Reverse Engineering

Dhairya Shah[1], Vrutik Shah[2], Anuj Sarda[3]

[1]Student, Dept. of Information Technology, K. J. Somaiya College of Engineering, Maharashtra, India
[2]Student, Dept. of Information Technology, K. J. Somaiya College of Engineering, Maharashtra, India
[3]Student, Dept. of Information Technology, K. J. Somaiya College of Engineering, Maharashtra, India

---***---

Abstract - *Conventional software engineering is mainly centered on the development and testing of new software. Reverse engineering allows programmers to study and reevaluate software developed by other people. For system renovation and program understanding, the information that we need can be drawn out from existing software and this approach is called Reverse Engineering. The goal of reverse engineering is to create a new item based on an existing one for which 3D CAD is not accessible. The purpose of this article is to present reverse engineering, what we need to know about it, and how we may re-manufacture any product. Reverse engineering is a form of engineering that prioritises the use of an existing item. It is possible to gain a better knowledge of requirement elicitation and a clear concept of what must be kept and what must be re-used by employing Reverse Engineering. The end goal is to construct a new item that is comparable to the old one. The RE method is now widely used in a variety of disciplines, including manufacturing engineering, software engineering, the film and entertainment sector, chemical engineering, technical industry, etc.*

Key Words: **Reverse Engineering, Legacy Systems, 3D CAD, Software, Manufacture, Testing, Software-Engineering**

1. INTRODUCTION

Engineering practice is prone to focusing on the execution and design of a product without taking into account its long-term viability. After growth, software engineering organizations focus their efforts on maintaining systems to eliminate existing errors and adapt them to changing requirements.

Since the corporate process has become so interconnected with computer programs, any disruption or downtime of the computing system will result in massive losses. The legacy systems are the systems that are usually very complex and large and they have developed over time.

Businesses cannot get rid of these legacy systems very easily. But, due to the change in the structuring of the system and rules of business, the requirements get altered over intervals or periods. Regrettably, design documentation for mature systems is often wrong, incomplete, or even non-existent. It is very tough to find out what the machine is doing, why it's doing it, how work is done, and why it's designed the way it is. Therefore, there is a requirement for these systems to be preserved and sustained.

Reverse engineering is the practice of replicating an original component or system without the use of templates, manuals, or a virtual model. It is useful for the betterment of own products as well as analyzing a competitor's product. It fundamentally is based on the reconstruction of design models related to a real product. The aim of reverse engineering is to replicate a product by going back to the effects of the initial design process. It also helps the systems to have proper upkeep of these legacy systems.

There are many tools, logical structures, and physical models that can be used in the process of Reverse Engineering. In software development, reverse engineering is called a maintenance phase in the whole life cycle of development.

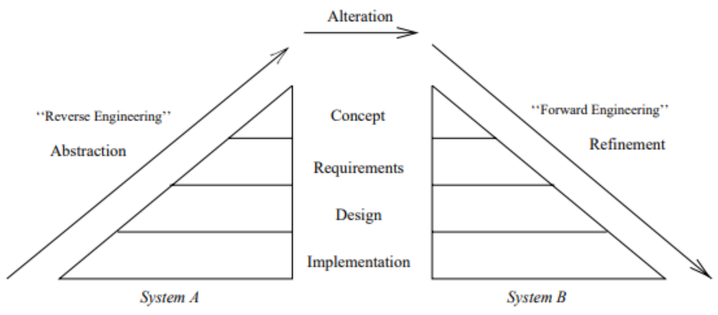


Fig - 1: Reverse Engineering Process Model [15]

2. PURPOSE OF REVERSE ENGINEERING

Reverse engineering is used for a variety of applications, including creating innovative and simple consumer options. It is used to make applications more efficient to use and discover undocumented functionality in legacy programs. The purpose of Software Reverse Engineering is to understand the design principles of source code to make systems compatible so they can work together. It is done to test one's own code to consider its shortcomings and to see if someone else has simply copied aspects of one's own technology.

- **Interoperations among Code**

Code that needs to be analyzed is referred to as interoperability. Examples include applications that require operating system interoperability and software-controlled exchanges.

- **Correcting and Updating**

Software must be modified or corrected to meet current needs, as determined by the programmer, particularly in the case of insufficient documentation.

- **Documentation misplaced**

It's sometimes done when a system's documentation is lost or when the developer is no longer around.

