Re-engineering – Software Engineering

Software Re-engineering is a process of software development that is done to improve the maintainability of a software system. Re-engineering is the examination and alteration of a system to reconstitute it in a new form. This process encompasses a combination of sub-processes like reverse engineering, forward engineering, reconstructing, etc.

Re-engineering

Re-engineering, also known as software re-engineering, is the process of analyzing, designing, and modifying existing software systems to improve their quality, performance, and maintainability.

1. This can include updating the software to work with new hardware or software platforms, adding new features, or improving the software's overall design and architecture.
2. Software re-engineering, also known as software restructuring or software renovation, refers to the process of improving or upgrading existing software systems to improve their quality, maintainability, or functionality.
3. It involves reusing the existing software artifacts, such as code, design, and documentation, and transforming them to meet new or updated requirements.

Objective of Re-engineering

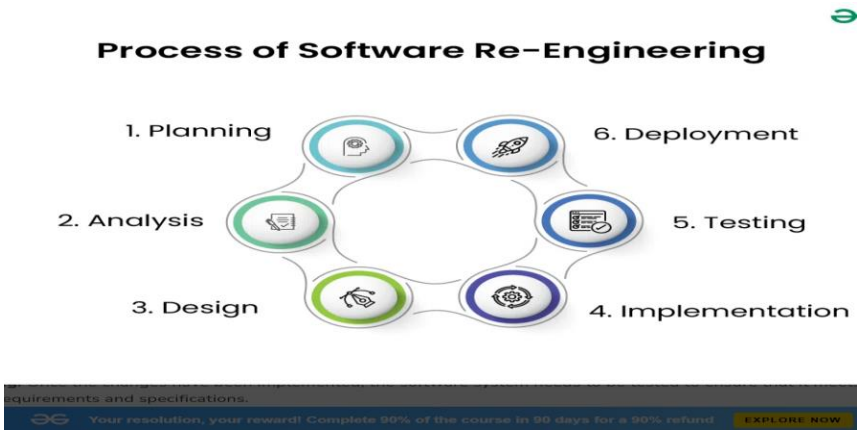
The primary goal of software re-engineering is to improve the quality and maintainability of the software system while minimizing the risks and costs associated with the redevelopment of the system from scratch. Software re-engineering can be initiated for various reasons, such as:

1. To describe a cost-effective option for system evolution.
2. To describe the activities involved in the software maintenance process.
3. To distinguish between software and data re-engineering and to explain the problems of data re-engineering.

Overall, software re-engineering can be a cost-effective way to improve the quality and functionality of existing software systems, while minimizing the risks and costs associated with starting from scratch.

Process of Software Re-engineering

The process of software re-engineering involves the following steps:



1. **Planning:** The first step is to plan the re-engineering process, which involves identifying the reasons for re-engineering, defining the scope, and establishing the goals and objectives of the process.
2. **Analysis:** The next step is to analyze the existing system, including the code, documentation, and other artifacts. This involves identifying the system's strengths and weaknesses, as well as any issues that need to be addressed.
3. **Design:** Based on the analysis, the next step is to design the new or updated software system. This involves identifying the changes that need to be made and developing a plan to implement them.
4. **Implementation:** The next step is to implement the changes by modifying the existing code, adding new features, and updating the documentation and other artifacts.
5. **Testing:** Once the changes have been implemented, the software system needs to be tested to ensure that it meets the new requirements and specifications.
6. **Deployment:** The final step is to deploy the re-engineered software system and make it available to end-users.

Why Perform Re-engineering?

