7 Software Re-engineering

Syllabus

Software Maintenance Problems, Redevelopment vs. Reengineering, Business Process Reengineering, Software Reengineering Process Model, Technical Problems of Reengineering.

Contents

7.1 Software Maintenance and Software Maintenance Problem

7.1.1 Software Maintenance

- It is a vast management process that takes place once the procedure is done. Software maintenance includes optimization of software performance by applying advanced development, reducing errors, and eliminating useless development. Solution development can take about 2 years to build a system, while software maintenance management is an ongoing activity.

7.1.2 Types of Software Maintenance Services Categories

1. Adaptive Maintenance

- It is the process of conversion in the system to keep the software compatible with changing business needs and technical evolution. This type of software maintenance primarily focuses on software frameworks. Adaptive maintenance is made in response to new operating systems, platforms, and hardware to retain continuity with the software.

2. Perfective Maintenance

- It is a process of modifying all elements, functionalities, and abilities to enhance system operations and performance. The software's receptiveness and usability are solved by perfective software maintenance. It includes altering current software functionality by improving, removing, or inserting new features or functions.

3. Corrective Maintenance

- Identifying errors in the existing solution and correcting them to make it works more accurately. This software maintenance activities aim to eliminate and fix bugs or issues in the software. Corrective software maintenance is usually done in the form of small updates frequently.

4. Preventive Maintenance

- This service helps in preventing the system from any forthcoming vulnerabilities. Preventive maintenance defines improvements of the software, which is done to safeguard the software for the future. It is carried out to prevent the product from any potential software alteration. Preventive maintenance also makes it easier to scale or maintain your code and handle your legacy system.

7.1.3 Need of Software Maintenance

Bug Fixing

- In maintenance management, bug fixing comes at priority to run the software seamlessly. This process contains search out for errors in code and corrects them. The issues can occur in hardware, operating systems, or any part of the software. This must be done without hurting the rest of the functionalities of existing software.

Capability Enhancement

- This comprises an improvement in features and functions to make solutions compatible with the varying market environment. It enhances software platforms, work patterns, hardware upgrades, compilers, and other aspects that affect system workflow. Boost your business using a technically updated solution applying software maintenance services regularly.

