Improved Software Quality Using Software Bug Triaging and Prioritizing

Vijay Kumar Burugari
Associate Professor
Department of Computer Science and Engineering
K L University, Vijayawada, Andhra Pradesh, India

M Sai Koushik
Department of Computer Science and Engineering
K L University, Vijayawada, Andhra Pradesh, India

K Gowtham Sai
Department of Computer Science and Engineering
K L University, Vijayawada, Andhra Pradesh, India

N Jagadeesh
Department of Computer Science and Engineering
K L University, Vijayawada, Andhra Pradesh, India

Abstract- Bugs are most tedious software problem that faces by many of the software organizations. Many companies firstly release beta version. After checking of many bugs in the software and solve that and releases the new software. In this paper, the specialized bug Triaging software (SBTS) which checks the issues obtained in the system. The synthetic dataset is used which is taken from various organizations for testing. The proposed system works very accurately and improves the performance of bug Triaging. Also increases the quality of the software.

***Index Terms*—bug report, bug-report, bug localization, bug fixing.**

I. INTRODUCTION

As software engineers can scarcely compose programs with no bugs, it is inescapable to discover bugs and perform bug fixing in programming advancement. In addition, it is expensive and tedious to discover and fix bugs in programming advancement. Programming testing and troubleshooting is assessed to devour more than 33% of the all out expense of programming advancement [1, 2].

To ensure the nature of programming productively, numerous activities use the reports regarding the bugs in order to collect and record the details about the bugs by the designers, analysers, and also end-users. In reality, with the quick advancement of many open source software, many bug reporting software's are used to measure it. For instance, open-source variant of Eclipse1 reported 29 reports presented every day [3]. Albeit an enormous bug reports allow us to improve the nature of programming, it is likewise bug reports are required to be tested for examining. To overcome this functional issue, numerous specialists proposed different methods to encourage bug-report2 investigation.

II LITERATURE SURVEY

This section explains about the various existing systems that perform operations on software bugs. The smart framework [4] that intuitively directs the bug task. The vector model is utilized to recover the title and the portrayal from the report to assemble a vector which later can be utilized to discover comparative reports by mining the information in the bug vault.

Xia, et. al introduced a bug forecast calculation [5], the reason for which is to anticipate the quantity of bugs which are to be distinguished and announced in every month. The monthly forecast of bug basically relies on the bug tally of the forerunner month. This expectation is accomplished through the forecast algorithm actualized in the separate paper [5].

The various exception handling systems that will find the connection between bug types and bug fixing arrangements are explained [6]. The exception handling methodologies are discussed with various scenarios and find the accurate bugs within the system. If the clients find the bug, then the developers solve that bug.

An incremental learning called TopicMiner which analyses the distribution of topic in order to allocate a fixer which is appropriate to it based on the affinity towards the topics of a new bug [10].

A graph based on Markov chains, that identifies the bug tossing history to better report about bugs [11]. A bug triage by using text categorization and machine learning techniques in order to predict the developer based on the description of bugs is discussed. [12]

The authors introduced a model that naturally demonstrates whether a showing up bug report is unique or copy of a previously existing report [7]. It spares the designer's time and endeavours. To foresee bug duplication, framework they have utilized simple strategies, for example, literary semantics, grouping in charts and surface highlights.

From the past many years the bug-report examination centres on how to utilize bug reports proficiently. The different data as the bug report, including its partners, depiction, etc., some work centres around the way it enhances bug reports [4–6] by tending to the issues in the already available bug reports, e.g., misclassifying another component as a bug and allocate seriousness levels to bug reports. To gather and relegate bug reports consequently, few work centres around the way to mechanize bug-report triage [3,7] by recognizing copy bug provides details regarding a similar bug, foreseeing the need of it, and allocating with high precision bug reports. Moreover, to assist designers with fixing the bugs revealed by bug reports, few work centres around fixing the bugs by considering the bug reports.

