A High Quality Software after Maintenance Depend on Effectiveness measures

Mohammad Aloran
Hashemite University
Software Engineering Department
Zarqa, 13133, Jordan
P.O.Box 33127,00962-785050797
mohammadaloran@itc.hu.edu.jo

Hazem Eid
Hashemite University
Software Engineering Department
Zarqa, 13133, Jordan
P.O.Box 33127,00962-786041112
hazemeid@itc.hu.edu.jo

Khalid T. Al-Sarayreh
Hashemite University
Software Engineering Department
Zarqa, 13133, Jordan
P.O.Box 33127,00962-798471991
khalidt@hu.edu.jo

ABSTRACT
Providing high quality software after maintenance is one of the most important topics, especially in long-term projects or an open source projects. In this paper we measure the quality of software after maintenance based on quality in use main measure effectiveness which has three measures task completion, task effectiveness and error frequency. This paper present an approach to extract developer experience where the experience of developer is a main factor that help to improving the maintenance process We analyzed these measures to the extent of their contribution in the achievement of high quality after maintenance tasks.

Categories and Subject Descriptors
Knowledge Engineering, Information and Communication for developing, Technologies as Teaching Strategy and Learning Styles.

General Terms
Measurement, Quality, Maintenance, Standardization.

Keywords
Maintenance, quality in use, effectiveness, developer experience

1. INTRODUCTION
Once the software is shipped to the user maintenance Become an important subject in the life cycle of the software, maintenance tasks can be done at any time so maintenance is a reason behind a change on requirements or for improving performance, meeting users satisfactions, fixing a bug, adding features and functionalities etc. noting that these alternations should be done immediately and should be delivered back to the user on time with high quality.

Currently, making sure that the maintenance process has achieved high quality is difficult, because the nature of the development differs from one project to another.

To insure a high quality software we analyzed the maintenance process and its types and how to define the developers expertise in certain maintenance workflows, then we tried to apply the effectiveness standards on the outputs to measure the quality of any modifications done to the software after completing the process of maintenance.

Furthermore, Long term projects and multifunctional software face problems concerning completing tasks in precise time frames, delivering required outputs to users. so receiving new change requests is one the responsibilities of the project manager who decides what type of maintenance is to be done, and who is the right developer to do such change requests.

According to IEEE standard the following concepts are defined and used: software maintenance definition "(A) 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. (B) The set of activities that takes place such that software installed for operational use continues to perform as intended and fulfill its intended role in system operation. Software maintenance includes improvements, aid to users, and related activities."[1], and according to the standard the following concepts are defined and used:" a process is activated when the software product undergoes any modification to code or associated documentation as a result of correcting an error, a problem or implementing an improvement or adaptation."[2]. The Maintenance enhancement definition "A modification to an existing software product to satisfy a new requirement" [3].

Recently, software maintenance types are defined as any maintenance task on the basis of the IEEE standards can be one of these types of software maintenance

- Adaptive maintenance: "The modification of a software product, performed after delivery, to keep a software product usable in a changed or changing environment"[3].
- Corrective maintenance :" The reactive modification of a software product performed after delivery to correct discovered problems"[3].
- Emergency maintenance :"An unscheduled modification performed to temporarily keep a system operational pending corrective maintenance"[3].
- Perfective maintenance : "The modification of a software product after delivery to detect and correct latent faults in the software product before they are manifested as failures"[3].
- Preventive maintenance :"The modification of a software product after delivery to detect and correct latent faults in the software product before they become operational faults"[3].

IPAC '15, November 23-25, 2015, Batna, Algeria
 ISBN 978-1-4503-3458-7/15/11…$15.00
DOI: http://dx.doi.org/10.1145/2816839.2816886

To ensure that maintenance process is performed in a good way as much as possible we should have an expert developer, the expert of developer can be extracted in many ways based on expert in a specific class or based on vocabulary etc. when there is a maintenance task the manager can give it to the expert maintainer whom will do it good as possible as he can.

Software quality definition "(A) The totality of features and characteristics of a software product that bear on its ability to satisfy given needs, such as conforming to specifications. (B) The degree to which software possesses a desired combination of attributes. (C) The degree to which a customer or user perceives that software meets his or her composite expectations. (D) The composite characteristics of software that determine the degree to which the software in use will meet the expectations of the customer. "[4].

