Reduction of enhanced maintenance effort using ARM model and RMMM plan

Muhammad Inayat Ullah
NWFP, University of Engineering & Technology, Peshawar, Pakistan

Muhammad Shahid
University of Technology Malaysia (UTM)

Nazir Muhammad
ICMS University Campus Hayatabad Peshawar, Pakistan

Muhammad Zeeshan
ICMS University Campus Hayatabad Peshawar, Pakistan

ABSTRACT: Software maintenance effort is playing a very important role for the development of the software. In maintenance phase user request for change and effort required for the maintenance of software is more as compare to the over all development of the software. Enhanced maintenance is very important factor among all factors in the software maintenance. In maintenance effort distribution, about fifty percent (50%) of effort consume by the Enhanced maintenance. To reduce these effort additional requirements manipulation model is used. Additional requirements manipulation model is the modular library of Design, coding and testing of functional, non-functional and the domain requirements. Although additional requirement manipulation is reduce the Enhanced maintenance model effort. But highly risk is involve in addition of a new module or replacement of the module form the existing software on user request to reduce the risk RMMM (Risk Mitigation, Monitoring and Management) Plan as to use for the review of the risk and for the risk planning.

KEYWORDS: Enhanced Maintenance, Risk Mitigation, Risk Monitoring, ARM, RMMM.

Introduction

Successful software is the one that fulfill the needs of the people who use it. It will remain in the market for a long time if it is modifiable and easier to use, achievement of such software is the result of the good effort. We need discipline for designing and development of the software. We need engineering approach when a software couldn't modify and harder to use. Such software should be the result of failed efforts and undisciplined approach.

Over all development effort of software is divided in to two parts.

- Software development effort
- Maintenance effort

More effort is required for the maintenance of the software as compare to the software development [Li7]. Risk analysis is a series of steps, which are used to understand and manage uncertainty in software development process. The risk may or may not happen; i.e., there are no 100% probable risks. If the risk becomes veracity, unwanted penalty or losses will de occurred [HP4]. Unpredictable risks are the joker in the deck [Cha089].

When user request for change in the software then maintenance phase is categorized into four parts.

- Correction
- Adaptation
- Enhancement
- Prevention

But in the maintenance distribution, Enhanced maintenance is more approximately fifty percent (50%) as compare to correction, adaptation and prevention [CC00]. Mathematically risk exposure is determined by the following relationship.

$RE = P \times C$

where "C" in the risk equation is the cost of project and "P" is the probability of occurrence for the risk [Hal98].

To deal with the risk in the project consider the three issues:

- Risk Mitigation
- Risk Monitoring
- Risk Management

Risk information sheet is used to document each risk [WW97]. A typical distribution of effort among the different phases of the software development is shown in figure-1.

- Requirement - 10%

- Design - 20%
- Coding - 20%
- Testing - 50%

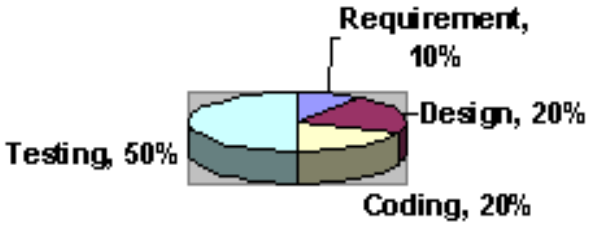

Figure 1. Effort distribution in software development phases

In the figure-1 testing phase effort is maximum among all the other phases of software development witch is 50% so more effort is require for testing of a software [fre**].Distribution of the error occurrence in the different phases of the software development is shown in the figure-2.

- Requirements Analysis-20%
- Design-30%
- Coding-50%

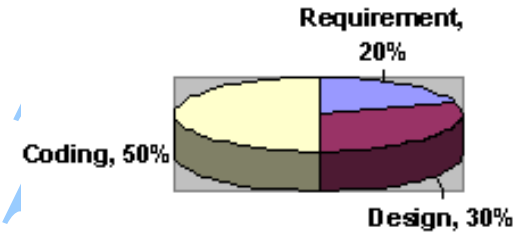

Figure 2. Error occurrence in software development phases

In the figure-2 error occurrence in the coding phase of the software development is maximum as compare to the requirement analysis and design phase so more effort is require for reducing errors in the coding phase [fre**].

Maintenance phase takes maximum effort among all the phases as shown in the figure-3.