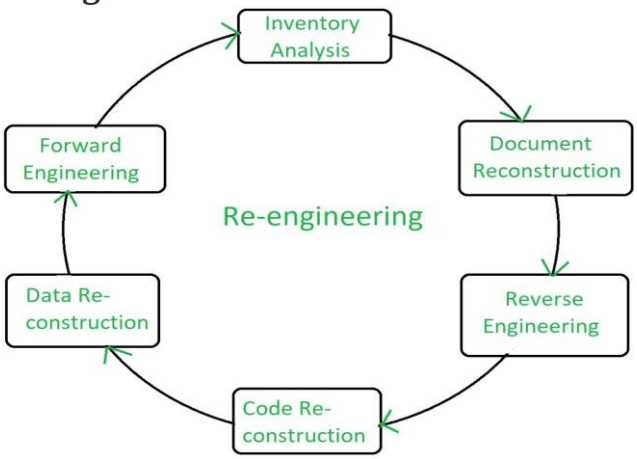
Re-engineering can be done for a variety of reasons, such as:

1. **To improve the software's performance and scalability:** By analyzing the existing code and identifying bottlenecks, re-engineering can be used to improve the software's performance and scalability.
2. **To add new features:** Re-engineering can be used to add new features or functionality to existing software.
3. **To support new platforms**: Re-engineering can be used to update existing software to work with new hardware or software platforms.
4. **To improve maintainability:** Re-engineering can be used to improve the software's overall design and architecture, making it easier to maintain and update over time.
5. **To meet new regulations and compliance**: Re-engineering can be done to ensure that the software is compliant with new regulations and standards.
6. **Improving software quality:** Re-engineering can help improve the quality of software by eliminating defects, improving performance, and enhancing reliability and maintainability.

7. **Updating technology:** Re-engineering can help modernize the software system by updating the technology used to develop, test, and deploy the system.
8. **Enhancing functionality:** Re-engineering can help enhance the functionality of the software system by adding new features or improving existing ones.
9. **Resolving issues:** Re-engineering can help resolve issues related to scalability, security, or compatibility with other systems.

Steps involved in Re-engineering

1. **Inventory Analysis**
2. **Document Reconstruction**
3. **Reverse Engineering**
4. **Code Reconstruction**
5. **Data Reconstruction**
6. **Forward Engineering**



Steps of Re-Engineering

Re-engineering Cost Factors

1. The quality of the software to be re-engineered.
2. The tool support available for re-engineering.
3. The extent of the required data conversion.
4. The availability of expert staff for re-engineering.

Factors Affecting Cost of Re-engineering

Re-engineering can be a costly process, and there are several factors that can affect the cost of re-engineering a software system:

1. **Size and complexity of the software:** The larger and more complex the software system, the more time and resources will be required to analyze, design, and modify it.
2. **Number of features to be added or modified:** The more features that need to be added or modified, the more time and resources will be required.
3. **Tools and technologies used:** The cost of re-engineering can be affected by the tools and technologies used, such as the cost of software development tools and the cost of hardware and infrastructure.

4. **Availability of documentation:** If the documentation of the existing system is not available or is not accurate, then it will take more time and resources to understand the system.
5. **Team size and skill level:** The size and skill level of the development team can also affect the cost of re-engineering. A larger and more experienced team may be able to complete the project faster and with fewer resources.
6. **Location and rate of the team:** The location and rate of the development team can also affect the cost of re-engineering. Hiring a team in a lower-cost location or with lower rates can help to reduce the cost of re-engineering.
7. **Testing and quality assurance:** Testing and quality assurance are important aspects of re-engineering, and they can add significant costs to the project.
8. **Post-deployment maintenance:** The cost of post-deployment maintenance such as bug fixing, security updates, and feature additions can also play a role in the cost of re-engineering.

In summary, the cost of re-engineering a software system can vary depending on a variety of factors, including the size and complexity of the software, the number of features to be added or modified, the tools and technologies used, and the availability of documentation and the skill level of the development team. It's important to carefully consider these factors when estimating the cost of re-engineering a software system.