Quality in use is defined as ""the capability of the software product to enable specified users to achieve specified goals" [5] . (ISO/IEC 9126-1)." quality in use is the user's view of the quality of a software, measuring the result of using the software ,the quality in use measures are used for the measurement of effectiveness, efficiency, Satisfaction and freedom from risk.

- Effectiveness measures:" assess the accuracy and completeness with which users achieve specified goals."[5]
- Effectiveness measures: according to Smith and Clark, (2004) definition: "A measure of the ability of a system to meet its specified needs (or requirements) from a particular viewpoint. This measure may be quantitative or qualitative and it allows comparable systems to be ranked. These effectiveness measures are defined in the problem-space. Implicit in the meeting of problem requirements is that threshold values must be exceeded." [6] Using this measure allows users to know if the goals behind using the software are completed and to reach needed accuracy.
- Efficiency measures: "assess the resources expended in relation to the accuracy and completeness with which users achieve goals."[5] Using this measure allows users to define if any of the functions made by the software are done in precise time.
- Satisfaction measures:" assess the degree to which user needs are satisfied when a product or system is used in a specified context of use"[5] Using this measure allows users to specify if there expected outputs are meet.
- Freedom from risk measures: "assess the degree to which a product or system mitigates the potential risk to economic status, human life, health, or the environment."[5] Using this measure allows users if any of the outputs affect the real life.

This paper mentions the definitions for each of the maintenance, maintenance types, quality in use, quality in use measures and effectiveness measures.

To scope the area of our research we will discuss the main factor for maintenance process (developer experience) and how to extract it from commit log file and we will discuss the three main measures for effectiveness and show how to apply it on the output of maintenance process, finally this paper provide procedural model to insure high quality after maintenance.

This paper is organized as following section 1 introduction contains general understanding of our paper, Section 2 related work shows other studies about this topic, section 3 developer expertise shows how other researchers extract developer expert section 4 effectiveness measures to discuss how to measure high quality, section 5 discuss the Proposed procedural Model, section 6 conclusion and future work and section 7 references.

2. RELATED WORK

There are many early studies about quality after maintenance and how to extract developer experience topics, in this suction we will mentions other researchers how discuss these two topics

2.1 Related work for quality after maintenance

Leopairote et al. [7] present an approach to recommend a methodology that measures software quality attributes and characteristic from reviews. Use ontology technique for sentence mapping onto attributes and characteristics in quality in use, composed of effectiveness, productivity, safety and satisfaction. This approach measures the quality in use depends on ISO/IEC 9126, and takes all measures to prove the methodology using reviews. Then, review scores are computed by using sentences appeared in each review. Finally, outcome of this approach is the quality in use scores of software product that are calculated by using its reviews' scores., in our approach we measure the quality of software after maintenance based on quality in use using effectiveness measures after maintenance.

Garcia et al. [8] present an approach to describe a framework for the measure and evaluation of the user experience based on the standard ISO/IEC 9126-4. The evaluation is measured based on four main aspects: effectiveness, productivity, safety and satisfaction. The aim of their paper is to prove that the evaluation of the quality in use of mobile services can be significantly sped up by the automation of its capture and analysis processes. But in our approach we measure the quality of software based on quality in use main measure effectiveness depends on the result of software after maintenance.

Welch et al. [9] present an approach to develop a model to analyze (1) the organizational and environmental factors that determine website effectiveness and (2) how website effectiveness contributes to overall service quality of the agency. Their paper discusses three website measures indicate that effectiveness is a multidimensional concept: effectiveness depends upon the referent and the analyst's perspective. Their paper measured the quality for website only. Our approach measured the quality in general and specific case (after maintenance).

2.2 Related work for extract developer experience

Hammad et al. [10] presents an approach to identify designers. They extract experience for designers by studying the type of design changes provided by designers. They distinguish between three levels of knowledge in design; the depth knowledge in design, the broad knowledge in design, and the architecture knowledge in design. We use two measures to categories the expert of developer, repeated vocabulary and unique vocabulary, and we distinguish two type of expert of developer the first one is specialized expert and the second one is specific expert.

