How Accountability Improves Software Reliability?

Pawan Kumar Chaurasia
Department of Information Technology
Babasaheb Bhimrao Ambedkar University
Lucknow (U.P), India.
pkc.gkp@gmail.com

Abstract—This paper encourage accountability as a principle for software reliability. There are various techniques to estimate software reliability which is tested. This paper treats to identify the faults which are put in the design into the system. It facilitates to detect, isolate and prevent such faults to inducements for spitefulness deportment. A major problem which comes in future is to fix the accountability, quantified by the probability to be exposed the weak section of the software reliability.

I. INTRODUCTION

Today, every field of corporate, organization and institution is change through advancement of technology. Through advancement, it is beneficial for the society and technology. But from these developments, it is observe that there are some dangers and harms for the society, which outcome. These harms affect a small change in a large and explosive in every part of public and private life. People day-by-day depends on technology like air-craft, train, appliances, equipments, communications media devices etc. These items make society dependent and consent about the safety, security, accountability and reliability.

Software reliability is a key factor for software quality [1]. Software and hardware reliability both have different mechanism and parameters to identify the cause of failure. Hardware faults are physical faults while in software, it comes from design. Hardware faults are easily identified and correct, while software faults are hidden, which are harder to identify, correct, detect and classify [2]. One of the most important attributes of software reliability engineering is 'reliability'. It is to identify the customer satisfaction factors regarding performance, usability, efficiency, portability, functionality, capability, maintainability and documentation etc. Software reliability engineering played an important role in the quantitative measurement / assessment of software quality. Software reliability engineering is defined as the probability of failure free operation for a specified period of time, in specified environment used by the user.

In this paper we define and search various factors of accountability for software reliability. These properties help to determine if a system performance is satisfactory but the design input perform wrong as required by the user. Which reflect that the system is not performing as required by the system designed? They can ponder on the reliability of the software. Accountability is important; when the system is developed in a group of development organization in various phases. It is essential to fix the accountability of developers who partially or fully involves in the whole development. We here indicate that system builders should consider accountability as a reliability factor to estimate the software reliability. Contribution of this paper is to focus on various factors of accountability which can improve the software reliability estimation at the time of testing and find the faults from the developer/user.

II. ACCOUNTABILITY ATTRIBUTES

It refers to make assurance of semantic behavior that improves the basic perimeter of software reliability and quality of the software product. A real example of an accountability service is transition through SMS, create password through mobile SMS can provide for the privacy, integrity, security as well as accountability of the user [3]. Through SMS service, accountability of a user is to accept the message for transmission, and response the receiver accountable for accepting its transmission [4]. From these facilities we can easily identify the behavior, knowledge, expression and actions of accountable system. Accountable system should be:

- **Authentic:** Actions of an accountable user are authentic; provide id and password to all the access. Through authentication; increase the accountability of the user. They cannot deny, if any hazards are occurred from the user login.
- **Certified:** A client, correspond, monitor or external member may verify that an accountable service is deporting in an accurate manner, and prove any misbehave from the arbitrator.
- **Observable:** It is observe that if the user demoralized the services state obtains a high probability of detection. An external attainder may identify the sequence of operations issued on the service.
- **Secured:** From authentication, certification and observation make system more secured and increased the accountability of the software from the user. Increased of secured data through SMS; may increase

the accountability of the user. If any data loss from the user login; it can be easily identify and track from the login details.

- **Confidentiality:** When the development is started every module of the software accountability is fixed. This increased the confidence of the development team members.

From these properties; accountability also satisfied the system dependability properties which are described here [5]:

- **Fault Prediction:** It is the first step to evaluate no of faults and the probability of occurrence of failure. Fault prediction is to formulate the relationship between failure data and the operational profile [6][7]. At the initial level prediction of software errors and failure are decreased.
- **Fault Identification:** It is the next phase after prediction. Error and failure is used to identify the number of faults in the software. There are several methods to identify the fault from the software.
- **Fault Prevention:** It is implemented at the design level, to make software demonstrated fault-free. If faults cannot be identify, then it can be prevent from faults in the software by using good software, good writing code, understand the function and user friendly of the software.
- **Fault Tolerate:** It deals with all the faults which are stay after the system is developed and established. It is the last step to identify the reliable software. Fault tolerant techniques have two different groups, single version and multi-version techniques. Fault-tolerance that has been dealt with critical applications like aerospace system, railway systems and nuclear power plant.
- **Fault Removal:** The aim of this phase is, to remove the faults after the development of the software by verification & validation. Exhaustive testing is applied and removed all the faults from the software.

