SWE-417 SOFTWARE REENGINEERING

Software Engineering Department

Sir Syed University of Engineering & Technology

Week No. 2

CONTENTS

SOFTWARE EVOLUTION MODELS AND PROCESSES (SEM&P)

- Software maintenance have its own software maintenance life cycle (SMLC) model.
- Three popular SMLC models:
  - Iterative models.
  - Change mini-cycle models.
  - Staged model of maintenance and evolution.

ITERATIVE MODELS

- The iterative models share the ideas that a complete set of requirements for a system cannot be completely understood, or the developers do not know how to build the full system.
- Therefore, systems are constructed in builds, each of which is a refinement of requirements of the previous build.
- A build is refined by considering feedback from users.

ITERATIVE MODELS

Key Features:

- Divides the development/maintenance process into smaller, manageable iterations.
- Each iteration builds on the last, allowing feedback and refinements after every cycle.
- Helps address evolving user requirements and changing environments over time.

- Example: Agile Methodology: Agile development and maintenance processes rely on iterative cycles or "sprints," where functionality is delivered incrementally, ensuring ongoing adaptability.

CHANGE MINI-CYCLE MODELS

- "Change mini-cycle models" refers to a set of structured and incremental steps used to guide the process of making changes or improvements to a software system.
- These models are designed to break down the reengineering process into manageable phases making it more structured and less disorderly.
- Change mini-cycle models are often employed to modernize, update, or enhance existing software systems. They typically consist of the following steps: Identification of Problem Areas , Analysis, Design , Implementation, Testing, Deployment , Monitoring and Feedback, Iteration.

CHANGE MINI-CYCLE MODELS

Key Features:

- Cycles are short and targeted, each dealing with a single change or a small set of changes.
- Maintains focus on incremental updates and rapid release cycles.
- Designed to minimize disruption while addressing small-scale maintenance tasks.

Continuous Integration/Continuous Deployment (CI/CD) Pipeline:

- Overview: In modern development environments, CI/CD models allow for frequent, small-scale changes to be integrated and deployed automatically. Each integration follows a mini-cycle of testing and deployment, enabling rapid updates and fixes.

CHANGE MINI-CYCLE MODELS

Key Features:

- Frequent Changes: Developers integrate small changes into the codebase multiple times a day.
- Automated Testing: Each change is tested automatically to ensure it doesn’t break the system.
- Continuous Deployment: Once a change passes the automated tests, it is deployed to production without manual intervention.

Example: A CI/CD system for a web application where small updates (like feature tweaks or bug fixes) are pushed live multiple times per day.

STAGED MODEL OF MAINTENANCE AND EVOLUTION

- The model is descriptive in nature, and its primary objective is to improve the understanding of how long-lived software evolves.
- The model considers four distinct, sequential stages of the lifetime of a system:
  - Initial Development
  - Evolution
  - Servicing
  - Phase-out

STAGED MODEL OF MAINTENANCE AND EVOLUTION

- **Initial Development:** When the initial version of the system is produced, detailed knowledge about the system is fresh. Before delivery of the system, it undergoes many changes.
- **Evolution:** After the initial stability, it is easy to perform simple changes to the system. Significant changes involve higher cost and higher risk. In the period immediately following the initial delivery, knowledge about the system is still almost fresh in the minds of the developers.

STAGED MODEL OF MAINTENANCE AND EVOLUTION

- **Servicing:** When the knowledge about the system has significantly decreased, the developers mainly focus on maintenance tasks, such as fixing bugs, whereas architectural changes are rarely effected.
- **Phase-out:** Moving from an existing, difficult-to-maintain system to a modern solution system has its own challenges involving wrapping and data migration. After the new system keeps running satisfactorily, sometimes in parallel with the old system, the old system is finally completely shut down.

STAGED MODEL OF MAINTENANCE AND EVOLUTION

- Key Features:
  - Divides the process into stages: Initial Development, Evolution, Servicing & Phase-out.
  - Emphasizes that maintenance and evolution are ongoing activities that occur over time.
  - Allows for gradual evolution while maintaining the system's integrity and addressing immediate needs.
- Example: ISO/IEC 14764; This international standard outlines different types of maintenance activities, aligning well with the staged model of how software evolves over time through these stages.

(SEM&P) - SOFTWARE MAINTENANCE STANDARDS

- IEEE and ISO have both addressed s/w maintenance processes.
- IEEE/EIA 1219 and ISO/IEC as a part of ISO/IEC12207 (life cycle process).
- IEEE/EIA 1219 organizes the maintenance process in seven phases: problem identification, analysis, design, implementation, system test, acceptance test and delivery.
- ISO/IEC describes s/w maintenance as an iterative process for managing and executing software maintenance activities.
- ISO/IEC 14764 is an international standard that provides guidelines for Software Maintenance.