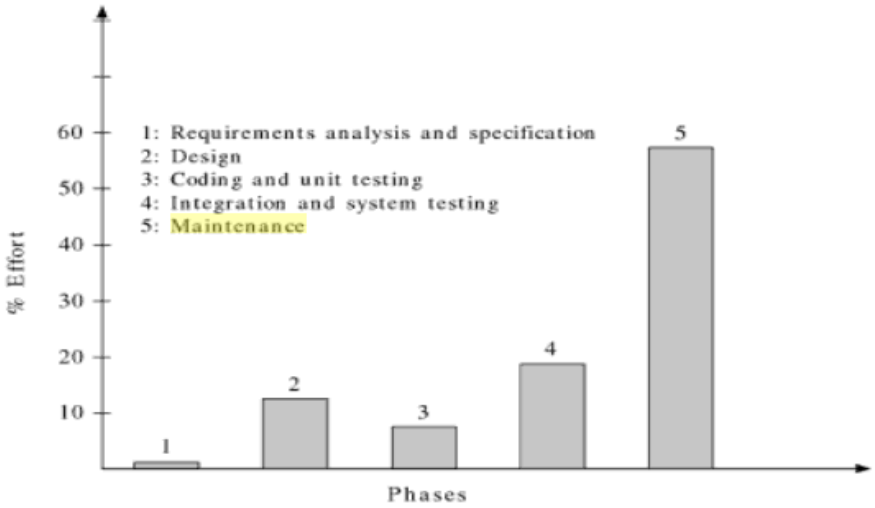

Figure 3. Effort Distribution between different activities of software development

From the figure-3 it is shown that maintenance phase require more effort as compare to the software development phases [BR00]. Maintenance phase is the last activity of the software development, it starts when the software is released. It begins with the feedback and error report of the users. When feedback and error reports are received from the users then the code of the software is updated and test cases are applied to the updated code after testing phase updated code is then implemented in the implement phase before the release of the new version of code [Sta03]. Maintenance phase is further divided into four parts which are following.

- Adoptive Maintenance
- Corrective Maintenance
- Preventive Maintenance
- Enhanced Maintenance

In the above-mentioned maintenance factors enhanced maintenance is require more effort among all the other factors. Distribution of efforts among the four factors of the maintenance phase is shown in the figure-4.

- Adoptive Maintenance-25%
- Corrective Mintenance-21%
- Preventive Maintenance-4%
- Enhanced Maintenance-50%

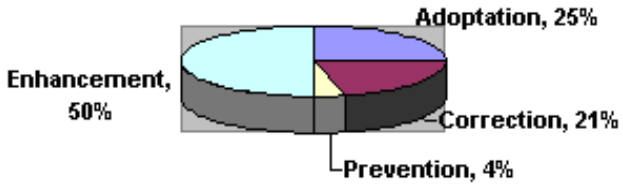

Figure 4. Effort distribution in the Maintenance Phase

In the figure-4 enhanced maintenance effort is maximum among the other entire maintenance factor. In our research we are reducing the enhance effort by using ARM (Additional Requirement Management) model and RMMM (Risk Mitigation and Monitoring and Management) plan in the enhanced maintenance.

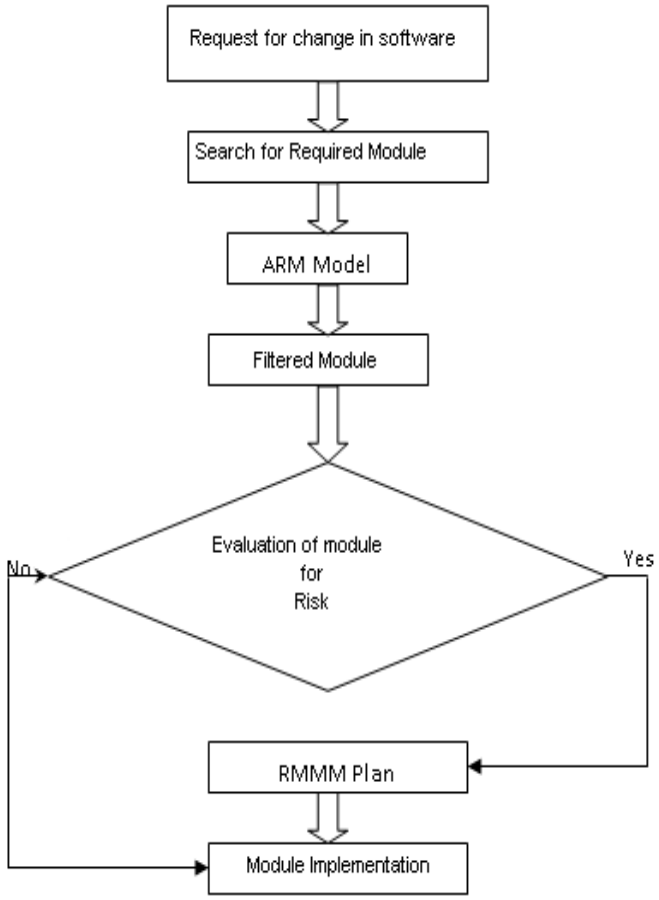

Figure 5. ARM and RMMM model for the reduction of Enhanced Maintenance Effort

1 ARM and RMMM Model

In overall effort of the software development maintenance effort is more as compare to the development effort. In this research proposed Model is designed for the reduction of the effort in the enhanced maintenance which shown in the figure-5.

In the proposed model ARM (Additional Requirement Management) is a library consisted of modules. This library is behaves like a database of functional, non-functional and domain functional components of the software. This library consisted of modules, which are coded, tested and designed. And this library is call on demand on the user request. Where RMMM (Risk Mitigation, Monitoring and Management) is a plan that is used to solve the problem of risk that is occurred during the module implementation. This model is reduced the maintenance effort when user request for change in the existing software.