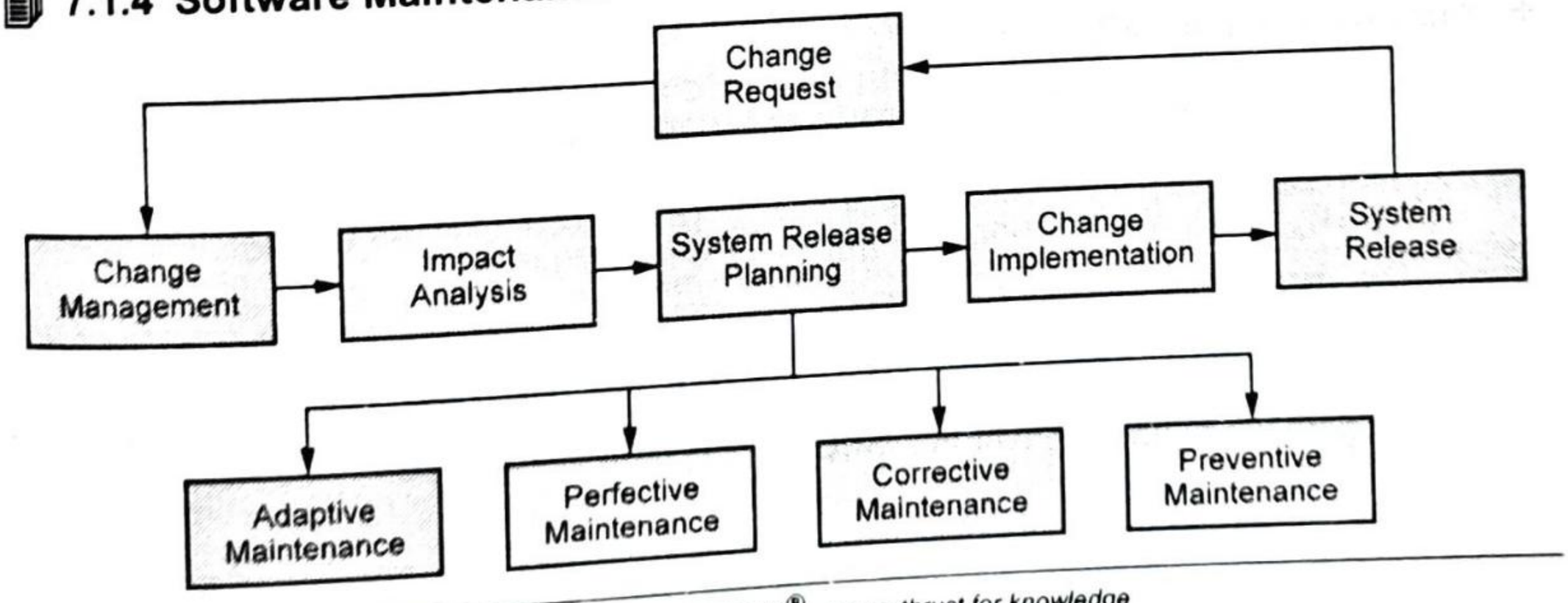
Removal of Outdated Functions

- The unwanted functionalities are useless. Moreover, by occupying space in the solution, they hurt the efficiency of the solution. Using software maintenance guide, such UI and coding elements are removed and replaced with new development using the latest tools and technologies. This elimination makes the system adaptive to cope with changing circumstances.

Performance Improvement

- To improve system performance, developers detect issues through testing and resolve them. Data and coding restricting as well as reengineering are part of software maintenance. It prevents the solution from vulnerabilities. This is not any functionality that performs in operations, but it develops to stop harmful activities like hacking.

7.1.4 Software Maintenance Process



- In the life cycle of software development, software maintenance plan is a very crucial phase. Hence, it is executed in the system through a well-planned software maintenance process which is known as Software Maintenance Life Cycle (SMLC). SMLC is implemented in seven different phases. Those are :

✦ **Phase 1 - Identification**

- As the name goes, in this phase, the modifications are 'identified'. Before implementing the changes for the requests raised, the modifications are first analyzed and classified according to the attention or maintenance it requires. This phase can be automated or manually done by a user.

✦ **Phase 2 - Analysis**

- The practicality and feasibility of each verified modification request are planned to incorporate changes in the software. The analysis includes validated changes or input where the cost of modification is also estimated.

✦ **Phase 3 - Design**

- The new framework of the software is determined according to the result of the analysis. Survey or test software is also developed for the purpose of safety and security.

✦ **Phase 4 - Implementation**

- This is where the main or new software framework is implemented; as in, the codes are crafted, and in the new support system, specifications are added.

✦ **Phase 5 - System testing**

- In this testing, the implementation of codes and specifications are tested. This stage determines if any further changes or additions are required in the new model of software.

✦ **Phase 6 - Acceptance Testing**

- This stage is performed by third-party end-users. They run a dummy software test, also known as a dry run test, to check if the implemented specifications are working properly, which was mentioned in the modification request.

✦ **Phase 7 - Delivery**

- As and when the testing phase is cleared and the developers get a green signal from the third party users, they deliver the software to the primary users.

7.1.5 Software Maintenance Problem

Causes for Software maintenance problem

Lack of Traceability

- Codes are rarely traceable to the requirements and design specifications.
- It makes it very difficult for a programmer to detect and correct a critical defect affecting customer operations.
- Like a detective, the programmer pores over the program looking for clues.
- Life Cycle documents are not always produced even as part of a development project.

Lack of Code Comments

- Most of the software system codes lack adequate comments. Lesser comments may not be helpful in certain situations.

Obsolete Legacy Systems

- In most of the countries worldwide, the legacy system that provides the backbone of the nation's critical industries, e.g., telecommunications, medical, transportation utility services, were not designed with maintenance in mind.
- They were not expected to last for a quarter of a century or more!
- As a consequence, the code supporting these systems is devoid of traceability to the requirements, compliance to design and programming standards and often includes dead, extra and uncommented code,which all make the maintenance task next to the impossible.

