Ch 1: Basic Concepts

Overview

Introduction to Software Maintenance

Definition

- Changes related to a software system after delivery.
- Set of activities and processes undertaken to manage a software system after its initial deployment.

Set of Activities

- Fixing bugs,
- Making enhancements,
- Adapting to new hardware or software platforms
- Ensuring the software's continued functionality over time.

Other Definitions

Evolution - a process of continuous change from a lower, simpler, or worse to a higher, more complex, or better state.

Maintainability - the ease with which maintenance can be carried out.

Maintenance - modification of a software product after delivery, to correct faults, to improve performance or other attributes, or to adapt the product to a modified environment

Difference Between Maintenance and Evolution

Software Maintenance typically does not involve major changes to software architectures. Changes are made by modifying existing components and adding new component to the system.

Software Evolution is a broader term that encompasses software maintenance and bigger changes. Evolution is very intrinsic to the very nature of software development.

Software maintenance to mean day-to-day changes implemented into a software system

Software evolution refers to what happens to software in the long term, during the whole of its life span.

Misconception

- Common misconception to believe that software is programs and maintenance activities are carried out in software programs. programs.
- Software comprises not only programs
  - source and object code
  - documentation of the program, such as requirements analysis, specification, design, system and user manuals,

Components of a Software System

- Program
    - Source Code
    - Object Code
- Documentation
    - Analysis Specification (Formal Specification, Context Diagram, Data Flow Diagram)
    - Design (Flow charts, Entity Relationship Charts)
    - Implementation (Source Code Listing, Cross-Reference Listing)
    - Testing (Test data, Test results)
- Operating Procedures
    - Manual to set up software System
    - Instructions in case of System Failure

Maintainability of Software

Def: It is the characteristic of software that allows to draw conclusions about how well software can be maintained. It can be used for assessing, controlling and predicting the effort needed to modify the software product.

Maintainable software has five characteristics according to ISO 9126. (ACSTC)

1. **Analyzability**: This sub-characteristic tells us about how well software can be analyzed. It correlates with metrics which tells about the effort needed for diagnosis of deficiencies or causes of failures, or for identification of parts to be modified. Ex:LOC, NOC.
2. **Changeability**: This sub-characteristic tells us how well software can be changed. It correlates with metrics which measure about the effort needed for modification, fault removal or for environmental change. Ex: LOC, NOM, NOC
3. **Stability:** This sub-characteristic tells us about how stable software is. It correlates with metrics which measure attributes of software that allow to conclude about the risk of unexpected effects as result of modifications. Ex: CBO, CDBC, CDOC
4. **Testability**: This sub-characteristic tells us about how well software can be tested and is tested. It correlates with metrics which measure attributes of software that allow to conclude about the effort needed for validating the software and about the test coverage. Ex: NOC, WMC etc
5. **Compliance:** The compliance sub-characteristic tells us about how well software adheres to application related standards, conventions, and regulations in laws and similar prescriptions. It correlates with metrics which measure attributes of software that allow to conclude about the adherence to application related standards, conventions, and regulations in laws and similar prescriptions.

New Developments vs Maintenance

Difference On	New Development	Maintenance
Activities	Development of a software from the beginning.	Changes related to a software system after delivery.
Impact of new features	Impacts on changing requirements and and designs.	Aligning and co-ordinating with existing features
Cost	Development Cost	Estimated expenditure at 40-90% of the costs of the entire life-cycle of the software system.

Need for Software Maintenance

- Companies spend more time maintaining existing software than developing new software.
- Maintenance accounts for 60-70% of the total software lifecycle costs, making it a significant expense.
- Effective maintenance is essential to ensure software reliability, usability, and performance over time.
- Neglecting maintenance can lead to increased defects, reduced flexibility, and system failures.

Importance of Software Maintenance

- **To provide continuity of service:** Systems need to keep running. For example, software controlling aeroplanes in flight or train signalling systems cannot be allowed just to stop if an error occurs. Unexpected failure of software can be life threatening.

- **To support mandatory upgrade:** This type of change would be necessary because of such things as amendments to government regulations e.g. changes in tax laws
- **To support user requests for improvements:** Users will request enhancements in functionality. There may also be requirements for better requirements for better performance and customisation to local working patterns.
- **To facilitate future maintenance work:** Shortcuts at the software development stage are very costly in the long run. It is often financially and commercially justifiable to initiate change solely to make future maintenance easier.

Case Study: NASA – Space Shuttle Software Maintenance

- **Context**: The Space Shuttle's onboard software had over **400,000 lines of code** and required extremely high reliability.

- **Maintenance Practices**:
  - Code was maintained for **over 20 years**.
  - Followed strict processes involving formal methods, inspections, and continuous regression testing.
  - Achieved an *extraordinarily low defect rate* of about 1 per 500,000 lines of code.

- **Takeaway**: Rigorous, well-documented maintenance processes are critical for safety-critical systems.

Categories of Software Change/Maintenance

- **Corrective Maintenance:** Modification initiated by defects in the software.
- **Adaptive Maintenance:** Change driven by the need to accommodate modifications in the environment of the software system.
- **Perfective Change:** Change undertaken to expand the existing requirements of a system.
- **Preventive Change:** Change undertaken to prevent malfunctions.

Correlation of Maintenance Types and Tasks

Corrective Maintenance

Task: Bug tracking and analysis, Program comprehension, Debugging, Defect repair, Regression testing

Adaptive Maintenance

Task: Impact Analysis, Dependency analysis, Configuration management, Interface adaptation

Perfective Maintenance

Task: Code refactoring, Performance tuning, UI/UX improvements, Adding non critical features, documentation “ updates, program restructuring

Preventive Maintenance

Task: Code reviews, Eliminating dead code, Improving test coverage, Refactoring to reduce complexity, Updating outdated libraries or frameworks

*Often overlaps with perfective maintenance but is more **proactive**