- **Analysis of Source Code**

It is necessary to study a code to see how it runs, what components it contains, and how it operates.

- **Knowledge**

It should be used to learn from the errors of others. The analyzer should not make these errors so that they can be corrected.

- **Commercial or Military espionage**

It may be used to steal or capture a prototype and dismantle it to learn about an enemy's or competitor's new study.

- **Cracking**

Security is an activity undertaken by crackers in which it plays an important part and is often used by crackers.

- **Creating unapproved and unlicensed duplicates**

It is used to make unauthorized duplicates. [14]

3. OBJECTIVES

The primary goal of reverse engineering a software framework is to improve the general readability of the system for both maintenance and new growth. There are five primary goals:

1. Work with complexity

We must build different methods to deal better with the large volume and intricate designs of systems. A way to control these features is automated support. When reverse-engineering techniques and technologies are combined with CASE conditions, we will be able to extract valuable and related knowledge, allowing decision-makers to monitor the process and product in the evolution of systems.

2. Generate new views

Building and maintaining graphical representations still remains to be a hindrance in the process. Reverse-engineering tools enable you to create or re-create graphical representations from other formats. While many designers work from a single, first viewpoint (such as dataflow diagrams), reverse-engineering techniques can help the analysis and verification process by generating additional new views from other perspectives (such as control-flow diagrams, structure maps, and entity- relationship diagrams). Reverse engineering techniques can also be used to create non-graphical representations, which are an important part of machine documentation.

3. Detection of side effects

Unintended outcomes and side effects can obstruct a system's output in a variety of ways due to both chaotic or unplanned initial design and constant modifications. RE will assist in the identification of anomalies and issues until they are identified as bugs by users.

4. Synthesize higher abstractions

RE necessitates methods and techniques for producing various views that cut through higher levels of abstraction. In the software community, there is debate over how fully the system can be automated. Expert device technology will undoubtedly play a key role in realizing the full potential of high-level abstraction generation.

5. Facilitate reuse

The large body of accumulated computing infrastructure is a significant barrier to software reuse. RE may assist in the selection of applicants for reusable device elements of existing programs. [9]

4. PHASES OF REVERSE ENGINEERING

There are four phases of Reverse Engineering:

1) Context Parsing Phase

Context parsing is the first step in evaluating source code, and it collects program knowledge related to syntax and other semantics. Compiler methods such as syntax processing and token parsing are discussed here. The parser compiles programming source code and then converts it into a more formal representation as an intermediate form during this process.

2) Component Analyzing Phase

This phase uses the technique of the intermediary functions or programs, i.e., the feedback from the previous phase, to uncover the component's objects. Call graphs, function information, structure chart, program slices, variables, data flow, attributes, control dependencies, and definition-use graphs are examples of these objects.

During this stage, several previous researchers have given methods and techniques for obtaining control-flow diagrams, data-flow diagrams, and other related information. The rationale for this is that the data was filtered during implementation. Most reverse engineering technologies support this phase with methodologies or systemic techniques.

3) Design Recovery Phase

The implicit knowledge information scattered throughout the program code is one of the major challenges to developing automated tools for extracting design and domain knowledge from programs. In a nutshell, extracting original specifications and/or design experience from program source code is often a difficult task. Now, using methods that blend knowledge and structural representation, analyzers can effectively extract some wide-ranging information from the results of the previous phase.

4) Design Reconstructing Phase

System models and design requirements obtained in the previous phase can be analyzed and integrated with this phase to recreate a brief view of the model. This model provides not only a limited set of features and behavior of the system but also the correct art and technology that may have been present in the original execution, resulting in destined and re-development work. Therefore, to fulfill this task, reverse engineering provides a fourth phase to entirely accomplish the work. [1,14]

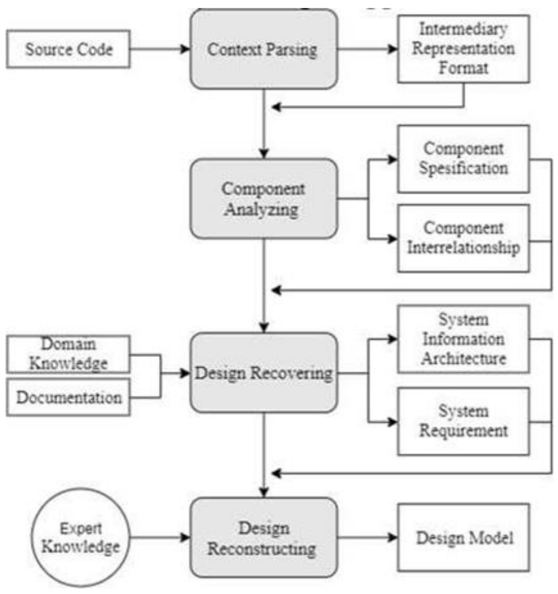


Fig -2: Phases of Reverse Engineering [14]