7.2 Software Re-engineering

7.2.1 Meaning and Objectives of Re-engineering

- Software Re-engineering is a process of software development which 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 is the reorganizing and modifying existing software systems to make them more maintainable.*

❑ Objectives of Re-engineering

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

7.2.2 The Need of Software Re-engineering

- Software re-engineering is an economical process for software development and quality enhancement of the product. This process enables us to identify the useless consumption of deployed resources and the constraints that are restricting the development process so that the development process could be made easier and cost-effective (time, financial, direct advantage, optimize the code, indirect benefits, etc.) and maintainable. The software reengineering is necessary for having -

a) **Boost up productivity :** Software reengineering increase productivity by optimizing the code and database so that processing gets faster.

b) **Processes in continuity :** The functionality of older software product can be still used while the testing or development of software.

c) **Improvement opportunity :** Meanwhile the process of software reengineering, not only software qualities, features and functionality but also your skills are refined, new ideas hit in your mind. This makes the developers mind accustomed to capturing new opportunities so that more and more new features can be developed.

d) **Reduction in risks :** Instead of developing the software product from scratch or from the beginning stage here developers develop the product from its existing stage to enhance some specific features that are brought in concern by stakeholders or its users. Such kind of practice reduces the chances of fault fallibility.

e) **Saves time :** As we stated above here that the product is developed from the existing stage rather than the beginning stage so the time consumes in software engineering is lesser.

f) **Optimization :** This process refines the system features, functionalities and reduces the complexity of the product by consistent optimization as maximum as possible.

7.2.3 Re-engineering Cost Factors

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

7.2.4 Advantages of Re-engineering

- **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 which may arise in new software development.
- **Reduced Cost :** The cost of re-engineering is less than the costs of developing new software.
- **Revelation of Business Rules :** As a system is re-engineered , business rules that are embedded in the system are rediscovered.
- **Better use of Existing Staff :** Existing staff expertise can be maintained and extended accommodate new skills during re-engineering.

7.2.5 Disadvantages of Re-engineering

- Practical limits to the extent 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.

7.3 Business Process Re-engineering

7.3.1 Definition and Meaning

- The **Business Process Reengineering** or **BPR** is the analysis and redesign of core business processes to achieve the substantial improvements in its performance, productivity, and quality. The business process refers to the set of interlinked tasks or activities performed to achieve a specified outcome.
- Simply, the business process reengineering means to change the way an individual performs the work such that better results are accomplished. The purpose of business process reengineering is to redesign the workflows in order to dramatically improve the customer service, achieve higher levels of efficiency, cut operational costs and become a world-class competitor.

❑ Companies use Business Process Re-engineering to :

- **Reduce costs and cycle times.** Business Process Reengineering reduces costs and cycle times by eliminating unproductive activities and the employees who perform them. Reorganization by teams decreases the need for management layers, accelerates information flows and eliminates the errors and rework caused by multiple handoffs.
- **Improve quality.** Business Process Reengineering improves quality by reducing the fragmentation of work and establishing clear ownership of processes. Workers gain responsibility for their output and can measure their performance based on prompt feedback.

7.3.2 Need of Business Process Re-engineering

The business process is required to be reengineered because of the following reasons :

- The processes the company is using might have become outdated or holds no relevance in the current market scenario.
- Often, the sub-divisions in the organization aims at improving their respective division performance and overlook the resultant effects on the other departments. This might lead to the underperformance of the firm as a whole.
- Due to the departmentalization, each employee focuses on the performance of his respective department and may overlook the critical issues emerging in other areas of the firm, and therefore, the need for re-engineering arises so that the role of the employees could be broadened and shall be made more responsible towards the firm.
- The existing business process could be lengthy, time-consuming, costly, obsolete, therefore, is required to be redesigned to match it with the current business requirements.
- The technology keeps on updating and in order to catch up with it, reengineering is a must.