Kagdi et al. [11] present an approach to recommend a ranked list of developers to assist in performing software changes to a particular file is presented, they use a combination of the three contribution measures to infer candidate developers who could best assist for a change in a given source code file. The contribution measures that are used are the commit contribution, the most recent activity date, and the number of active workdays. In our approach we extract developer expertise based on vocabulary.

3. DEVELOPER EXPERIENCE

Developer expertise very important in the maintenance and evolution of software, and it's helpful in transfer the knowledge between developers. The team manager can use developer expertise if he have a change request or a maintenance tasks and don't know which developer can give help to solve it.

In this paper we present an approach to extract developer expert based on the repeated use of vocabulary from analysis of the developer's commits. In this approach to extract developer expert we use the commit log file manually to extract:

1. Extract set of vocabulary from a software repository for each developer.
2. Determine repeated vocabulary.
3. Determine unique vocabulary.

Developers have varies expert on the software repository, developers use different vocabulary in their commit and use same vocabulary in different number of repetition from the developer to another, based on their activity on the software repository. These experts of developer we can identify by vocabulary's are extract from software repository we distinguish two type of expert of developer the first one is specialized expert and the second one is specific expert.

We use two measures to categories the expert of developer, repeated vocabulary which how much a developer use the same vocabulary and unique vocabulary which vocabulary used from single developer.

- **Specific experience**

  This type of experience for developers who have a good knowledge of the details of the vocabulary used by developers in their Contributions on software repositories. Determine by measure Number of repeat vocabulary committed by the developer, more unique vocabulary lead to more Specific experience for developer.

  For example, consider a developer who used high number of repeated vocabulary in his Contribution on software so he has details about that vocabulary. Our assumption that a developer with high specific expert can help in any change request or bugs requires dealing with this vocabulary.

- **Special experience**

  This type of experience for developers who have special knowledge of the vocabulary whether (class, object, method….) used by single developer in their Contributions on software repositories. Determine by measure unique vocabulary committed by a single developer, more unique vocabulary lead to more special knowledge for developer. For example, a developer used vocabulary no one else used same vocabulary in their commit, so this developer have special knowledge about this vocabulary. Our assumption that a Developer with high special expert can help in any change request or bugs requires.

There are many ways to infer the developer experience some of which is based on the maintenance type, vocabulary and the other based on design changes and design pattern used by the developer etc. Having an experienced developer plays an important role in the maintenance process where he could complete the process without any mistakes and high quality in proportion to needs of the user. Therefore the developer experience is main factor in software maintenance enhancement. In our paper we relied on the developer experience as a main factor to help improving the maintenance process.

Figure 1 shows that the first level of our approach is vocabulary from this level as mention above we find two main experience types the specific experience and special experience as a second level, the third level is developer experience based on the two levels mentioned before.

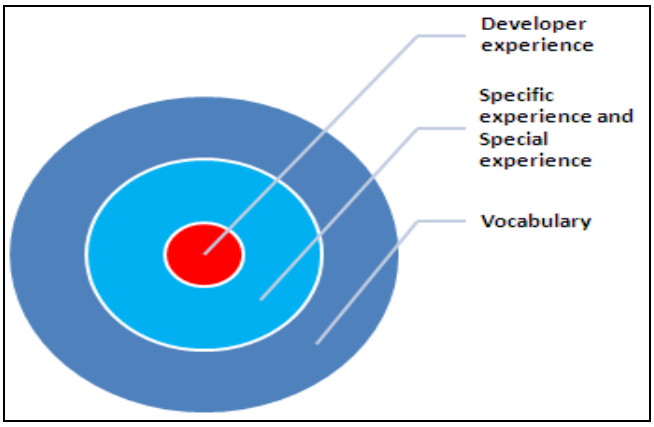

Fig. 1: Developer experience based on vocabulary

4. Effectiveness measures

In this section, we introduce three independent effectiveness measures - Task completion, Task effectiveness, and Error frequency measures - to address the effectiveness issues of a maintenance from different perspectives.

