ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijareeie.com

Vol. 6, Issue 2, February 2017

The Role of Software Re-Engineering in Software Engineering

Anurag Dixit

Department of Computer Science and Engineering, Galgotias University, Yamuna Expressway Greater Noida, Uttar Pradesh, India

Email Id: anurag.dixit@Galgotiasuniversity.edu.in

ABSTRACT: Re-engineering is the most commonly used simulation technique to prevent different threats in the design process, in almost every area of the design process. This will be very beneficial in application development processes to get ahead of time uncertainty, expense complexity and most importantly the final product using such methods will be successful for application development processes. Re-engineering is the only way to properly exploit the software and address the software bottleneck issue. Re-engineering is examining, analyzing and changing an existing technology system in order to reconstruct it in a particular form and then implementing the unique form. The paper highlights the importance of software re-engineering and the factors behind such importance accompanied by a review on each of these factors with illustrations to demonstrate that a re-engineering method is a valuable tool for transforming existing, outdated systems into more functional, simplified systems. And re-engineering is used to enhance maintenance, interoperability, efficiency, and testing. Re-engineering is also used to reduce social dependence.

KEYWORDS*:* Interoperability, Maintainability, Software Engineering, Software Re- Engineering

I.INTRODUCTION

Re-engineering is a method that utilizes not only the conventional activities but also a mixture of such operations and a set of spiral model implementation activities to transform an existing structure into a target system.Usually, the process includes a mixture of other procedures like reverse engineering, re-documentation, reorganization, interpretation, and forward design [1].Software life expectancy can be doubled by software program engineers eventually grow old (becomes complicated) and retire but software can be re-produced in the era of rapidly changing through re-engineering, system models are rapidly evolving.Software re-engineering enhances the software's quality, effectiveness and dependability [2]. In the current set of standards Re engineering makes the system unique to an extent.Upkeep of the software begins after software creates correct information to improve the efficiency and other characteristics of the software system, the constant software upkeep affects the software performance and upkeep costs are high, then re-engineering is used.
Management of software project scheduling is the obvious necessity in a day now. Price, performance is the most important thing in projects for controlling the software development time. There is the danger of software failure in every area of the application development phase.So the aim of the process management process should be to prevent failure of the software.Barry Bohem states that "bad management can raise software expenses more quickly than any other aspect." There must be a method for assessing the efficiency of software development, as it is difficult to control any operation without any traditional method for assessing that operation.There are four re-engineering goals: preparedness for stable improvement, improvement of maintenance, integration, and reliability improvement.
Need for Software Re- Engineering
There is requirement for software re- engineering for following reasons-

- Develop legacy software.
- Technology evolving.
- Number of successful projects decreased.
- Increasing the number of organizations competing.
- More competition for the characteristics of quality.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijareeie.com

Vol. 6, Issue 2, February 2017

- People's attitudes shift.
- Assumption on maintaining the Software.

Software Re-engineering is an application development process that is done to change a software system's maintainability [3]. Re-engineering is the analysis and modification of a system that will reconstruct it in a new way.Re-engineering becomes a useful tool for transforming ancient, outdated systems into more effective, simplified systems. But the growth of projects is often strapped for time and resources, making it necessary to consider the alternatives.Reengineering is usually mentioned as "changing business processes." Such a shift places new demands on systems. Reengineering specialist is much increasingly rare than architecture specialists and most technicians have little study experience in this field.

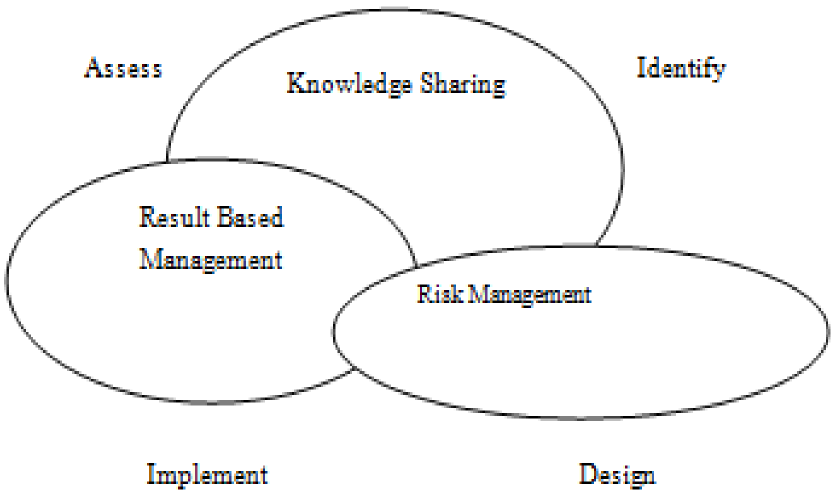

Fig. 1: Software Re- Engineering

II.SOFTWARE RE- ENGINEERING PROCESS

Reengineering takes some time; it charges significant amounts of money, and it consumes resources that might otherwise be taken up for important problems [4]. Reengineering will not be completed in a few years or even a few months, for all the purposes.Information technology reengineering is an exercise that will soak up IT assets for many decades. That is why each company needs a pragmatic software reengineering approach. In a reengineering process template, a functional strategy is contained.Such events happen in a linear series in certain instances but that is not always the situation. For instance, reverse engineering (comprehending a program's inner workings) will have to happen before software reconstruction can begin. It is a cylindrical model this implies that everyone can re-examine each of the operations described as part of the model. After either of these operations, the method may cancel for any specific process.