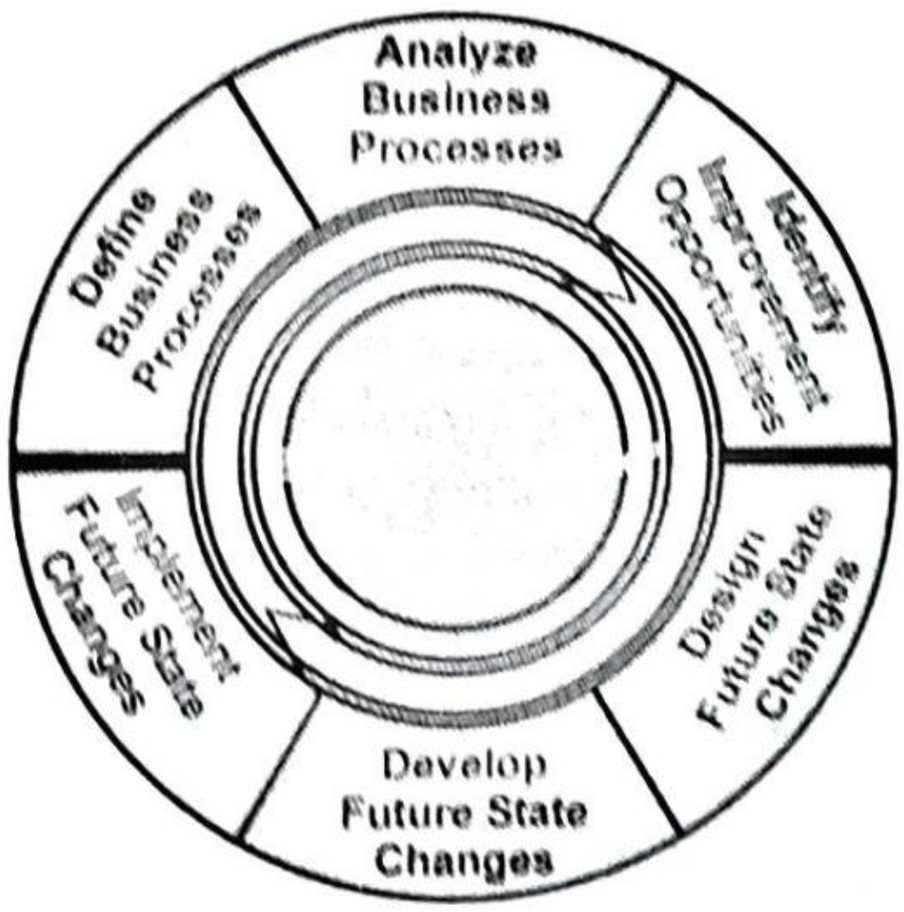
7.3.3 Steps Involved in Business Process Re-engineering

1. **Define Business Processes**. Map the current state (work activities, workflows, roles and reporting relationships, supporting technology, business rules, etc.).
2. **Analyze Business Processes**. Identify gaps, root causes, strategic disconnects, etc. in the context of improving organizational effectiveness, operational efficiency and in achieving organizational strategic objectives.
3. **Identify and Analyze Improvement Opportunities**. Identify, analyze and validate opportunities to address the gaps and root causes identified during analysis. This step also includes identifying and validating improvement opportunities that are forward facing.

4. **Design Future State Processes**. Select the improvement opportunities identified above that have the most impact on organizational effectiveness, operational efficiency, and that will achieve organizational strategic objectives. Make sure to select opportunities for which the organization has the budget, time, talent, etc. to implement in the project timeframe. Create a forward-facing future-state map that comprehends the selected opportunities.

5. **Develop Future State Changes**. Frequently overlooked (and a key root cause in failed BPR initiatives), this is where the above opportunities are operationalized before implementation. New workflows and procedures need to be designed and communicated, new/enhanced functionality is developed and tested, etc. Changes and opportunities cannot be implemented until they are operationalized.

6. **Implement Future State Changes**. Classic implementation based on dependencies among changes/opportunities, change management, project management, performance monitoring, etc.



7.3.4 Benefits of Business Process Re-engineering

BPR plays a major role in organizational performance improvement in terms of cost, quality, delivery, employee productivity, etc. It also helps

- Streamline business processes and systems
- Companies easily adapt to changing times and reduce operating expenses
- Improve company profitability and sustain competitive advantage
- Boost employee productivity
- Increase customer satisfaction by improving the quality of products and services

7.3.5 Principles of Business Process Re-engineering

Following are the 7 principles of reengineering :

1. Identify all the organization's processes and prioritize them in order of redesign urgency
2. Integrate information processing work into the real work that produces the information
3. Treat geographically dispersed resources as though they were centralized
4. Link Parallel activities in the workflow instead of just integrating their results
5. Put the decision point where the work is performed, and build control into the process
6. Capture information once and at the source.

7.3.6 BPR Example

❑ A real-life example of BPR

- Many companies like Ford Motors, GTE, and Bell Atlantic tried out BPR during the 1990s to reshuffle their operations. The reengineering process they adopted made a substantial difference to them, dramatically cutting down their expenses and making them more effective against increasing competition.