5. APPLICATION

Reverse engineering is used in a wide range of computer science fields as well as other fields. It is mainly used because of its efficiency, cost-saving, and time-saving scheme.

- **For military applications, reverse engineering is widely used**

Reverse engineering is often used by militaries to replicate other countries' inventions, gadgets, and knowledge for intelligence operations.

- **Reverse Engineering Java Applications Using UML**

Importing data from a Java application into a UML model allows for reverse engineering. Learned how to draw an application diagram, sequence diagram, a viewing tool, a class diagram, and a dependency diagram.

- **Reverse Engineering for Automated Protocol**

Some approaches try to create such a protocol specification by both dynamic and static analysis of the software that builds and delivers that data, rather than by reverse-engineering the protocol as it is.

- **Machines using Reverse Engineering**

Reverse engineering has become a feasible technique for creating a 3D virtual model of an existing physical component of any system as the computer-aided design has grown in popularity. The process of reverse engineering entails measuring and reconstructing an object as a 3D model. Various devices, such as CMMs, 3D scanning technologies, organized light digitizers, and laser scanners can be used to measure physical objects.

- **Software reverse engineering**

It's also possible to think of it as going backward in the growth cycle. In an inversion of the conventional waterfall model, the output of the implementation phase is reverse engineered back to the analytical phase.

- **Reverse Engineering in Industries**

The production rate of the production system is directly dependent on the resiliency and quality of the tool material. The structure of the component can be changed by using reverse engineering to get the optimum production rate. The engineer can complete a product concept design based on the requirements of its function and then use some soft materials, such as plaster or wood, etc., to fabricate models.

- **Integrated circuits/smart cards reverse engineering**

Reverse engineering is a method of reviewing a smart card that is harmful. The attacker removes the magnetic stripe layer by layer and captures it with an electron microscope. The attacker's biggest challenge is to place everything in the proper order to figure out how it works.

- **Legality**

Even though a trade secret protects an artifact or process, reverse engineering is frequently legal as long as it is acquired legally. Patented objects, on the other hand, do not require a full release of an invention, so they do not have to be reverse-engineered to be analyzed. [10,14]

6. TOOLS FOR REVERSE ENGINEERING

To study a process or technique scientists have developed different tools. Tools can be ranging from a small prototype to tools applied in real-life applications.

Tool building is an important step in understanding the feasibility of any process. It is a costly and difficult task; hence requires some focus and understandability.

Earlier, tools were built with little or no reuse. Tool building turned out to be a high-cost process as everything was created from scratch. A new approach called component-based tool building was devised to overcome this. In component-based, ready-to-use software components were used.

A. Requirements

Scalability: The tool must be able to handle an enormous amount of data competently. A graph visualizing software should have the potential to display hundreds of curves and nodes.

Interoperability: An interoperable tool should be capable of exchanging information with other tools. A set of tools will be at hand to execute a specific reverse engineering task.

Customizability: Every reverse engineering technique is different in its own way and application. Therefore, it is significant that the tools be customizable and meet the on-going process requirements.

Usability: Tools that are often easy to use are referred to as usable. User-friendly tools are more popular and do not infer additional development costs from continuous user grievances.

Adoptability: A useful tool is adopted by the customers or users. Tools not supporting important tasks do not progress further as they fail to meet user requirements. [2]

B. Examples

I. Scanning

Recognizing implementation choices necessitates non-linear, in- depth access to the source code. When a decision is guessed, it is often important to validate it by checking relevant pieces of code. Furthermore, the coding must be addressed so that the decision's particulars can be protected and a brief shown in its place.

A large portion of the choices is made by understanding syntactic variations in program structures and variable use. Their identification requires the same processing that happens during the early stages of a software compilation. In particular, the objects of program interpretation, such as the abstract syntax tree and the symbol table, can be used as a source of data.