Inventory Examination- Any company of software must have an inventory of all programs. The inventory can only be a database model with details that offer a complete description of each successful request (e.g., scale, age, market robustness).

Document Restructuring- Failure to report is the hallmark of several existing systems.What are the options offered to address it? It is far too tedious to build the documents.If the system is working everyone is going to be living with what everyone has.Documentation needs to be changed but options are limited. A strategy called "document when touched" is used.The program is essential to company and needs to be properly reconfigured.

Reverse Engineering- Reverse engineering is the method of evaluating software with a view to constructing a program depiction at a greater level of complexity than the source code [5].Reverse engineering is the layout building process.Reverse engineering instruments acquire information from an existing program from information, structural, and procedural layout.

Code Restructuring- Code restructuring is the most popular type of reengineering (in this situation, the use of the word reengineering is controversial, simply).Many computer systems have a relatively strong software design, but specific components have been programmed to make it difficult to recognize, check, and manage it.In such situations, the code can be reorganized within the defendant's components. To perform this task, a debugging tool is used to evaluate the source code. Organized programming build breaches are reported and code reorganized (this can be done manually).The resulting reorganized software is checked and designed to ensure there have been no irregularities. Documentation of the internal code is revised.

Data Restructuring- A program with a feeble design of the data will be difficult to relate and improve.In fact, information design had more to do with an organization's long-term feasibility than the source code itself, for several implementations.Unlike code reorganization, which takes place at a low level of subjectivity, information structuring is a reengineering operation on a large scale.Data reorganization mostly starts with a reverse engineering exercise.Data elements and characteristics are recognized, and performance is checked for current data structures.When the data framework is fragile (e.g., flat documents are being introduced at the moment, when a transactional method would significantly streamline computation), the information will be revised.Because data design has a significant influence on the design of the program and the methodologies that inhabit it, alterations in the information will always lead to changes either architectural or software-level.

Forward Engineering- In a perfect world, programs would be redesigned using an integrated "reengineering engine." The existing system would be fed into the engine, evaluated, reorganized and then revived in a form with the best technology quality features.More importantly, such reengineering instruments are progressively becoming more advanced. Forward engineering is the method of constructing complexities and reduced-level information from a high-level design, or definition. In Information Technology, forward engineering is usually essential, as it reflects the ' normal ' process of creation. This can often lead to loss of technicalities, if the designs are much more comprehensive semantically, or complexity levels. The Steps of Software Re- Engineering Process is shown below in figure 2 Steps of Software Re- Engineering Process.