Advantages of Re-engineering

1. **Reduced Risk:** As the software is already existing, the risk is less as compared to new software development. Development problems, staffing problems and specification problems are the lots of problems that may arise in new software development.
2. **Reduced Cost:** The cost of re-engineering is less than the costs of developing new software.
3. **Revelation of Business Rules:** As a system is re-engineered , business rules that are embedded in the system are rediscovered.
4. **Better use of Existing Staff:** Existing staff expertise can be maintained and extended accommodate new skills during re-engineering.
5. **Improved efficiency:** By analyzing and redesigning processes, re-engineering can lead to significant improvements in productivity, speed, and cost-effectiveness.
6. **Increased flexibility:** Re-engineering can make systems more adaptable to changing business needs and market conditions.
7. **Better customer service:** By redesigning processes to focus on customer needs, re-engineering can lead to improved customer satisfaction and loyalty.
8. **Increased competitiveness:** Re-engineering can help organizations become more competitive by improving efficiency, flexibility, and customer service.
9. **Improved quality:** Re-engineering can lead to better quality products and services by identifying and eliminating defects and inefficiencies in processes.
10. **Increased innovation:** Re-engineering can lead to new and innovative ways of doing things, helping organizations to stay ahead of their competitors.

11. **Improved compliance:** Re-engineering can help organizations to comply with industry standards and regulations by identifying and addressing areas of non-compliance.

Disadvantages of Re-engineering

Major architectural changes or radical reorganizing of the systems data management has to be done manually. Re-engineered system is not likely to be as maintainable as a new system developed using modern software Re-engineering methods.

1. **High costs:** Re-engineering can be a costly process, requiring significant investments in time, resources, and technology.
2. **Disruption to business operations:** Re-engineering can disrupt normal business operations and cause inconvenience to customers, employees and other stakeholders.
3. **Resistance to change:** Re-engineering can encounter resistance from employees who may be resistant to change and uncomfortable with new processes and technologies.
4. **Risk of failure:** Re-engineering projects can fail if they are not planned and executed properly, resulting in wasted resources and lost opportunities.
5. **Lack of employee involvement:** Re-engineering projects that are not properly communicated and involve employees, may lead to lack of employee engagement and ownership resulting in failure of the project.
6. **Difficulty in measuring success:** Re-engineering can be difficult to measure in terms of success, making it difficult to justify the cost and effort involved.
7. **Difficulty in maintaining continuity:** Re-engineering can lead to significant changes in processes and systems, making it difficult to maintain continuity and consistency in the organization.

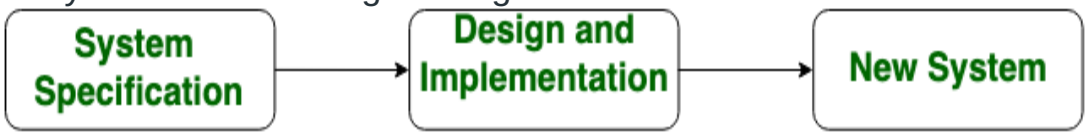
What is Forward Engineering?

Forward Engineering is a method of creating or making an application with the help of the given requirements. Forward engineering is also known as Renovation and Reclamation. Forward engineering requires high proficiency skills. It takes more time to construct or develop an application. Forward engineering is a technique of creating high-level models or designs to make complexities and low-level information. Therefore this kind of engineering has completely different principles in numerous packages and information processes. Forward Engineering applies all the software engineering process that contains SDLC to recreate associated existing applications. It is near to full fill new needs of the users into re-engineering.

Characteristics of forward engineering

1. Forward engineering is a variety of engineering that has different principles in numerous package and information processes.

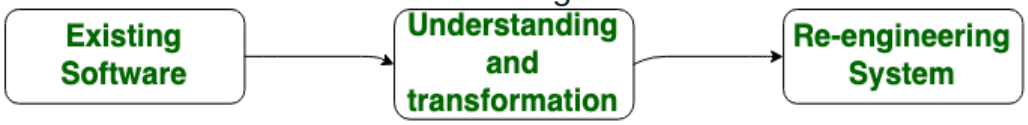
2. Forward engineering is vital in IT as a result of it represents the ‘normal’ development process.
3. Forward engineering deals with the conversion of business processes, services, and functions into applications.
4. In this method, the business model is developed first. Then, a top-down approach is followed to urge the package from the model developed.
5. Forward engineering tools are accustomed to moving from implementation styles and logic to the event of supply code.
6. It essentially permits the user to develop a business model which may then be translated into data system components.
7. These tools follow the top-down approach. System creator and visual Analyst is a forward engineering CASE tool.