II. Browsers/Hypertext

Browsing operations are carried out whenever a program is correctly evaluated and processed in an organized manner. The abstract syntax tree, in particular, will assist in understanding the hierarchical design of the program code. Similarly, symbol table information can be used to help reinforce cross-reference-related suspicions.

High bandwidth screens and direct modulation interfaces are used to evaluate non-linear text organization. The text is source code in this case, and the non-linear relationships are given by the 5 parsers.

III. Object Server

The knowledge framework resulting from detected design decisions must be saved in an archive for use by program maintainers. While current database structures will serve some of the organization, they are insufficient. In the same way that CASE tools are turning to preventive-oriented databases and object repositories to aid in the advancement of engineering practices.

IV. Task-Oriented Tools/Debugging

Once an understandable information system has been filled with application information, resources applicable to a specific software management activity may be used. The data model and knowledge structure are required for an integrated set of resources. Find the following debugging tools as an example.

The software maintainer begins with a problem report indicating that a program is generating unpredictable results on a given run. The maintainer wants to rapidly isolate the issue to a small section of code. He employs a method that shows which statements are theoretically responsible for the issue for a given range of true and invalid performance values.

To control the required statements, the tool explores the dependence relationships between program statements and the implementation history of the program. In a given case, the tool has a feature that shows only similar phrases. In this way, the maintainer will concentrate on the relevant code.[7]

C. Case Studies

REOffice and REVisio

Their major application is to analyze and display graphs and statistics of reverse engineering data. As the name suggests REOffice is built on Microsoft products like Excel and PowerPoint while REVision is built on Vision.

SVG graph editor

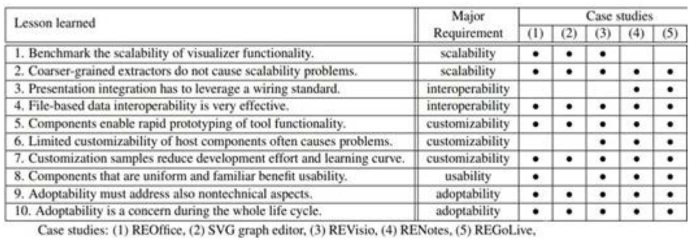
The SVG chart manager controls the SVG vector graphic format to embed diagrams into assorted reports like PPT, Word and so on With around 6000 lines of Java code, the SVG diagram manager is executed.

RENotes

Their major application is to analyze and display graphs and statistics of reverse engineering data. As the name suggests REOffice is built on Microsoft products like Excel and PowerPoint while REVision is built on Vision.

REGoLive

It is a tool that is developed for the reverse engineering of websites, unlike others that were designed for traditional reverse engineering tasks. [2]



Lesson learned	Major Requirement	Case studies				
		(1)	(2)	(3)	(4)	(5)
1. Benchmark the scalability of visualizer functionality.	scalability	•	•	•		
2. Coarser-grained extractors do not cause scalability problems.	scalability	•	•	•	•	•
3. Presentation integration has to leverage a wiring standard.	interoperability				•	•
4. File-based data interoperability is very effective.	interoperability	•	•	•	•	•
5. Components enable rapid prototyping of tool functionality.	customizability	•	•	•	•	•
6. Limited customizability of host components often causes problems.	customizability			•	•	•
7. Customization samples reduce development effort and learning curve.	customizability	•	•	•	•	•
8. Components that are uniform and familiar benefit usability.	usability	•		•	•	•
9. Adoptability must address also nontechnical aspects.	adoptability	•	•	•	•	•
10. Adoptability is a concern during the whole life cycle.	adoptability	•	•	•	•	•

Case studies: (1) REOffice, (2) SVG graph editor, (3) REVisio, (4) RENotes, (5) REGoLive.

Fig -3: Lessons Learned from Case Studies [2]

7. SPECIALIZATION OF REVERSE ENGINEERING

I. Redocumentation

This is not the strongest form of reverse engineering. It includes just the development of programming documentation at a similar degree of deliberation.

II. Design Rediscovery

It is used to create a model of the higher level of abstraction. It involves redocumentation but requires field knowledge and external information.

III. Restructuring

Lateral change of the system inside a similar degree of abstraction. Maintains the same degree of functionality and semantics.