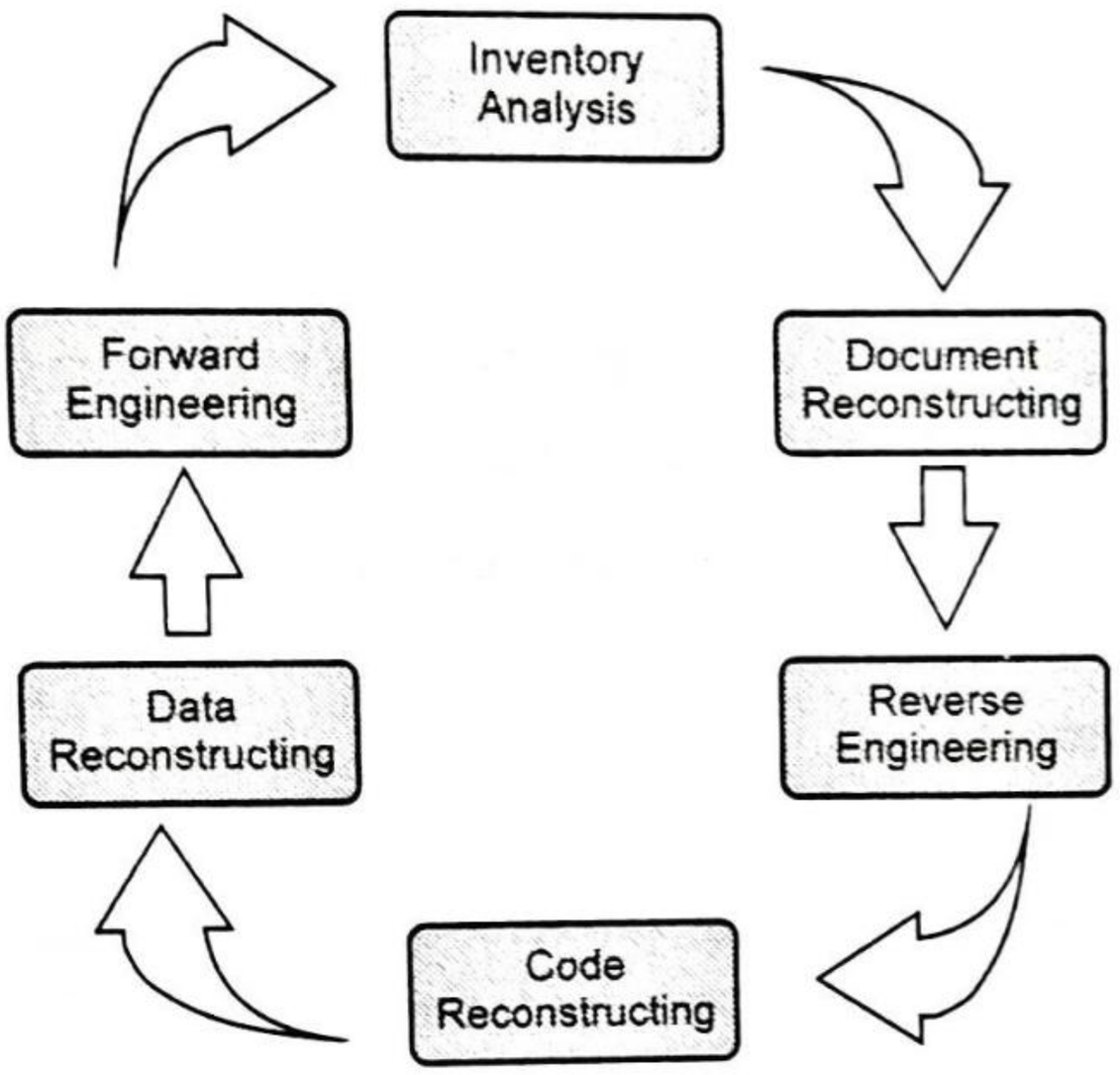
❑ The story

- An American telecom company that had several departments to address customer support regarding technical snags, billing, new connection requests, service termination, etc. Every time a customer had an issue, they were required to call the respective department to get their complaints resolved. The company was doling out millions of dollars to ensure customer satisfaction, but smaller companies with minimal resources were threatening their business.
- The telecom giant reviewed the situation and concluded that it needed drastic measures to simplify things - a one - stop solution for all customer queries. It decided to merge the various departments into one, let go of employees to minimize multiple handoffs and form a nerve center of customer support to handle all issues.
- A few months later, they set up a customer care center in Atlanta and started training their repair clerks as 'frontend technical experts' to do the new, comprehensive job. The company equipped the team with new software that allowed the support team to instantly access the customer database and handle almost all kinds of requests.
- Now, if a customer called for billing query, they could also have that erratic dial tone fixed or have a new service request confirmed without having to call another number. While they were still on the phone, they could also make use of the push-button phone menu to connect directly with another department to make a query or input feedback about the call quality.

