A STUDY OF SOFTWARE RE-ENGINEERING FOR IMPROVEMENT IN SOFTWARE ARCHITECTURE

Mohit Kumar Sharma
Head and Assistant Professor
Post Graduate Department of Computer Science
J.C. D.A.V. College, Dasuya, Punjab, India

Abstract :* *Software Re-engineering is the procedure of updating the internal mechanisms of a software or program or structure of a software. The purpose of software re-engineering is for improvement in the software internal structure to make it easier to understand. Software Re-engineering involves source code translation, reverse engineering, program structure improvement, program modularization and data re-engineering in process. Existing software and applications may be converted to enhance their functionality to take benefit of present better practice techniques. In this study has been undertaken to analysis of software re-engineering to enhance reliability, robustness, efficiency and lower costs for improvement in software structure in present digital era.

Index Terms – Re-engineering, Benefits, Process, Model

I. INTRODUCTION

Software Re-engineering is the procedure of updating the internal mechanisms of a software or program or structure of a software. The purpose of software re-engineering is for improvement in the software internal structure to make it easier to understand. In re-engineering, working and software internal architecture remains the common, but it involves re-documentation, organization, modification and updating of software [1]. Software re-engineering describes having a re-look at an entity, such as a process, task, designs, approach using software engineering principles for improvements. Software re-engineering used to re-implement software to make more maintainable. This concept concerns five parameters as management philosophy, pride, policy, procedures, and practices for improvements impacting cost, quality, service, and speed [2].

Software Re-engineering involves source code translation, reverse engineering, program structure improvement, program modularization and data re-engineering in process. Source code translation is an automatic conversion of program in one language to another. Restructuring of a system without changing its functionality [3]. Reengineering involves putting in the effort to make it easier for maintenance and re-engineered system may also be restructured and should be re-documented. When requirement to update the software to keep it to the current need, without impacting its working is a software re-engineering.

It is a thorough procedure, where software design is modified and programs are re-coded. Legacy software cannot keep tuning with the latest technology available in the market. When hardware changes then updating of software becomes necessary. Even if software grows old with time, its functionality need to update [4]. When initially Unix was developed in assembly language then C language came into existence, Unix was re-engineered in C, because working in assembly language was difficult. Sometimes developers study a few parts of software need more maintenance than others and requirement of software re-engineering.

II. OBJECTIVES

Every software application has limited age of usage, so software is required to update or re-structure according to need of the user requirements after some time or demand [5]. Software Re-engineering is needed due to -

- Software Maintenance for efficiency
- Software Quality attributes are demanding
- New technology and applications Increasing
- Organizations Competition
- Successful Projects Demanding
- Legacy Software Increasing
- More Robustness
- Cost Effectiveness

III. BENEFITS AND LIMITATIONS OF SOFTWARE RE-ENGINEERING

Software Re-engineering has following benefits [6]-

1. **Reduced Risk –** It is based on incremental improvement of software. There is a more risk in new software development as coding problems, efficient staffing problems and user specification problems.
2. **Reduced cost –** The cost of software re-engineering is often significantly less than the costs of developing new software applications
3. **Incremental Creation -** When hardware changes then updating of software becomes necessary. Even if software grows old with time, its functionality need to change for improvements.

Software Re-engineering has following limitations-

1. **Programming Limits -** There are programming limits to the extent that a software can be improved by re-engineering. It is not possible, for example, to convert a system written using a functional approach to an object-oriented system.
2. **High Cost for Major architectural changes –** If more structural changes of the software data management, cannot be carried out automatically involves high additional cost.

IV. SOFTWARE REENGINEERING PROCESS

The activities in Software Re-engineering process are:

1. **Source code translation –** Software or Program is converted from an old programming language to a new modern version of the same language or to a different language for efficiency [7].
2. **Reverse Engineering –** Software or program is analyzed and information, extracted from it which helps to document its organization and functionality [8].
3. **Program Restructuring –** Internal Control structure of the program is analyzed and updated to make it easier to read and understand.