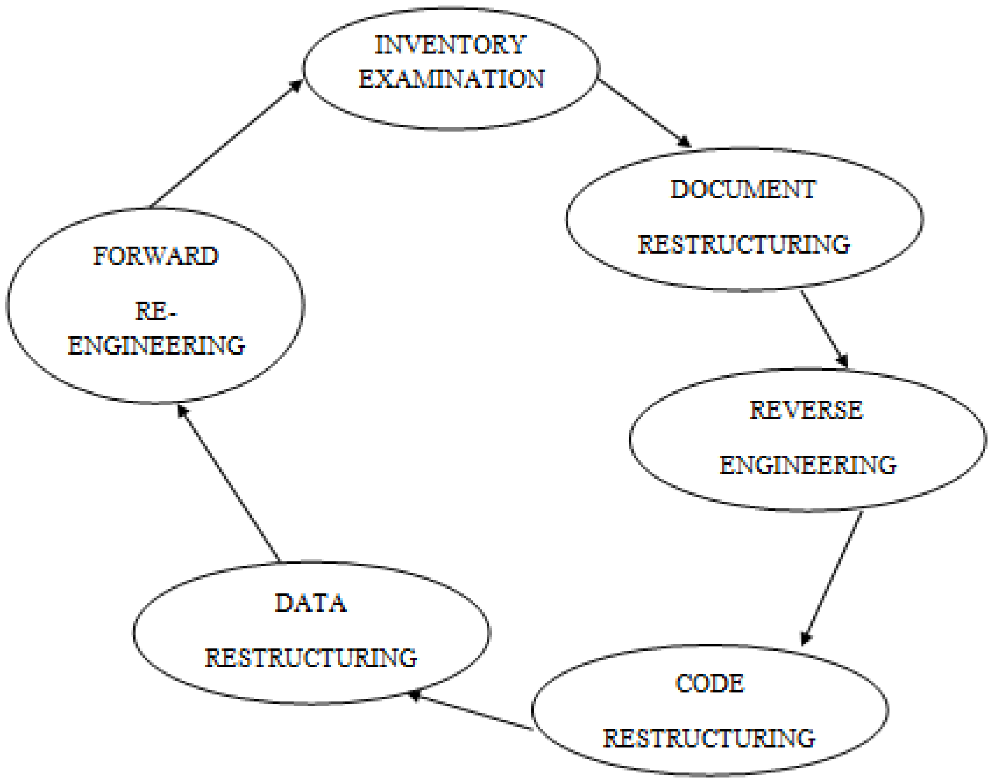

Fig. 2: Steps of Software Re- Engineering Process

Business Process of Re- Engineering-
Legacy software occurs wherever it becomes intolerable maintenance costs, and instead, re-engineering will be the only opportunity to reduce overhead and stop software development. The re-development will reduce the institution's overall portfolio on the software [6].

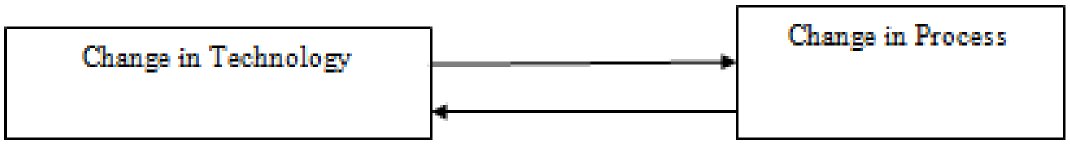

Fig. 3: Business Process of Re- Engineering

III.BENEFITS OF SOFTWARE RE- ENGINEERING

Improved Maintainability- Lots of money are expended on maintaining applications.Production of simple-to-maintain software can thus possibly save enormous costs.In business sector, the issue of keeping software is generally recognized, and much has been published about how e.g. instruments and procedures can promote maintainability.It is not possible to manage what can't be measured, and there is no widespread measure of maintainability yet [7]. Although some proposals have been submitted, the very concept of calculating sustainability has fundamental problems.Maintainability is the simplicity with which it is possible to modify a software package or feature to fix bugs, boost performance or other characteristics or adjust to a modified world.Often maintainability rises and quite often declines which was anticipated. For instance, the logical "system reform" shift would lead to an increase in the metric of maintainability.
Improved Performance- Performance (sensitivity and usability) is performance software make-or-break.Bad performance charges billions of dollars yearly in lost income for the industry, decreased efficiency, increased investment and equipment costs, and broken customer relationships.In reaction, the project often enters "crisis mode" in an effort to crank or even design the software to achieve performance targets [8]. Maximizing the performance and capability reward of the tuning attempts is essential in these circumstances.Using this method, a comparatively small expense has been used to provide a high return.When the problems are encountered the only option is probably to change the program.Nevertheless, it's important to remember that a calibrated system can occasionally if ever, show the performance you might have obtained by re-engineering the process.
Increased Interoperability- System interoperability includes the reliability of software production and processing to communicate with existing and new equipment to share information between structures in a reliable manner.In particular, system interoperability includes the idea of software processing and software engineering including existing systems and firmware upgrades/software that must communicate to ensure trustworthy news sharing among systems [9].Dynamic systems engineering involves flexibility from both procedures and designs to enable the procedures and designs themselves to be re-engineered and re-evaluated to enhance compatibility.
Reduced Individual Dependency- Developing huge-scale software requires cooperation between and within very big engineering team members that may be situated in various buildings, on different business universities, and in time zones.Cooperation between application development groups is one of the software development factors which are most hard to improve.Cooperation is a choice-making process which demands interaction, capacity and collaboration. These coordinating elements are essential, but inadequate in themselves, for cooperation to occur.Interaction is important as individual X needs to convey what needs to be done to individual Y in some manner and individual Y need to understand the interaction. Capacity is needed since Y needs to be able to do what he wants.Therefore, personal reliance is critical and a great deal of effort is made to organize the function of the software development community so that software re-engineering is used to minimize individual reliance by reforming the existing system in a way that reduces the dependence of individual X on individual Y where (both X & Y) are representatives of the team's work.

Enhanced Testability- The objective of testability reengineering is to reorganize the code in such a way that testing requirements can be encountered while reducing the size and difficulty of the system. Testability Re-engineering technology is certainly a worthwhile endeavour.The detection and elimination of clones the refactoring of tightly embedded code and the design, redesign are activities which can be standardized [10]. There are several resources that support this. The re-engineering expenses can be reduced by their use.The isolation of the ui from the application logic is a very key factor in testability.This isolation of the display from the processing is a requirement for checking the application, i.e. the operational logic, without having to open the information in the user interface, which takes a long time and is difficult to accomplish.

