Software Maintenance

Software Maintenance

Software maintenance is a part of the Software Development Life Cycle.

- The primary goal is to modify and update software applications after delivery to correct errors and to improve performance.

Software is a model of the real world.

- When the real world changes, the software requires alteration wherever possible.

Software Maintenance is an inclusive activity that includes

- Error corrections,
- Enhancement of capabilities,
- Deletion of obsolete capabilities, and
- Optimization.

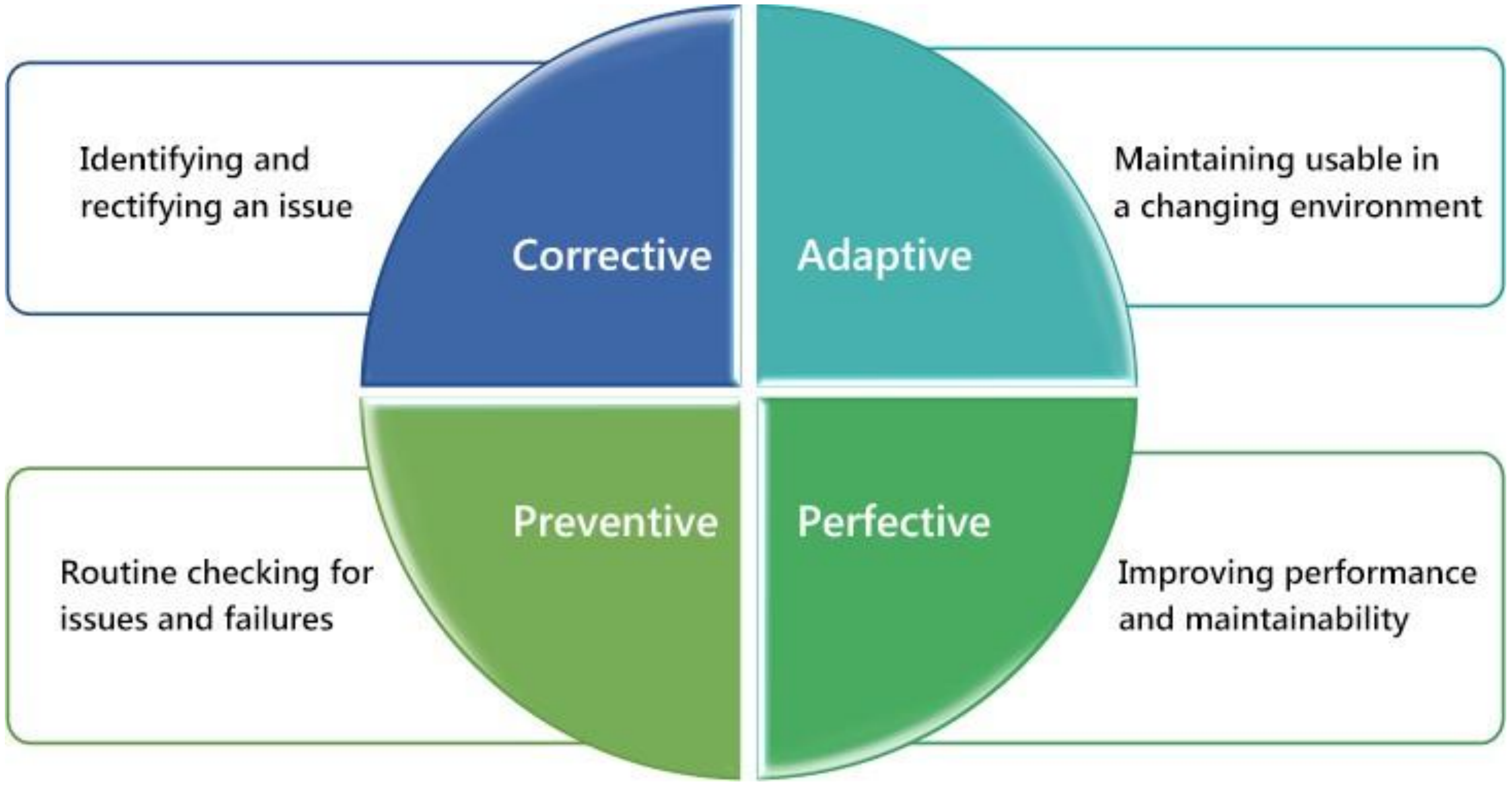
Software Maintenance

Software Maintenance is needed for

- Correct errors
- Change in user requirements with time
- Changing hardware/software requirements
- To improve system efficiency
- To optimize the code to run faster
- To modify the components
- To reduce any unwanted side effects.
- Thus maintenance is required to ensure that the system continues to satisfy user requirements

Software Maintenance

Categories of Software Maintenance


Identifying and rectifying an issue
Corrective
Adaptive
Maintaining usable in a changing environment
Routine checking for issues and failures
Preventive
Perfective