III. ACCOUNTABILITY TYPES

When software development is commenced, each individual on the project team should have responsibility of the results, which he or his team is accountable, and the future benefits depend on how well he does in producing those results. For large project it is necessary to split the project into small module and organize the project into these modules in such a way that each individual should be accountable for his job. In software development, software development life-cycle is generated, and it is divided into various phases in fig. 1. Each phase have its own accountability to fulfill the requirement of the project within due time.

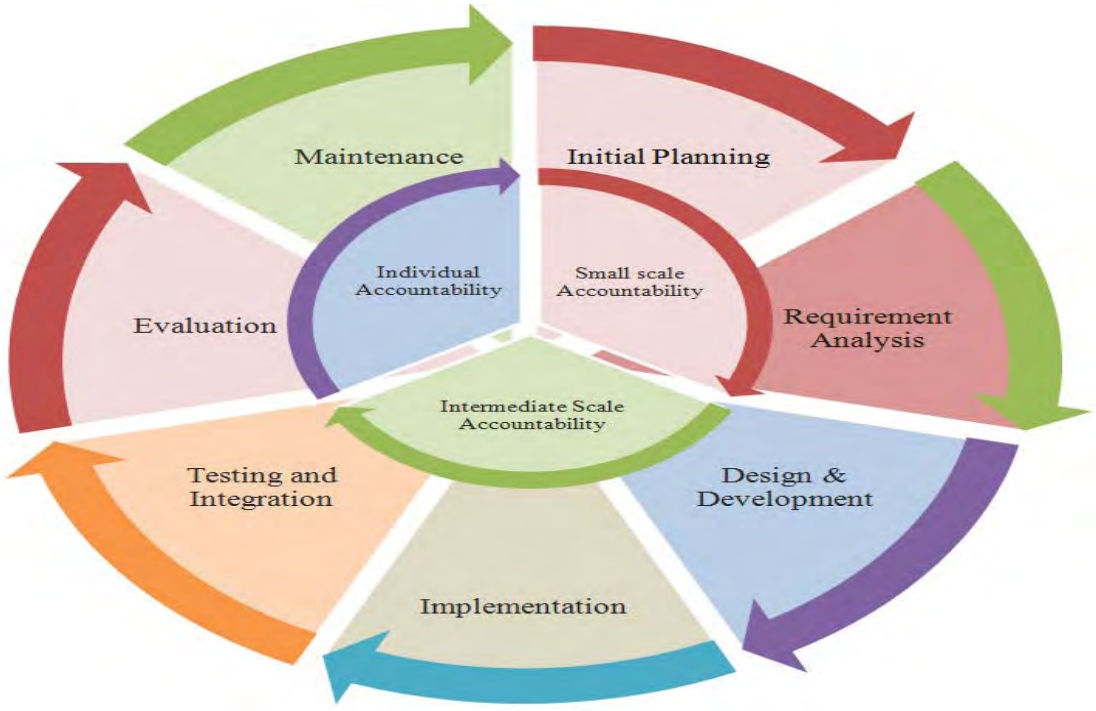

Figure 1: Accountability Software Development Life-Cycle

On the base of above software development life cycle, accountability is categorized into three parts on the base of module distribution, which is as follows:

Small-scale Accountability: On a small-scale within the project, first phase is the most important phase at the time of development. In requirement phase project manager and team have cleared the objectives (Modules, Schedules and Budgets) of the user. They also include the various assumptions and interface which are

applicable at the time of development. Software requirements, computer availability, other requirements and influenced by how well they perform their responsibility.

Intermediate-Scale Accountability: It is the mechanism, which include analysis and design of the software but also formal agreement between user and developer. In this agreement, Project Work Authorization between the Project Organization and the Development Organization, identifying the specific subproject products to be developed, and their associated modules and budget. Parallel team prepared a standard to accept the module from another team. After analysis, design and coding, the independent test team will not accept a module for test until it fulfills all the test objectives [8]:

- Properly test every branch and statement.
- Provide a test case list, based on a list prepared from the design specification prepared the design by the independent test team.
- Prepare standards for all projects through a standard test phase.
- All the test case, modules, function and statement are tested and execute as require by the user and standard prepare by the test time.

Individual Accountability: It is the most effective device for ensuring individual accountability. The whole software is divided into various phases and each phase is allotted for individual team on its own accountability to be submitted after proper execution from the team. If there is any defect found then the accountable person is responsible [9]. Every software development module accountability, which defines all the requirements, design, coding and testing are associated with everyday. Profile and accountability are assured is presented in figure 2. Each phase of the figure describes each of the functions: requirement, design, assumptions and test cases etc. This finds his personal consignment to meet all the due dates within time. Each phase is reviewed independently by a supervisor or tester to assure that it has been satisfactory finished.

