IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

Issues and Challenges in Software Quality Assurance

Himangi[1], Surender Singh[2]

Dept of Computer Science and Engineering, Om Institute of Technology & Management, Hisar, India[1, 2]

Abstract: Small scale software product can be developed with the contribution of a single participant but development at large scale requires the multiple contributors for a single project. As the number of participant team size increases, ratio of ownership over the developed code for each person also increases with the probability of bugs over specific LOC. Researchers have developed various Metric tools for different development processes and in this survey paper, various solutions related to software quality are discussed.

Keywords: Software Quality, Metric, Program Size, SLOC.

I. INTRODUCTION

Developer Risk and Software Quality

Software Quality may suffer from the efforts, skill set and experience of development team and developer may perform the number of commits and with each commit, he can also introduce the bugs also thus results in the risk of producing the software with degraded quality. Ii is necessary to measure the impact of individual developer's contribution. Selecting an incompatible team for highly complex projects increases the risk of producing a product with large scale of bugs and it will also increase the cost of maintainability[12][13].

Ownership

More than one developer can participate in development process and in it is important to identify the exact code contribution ratio for each developer, in order to predict the bug ratio over per Line of Code.

It can be defined as measurement of the contribution made by a developer, in order to develop a software module. It can be calculated on the basis of number of commits performed by a developer.

$$\text{Owenership P}_{j}(d_i) = \frac{\text{Commits j } (d_i)}{\sum_{i=1}^{N} \text{Commits j } (d_i)}$$

Where d_i denotes the developer and Commits J is the number of commits performed by developer.

Developer's quality

It can be represented in terms of the quality of code being produced by a specific developer and number of bugs in a given source code directly depends upon the developer's experience, skill set and exposure to the technology being used for development.

Quality can be measure in terms of the ratio of number of bugs introduced by developer over the number of commits performed by same developer.

$$\text{Quality}(d_i) = 1 - \frac{\text{introduced j } (d_i)}{\text{Commits (di)}}$$

Issues related to developer's quality [11]

- Work Environment
- Organization structure
- Organization Culture
- Developer Skill Set
- Exposure to Technology
- Decision Making
- Developer's Contribution

II. LITERATURE SURVEY

Soliman et al. [1] developed automated tool to compute the six CK metrics by gathering the required information from class diagrams, activity diagrams, and sequence diagrams. In addition, extra information is collected from system designer, such as the relation between methods and their corresponding activity diagrams and which attributes they use. The proposed automated tool operates on XMI standard file format to provide independence from a specific UML tool. To evaluate the applicability and quality of this tool, it has been applied to two examples: an online registration system and one of the bioinformatics Microarray tools (MIDAS).

M. Greiler et al. [2] did a survey and explored the relationship of code ownership and software quality using Microsoft's software. Analysis shows that the impact of code quality over the quality of end product and this relationship can be used to develop a metric which can be used to reduce the number of bugs. Researchers can utilize this analytical data for their production teams.

J. H. Hayes et al. [3] investigated the impact of testing over the quality under the constraints of bugs in a given code w.r.t. ownership using a method based on supervised machine learning which can build a decision tree to

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 8, August 2016

identify the bug's frequency in code. Correlation shows the dependent relationship between testing and quality whereas Regression shows the relationship between measurements and requirements. Analytical and statistical results that measurements can be enhanced using trained learning sets.

F. A. Fontana et al. [4] explored the quality of software architecture using a metric which can be used to find out code Smells which can cause defilement of software architecture. Presence of code smells and their relationship defines the impact of degradation. Proposed scheme can be extended to measure its influence over quality.

S. Eldh et al. [5] explored the code ownership claimed by more than one participants, called contributors. In software development process, different people are engaged for various activities i.e. analysis, Coding, Testing, deployment and end user support. Efforts made by each participant can be defined as the total contribution made by individual person. It is quite complex to identify the exact contribution made by each member thus raises the requirement of code ownership metric which can be used to identify the various factors related to the claim of ownership and its impact over the overall development process.

C. Farag et al. [6] explored the issues related to code ownership, maintainability, code versioning, aging effects over the software product and proposed a metric to analyze these factors. Analysis show that changes made at ay stage may affect the code ownership as well as it can also introduce the bugs. If ownership cannot be defined then it is complex to identify the correlation values also. Frequency of code updates also affect the ownership ratio.

Z. Bukhari et al. [7] investigated the requirements of Metric tools for various purpose i.e. development, quality measurement etc. and identified the different ways of Metric selection on the basis of their limitations and scope. Analysis shows that perfect Metric tool can be selected on the various factors i.e. evaluation cost, measurement accuracy, dependency factors and Input/Output type etc. Current research work can be extended to minimize the human feedback related to Metric selection.

Anjana Gosain et al. [8] explored the different types of dynamic software metrics for object oriented approaches under the constraints of validation and software quality. They sub divided the metrics into different categories i.e. size which can be defined as the size of the code on disk as well as memory size occupancy by same code at run time, complexity which can be defined by frequency, cohesion and coupling can describe the inside and outside bounding/dependency of existing modules over each other at run time, inheritance describes the frequency of interaction between parent and child classes and polymorphism metric can describe behavior index as ratio of polymorphic and non polymorphic dispatches to be executed. They also represented the Software quality attributes, metric types and their validation graphically.

