SE 811: Software Maintenance

Toukir Ahammed

Another flaw in the human character is that everybody wants to build and nobody wants to do maintenance.

—Kurt Vonnegut, Jr.

Software Maintenance

- In circa 1972, R. G. Canning in his landmark article "The Maintenance 'Iceberg'," discussed the problems of software maintenance.
- R. G. Canning compared software maintenance to an "iceberg" to emphasize the fact that software developers and maintenance personnel face a large number of problems
- A few years later, in 1976, Swanson introduced the term "maintenance" by grouping the maintenance activities into three basic categories: corrective, adaptive, and perfective

Software Maintenance

- Practitioners took a narrow view of maintenance as correcting errors, and enhancing the functionalities of the software.
- Hence, maintenance can be inappropriately seen as a continuation of software development with an extra input—the existing software system

Software Maintenance

The ISO/IEC 14764 standard defines software maintenance as:

- "... the totality of activities required to provide cost-effective support to a software system. Activities are performed during the pre delivery stage as well as the post-delivery stage."
  - Post-delivery activities includes changing software, providing training, and operating a help desk.
  - Pre-delivery activities include planning for post-delivery operations.
- The process of modifying a software system or its components after delivery to correct faults, improve performances or other attributes, or adapt to a changed environment

Software Maintenance and Maintainability

The ISO/IEC 14764 standard defines software maintainability as

- "… the capability of the software product to be modified. Modification may include corrections, improvements or adaptation of the software to changes in environment, and in requirements and functional specification"

Software Maintenance and Software Development

- Maintenance is event driven, whereas development is requirements driven
- A process for software development begins with the objective of designing and implementing a system to deliver certain functional and nonfunctional requirements.
- On the other hand, a maintenance task is scheduled in response to an event. Reception of a *change request* from a customer is a kind of event that can trigger software maintenance.

Software Maintenance and Software Development

- Similarly, recognition of the needs to fix a set of bugs is considered another kind of event. Events originate from both the customers and from within the developed organization.
- Generally, the inputs that invoke maintenance activities are unscheduled events; execution of the actual maintenance activities might be scheduled according to a plan, but the events that initiate maintenance activities occur randomly.
- A maintenance activity accepts some existing artifacts as inputs and generates some new and/or modified artifacts.

Evolution vs Maintenance

- The terms evolution and maintenance are used interchangeably. However, there is a semantic difference.
- The concept of *software maintenance* means preventing software from failing to deliver the intended functionalities by means of bug fixing.
- The concept of *software evolution* means a continual change from a lesser, simpler, or worse state to a higher or better state.
- All support activities carried out *after* delivery of software are put under the category of *maintenance*.
- All activities carried out to effect changes in requirements are put under the category of *evolution*.

Evolution vs Maintenance

- Maintenance of software systems primarily means fixing bugs but preserving their functionalities.
- Maintenance tasks are very much planned. For example, bug fixing must be done and it is a planned activity.
- In addition to the planned activities, unplanned activities are also necessitated. For example, a new usage of the system may emerge.
- Generally, maintenance does not involve making major changes to the architecture of the system. In other words, maintenance means keeping an installed system running with no change to its design.

Software Maintenance vs Hardware Maintenance

- Maintenance of physical systems often requires replacing broken and worn-out parts. For example, owners replace the worn-out tires and broken lamps of their cars. Similarly, a malfunctioning memory card is replaced with a good one.
- On the other hand, software maintenance is different than hardware maintenance.
- In hardware maintenance, a system or a component is returned to its original good state.
- On the other hand, in software maintenance, a software system is moved from its original erroneous state to an expected good state.

Evolution vs Maintenance

- Evolution concern whatever happens to a system over time.
- Evolution of software systems means creating new but related designs from existing ones. The objectives include supporting new functionalities, making the system perform better, and making the system run on a different operating system.
- Basically, as time passes, the stakeholders develop more knowledge about the system. Therefore, the system evolves in several ways. As time passes, not only new usages emerge, but also the users become more knowledgeable.
- As Mehdi Jazayeri observed: “Over time what evolves is not the software but our knowledge about a particular type of software”.