Improving performance and maintainability

Software Maintenance

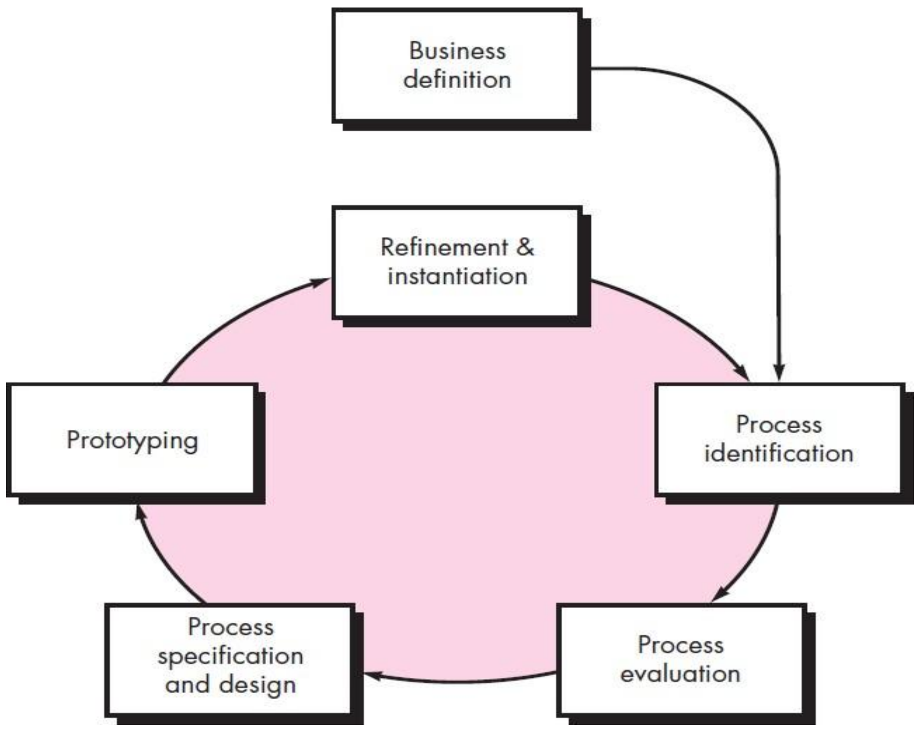
Software Re-engineering

- Software re-engineering is concerned with re-implementing legacy systems to make them more maintainable.
- Re-engineering may involve re-documenting the system, organizing and restructuring the system, translating the system to a more modern programming language and modifying and updating the structure and values of the system's data.
- The functionality of the software is not changed and, normally, the system architecture also remains the same.
- When re-engineering principles are applied to business processes then it is called Business Process Reengineering (BPR).

Software Maintenance

Business Process Reengineering (BPR) Model

- Business process reengineering (BPR) extends far beyond the scope of information technologies and software engineering.



Business definition:

- Business goals are identified within the context of four key drivers: cost reduction, time reduction, quality improvement, and personnel development and empowerment.

Process identification:

- Processes that are critical to achieving the goals defined in the business definition are identified.

