International Journal of Engineering Research and General Science Volume 3, Issue 6, November-December, 2015
ISSN 2091-2730

Effective Bug Triage and Recommendation System

Manisha Bedmutha [1], Megha Sawant [2], Sushmitha Ghan[3]
Department of Computer Engineering,
P.E.S. Modern College of Engineering,
Pune University (MH), India.
[1]m.bedmutha13@gmail.com
[2]meghasawant93@gmail.com
[3]sushghan11@gmail.com

Abstract— Dealing with the bugs is an important thing in software industry. For fixing these bugs lots of time is required and each developer have to go through bug report which aims at wasting the time and ultimately the money. To solve this problem an important step which we can take is bug triage, which aims as correctly assign a developers to new bug. As well as to decrease time cost in manual work of developer text classification technique is applied to conduct the automatic bug triage. Ultimatum is to reduce scale and improve the quality of bug data. After donning with triage process, we are recommending bugs to the respective developers depending on their profile and previous work history which will help the system to predict which bug to assign to which developer. If any particular developer failed to fix the bug in the given time the triage method will be used to predict the new developer.

Keywords— Mining Software Repositories, Data Management in bug Repositories, Bug Data Reduction, Feature Selection, Instance Selection, Bug Triage, Bug Report ,Prediction for assigning bugs , Frequent item selection, Recommendation of bugs.

1. INTRODUCTION

Bugs are the programming errors that cause significant performance degradation. Bugs lead to poor user experience and low system throughput. Large open source software development projects such as Mozilla and Eclipse receive many bug reports. They usually use a bug tracking system where users can report their problems which occurred in their respective projects. Each incoming bug report needs to be triaged. For example, as of October 2012, the Mozilla project had received over 80,000 reports, averaging 300 new bug reports each day [2].

In appropriate developers may delay the bug resolution time as In manual bug triage in Eclipse, 44 percent of bugs are assigned by mistake while the time cost between opening one bug and its first triaging is 19.3 days on average [1], [16].

Selecting the most appropriate developer to fix a new bug report is one of the most important stages in the bug triaging process and it has a significant effect in decreasing the time taken for the bug fixing process [16] and the cost of the projects [2], [11]. Software companies spend over 45 percent of cost in fixing bugs [1], [8], and [17]. In traditional bug triage systems, a developer who is dominant in all parts of the project as well as the activities plays the role of bug triager in the project. The triager reads a new bug report, makes a decision about the bug, and then selects the most appropriate developer who can resolve the bug. Fixing bug reports through the traditional bug triage system is very time consuming and also imposes additional cost on the project [2].

One of the important reasons why bug triaging is such a lengthy process is the difficulty in selection of the most competent developer for the bug kind. The bug triager, the person who assigns the bug to a developer, must be aware of the activities (or interest areas) of all the developers in the project. Bug triaging normally takes 8 weeks to resolve a bug if the developer, to whom the bug report is assigned, could not resolve it, it is assigned to another developer. This would consume both time and money. Thus, it is really important on part of bug triager to assign the bug report to a developer who could successfully fix the bug without need of any tossing. Hence, the job of bug triager is really crucial [2].

In this paper, we present an approach for automatically recommending the bugs to the respective developers. The goal is to reduce the bug report and analyze it using bug instance selection and Feature Selection. To use the term frequency mining technique to predict the most suitable developer for new bug report depending upon the information available from historical bug reports and the profile of the developer. The term extracted from previously fixed bug reports and area in which developer is expertise are used for updation of profiles of developers. The frequency of each term corresponding to each topic indicates the expertise factor of each developer relating that particular topic. If the frequency of solving a bug based on particular topic is more, then system should

recommend the bugs related to that topic first to the developer.

2. RELATED WORK

Xuan et al. [1] combined feature selection with instance selection to reduce the scale of bug data sets as well as improve the data quality. To determine the order of applying instance selection and feature selection for a new bug data set, they have extracted attributes of each bug data set and train a predictive model based on historical data sets. They empirically investigate the data reduction for bug triage in bug repositories of two large open source projects, namely Eclipse and Mozilla.

Matter et al. [3] proposed an information retrieval based technique that computes the expertise of developers based on vocabulary used in source code. They extracted the vocabulary of developers from the version control repository of the project. Their approach finds a relationship between the vocabulary extracted from new bug reports and the vocabulary extracted from source code and then uses the relationship between them for recommendation purpose.

Hu el al. [4] they have proposed Bug Fixer, an automated bug report assignment method that utilizes historical bug fix data. Bug Fixer adopts a new method for computing bug report similarity. Bug Fixer also constructs a novel network structure, called Developer-Component-Bug (DCB), to model the relationship between the developers and the source code components they worked on, as well as the relationship between the components and the associated bugs. For a new bug, Bug Fixer calculates its similarity to existing bugs, and recommends developers based on the structure of DCB network.

Hosseini et al. [10] proposed an auction based bug allocation mechanism. When a new bug report comes, the bug triager extracts some basic information such as type, severity, etc. and then auctions off the bug. The developers place their bids on the bugs. The bug triager receives the bids and based on the developer's experience, expertise, historical preferences and current work queue, he announces the final decision and assigns the bug report to the winner developer.

Park el al. [14] they proposed a new bug triaging technique, COSTRIAGE, by (1) treating the bug triage problem as a recommendation problem optimizing both accuracy and cost and (2) adopting CBCF combining two recommender systems. A key challenge to both techniques is the extreme sparseness of the past bug fix data. They addressed the challenge by using a topic model to reduce the sparseness and enhanced the quality of CBCF.