The various studies that explains about bugs in machine learning (ML). There are many open source ML's such as Apache Mahout, Lucene, and OpenNLP is discussed [8].

A model developed an Android Applications which reports the bug and fixes it [9].

Kanwal developed a recommender which prioritises the bug reports automatically by using the Naïve Bayes and Support Vector Machine (SVM) classifiers [13,14].

Chalapathi Rao implemented a Software Bug Complexity Cluster (SBCC) using Self Organizing Maps and it improved the performance and efficiency in terms of proximity error and fitness [15].

III. PROPOSED ALGORITHM

The framework proposed a computerized water system utilizing Twilio. This framework contains MCP3208, soil dampness, temperature sensor, and raspberry pi. In this system, it can be operated the water motor through the web page without human intervention. By using Twilio send an alert message to the farmer regarding the water content in the field. Based upon the alert message farmer take an action on the field to control the water system for the crops.

Triaging of bugs and allocating a programmer to fix them is a difficult and very time taking. Programmers are commonly master in some specific zone. For instance barely any programmers are master in GUI while some are in unadulterated java usefulness and so forth; in this way doling out a particular bug to permissible engineer could spare time. It would likewise assist with keeping up the intrigue level of the engineers by allotting bugs as per their

advantage. It's anything but a simple errand to dole out right bug to the correct designer for tri-ager without knowing the genuine class a bug has a place with. A strategy for arrangement of open source programming bugs utilizing the rundown and depiction of the bugs gave by the bug journalists is proposed right now.

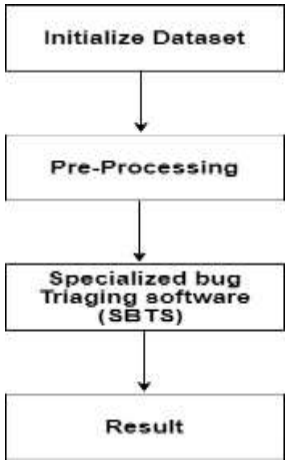

Figure: 1 System Architecture

The objective of the proposed work is as follows:

1. the priority should be given based on new bug report.
2. to predict the list of developers the ranking should be assign.
3. Immediate selection should be removed.
4. from the un-formatted words the selection of keyword should be removed.

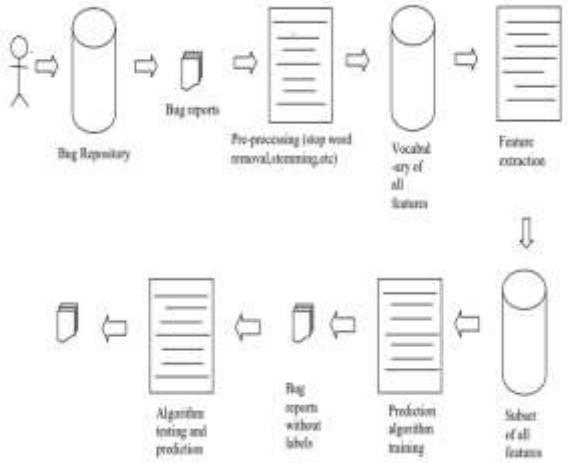

Figure: 2 Flow Diagram

IV. Experiment and Result

The implementation is done using JAVA and jdk1.8 which will give the better results for bug triaging. The datasets consist of 30 attributes and 10k records are available. Some of the attributes are Bug ID, Product, Component, Assignee, Status.