(SEM&P) - SOFTWARE MAINTENANCE STANDARDS

- The activities which comprise the maintenance process are:
  - Process Implementation, Problem and Modification Analysis, Modification Implementation, Maintenance Review/Acceptance, Migration and Retirement.
  - Each of these maintenance activity is made up of tasks.
  - Each task describes a specific action with inputs and outputs.

(SEM&P) - SOFTWARE CONFIGURATION MANAGEMENT (SCM)

- SCM is the discipline of managing and controlling change in the evolution of software system.
- It ensures that the released software is not contaminated by uncontrolled or unapproved changes.
- An SCM system has four different elements, each element addressing a distinct user need as follows:
  - Identification of software configurations.
  - Control of software configurations.
  - Auditing software configurations.
  - Accounting software configuration status.

(SEM&P) - SOFTWARE CONFIGURATION MANAGEMENT (SCM)

- **Identification of software configurations:** This includes the definitions of the different artifacts, their baselines or milestones, and the changes to the artifacts.

- **Control of software configurations:** This element is about controlling the ways artifacts or configurations are altered with the necessary technical and administrative support.

(SEM&P) - SOFTWARE CONFIGURATION MANAGEMENT (SCM)

- **Auditing software configurations:** This element is about making the current status of the software system in its life cycle visible to management and determining whether or not the baselines meet their requirements.

- **Accounting software configuration status:** This element is about providing an administrative history of how the software system has been altered, by recording the activities necessitated by the other three SCM elements.

REENGINEERING

- Reengineering is the examination, analysis and restructuring of an existing software system to reconstitute it in a new form, and the subsequent implementation of the new form.

- Chikofsky and Cross II defines reengineering as: "the examination and alteration of a subject system to reconstitute it in a new form and the subsequent implementation of the new form."

REENGINEERING

- Jacobson and Lindstorm defined following formula:

  Reengineering = Reverse engineering + Δ + Forward engineering

- Reverse engineering is the activity of defining a more abstract, and easier to understand, representation of the system. The core of reverse engineering is the process of examination, not a process of change, therefore it does not involve changing the software under examination.

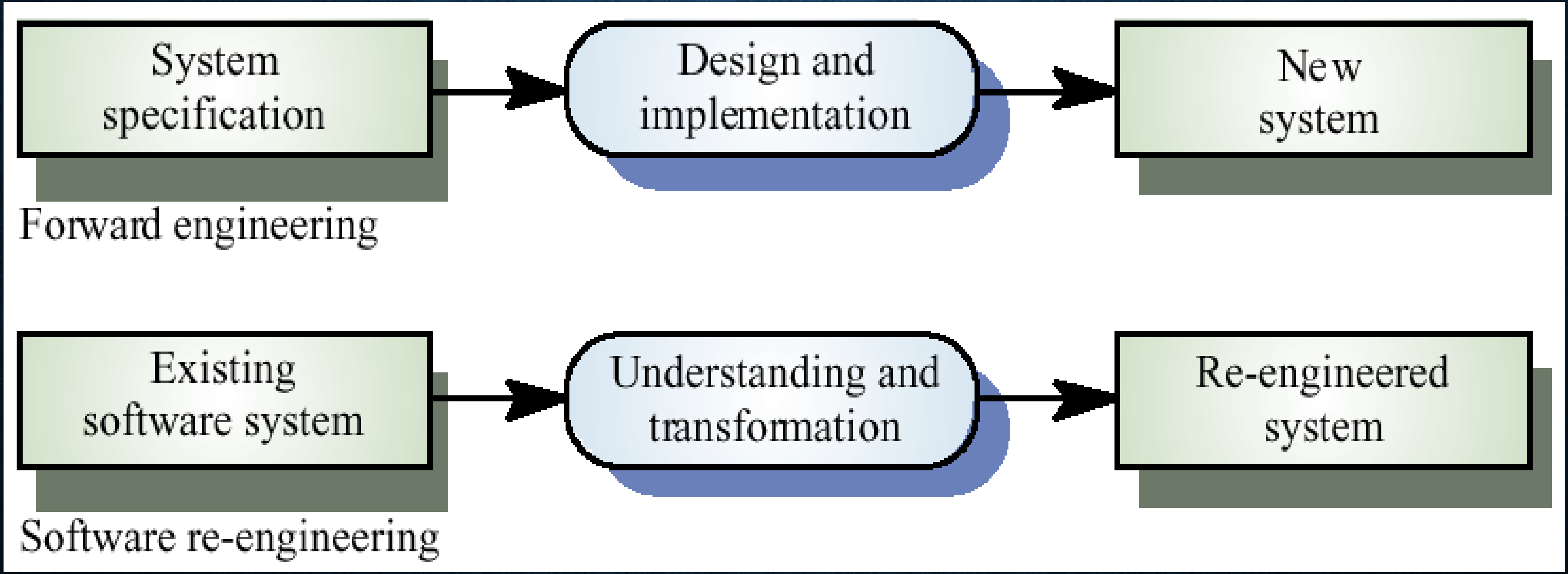
REENGINEERING

- The second element captures alteration that is change of the system.
- The third element "forward engineering," is the traditional process of moving from high-level abstraction and logical, implementation-independent designs to the physical implementation of the system.