Process evaluation:

- The existing process is thoroughly analyzed and measured.

Process specification and design:

- Based on information obtained during the first three BPR activities, use cases are prepared for each process that is to be redesigned.

Prototyping:

•A redesigned business process must be prototyped before it is fully integrated into the business.

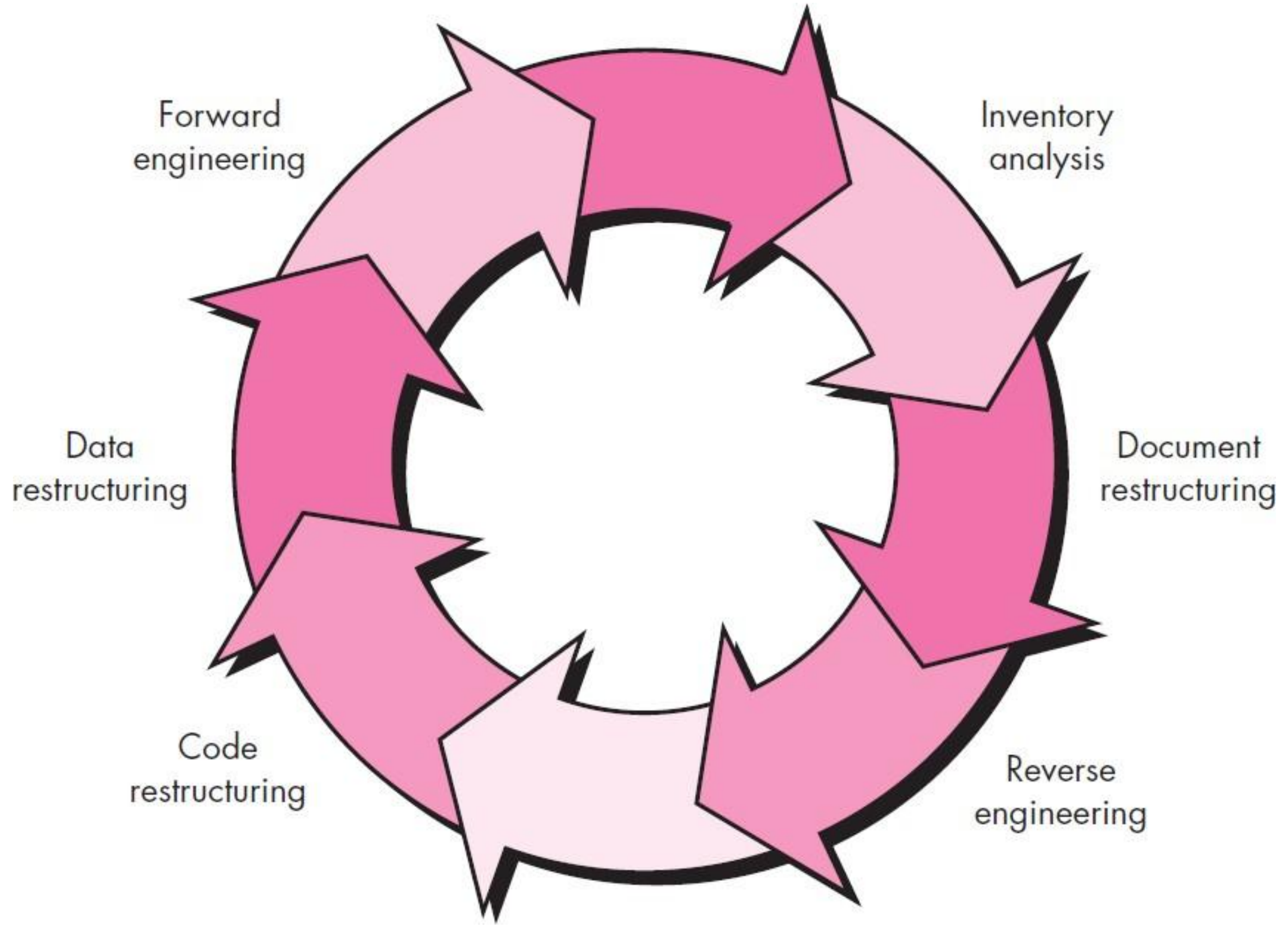
Refinement and instantiation:

•Based on feedback from the prototype, the business process is refined and then instantiated within a business system.