- The redefined customer-contact process enabled the company to achieve new goals.
  - Reorganized the teams and saved cost and cycle time
  - Accelerated the information flow, minimized errors, and prevented reworks
  - Improved the quality of service calls and enhanced customer satisfaction
  - Defined clear ownership of processes within the now-restructured team
  - Allowed the team to evaluate their performance based on instant feedback

7.4 Software Re-engineering Process Model

The Software Re-engineering process basically undergoes six main phases.



1. Inventory Analysis :

- Every software organisation should have an inventory of all the applications.
- Inventory can be nothing more than a spreadsheet model containing information that provides a detailed description of every active application.
- By sorting this information according to business criticality, longevity, current maintainability and other local important criteria, candidates for re-engineering appear.
- The resource can then be allocated to a candidate application for re-engineering work.

2. Document Reconstructing :

- Documentation of a system either explains how it operates or how to use it.
- Documentation must be updated.

- It may not be necessary to fully document an application.
- The system is business-critical and must be fully re-documented.

❑ 3. Reverse Engineering :

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

❑ 4. Code Reconstructing :

- To accomplish code reconstructing, the source code is analysed using a reconstructing tool. Violations of structured programming construct are noted and code is then reconstructed.
- The resultant restructured code is reviewed and tested to ensure that no anomalies have been introduced.

❑ 5. Data Restructuring :

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

❑ 6. Forward Engineering :

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

🕮 7.5 Technical Problem of Re-engineering

- Practical limits to the extent 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.

❑❑❑

8 Project Closure

Syllabus

Project Closure Analysis. Case Study of Software Company's Project Closure Analysis Report.

Contents

8.1 Project Closure Analysis

8.1.1 Meaning

- Project closure analysis is the key to learning from the past so as to provide future improvements. To achieve this goal, it must be done carefully in an atmosphere of safety so that lessons can be captured and used to improve the process and future projects. Before we describe the details of the closure analysis report, we briefly discuss the role of closure analysis and its implementation.

8.1.2 The Role of Closure Analysis

- The objective of a postmortem or closure analysis is "to determine what went right, what went wrong, what worked, what did not, and how it could be made better the next time." Relevant information must be collected from the project, primarily for use by future projects. That is, the purpose of having an identified completion analysis activity, rather than simply saying, "The project is done," is not to help this project but rather to improve the organization by leveraging the lessons learned. This type of learning can be supported effectively by analysis of data from completed projects. This analysis is also needed to understand the performance of the process on this project, which in turn is needed to determine the process capability.

8.1.3 Importance of Closing a Project

- At first glance, it might seem like completing the first four phases of the project lifecycle would be all you need to do to tie up your project and call it good.
- However, without a formal closing process, you risk letting crucial details fall through the cracks, which can result in confusion, a never-ending project, dissatisfied clients, and even liability issues.

❑ Project closure helps avoid :

- Repeating mistakes on future projects and objectives
- Having final products or deliverables without dedicated support and resources
- Failing to identify the team or individuals who will own and maintain the solution following final delivery
- Creating liability issues resulting from incomplete payments, contracts, or deliverables
- Following a clear project closure plan helps you properly transition your solution to the client or end-user. This process ensures the final stakeholders have the information, resources, and training to successfully manage and use the end product.

- The project closure process also ensures the project is formally completed and is no longer considered a project, allowing you to hand the reins over to the correct team in charge of managing and maintaining the project's outputs.
- By officially closing a project, you minimize risks, increase client satisfaction, and ensure all parties are on the same page. In other words, project closure is a process you can't afford to skip.

8.1.4 Project Closure Report

- The *Project Closure Report* is the final document produced for the project and is used by senior management to "tidy up" any loose ends and formally close the project.
- This template should be used for more complex projects and developed after the project has been reviewed.
- In less complex projects the review and closure processes can be combined into a *Project Review and Closure Report.*

8.1.5 Need to Develop a Project Closure Analysis and Project Closure Report

❑ Need to develop a Project Closure Analysis

A Project Closure analysis is developed to :

- Detail activities undertaken to close the project;
- Outline outstanding issues, risks, operational matters and recommendations.
- This document lists the closure activities and any outstanding matters and recommends how they should be addressed.

❑ Need to develop a Project Closure Report

- The *Project Closure Report* is usually developed once the project is completed and all the project outputs have been delivered to the Business Owner(s)
- Or it has been decided to close the project for some other reason.
- This may be the result of a recommendation from a review of the project where the findings are negative
- Or may be the result of changed priorities within the agency, division or business unit.