Table: 1 Performance Measure

Approach Used	Accuracy
Existing System	91.2 %
SBTS	97.6%

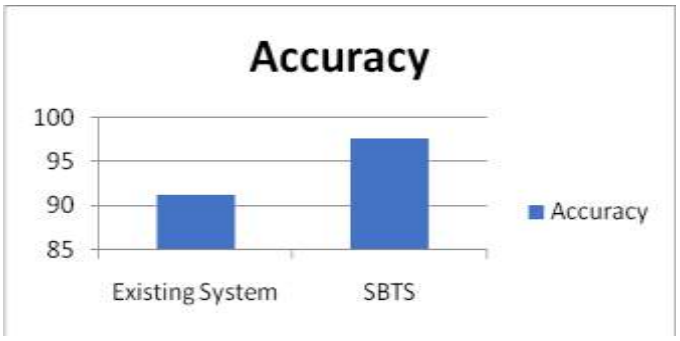

Figure: 3 Performance of proposed system

IV.CONCLUSION

The proposed system SBTS is used to find the bugs in the given dataset which is Bug triaging that contains details about bug prioritization or assign, defect analysis. SBTS check all the data and produced the accurate results compare with existing system. It also improves the software quality to overcome issues at runtime. The technique processes the high-quality bug data in software development and maintenance phase.

REFERENCES

[1] G. Bortis, A. van der Hoek, "Porchlight: A tag-based approach to bug triaging", Proc. 35th Int. CSE., pp. 342-351, 2013.
[2] W. M. Ibrahim, N. Bettenburg, "On the relationship between comment update practices and software bugs", J. Syst. Software, vol. 85, no. 10, pp. 2293-2304, 2012.
[3] M. Linares-Vasquez, K. Hossen, H. Dang, "Triaging incoming change requests: Bug or commit history or code authorship?", 28th IEEE Int. Conf. SM., pp. 451-460, 2012.
[4] H. Naguib, N. Narayan, B. Brugge, D. Helal, "Bug report assignee recommendation using activity profiles", 10th IEEE Work. Conf. MSR, pp. 22-30, 2013.
[5] X. Xia, D. Lo, M. Wen, E. Shihab, and B. Zhou, “An empirical study of bug report field reassignment,” WeekIEEE CSMRRE, 2014, pp. 174–183.
[6] X. Xie, W. Zhang, Y. Yang, and Q. Wang, “Dretom: Developer recommendation based on topic models for bug resolution,” in Proc.

8th, PMSE., 2012, pp. 19–28.
[7] G. Yang, T. Zhang, and B. Lee, “Towards semi-automatic bug triage and severity prediction based on topic model and multi-feature of bug reports,” in Proc.IEEE 38th ACSAC, 2014, pp. 97–106.
[8] F. Thung, S. Wang, D. Lo, and L. Jiang. An empirical study of bugs in machine learning systems. In ISSRE, pages 271–280, 2012.
[9] P. Bhattacharya, L. Ulanova, I. Neamtiu, and S. C. Koduru. An empirical analysis of bug reports and bug fixing in open source android apps. In CSMR, pages 133–143, 2013.
[10] X. Xia, D. Lo, Y. Ding, J. M. Al-Kofahi, T. N. Nguyen and X. Wang, "Improving Automated Bug Triaging with Specialized Topic Model," in IEEE Transactions on Software Engineering, vol. 43, no. 3, pp. 272-297, 1 March 2017.
[11] G. Jeong, S. Kim, T. Zimmermann, "Improving bug triage with bug tossing graphs", Proc. Joint Meeting Eur. Softw. Eng. Conf. ACM SIGSOFT Int. Symp. Found. Softw. Eng., pp. 111-120, 2009.
[12] D. Cubranic and G. C. Murphy. “Automatic bug triage using text categorization” In SEKE 2004: Proceedings of the Sixteenth International Conference on Software Engineering & Knowledge Engineering, pages 92--97, 2004.
[13] Kanwal, J., Maqbool, O. “Bug Prioritization to Facilitate Bug Report Triage”, Journal of Computer Science and Technology, 27,397–412, 2012.
[14] Kanwal J, Maqbool O. Managing open bug repositories through bug report prioritization using SVMs. In Proc. International Conference on Open-Source Systems and Technologies, Lahore, Pakistan, December 2010.
[15] M Chalapathi Rao, P Suryanarayana Babu, “Assessment of Software Bug Complexity and Severity using Evolutionary SOM Scheme”, International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249 – 8958, Volume-8 Issue-6, August 2019.