I Additional Requirements

In the maintenance phase the user requests for change in software such requests are known as additional requirements. In the request, user wants to add or replace software component. These requests may be functional, non-functional and domain functional.

Functional requirement are the requirement related to the software or its components it is consisted of inputs, outputs and behavior. Where non-functional requirements are quality attributes, quality goals, constraints and quality of service and non-behavioral requirements. Domain requirements are the subactivity in which reusable components of the software are consider for the development of the software.

II Additional Requirement Management Model

When user want to add or replace functionality of the software. Additional requirement management is consisted of functional, non-functional and domain module of the software. It is like a library .in additional requirement management designing, coding and testing of each module is stored. And user select the required module form the additional requirement management library. This library plays main role to reduce the enhanced maintenance effort.

III Module Registration

When the specified module is located from the additional requirement management library. Then the selected module is registered for further implementation. It takes out the code of the component of the software that is already designed and tested in the additional requirement management library. After module registration selected module is then send to the RMMM plan for risk planning and management that is used to reduce the risk in the module implementation means when add any component of the software on the user request.

IV RMMM Plan

Main issues in the software development are time tradeoff and space tradeoff. To resolve these types of issues, which occur in the module implementation, we need RMMM Plan. In the RMMM plan risk mitigation is the problem avoidance activity and risk monitoring is the project tracking activity. Three steps to monitor the risk are as under.

- Identify the risk where the risk is occurring.
- Proper management for controlling the risk
- Collect information of future risk analysis

All the three activities of RMMM plan are used to reduce the risk occurred in the software. When all the risks have been identified these risks are evaluated to determine their probability of occurrences. If more risk is occurred in the module implementation then track out the risk where it is occurred. If risk occurrence is less for the module implementation then avoid it.

2 Working of the model

In the research, proposed model shown in figure-5 is divided into seven major activities. Description of each activity of the model is following.

- User request for the addition or replacement of the functionality in the software.
- Search for specific module from the ARM model
- ARM model is the library of designing, coding and testing of the functional, non-functional and domain requirements.

- Specified module is then registered for implementation
- If risk is occur for the implementation of the module.
- Apply RMMM Plan for the reduction of the module implementation risk.
- If no risk for the module implementation then implements.

These are the overall working of the proposed model. In the first step users request for replace or add the functionality in the software then the specified module is then send to the ARM (Additional Requirement Management). If the required module is available in the ARM then the specified module is register for implementation after that selected module is then pass to RMMM Plan for checking the risk in the module implementation. If more risk is occurring for the module implementation then track out the risk and evaluate the probability of occurrence. If risk is less then avoid it and implement the module.

Conclusion

Maintenance phase is one of the most difficult activity of the software engineering since it includes as a part of every phase. Fault could lie in anywhere in the software. The main reason of the fault of the software maybe lies in the design documents or non-existent specifications. As the maintenance activity of the software is very complicated so more effort is required to maintain the software. To reduce the effort of the enhanced maintenance factor of the maintenance phase in research paper we have introduced a new model. In this Model user request for change in the software. And additional requirements from the users are stored in the ARM (additional requirement management) Model in the form of modules. These modules are coded, tested and designed during the software development with collaboration of users and software developers. RMMM plan is used to reduce the module implementation risk. Main target of the research was to reduce the risk and effort in the enhanced maintenance.

References

[BR00] **K. H. Bennett, V. T. Rajlich -** *Software Maintenance and Evolution: a Roadmap*, ACM Press, 2000.

[Cha892] **R. N. Charette** - *Software Engineering Risk Analysis and Management*, McGraw. Hill/Intertext, 1989.

[CC00] **G. Canfora, A. Cimitile** - *Software Maintenance*, University of Sannio, Faculty of Engineering at Benevento,2000.

[fre**] http://www.freetutes.com/systemanalysis/sa2-error-distribution-with-phases.html

[Hal98] **E. M. Hall** - *Managing Risk: Methods for Software Systems Development*, Addison-Wesley, 1998.

[Hig95] **R. P. Higuera** - *Team Risk Management*, Cross Talk, U.S. Dept. of Defence, P. 2-4, January 1995.

[Lin02] **J. Lindroos** - *The Role of Program Structure in Software Maintenance*, University Of Helsinki, Feb 2005.

[Sca01] **W. Scacchi** - *Process Models in Software Engineering*, Institute for Software Research, University of California and Irvine, 2001.

[Sta03] **Prof. Stafford** - *Software Maintenance As Part of the Software Life Cycle*, Tufts University, December 2003.

[SG07] **Y. Singh, B. Goel** - *A Step Towards Software Preventive Maintenance*, Guru Gobind Singh Indraprastha University, ACM SIGSOFT, July 2007.

[Ver00] **Chris Verhoef** - *Software Evaluation*: A Taxonomy,University of Amsterdam, 2000.

[Web05] **R. Weber** - Fit *for Change: Steps towards Effective Software Maintenance*, IEEE, Sept 2005.

[WW97] **R. C. Williams, J. A. Walker** - *Putting Risk Management into Practice*, IEEE Software, pp, 75-81, May 1997.