IV. Reengineering

The most revolutionary and expansive extension. By and large, incorporates a blend of figuring out for perception, and a reapplication of forwarding designing to re-survey which functionalities should be held, erased, or added. [3]

8. CHALLENGES AND DIFFICULTIES

A. Challenges

Domain of Application – Language of Programming
Devices are available to comprehend what the code is doing from the code's perspective however there isn't anything to help the reverse engineer figure out what's going on with the code from the application's perspective.

Machines and Programs – Abstract, High-Level Design
Writing computer programs is essentially about planning from the theoretical to the definite execution, yet there isn't anything to help in the opposite planning.

Unique Coherent, Structured System – Actual System
Keeping up the design for quite a while makes the construction stray from the first detail. The reverse engineer should have the option to unite and coordinate with the plan in the record and the one carried out by him.

Hierarchical Programs – Cognitive Association

People don't think about everything simultaneously except in "pieces" of information. A reverse engineer should have the option to create the right significant level lumps from the low-level realities expressed in the program. [3]

B. Difficulties

Reverse engineering has many difficulties. RE has difficulties because it has to do the work of bridging different worlds. The important bridges are the following five gaps.

- The distance between a problem in one application domain and its solution in another.
- The chasm that exists between the solid reality of real machines and computer systems and the vague world of high-level explanations.
- The discrepancy between the ideal cogent and highly organized system definition and the real system, the structure of which may have decomposed over time.
- The discrepancy between the world of ranked systems and the associational essence of human cognition.
- The distinction between a bottom-up study of the source code and a top-down synthesis of the program summary [7]

9. INNOVATIVE TECHNIQUES

Digitizing

Digitizing processes allow the conversion of real-world component surfaces to digital representations. The primary aim of digitizing is to scan the part in space points and output the results in CAD applications. An important type of digitizing process is 3D scanning. The 3-axes mechanical configuration, probe head, control unit, and desktop are vital device components. CMMs have become a crucial part of measuring tools. The CMM is a type of Cartesian robot, which includes a touch-trigger probe instead of a gripper.
The Pro/Engineer Wildfire CAD/CAE/CAM embedded framework is the most dependable technology program used. In this process, real objects are scanned using a scanner, which calculates the dimensions of the object and displays it on a laptop monitor in the form of a point cloud. The point cloud is a set of the main points on the object's surface.
As a consequence, in this process, the spatial geometry of the product is converted into a 3D point cloud of the object's shape. 3D scanning is a method that helps one to migrate and use scanned points from space to CAD applications. There are other forms of digitizing equipment that can promote this conversion process. The main types are Optical, Laser, Contact, Destructive4.

Segmentation of the acquired data

The CMM's output data are the coordinate values of the probe's middle and normal vectors in the X, Y, and Z directions at the location of the tactual point. Furthermore, the calculation data performance pattern does not satisfy the criteria for the generation of a CAD model. The format of the calculation data result must be translated and segmented into small sections that Pro/Engineer applications can tolerate. The created data can be used directly to create a CAD model of the component.

Knowledge extraction (feature recognition)

After analyzing the measurement data and transforming the data pattern, the CAD model is constructed directly from the measurement data using Pro/Engineer CAD/CAE/CAM tools. To model the surface of the component in a CAD model, we must choose surface features from the cloud of points collected by digitization. Ground segments and borders are used in the surface features. The 2 models are the Curve mode and Surface model.

Reconstruction of the 3D model

Divide the item into two sections, upper and lower, to make it more feasible and simpler to deal with. The mesh isn't perfect; it has some holes and some surface damage. The mesh can be restored using a feature in the Strong Works app. It consists of filling the gaps and repairing the damaged bits. Finally, the mesh seems to be much smoother than before, and it is the revised edition. [10]

10. CURRENT AND FUTURE SCOPE

Software reverse engineering is now at a very advanced stage. Because technological advancements are advancing at such a fast pace, such types of projects are becoming more common. Many software firms currently use reverse engineering to reduce the cost and time required to implement new techniques.