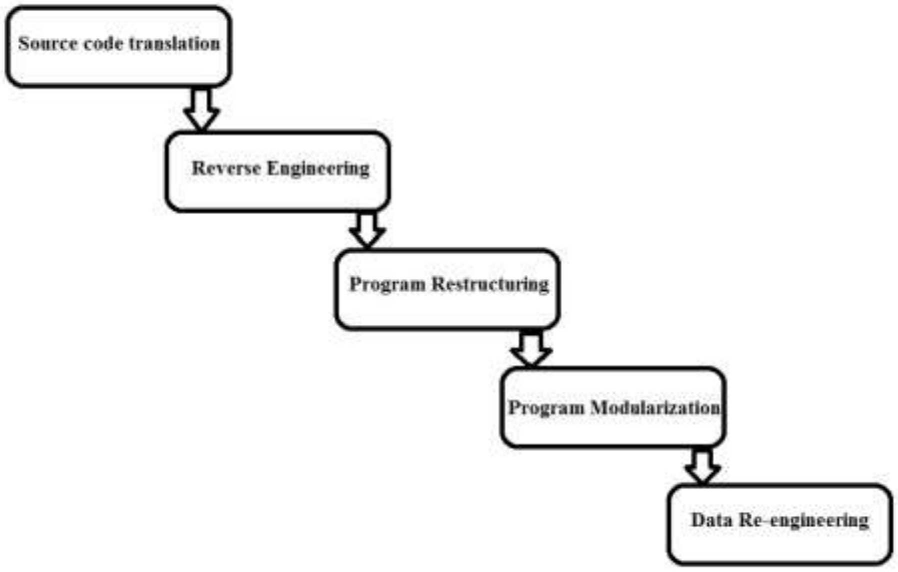

Figure 1 – Software Re-Engineering Process

4. **Program Modularization –** Software modified and related parts of the program are grouped together and, where appropriate, redundancy is removed from structure.
5. **Data Re-engineering -** The data processed by the software or program is modified to reflect program changes for reliability and efficiency.

V. SOFTWARE REENGINEERING MODEL

The activities of the software re-engineering process model are described as follows:

1. Inventory Analysis

Every software company should have an inventory of all software applications. The inventory can be nothing more than spreadsheet model containing information that provides a detailed description of every active application.

2. Document Restructuring

Weak documentation is the trademark of many legacy systems. Creating documentation is far too time consuming. Documentation must be updated, but have limited resources. The system is business critical and must be fully re-documented.

