design of some part of a software system, in general so that it has higher maintainability. In general, no *new* features are added *for* users
	has purpose to increase maintainability of a system
	has part forward engineering
	has part refactoring
	includes • cleaning up the code to make it more readable • completely replacing a layer • re-factoring part of the design
	is a kind of maintenance
	is a kind of software engineering
	reduces long-term costs
	should make the system more amenable to adding features in the future
	should not include adding any *new* features *for* users
maintenance	**is part of** software engineering

software engineering	**has challenge** accurately forecasting how much time it will take either to develop a system, or to make a specific set of changes
	has definition from the Canadian Standards Association The systematic activities involved in the design, implementation and testing of software to optimize its production and support
	has definition from the *IEEE* (1) The application of a systematic, disciplined, quantifiable approach to the development, operation, maintenance of software; that is, the application of engineering to software. (2) The study of approaches as in (1)
	has goal solving customers' problems
	has part ensuring that maintenance and evolution of software is done in a systematic way
	involves applying well-understood techniques in an organized and disciplined way
	involves the translation of higher-level designs into particular programming languages
	is / labour-intensive
	is highly iterative
	is undergoing development in its technology and techniques
	is normally organized into software projects
	sometimes consists of developing completely *new* software
	uses resources such as the time and money of the stakeholders, and the CPU-time and memory of computers
	usually consists of modifying software that has been already written - *this* is because software is normally continually changed over a period of years until it becomes obsolete

Forward Engineering	
subject	**fact**
forward engineering	**has definition** Moving from requirements to design or design to implementation
	is part of reengineering
	is performed during reverse engineering
	is a kind of software engineering
software	**has challenge** accurately forecasting how much time it will take either to

engineering	develop a system, or to make a specific set of changes
	has definition from the Canadian Standards Association The systematic activities involved in the design, implementation and testing of software to optimize its production and support
	has definition from the *IEEE* (1) The application of a systematic, disciplined, quantifiable approach to the development, operation, maintenance of software; that is, the application of engineering to software.
	has goal solving customers' problems
	has part ensuring that maintenance and evolution of software is done in a systematic way
	has part evolution
	has part maintenance
	has part managing software projects
	has part programming
	has part programming
	has part project management
	involves applying well-understood techniques in an organized and disciplined way
	involves the translation of higher-level designs into particular programming languages
	is / labour-intensive
	is highly iterative
	is undergoing development in its technology and techniques
	is normally organized into software projects
	sometimes consists of developing completely *new* software
	uses resources such as the time and money of the stakeholders, and the CPU-time and memory of computers
	usually consists of modifying software that has been already written - *this* is because software is normally continually changed over a period of years until it becomes obsolete

EXTRA INFO: REVERSE ENGG

Reverse engineering is taking apart an object to see how it works in order to duplicate or enhance the object. The practice, taken from older industries, is now frequently used on computer hardware and software. Software reverse engineering involves reversing a program's machine

code (the string of 0s and 1s that are sent to the logic processor) back into the source code that it was written in, using program language statements.

Software reverse engineering is done to retrieve the source code of a program because the source code was lost, to study how the program performs certain operations, to improve the performance of a program, to fix a bug (correct an error in the program when the source code is not available), to identify malicious content in a program such as a virus or to adapt a program written for use with one microprocessor for use with another. Reverse engineering for the purpose of copying or duplicating programs may constitute a copyright violation. In some cases, the licensed use of software specifically prohibits reverse engineering.

Someone doing reverse engineering on software may use several tools to disassemble a program. One tool is a hexadecimal dumper, which prints or displays the binary numbers of a program in hexadecimal format (which is easier to read than a binary format). By knowing the bit patterns that represent the processor instructions as well as the instruction lengths, the reverse engineer can identify certain portions of a program to see how they work. Another common tool is the disassembler. The disassembler reads the binary code and then displays each executable instruction in text form. A disassembler cannot tell the difference between an executable instruction and the data used by the program so a debugger is used, which allows the disassembler to avoid disassembling the data portions of a program. These tools might be used by a cracker to modify code and gain entry to a computer system or cause other harm.

Hardware reverse engineering involves taking apart a device to see how it works. For example, if a processor manufacturer wants to see how a competitor's processor works, they can purchase a competitor's processor, disassemble it, and then make a processor similar to it. However, this process is illegal in many countries. In general, hardware reverse engineering requires a great deal of expertise and is quite expensive.

Another type of reverse engineering involves producing 3-D images of manufactured parts when a blueprint is not available in order to remanufacture the part. To reverse engineer a part, the part is measured by a coordinate measuring machine (CMM). As it is measured, a 3-D wire frame image is generated and displayed on a monitor. After the measuring is complete, the wire frame image is dimensioned. Any part can be reverse engineered using these methods.

DISTINCTION BETWEEN REVERSE , FORWARD and RE- ENGG(Brief Overview)

REVERSE ENGG	FORWARD ENGG.	RE ENGINEERING
It performs transformation from a lower abstraction to higher one.	The traditional process of moving from high-level abstractions and logical, implementation-independent designs to the physical implementation of a system.	Re-engineering is to examine the finished product and build it again, but better
Usually done when docs are not appropriate or is missing	Modification of the system is done. E.g 1) use of different programming lang. 2) introduction of new DBMS 3) Transfer of s/w to new h/w platform.	Usually done for improvements in the system.
Reverse engineering is finding out how a product works from the finished product.	*In **forward-engineering**, one designs a machine to do something*	After finding out how the product works it is done or implemented with new technology or methodology.
Reverse Engineering is trying to recreate the source code from the compiled	Forward engineering is normal engineering. It builds devices that can do	Re-Engineering on the other hand is creating a new piece of software

code. That is trying to figure out how a piece of software works given only the final system.	certain useful things for us: bridges, furnaces, cars, planes. It is forward, because WE build the devices (by applying the principles of physics and previous engineering).	with similar functionality as an existing one. But you may be "improving" the way it was built.
Reverse-engineering is taking apart a finished product for the purposes of learning how it works.	**usually consists of** modifying software that has been already written - *this* is because software is normally continually changed over a period of years until it becomes obsolete.	Re-engineering is simply a new re-implementation of a product with better engineering.
Reverse engineering is to take a bridge apart to see how it was built.	**sometimes consists of** developing completely *new* software	Re-engineering is to throw a bridge away and rebuild it from scratch
Any activity that requires program understanding at any level may fall within the scope of reverse engineering.		REENGINEERING may be useful for the modification of the legancy code or software