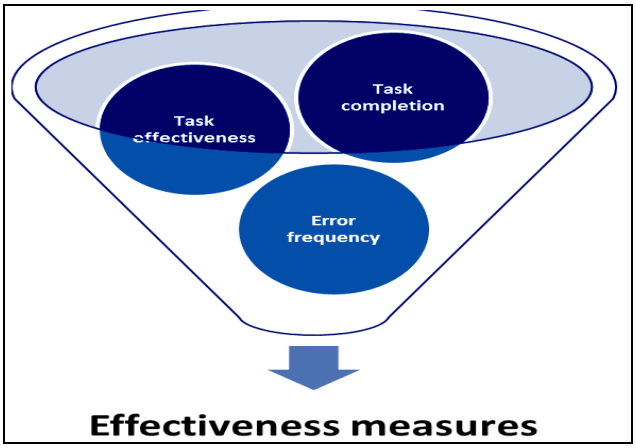

Fig. 2: Measures of effectiveness

4.1 Task completion

The Task completion measures evaluates What proportion of the tasks after maintenance are completed correctly, that means the Proportion between number of tasks completed and total number of tasks attempted. This measure helps to insure the total tasks attempted after maintenance is completed correctly.

$$X = A/B \qquad (1)$$

- Where:

A = number of tasks completed.

B = total number of tasks attempted.

4.2 Task effectiveness

Task effectiveness is another important effectiveness measure for quality of software, it evaluates What proportion of goals to the tasks achieved correctly, that means the proportion of the goals to the change is achieved correctly after maintenance this measure give a weight Ai based on the extent to which it detracts from the value of the output .

$$\{X = 1-\square Ai \mid X>0\} \quad (2)$$

- Where:

Ai= proportional value of each missing or incorrect component in the task output (maximum value = 1)

4.3 Error frequency (failure frequency)

The Error frequency evaluates what the frequency of errors is made by the user compared to a target value, that means is the frequency of errors made by the user which is introduce after maintenance compared to number of tasks attempted.

This measure helps to provide the proportion of failure frequency of product after maintenance.

$$X = A/B \quad (3)$$

- Where:

A = number of errors made by the user

B= number of tasks (or can be time)

User group/time line/volume

5. PROPOSED PROCEDURAL MODEL

In this section we suggested a procedural model showing how to apply the measures of effectiveness on outputs of the maintenance process.

This paper discussed a procedural model shows in figure 3 that explain the process of maintenance and its relation with effectiveness measures to reach a high quality software after maintenance. This procedural model works as follows: when a new change requests Became required, the first step is the responsibilities of the project manager are decides what type of maintenance based on his experience is to be done, and who is the right developer can help to fix this maintenance type. The developer experience will be extracted based on vocabulary with two types of experience specific experience and special experience. After that when the expert developer finished the maintenance task the effectiveness measures should be applied on the output of maintenance process.

From the above model after implementing the first measure of Task Completion on outputs we came with the proportion of tasks to completed task correctly for example, the expected functions of E-mails is to send new E-mail, receive, send E-mail with attachments, forward and reply so the number of task is 5, in order to achieve high quality the percentage of completing expected tasks has to be 5/5=100%.

Otherwise if one of the tasks fails like the E-mail sent without the attached file the percentage will be 4/5=8% thus lower quality and further maintenance is required not for the 5 task but just to the task who failed in this case maintenance for the task send attached file.

After implementing the second measure of Task Effectiveness on outputs. For example the E-mail is sent without attached file, thus lower quality and further maintenance is required not for the 5 task but just to the task who failed in this case maintenance for the task send attached file.

After implementing the third measure of error frequency on outputs we came with the proportion of errors made by user for example, the expected functions of E-mails is to send new E-mail, receive, send E-mail with attachments, forward and reply so the number of task is 5 so in order to achieve high quality the percentage of errors has to be 0/5=0%.

Otherwise if one of the tasks fails the percentage will 1/5=2% thus lower quality, and further maintenance is required for the task who had an error.

After applied effectiveness measures on the output of maintenance process it will be a percentage that show the quality of the software after maintenance, if the percentage is lower further maintenance process is required for the task who had an error, but if the percentage is high then we will have software after maintenance with high quality.