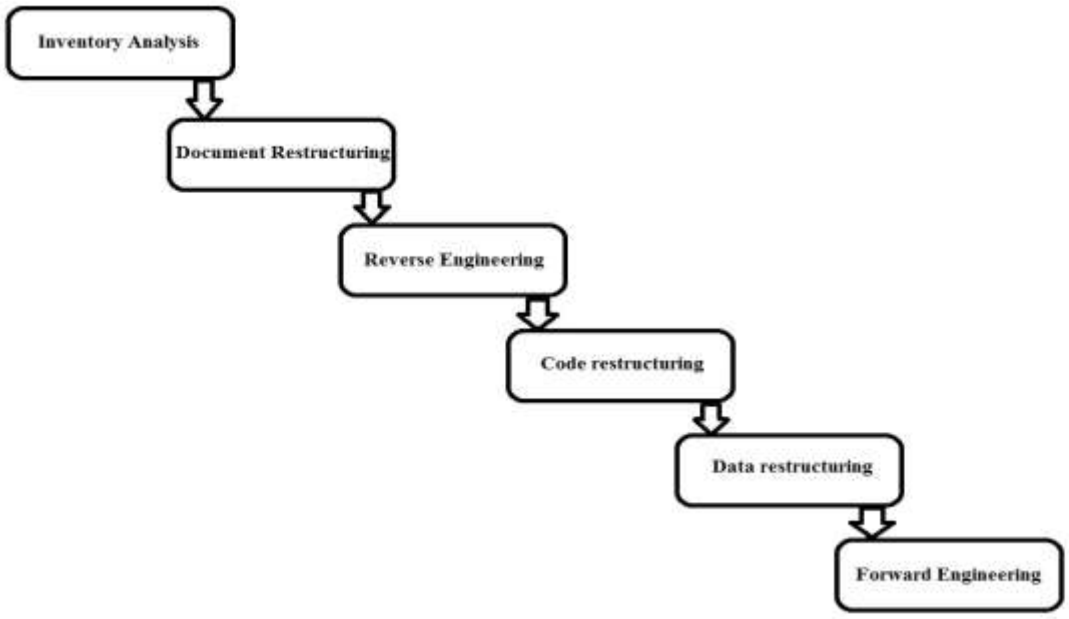

Figure 2 - Software Reengineering Model

3. Reverse Engineering

Reverse engineering is a process of design recovery. Reverse-engineering tools extract data and architectural and procedural design information from an existing program [9,10].

4. Code Restructuring

Some legacy systems have relatively solid program architecture, but individual modules were coded in a way that makes them difficult to understand, test, and maintain.

5. Data Restructuring

Data restructuring begins with a reverse engineering activity. Current data architecture is dissected, and necessary data models are defined. Data objects and attributes are identified, and existing data structures are reviewed for quality.

6. Forward Engineering

Forward engineering, also called renovation or reclamation, not only recovers design information from existing software, but also uses this information to alter or reconstitute the existing system in an effort to improve its overall quality.

VI. CONCLUSION

Software Re-engineering plays very vital role in software maintenance and improvement to enhance efficiency and reliability. Reclamation and reuse engineering emphasize reengineering activities that make the software more reusable. In this study, Software Re-Engineering concepts, benefits, process and model discussed to explore the importance of reusability and re-engineering for Software developing organizations. It helps to reduce cost, reduce risks as well as better use of existing staff. Source code translation is an automatic conversion of program in one language to another. Restructuring of a system without changing its functionality. Reengineering involves putting in the effort to make it easier for maintenance and re-engineered system may also be restructured and should be re-documented.

REFERENCES

[1] Roger S. Pressman, Software Engineering, Tata-McGraw Hill Publishing House, 2009
[2] Ali Behforooz, Software Engineering Fundamentals, Oxford University Press. Frederick J.H, 1996
[3] E.M. Awad, Systems Analysis and Design, Galgotia Publications Ltd, 1993
[4] Anirban Basu, Software Quality Assurance and Testing, Prentice Hall of India, 2015
[5] Glenford J. Myers, Software Reliability Principles, John Wiley & Sons, 2002
[6] Nasib S. Gill, Software Engineering-Software Reliability and Testing, K.B. Publishing, 2016
[7] IEEE, "IEEE Standard Glossary of Software Engineering Terminology", 1990
[8] Ramandeep Singh, "A Review of Reverse Engineering Theories and Tools", International Journal of Engineering Science Invention ISSN (Online): 2319 – 6734, ISSN (Print): 2319 – 6726, Volume 2 Issue 1, January. 2013, PP.35-38
[9] Yijun Yu, "Software Re-engineering", Software Engineering, Spring 2005
[10] Alexandru C. Telea, "Reverse Engineering – Recent Advances and Applications", Published by InTech Janeza Trdine, Rijeka, Croatia
[11] Eilam, Eldad, Reversing: Secrets of Reverse Engineering. Wiley Publishing
[12] James, Dick, "Reverse Engineering Delivers Product Knowledge; Aids Technology Spread". Electronic Design. Penton Media.
[13] Raja, Vinesh; Fernandes, Kiran J., Reverse Engineering - An Industrial Perspective. Springer.