Software Maintenance

Steps in Re-engineering

- Inventory Analysis
- Document Reconstruction
- Reverse Engineering
- Code Reconstruction
- Data Reconstruction
- Forward Engineering



Software Maintenance

Steps in Re-engineering Cont'd

1. Inventory Analysis:

- Every software organization 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.

Software Maintenance

Steps in Re-engineering Cont'd

3. Reverse Engineering:

- The process of analyzing a program in an effort to create a representation of the program at a higher level of abstraction than the source code
- It is a process of design recovery.
- These tools extract data and architectural and procedural design information from an existing program.

4. Code Reconstructing:

- The source code is analyzed using a reconstructing tool.
- Violations of structured programming construct are noted and code is then reconstructed or even rewritten in a more modern programming language.
- The resultant restructured code is reviewed and tested to ensure that no anomalies have been introduced.

Software Maintenance

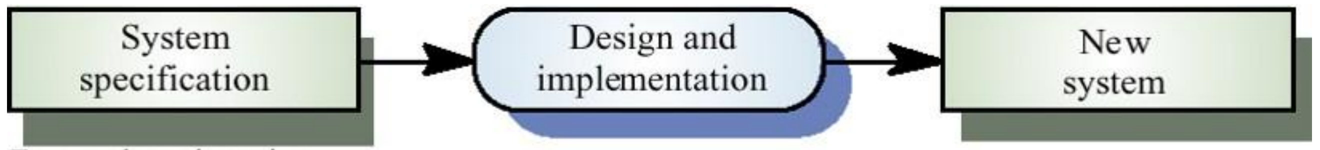
Steps in Re-engineering Cont'd

5. Data Restructuring:

- Begins with a reverse engineering activity and the current data architecture is dissected, and the 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 uses this information to alter or reconstitute the existing system in an effort to improve its overall quality.
- In most cases, reengineered software reimplements the function of the existing system and also adds new functions and/or improves overall performance.



Forward engineering

Software Maintenance

Re-engineering Cost Factors:

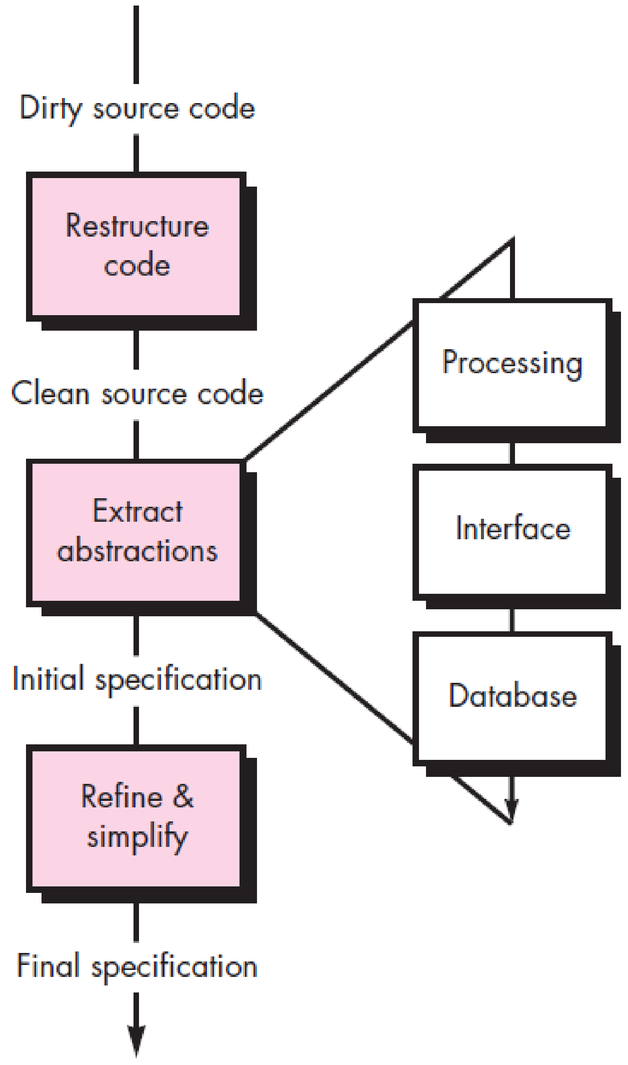
- The quality of the software to be re-engineered
- The tool support available for re-engineering
- The extent of the required data conversion
- The availability of expert staff for re-engineering