8.1.6 Objective of Closure Analysis Report

- The objective of Closure Analysis Report is :
  - Determine what went wrong
  - What went right
  - What worked
  - What did not work
  - How to make it better next time
- Relevant information needs to be collected from the project, for future projects.
- Also this analysis throws light on performance of the process on project and its capability.
- These analyzed results should be packaged such that they can be used by others effectively.
- Even if not used by others, closure analysis results will consolidate the experience gained by the project personnel.
- The lessons learnt can be carried forward by the project "personnel" for future projects.

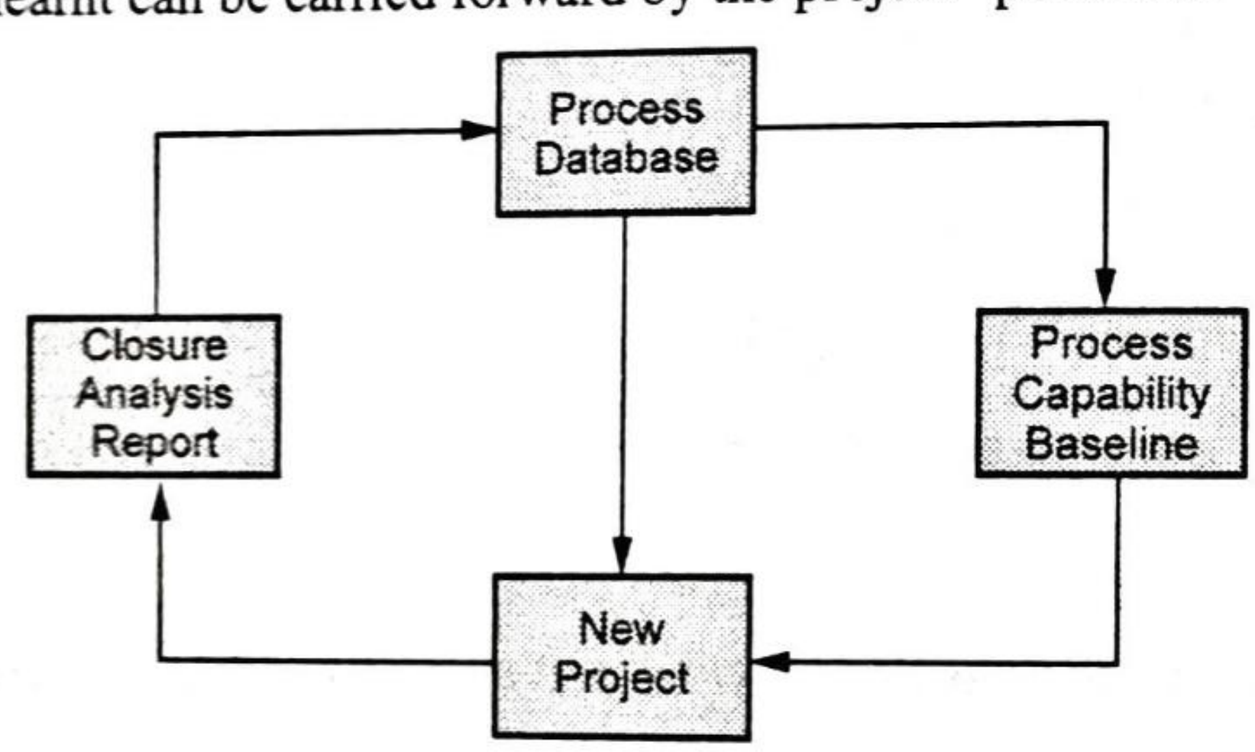


Fig. 8.1.1

8.1.7 Elements in Closure Analysis Report

- The major elements in a project closure analysis are :
  - General and Process - Related Information
  - Risk Management
  - Size
  - Effort
  - Defects

- Causal Analysis
- Process Assets

❑ General and Process - Related Information

- The closure report first gives
  - Some general information about the project.
  - Overall Productivity Achieved
  - Quality Delivered
  - Process used and Process Deviation
  - Estimated and Actual Start & End Date of Project
  - Tools used and their experience etc.

❑ Risk Management

- In this part the RISK anticipated for the project are given along with risk mitigation steps planned
- The top risk are reviewed in the post project analysis
- Notes are prepared on the effectiveness of the risk mitigation steps employed.