The leading reflect that the accountability and personal commitments have been producing a good result. Instead of wasting time in head-to-head communication; supervisor focused on each employee; how well each performer is doing his best and more effective. The commitment of each performer will lead more focus for thorough preparation or putting some useful inputs in order to meet out the output as per required within scheduled time.

Sr. No	Software Phases	Due Date	Completion Date	Originator	Reviewer	Comments
1	Requirements Specification	11-1-14	11-1-14	System Analyst	Analyst	On time
2	Design Description	11-1-14	11-1-14	System Analyst	Analyst	On time
3	Functional Flow Chart	25-1-14	25-1-14	System Analyst	Analyst	On time
4	Interface Design	28-1-14	28-1-14	System Designer	Analyst	On time
5	Constraints and Assumptions	02-02-14	02-02-14	System Designer	Analyst	On time
6	Code Module	06-02-14	06-02-14	Programmer	Analyst	On time
7	Test Cases	06-02-14	06-02-14	Testing Team	Analyst	On time
8	Review Test Cases	06-02-14	06-02-14	Testing Team	Analyst	On time
9	Test Case Output	15-03-14	15-03-14	Testing Team	Analyst	On time
10	Detailed Flow Chart	15-03-14	18-03-14	System Analyst	Analyst	Delay
11	Update Flow Chart	15-03-14	15-03-14	System Analyst	Analyst	On time
12	Update Design	25-8-14	25-08-14	System Analyst	Analyst	On time
13	Update Implementation	25-08-14	28-08-14	System Analyst	Analyst	Delay

Figure 2: Software Development Module Accountability

The above result represents accountability and commitments of the reviewer, which have been very useful for improvement. Instead of a time-consuming and undetermined situation, team-leaders can find out how well each participant is doing on his part, and thus he can take right decision in earlier stage, which is more effectively. The reviewer commitment made by the performer will often lead to do more preparation, or putting some extra effort, in-order to execute the module within schedule date and time.

IV. CONCLUSIONS

Today a person depends on software, which increases the reliability problem in future. These problems take place when faults are encountered during the execution of the program. Software failure occurs, if the behavior of the software differs from actual performance of the software. These failures can be minimized through accountability of every module for the project development team and each member, who involve in the development. The accountability can be reviewed by the reviewer team and assessed by promoting through incentives to the employee. The primary objectives of accountability are;

- To organize the project with distinct responsibilities and appoint authorized person with responsibility.
- To organize the goal and incentive organization in such a manner that they are mutually reinforces to achieve.
- When accountable person achieved the goal and execute the module without error; which minimize the software faults and improve software reliability.
- After assemble and execute the module and get error free, it means accountability increase in same proportion as the software reliability.

V. REFERENCES

[1] J. D. Musa, "Theory of Software Reliability and its Applications", IEEE Transaction on Software Engineering, SE-1, pp-312-327.
[2] P. K Chaurasia, "Operational Profile: A Critical Review", Journal of Global Research in Computer Science, ISSN 2229-371X, Vol 4, No8, Aug 2013, pp 57-61.
[3] A. R. Yumerefendi and J. S. Chase, "Trust but Verify: Accountability for Network Services", in Proceedings of the 11th ACM SIGOPS European Workshop, September 2004.
[4] A. Anagnostopoulos, M. T Goodrich and R. Tamassia, "Persistent Authenticated Dictionaries and their Applications" in 4th International Conference on Information Security, ISC, Oct 2001.
[5] P. K. Chaurasia, Software Reliability Chain Model", International Journal of Software and Web Services (IJSWS), ISSN: 2279-0063, Vol 8(1), May 2014, pp 46-50.
[6] R.A. Khan, K. Mustafa and S. I Ahson "Operational Profile Factor for Reliability Estimation", University Press, CIT 2004, pp 347-354.
[7] P. K. Chaurasia, "Developing Operational Profile for University Library Management System" ISSN 0031-4773, Vol 76, No 6, June 2014, pp 2-7.
[8] F. J Mullin, "Software Test Tools: Project Experience" Proceedings of the TRW Symposium on Reliabile, Cost Effective, secure Software, TRW-SS-74-14, 1914.
[9] R. D. Williams, "Managing the Development of Reliable Software" Proceedings of the 1975 International Conference on Reliable Software, IEEE Cat No 75, CH 0940-7CSR, 1975, pp. 3-8.