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 that 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 to accommodate new skills during re-engineering.

Software Maintenance

Reverse Engineering



- Reverse engineering is the process followed in order to find difficult, unknown and hidden information about a software system.
- May be part of a re-engineering process but may also be used to re-specify a system for re-implementation
- Builds a program database and generates information from this Program understanding tools (browsers, cross-reference generators, etc.) may be used in this process

Reverse engineering is the process of analyzing a subject system with two goals in mind:

(1) to identify the system's components and their interrelationships; and,

(2)to create representations of the system in another form or at a higher level of abstraction

Software Maintenance

Reverse Engineering

The purpose of reverse engineering is to facilitate maintenance work by improving the understandability of a system and producing the necessary documents for a legacy system.

Helps to

- Understand data (internal data structure and database structure)
- Understand processing (overall functionality of the entire application)
- Understand user interfaces (basic actions and behavioral response of the system to these actions)

Difference between Forward Engineering and Reverse Engineering

Key	Forward Engineering	Reverse Engineering
Definition	• Forward Engineering is the mode of creation or development in which the development is done on the basis of given requirements from client/consumer. In this the requirements are provided prior to the development of the application.	• Reverse Engineering is the mode of creation or development in which the development is done on the basis of requirements gathered from the developed application or the changes/enhancements that are provided from the client/consumer.
Execution Time	• As in case of Forward Engineering the application is to be developed from scratch and all approaches and planning need to be done before actual development get stared so it requires more time as compared to that in Reverse Engineering.	• Reverse Engineering takes less time to develop an application as only modifications and enhancements need to develop instead of the core functionality of the application.
Proficiency Skill	• As mentioned above application need to be developed from scratch so high skill proficiency is need to decide approaches and development planning in case of Forward Engineering.	• Reverse Engineering low or medium skill proficiency is also sufficient.
Nature	• On the basis of above-mentioned points we can state that the nature of Forward Engineering is Perspective.	• Reverse Engineering could be stated as of Adaptive nature.
Example	• Example of Forward Engineering could be any newly developing application or system that is started or about to start based on given requirements.	• Reverse engineering includes mainly research and RND work which could get implemented in the already developed application or system for providing more efficient performance with more functionality.

Software Maintenance

The cost-benefit analysis model for reengineering

- Nine parameters are defined

- P1 current annual maintenance cost for an application
- P2 current annual operations cost for an application
- P3 current annual business value of an application
- P4 predicted annual maintenance cost after reengineering
- P5 predicted annual operations cost after reengineering
- P6 predicted annual business value after reengineering
- P7 estimated reengineering costs
- P8 estimated reengineering calendar time
- P9 reengineering risk factor (P9 1.0 is nominal)
- L expected life of the system

- The cost associated with continuing maintenance of a candidate application (i.e., reengineering is not performed) can be defined as

$$C_{\text{maint}} = [P_3 - (P_1 + P_2)] \times L$$

- The costs associated with reengineering are defined using the following relationship

$$C_{\text{reeng}} = P_6 - (P_4 + P_5) \times (L - P_8) - (P_7 \times P_9)$$

- The overall benefit of reengineering can be computed as

$$\text{Cost benefit} = C_{\text{reeng}} - C_{\text{maint}}$$