Forward Engineering

What is Reverse Engineering?

Reverse Engineering is also known as backward engineering, is the process of forward engineering in reverse. In this, the information is collected from the given or existing application. It takes less time than forward engineering to develop an application. In reverse engineering, the application is broken to extract knowledge or its architecture.



Reverse Engineering

Key Elements of Forward engineering and reverse engineering

Here are some key differences between the two:

1. **Goal:** The goal of forward engineering is to develop new software from scratch, while the goal of reverse engineering is to analyze and understand an existing software system.
2. **Process:** Forward engineering involves designing and implementing a new software system based on requirements and specifications. Reverse engineering involves analyzing an existing software system to understand its design, structure, and behavior.
3. **Tools and Techniques:** Forward engineering often involves the use of software development tools, such as IDEs, code generators, and

testing frameworks. Reverse engineering often involves the use of reverse engineering tools, such as decompilers, disassemblers, and code analyzers.

4. **Focus:** Forward engineering focuses on the creation of new code and functionality, while reverse engineering focuses on understanding and documenting existing code and functionality.
5. **Output:** The output of forward engineering is a new software system, while the output of reverse engineering is documentation of an existing software system, such as a UML diagram, flowchart, or software specification.

In summary, forward engineering is focused on the creation of new software systems, while reverse engineering is focused on understanding and documenting existing software systems. Both approaches use different tools and techniques to achieve their goals and produce different outputs.

Difference between Forward Engineering and Reverse Engineering

Aspect	Forward Engineering	Reverse Engineering
Process	In forward engineering, the application are developed with the given requirements.	In reverse engineering or backward engineering, the information are collected from the given application.
Skill Level	Forward Engineering is a high proficiency skill.	Reverse Engineering or backward engineering is a low proficiency skill.
Development Time	Forward Engineering takes more time to develop an application.	While Reverse Engineering or backward engineering takes less time to develop an application.
Nature	The nature of forward engineering is Prescriptive.	The nature of reverse engineering or backward engineering is Adaptive.
Production Start Point	In forward engineering, production is started with given requirements.	In reverse engineering, production is started by taking the existing products.
Examples	The example of forward	An example of backward

Aspect	Forward Engineering	Reverse Engineering
	engineering is the construction of electronic kit, construction of DC MOTOR , etc.	engineering is research on Instruments etc.
Development Steps	Forward engineering Starts with requirements analysis and design, then proceeds to implementation and testing.	Reverse engineering Starts with an existing software system and works backward to understand its structure, design, and requirements.
Use Case	Forward engineering is used to create new software applications from scratch.	Reverse engineering is Used to modify and improve an existing software application.
Abstraction Level	Forward engineering is process of moving from a high-level abstraction to a detailed implementation.	Reverse engineering is a process of moving from a low-level implementation to a higher-level abstraction.
Requirements and Design Specifications	Requires a clear set of requirements and design specifications.	Requirements and design specifications may not be available, making it necessary to reconstruct them from the code itself.
Time and Cost	Forward engineering is generally more time-consuming and expensive.	Reverse engineering is generally less time-consuming and less expensive.
Final Product	The final product is completely new and independent of any existing software system.	The final product is typically a modified or improved version of an existing software system.
Development Steps	Involves a series of steps such as requirements gathering, design,	Involves steps such as code analysis, code understanding, design

Aspect	Forward Engineering	Reverse Engineering
	implementation, testing, and deployment.	recovery, and documentation.
Software Development Life Cycle Stage	Forward engineering is commonly used in the initial stages of software development.	Reverse engineering is commonly used in the maintenance stage of the software development life cycle.