Classification of Software Maintenance

- There will always be defects in the delivered software application because software defect removal and quality control are not perfect.
- Therefore, software maintenance is needed to repair these defects in the released software.
- Maintenance activities can be classified from different viewpoints:
  - Intention-based classification of software maintenance activities;
  - Activity-based classification of software maintenance activities; etc.

Intention-Based Classification of Software Maintenance

Maintenance activities are divided into four groups:

- **Corrective maintenance**
- **Adaptive maintenance**
- **Perfective maintenance**
- **Preventive maintenance**

Corrective Maintenance

- The purpose of corrective maintenance is to correct failures: processing failures and performance failures.
- Examples of corrective maintenance:
  - A program producing a wrong output is an example of processing failure.
  - Similarly, a program not being able to meet real-time requirements is an example of performance failure.

Corrective Maintenance

- The process of corrective maintenance includes isolation and correction of defective elements in the software.
- There is a variety of situations that can be described as corrective maintenance such as correcting a program that aborts or produces incorrect results.
- Basically, corrective maintenance is a reactive process, which means that corrective maintenance is performed after detecting defects with the system.

Adaptive Maintenance

- The purpose of adaptive maintenance is to enable the system to adapt to changes in its data environment or processing environment.
- This process modifies the software to properly interface with a changing or changed environment.
- Adaptive maintenance includes system changes, additions, deletions, modifications, extensions, and enhancements to meet the evolving needs of the environment in which the system must operate.

Adaptive Maintenance

Examples of Adaptive maintenance are:

- changing the system to support new hardware configuration;
- converting the system from batch to on-line operation; and
- changing the system to be compatible with other applications.

A more concrete example is:

- an application software on a smartphone can be enhanced to support WiFi-based communication in addition to its present Third Generation (3G) cellular communication

Perfective Maintenance

- The purpose of perfective maintenance is to make a variety of improvements, namely, user experience, processing efficiency, and maintainability.
- Examples of perfective maintenance are:
  - the program outputs can be made more readable for better user experience;
  - the program can be modified to make it faster, thereby increasing the processing efficiency;
  - and the program can be restructured to improve its readability, thereby increasing its maintainability.

Perfective Maintenance

- Activities for perfective maintenance include restructuring of the code, creating and updating documentations, and tuning the system to improve performance.
- It is also called “maintenance for the sake of maintenance” or “reengineering”.

Preventive Maintenance

- The purpose of preventive maintenance is to prevent problems from occurring by modifying software products.
- Basically, one should look ahead, identify future risks and unknown problems, and take actions so that those problems do not occur.
- For example, good programming styles can reduce the impact of change, thereby reducing the number of failures. Therefore, the program can be restructured to achieve good styles to make later program comprehension easier.
- Some researchers and developers view preventive maintenance as a subset of perfective maintenance.

Preventive Maintenance

- Preventive maintenance is very often performed on safety critical and high available software systems.
- The concept of “software rejuvenation” is a preventive maintenance measure to prevent, or at least postpone, the occurrences of failures (crash) due to continuously running the software system.
- It involves occasionally terminating an application or a system, cleaning its internal state, and restarting it.
- Rejuvenation may increase the downtime of the application; however, it prevents the occurrence of more severe failures.

Activity-based Classification of Software Maintenance

Maintenance modification activities are classified into two categories:

- **Corrections**: Activities in this category are designed to fix defects in the system, where a defect is a discrepancy between the expected behavior and the actual behavior of the system.
- **Enhancements**: Activities in this category are designed to effect changes to the system. It is further divided into three subcategories as follows:
  - enhancement activities that modify some of the existing requirements implemented by the system;
  - enhancement activities that add new system requirements; and
  - enhancement activities that modify the implementation without changing the requirements implemented by the system.