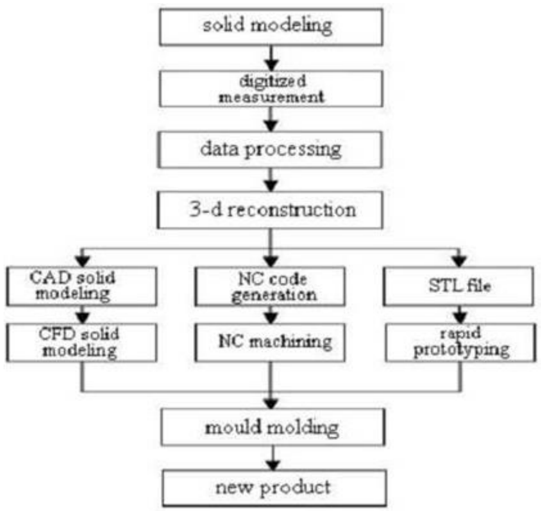


Fig - 4: Flowchart of Reverse Engineering [4]

Various tools have been developed in recent years that perform reverse engineering techniques partly or completely. Software restoration is the process of bringing yesterday's software up to date with today's technology.

When working on a legacy system, it's also necessary to understand the business rules, which necessitates some reverse engineering steps. [14]

11. CONCLUSIONS

Reverse engineering talks the elite territory of support, where cost is caused on the grounds that it was not done right the first run through. The objective is to improve the strength and security of the reverse engineering measure results. It is likewise used to recuperate design and significant information from a product framework.

This paper presents a few odds of utilization and benefit from using the RE strategies in assembling measure, particularly for those situations for which there is shortage of the item drawings-it must be re-established and there is absence of shortage of supply.

It also depicts the elements why Reverse Engineering is troublesome. By utilizing Reverse Engineering, it assists with improving comprehension of prerequisite elicitation and a reasonable comprehension of what should be held and what should be re-utilized.

REFERENCES

[1] Syed Ahsan Fahmi and Ho-Jin Choi, "Software Reverse Engineering to Requirements", International Conference on Convergence Information Technology,2007.

[2] Holger M. Kienle. "Building Reverse Engineering Tools with Software Components: Ten Lessons Learned", 14th Working, Conference on Reverse Engineering (WCRE)2007.

[3] Michael L. Nelson. "A Survey of Reverse Engineering and Program Comprehension". ODU CS 551 - Software Engineering Survey,1996.

[4] G.Sreeram Reddy1, Manzoor Hussian2, K.Srinivasa Rao3 "Latest Research on Reverse Engineering Technology: Review", Proceedings of the International Conference on Paradigms in Engineering Technology (ICPET),2016.

[5] Masaaki Komatsu, Hideaki Aburatani, Sanit Teawhim "Application of the Reverse Engineering as an early Engineering Education", IEEE REGION 10 CONFERENCE (TENCON)Osaka, Japan, November 16-19, 2020.

[6] Cristo´bal Costa-Soria1, Manuel Llavador2, Mar´ıa del Carmen Penade´s3. ”An Approach for Teaching Software Engineering through Reverse Engineering”, EAEEIE Annual Conference, 2009.

[7] Spencer Rugaber ”White paper on reverse engineering”. Georgia Institute of Technology,1994.

[8] Kevin Mobley” Reverse Engineering for Software Performance Engineering”, 14th Working Conference on Reverse Engineering (WCRE),2007.

[9] Elliot 1. Chikofsky, James H. Cross It ”Reverse engineering and Design Recovery: A Taxonomy ”. Index Technology Corp. and Northeastern University, 1990.

[10] Pankaj Kumar and Ashok Madan.“Review Paper on Reverse Engineering”, Journal of Basic and Applied Engineering Research, 2015.

[11] Shikun Zhou, Hongji Yang, Paul Luker, Xudong He.“ A Useful Approach to Developing Reverse Engineering Metrics”, Twenty-Third Annual International Computer Software and Applications Conference (Cat. No.99CB37032),1999.

[12] F D Milsom ”Why reverse engineering software?”, IEE Colloquium on Reverse Engineering for Software Based Systems,1994.

[13] Patricia Lutsky.“ Automating Testing by Reverse Engineering of Software Documentation”, Brandeis University, 1995.

[14] Ati Jain, Swapnil Soner, Anand Gadwal.“ Reverse Engineering: Journey from Code to Design”, 2011 3rd International Conference on Electronics Computer Technology, 2011.

[15] Gerald C. Gannod, Yonghao Chen and Betty H. C. Cheng ,“ An Automated Approach for Supporting Software Reuse via Reverse Engineering “, Proceedings 13th IEEE International Conference on Automated Software Engineering (Cat. No.98EX239), 2002.