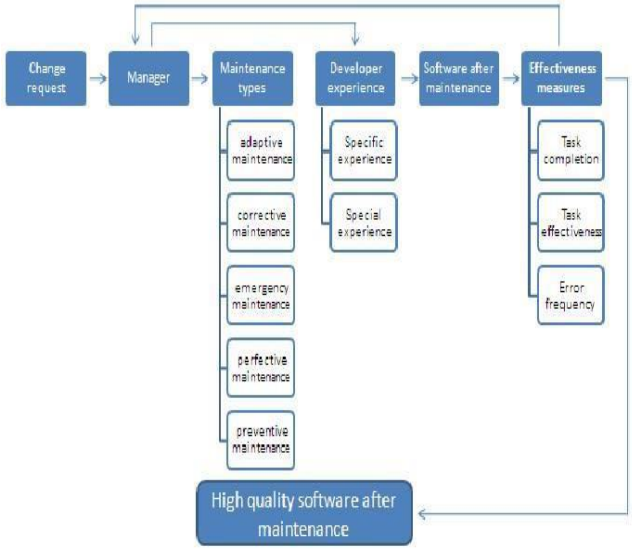

Fig. 3: The procedural model

6. Conclusion and future work

Quality after maintenance is one of the most important topic, as the software products getting more and more complex, maintain these products is becoming difficult especially in long-term projects or an open source projects.

in this paper we present an procedural model that showing how to apply effectiveness measures on the outputs of the maintenance process and discuss maintenance types and show how the maintenance types effect of maintenance process , and we present an approach to extract developer expert based on the repeated use of vocabulary from analysis of the developer's commits.

For future work, this paper could be enhanced by adding some measures that actually measure the quality after maintenance we will add Efficiency measures, Satisfaction measures and Freedom from risk measures. We will show the performance of the proposed model by using for example simulation, test, modification… etc.

We will improve a new approach for extracting developer expert. We will improve the proposed procedural model to be a generic model for measuring a high quality after maintenance.

7. Acknowledgment

Our thanks to IPAC 2015 conference team for their support and help our software engineering research groups at Hashemite University in Jordan.

8. REFERENCES

[1] IEEE STANDARDS COORDINATING COMMITTEE, et al. IEEE Standard Glossary of Software Engineering Terminology (IEEE Std 610.12-1990). Los Alamitos. CA: IEEE Computer Society, 1990

[2] ECSS, CONTACT. European Cooperation for Space Standardization. ECSS-Q-40A, ESA Publication, 2002.ISO/IEC-24570 (2005).

[3] STANDARD, I. Software engineering–software life cycle processes–maintenance. ISO Standard, 2006, 14764: 2006.

[4] IEEE Std 1633™-2008, "IEEE Recommended Practice on Software Reliability", Standards Committee of the IEEE Reliability Society 2008, Approved 27 March

[5] ISO/IEC-9126, "Software Engineering - Product Quality - Part 1: Quality Model ", International Organization for Standardization, Geneva (Switzerland), 2004.

[6] Chen, Yen-Fu, et al. "A systematic review of the effectiveness of adalimumab, etanercept and infliximab for the treatment of rheumatoid arthritis in adults and an economic evaluation of their cost-effectiveness." Health technology assessment 10.42 (2006): 1-266.

[7] Leopairote, Warit, Athasit Surarerks, and Nakornthip Prompoon. "Software quality in use characteristic mining from customer reviews." Digital Information and Communication Technology and it's Applications (DICTAP), 2012 Second International Conference on. IEEE, 2012.

[8] García, Iván Pretel, and Ana Belén Lago Vilarino. "Quality in use of service evaluation framework for mobile services." Information Systems and Technologies (CISTI), 2010 5th Iberian Conference on. IEEE, 2010.

[9] Welch, Eric W., and Sanjay Pandey. "Multiple measures of website effectiveness and their association with service quality in health and human service agencies." System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on. IEEE, 2007.

[10] Hammad, Maen, Mustafa Hammad, and Hani Bani-Salameh. "Identifying Designers and their Design Knowledge." International Journal of Software Engineering and Its Applications 7.6 (2013): 277-288.

[11] Kagdi, Huzefa, Maen Hammad, and Jonathan Maletic. "Who can help me with this source code change?." Software Maintenance, 2008. ICSM 2008. IEEE International Conference on. IEEE, 2008.