Chandan Kumar et al. [9] represented a error estimation scheme for software based on BBN metrics. Authors focused on the early development stages at which it is quite hard to use metrics and considered the reliability and uncertainty in these phases. Analysis results show accuracy of proposed method in terms of error detection at early stages.

Tao Yue et al. [10] developed a framework to produce quality metrics for MOF meta models to measure the quality of models. Comparison of newly produced metrics with manually defined quality metrics using UML classes and sequence diagrams shows that it automatic produced metrics are more efficient and can ensure the software quality in more accurate way. Proposed work can be extended to develop metric for MOF-based languages.

III. PROBLEM FORMULATION

Quality of a Software product is an important factor that is need to be measured accurately on the basis of its attributes but software exists only in virtual form so it is quite hard to predict the quality of a specific software. Many researchers have already developed various quality metrics to measure the quality of different attributes. As per survey, it can be observed that there is need to extend the existing research work as well as there is need to recognize a perfect tool to measure software quality.

Following are some metric tools which can be used to define the metrics related to Program Size, Ownership and Developer's quality etc.:

- Eclipse Metrics
- Analyst4j
- CCCC
- Dependency Finder
- Heap Analysis Tool
- Agile Metrics

IV. CONCLUSION

In this paper, various tools and techniques were explored which can ensure the quality of the software. Software Metrics can be used to identify the risk assessment w.r.t. developer's experience, coupling and cohesion impact over quality, dynamic metrics can be used for software complexity, relationship between development activities and their impact over quality can be identified, statistic analysis can be used to exploit the coding standard violations, service-oriented mining is used ensure the quality metric for web enabled applications, fault prediction capabilities can predict the software behavior in advance, six CK metrics can acquire the information from various classes, diagrams and can define the relationship among them etc. Researchers have also developed some

metrics for fault finding, ensuring process accuracy for object oriented software.

REFERENCES

[1] SOLIMAN, AHMED, S.H, "UTILIZING CK METRICS SUITE TO UML MODELS: A CASE STUDY OF MICROARRAY MIDAS SOFTWAR", INFOS, IEEE-2010, PP.1-6
[2] M. Greiler, Kim Herzig, Jacek Czerwonka, "Code Ownership and Software Quality: A Replication Study", IEEE/ACM 12th Working Conference on Mining Software Repositories, 2015 , pp.2-12
[3] J. H. Hayes, Wenbin Li, Tingting Yu, Xue Han, Mark Hays, Clinton Woodson, "Measuring Requirement Quality to Predict Testability", AIRE, IEEE, 2015, pp.1-8
[4] F. A. Fontana, Vincenzo Ferme, Marco Zanoni, "Towards assessing software architecture quality by exploiting code smell relations", IEEE/ACM 2nd International Workshop on Software Architecture and Metrics, pp.1-7
[5] S. Eldh, Brendan Murphy, "Code Ownership Perspectives", IEEE Journals & Magazines, 2015, Vol.32 (6), pp.18-19
[6] C. Farag, Péter Hegeď us, Gergely Ladányi, and Rudolf Ferenc, "Impact of Version History Metrics on Maintainability", 8th International Conference on Advanced Software Engineering & Its Applications, IEEE, pp.30-35
[7] Z. Bukhari, Jamaiah Yahaya, Aziz Deraman, "Software Metric Selection Methods: A Review", 5th International Conference on Electrical Engineering and Informatics, IEEE-2015, pp. 433 – 438
[8] ANJANA GOSAIN, GANGA SHARMA, "DYNAMIC SOFTWARE METRICS FOR OBJECT ORIENTED SOFTWARE: A REVIEW", INFORMATION SYSTEMS DESIGN AND INTELLIGENT APPLICATIONS, ADVANCES IN INTELLIGENT SYSTEMS AND COMPUTING, SPRINGER-2015, PP.579-589
[9] CHANDAN KUMAR, DILIP KUMAR YADAV, "SOFTWARE DEFECTS ESTIMATION USING METRICS OF EARLY PHASES OF SOFTWARE DEVELOPMENT LIFE CYCLE", SPRINGER-2014, PP. 1-9
[10] TAO YUE, SHAUKAT ALI, "A MOF-BASED FRAMEWORK FOR DEFINING METRICS TO MEASURE THE QUALITY OF MODELS", ECMFA, SPRINGER-2014, PP. 213-229
[11] Yangsong Wu ; State Key Lab. for Novel Software Technol., Nanjing Univ., Nanjing, China ; Yibiao Yang ; Yangyang Zhao ; Hongmin Lu, "The Influence of Developer Quality on Software Fault-Proneness Prediction", SERE, IEEE-2015, pp.11-19
[12] Lee, Shou-Yu ; Li, Yihao, "DRS: A Developer Risk Metric for Better Predicting Software Fault-Proneness", TSA, IEEE-2015, pp.120-127
[13] GREILER, M. ; MICROSOFT CORP., REDMOND, WA, USA ; HERZIG, K. ; CZERWONKA, J.," CODE OWNERSHIP AND SOFTWARE QUALITY: A REPLICATION STUDY", MINING SOFTWARE REPOSITORIES (MSR), 2015 IEEE/ACM, PP. 2 - 12