IV.RISKS/ CHALLENGES

There are number of risks which the re- engineering team has to experience they are as follows-
Technology Risks-

- Retrieved data is not beneficial or used.
- Reverse engineering to representations which cannot be communicated.
- Insufficient reengineering innovation to achieve reengineering objectives.

Tool Risks-

- Reliance on tools not performing as marketed.
- Never using instruments mounted.

Strategy Risks-

- Premature dedication to a complete system reengineering alternative.
- Incompetence to have a long-term view with immediate-term goals.
- Lack of global ambition: code, information, system reengineering.
- No schedule to use reengineering instruments.

Application Risks-

- Reengineering with no regional application specialist available.
- Existing market information is lost, rooted in software code.
- Reengineered device fails to work properly.

Process Risks-

- Exceptionally high manual reengineering expenses.
- Cost advantages not discovered in the prescribed time frame.
- Could not financially rationalize the reengineering achievement.
- Reengineering effort glides.
- Absence of management dedication to continuous reengineering method.

Personnel Risks-

- Programmers are inhibiting the reengineering process from starting.
- Less appropriate coders for making a controversial reengineering programme look less useful.

V.CONCLUSION

After the comprehensive analysis, it can be concluded that the re-engineering program is the mechanism by which the project expense, time, and commitment can be easily reduced. Given the advancement of the entire project the designers choose to use most of the prior software's currently existing parts. Because of the risk aspect, it is not mostly used in low-scale businesses, even though the use of developing entire venture rather than ploy the some parts of the venture which had previously built. Several complex software design techniques have been evolved to enhance software scalability and upkeep, and to shorten the time needed for upkeep and advancement activities, but many businesses have ancient or legacy software systems, such businesses are spending too much money to preserve the legacy systems.Such systems cannot be introduced to new systems since they involve indirect data and cannot be

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijareeie.com

Vol. 6, Issue 2, February 2017

destroyed in making decisions. Reengineering is becoming a valuable tool for such reasons in converting old, outdated structures into more profitable, simplified structures.

REFERENCES

[1]B. McMillin, “Software Engineering,” Computer. 2018.
[2]G. Hurlburt and J. Voas, “Software is driving software engineering?,” IEEE Softw., vol. 33, no. 1, pp. 101–104, 2016.
[3]B. A. Kitchenham, “Systematic review in software engineering,” 2012.
[4]J. Hayes and J. Hayes, “Business process re-engineering,” in The Theory and Practice of Change Management, 2018, pp. 385–393.
[5]G. Šag, Z. Lulić, and I. Mahalec, “Reverse engineering,” in Concurrent Engineering in the 21st Century: Foundations, Developments and Challenges, 2015.
[6]J. Sungau, “Business Process Re-Engineering,” 2018, pp. 15–33.
[7]P. T. Dube and D. G. Lubas, “NAVSEA reliability and maintainability engineering,” in Proceedings - Annual Reliability and Maintainability Symposium, 2016, vol. 2016-April.
[8]W. Suryn, Software Quality Engineering: A Practitioner’s Approach. 2014.
[9]P. Bourque and R. E. Fairley, Software Engineering - Body of Knowledge. 2014.
[10]ISO, IEC, and IEEE, “Systems and software engineering — Life cycle processes — Requirements engineering,” ISO/IEC/IEEE 26515 First Ed. 2011-12-01; Corrected version 2012-03-15, vol. 2012, pp. 1–36, 2012.
•K.Deepika, P.Andrew, R.Santhya, S.Balamurugan, S.Charanyaa, "Investigations on Methods Evolved for Protecting Sensitive Data", International Advanced Research Journal in Science, Engineering and Technology Vol 1, Issue 4, Decermber 2014.
•K.Deepika, P.Andrew, R.Santhya, S.Balamurugan, S.Charanyaa, "A Survey on Approaches Developed for Data Anonymization", International Advanced Research Journal in Science, Engineering and Technology Vol 1, Issue 4, December 2014.
•Vishal Jain and Dr. S. V. A. V. Prasad, “Mapping between RDBMS and Ontology: A Review”, International Journal of Scientific & Technology Research (IJSTR), France, Vol. 3, No. 11, November, 2014 having ISSN No. 2277-8616.
•Vishal Jain and Dr. S. V. A. V. Prasad, “Mining in Ontology With Multi Agent System in Semantic Web : A Novel Approach”, The International Journal of Multimedia & Its Applications (IJMA) Vol.6, No.5, October 2014, page no. 45 to 54 having ISSN No. 0975-5578.