SYSTEM RE-ENGINEERING

- Re-structuring or re-writing part or all of a legacy system without changing its functionality.
- Applicable where some but not all sub-systems of a larger system require frequent maintenance.
- Re-engineering involves adding effort to make them easier to maintain. The system may be re-structured and re-documented.

FORWARD ENGINEERING AND RE-ENGINEERING



WHEN TO RE-ENGINEER

- When system changes are mostly confined to part of the system then re-engineer that part.
- When hardware or software support becomes obsolete.
- When tools to support re-structuring are available.
- The term “Re-engineering” is often associated with Business Process Re-engineering (BPR), a specific approach to reengineering business processes within an organization, but the concept can be applied to various domains and contexts.

RE-ENGINEERING ADVANTAGES

- Reduced risk
  - There is a high risk in new software development. There may be development problems, staffing problems and specification problems.
- Reduced cost
  - The cost of re-engineering is often significantly less than the costs of developing new software.
- Preserves Existing Functionality; Maintains Business Continuity
  - Unlike redeveloping a new system from scratch, re-engineering retains the existing business logic, minimizing disruption to ongoing operations.

BUSINESS PROCESS RE-ENGINEERING (BPR)

- Business Process Re-engineering (BPR) is a fundamental redesign of business processes to achieve dramatic improvements in critical aspects such as cost, quality, service, and speed.
- It's a structured approach to reinvent how work is done within an organization, with the goal of optimizing processes to meet current business needs and adapt to changing conditions.
- **Goals:** The primary goal of BPR is to improve overall business performance by streamlining processes, eliminating redundancies, reducing costs, and enhancing customer satisfaction.

STEPS IN BPR

- **Identify Processes**: Identify the processes that need re-engineering.
- **Analyze and Document**: Understand how these processes currently work, document them, and analyze their inefficiencies.
- **Redesign**: Develop a new process design that is more efficient and effective.
- **Implement**: Implement the redesigned processes.
- **Monitor and Improve**: Continuously monitor the performance of the new processes and make improvements as needed.

CHALLENGES OF BUSINESS PROCESS RE-ENGINEERING (BPR)

- Resistance to Change
- High Implementation Costs
- Risk of Failure

Important points:

- Business Process Re-engineering (BPR) is a powerful approach for organizations looking to achieve dramatic improvements in their performance by fundamentally redesigning their business processes.
- By focusing on customer needs, leveraging technology, and radically rethinking how work is done, BPR can lead to significant cost reductions, faster operations, and better service quality.
- However, successful BPR requires strong leadership, effective change management, and careful planning.

EXAMPLES OF BPR

- Business Process Re-engineering (BPR) has been applied in various industries and organizations to streamline operations, improve efficiency, and adapt to changing market conditions.

Some real-world examples of BPR implementations:

- **IBM**: IBM is the example of a company that has used BPR to transform its processes. They have applied BPR to various areas, including customer support, and manufacturing, resulting in reduced costs and improved operational effectiveness.
- **Amazon**: Amazon, one of the world's largest e-commerce companies, continually uses BPR principles to innovate and improve its fulfillment and logistics operations, ensuring rapid order processing and delivery.

BENEFITS OF BUSINESS PROCESS RE-ENGINEERING (BPR)

- Cost Reduction: By eliminating redundant steps, automating processes, and streamlining workflows, organizations can significantly reduce operating costs.
- Enhanced Customer Satisfaction: By designing processes that are more responsive to customer needs, BPR improves customer service and satisfaction.
- Higher Quality & Employee Empowerment
- Better Use of Technology: BPR often incorporates new technologies such as automation, artificial intelligence (AI), or enterprise resource planning (ERP) systems, leading to more efficient processes and improved decision-making.

SUMMARY

- Three popular SMLC models; Iterative models, change mini-cycle models & staged model of maintenance and evolution has been discussed.
- Reengineering = Reverse engineering + Δ + Forward engineering
- Business Process Re-engineering (BPR) is a strategic approach to improving an organization's processes to achieve significant performance improvements in key areas such as cost, quality, service, and speed. Steps , challenges & benefits , all discussed in detailed.