❑ Size

- The project in size is captured in terms of simple/medium/ or complex modules, along with criteria used for classification
- Data on estimated and actual size is compared
- Productivity is measured in terms of function point per person month or LOC
- The above parameters are captured in closure analysis.

❑ Effort

- Total estimated effort and actual effort in person - hours are captured in closure analysis.
- The total estimated effort is obtained from project management plan.
- If there is a large deviation, reason for the variation are analyzed and recorded.
- These information are useful in forming PCB.
- The effort are separated for
  - Task
  - Review
  - Rework.

- The cost of quality for project is also computed and captured.
- The cost of quality can be defined as "the total effort spent in review, testing and rework to remove defects and project specific training.

❑ Defects

- A summary of defects found during the project is given in closure analysis report.
- The following information are captured,
  - Stage injected
  - Stage detected
  - Severity
  - Program and module where defect found.
- Information on defect distribution coupled with defection injection rate will be useful for defect prediction.

❑ Casual Analysis

- Casual analysis involves looking at large variations between estimated and actual values.
- The reason for variation is also analyzed
- They are also determined through discussion & brainstorming
- Once causes are identified, this is used as lessons learned.
- Casual analysis is done through Fishbone diagram and Pareto charts.

❑ Process Asset

- The metrics and other artifacts produced during the project can be used for future projects
- These artifacts are called **process asset**.
- These artifacts are available to other project by capturing them into PDB.

❑ Archiving

- Every time a software version is released, the source code along with relevant document is archived.
- This provides a long - term backup
- The list of artifacts that are archived are :
- Project proposal, contract and review document
- Requirement specification.

- Project plan, configuration management plan, installation and AT plan and maintenance plan.
- High - level design and review document
- Program specification, pseudocode and review
- Source code
- Testing related files, test case, test result
- Manuals and technical support materials.

8.2 Case Study of Software Company's Project Closure Analysis Report

8.2.1 Examples of Project Closing Oversight

Case-based example	Impact
The IT team has completed the development of an application. The application was fully tested and accepted by the business and users. A few months later, users look for basic "how-to guides" but never find them, because they were seen as a secondary product and of lesser importance than the application itself	Users are dissatisfied with the outcome of the project and view it as a failure. Users are not capable of fully using the application, as they are dissatisfied with the lack of documentation to help them achieve what they need through the application. Responsibility to correct the situation is diluted. Developers engaged in supporting users, as opposed to being in a position to work on new projects.
At the end of the application development project, the project manager is required to close the contract with the vendor who provided him with two HTML developers but hasn't on the premise that this is a minor administrative matter and everybody knows that the project is over.	Three months after the project, the finance department receives invoices for work that was completed during the project life cycle, with claims of extra time and effort. Because the work is a distant memory and the exact proceedings and requests were not documented at the time, the organization and the contractor enter into a dispute. Such a dispute not only harms the relationship between the two parties, but could also make one of them liable for reparations, damages and legal costs, hence costing the organization far more than necessary.

Case-based example	Impact
There is no formal end to the project application development project and hence developer's time is still allocated to that project and they are not free to work on other projects or tasks and stakeholders continue to view this as a long-term project.	Project manager, project teams and other resources are continuously engaged in post-project activities, though unnecessarily. Support staff is incapable of supporting the application due to the lack of a formal hand off. The organization is constrained in initiating new projects due to lack of confidence or lack of resources.

8.2.2 Impact of Project Closing Oversight

- Project closing is further explained in depth throughout this paper. A comprehensive project closing process would typically include all of the following processes, and may include others, depending on the size, magnitude, complexity, and impact of the project :

1. Making sure all the work that needed to be has been done.
2. Obtaining approval by the project's sponsor and customer (whether internal or external) for the work completed.
3. Reviewing whether or not all organizational governance processes have been executed.
4. Assessing whether or not the necessary project management processes have been applied.
5. Administrative closing of any and all procurements, reviewing that all work on the contract has been completed and that both parties have completed their contractual obligations toward each other.
6. Formally recognizing the completion of a project and its transition to operations.
7. Validating that the project achieved benefits identified in the business case.
8. Capturing of lessons learned : What was done well, and should be documented so it can be repeated in the future ? What could have been done better ? And if so, how can it have been done better ?
9. Disbanding project resources, freeing them to perform other projects and undertake other tasks as required within the organization.
10. Transitioning project deliverables to the customer organization in a manner that warrants seamless operations and support.

❑❑❑