Bhattacharya et al. [15] employed a fine grained incremental learning approach. They constructed multi feature tossing graphs and used it for predicting the relevant developers for the bug report. Their results showed high prediction accuracy for recommending developers and high reduction in tossing path lengths.

3. PROPOSED ARCHITECTURE

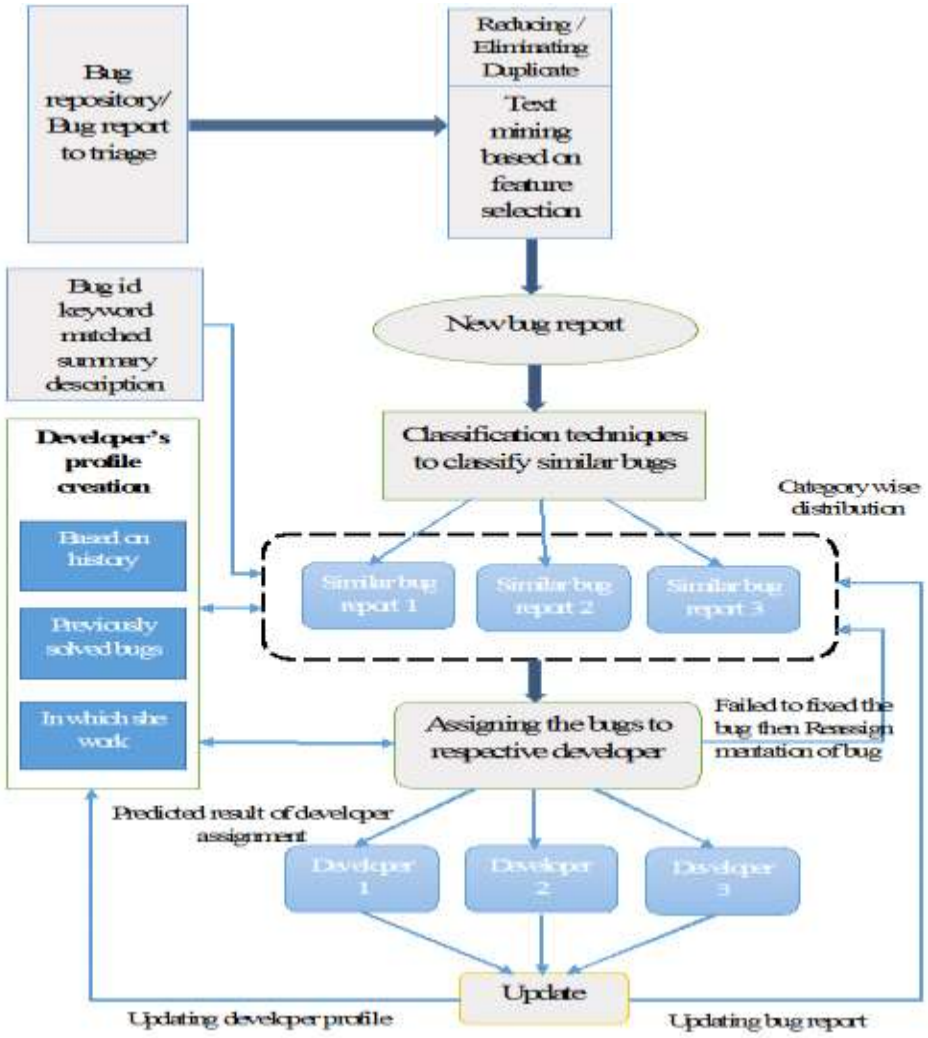

Figure 3.1: Architecture diagram of proposed system

3.1 Bug Triage

Bug Triage refers to assuring quality of bug reports. Bugs are inevitable in software development. The situation arise when we have certain number of bugs to be fixed and don't know their priority so, some standards are used in company that when certain number of bugs reached and not fixed then there should be meeting held and decided on the priorities of the bugs, so all the team members will gather and decided about each and every bug and finalize the priorities.

In daily testing and maintenance, bug reports are accumulated and stored in projects' bug tracking system. In our proposed system bug tracking system provides a platform for tracking the status of bug reports. Once a bug report is created, through the bug tracking system, developers can also collaborate on reproducing and fixing bugs. A bug report should be assigned to an appropriate developer before the process of bug fixing. The step of assigning bug reports to developers is called bug triage.

3.2 Bug Reduction

Here we are reducing the bug data by using instance selection and feature selection which will give low scale data and the qualitative data. Instance selection associated with data mining tasks such as classification and clustering. Feature selection is to choose a subset of input variables by eliminating features with little or no predictive information.

3.3 Profile Updation

The reduced bug report will be automatically assigned to the developer based on his previous work history and experience. After the bugs get solved by the developer the developer's profile will get update.

3.4 Bug Recommendation

If the frequency of solving a bug based on particular topic is more, then system will recommend the bugs related to that topic first to the developer.

3.5 Reassignment of bugs

If a particular developer failed to solve the bug in the given time then that bug will be reassign to the other developer.

4. CONCLUSION

In this paper we have focused on reading the bug report which will remove the redundant and noisy data. The system will automatically recommend the bugs to the developer which will save the time and cost in finding the bugs for fixing. As the system is time based if particular developer failed to solve the bug in given time the bug will be reassign to the other developer, so ultimately the project will get complete prior to the dead line. Also it will be easy to find out which developer is expertise in which area.

REFERENCES:

[1] Jifeng Xuan, He Jiang, Yan Hu, Zhilei Ren, Weiqin Zou, Zhongxuan Luo, and Xindong Wu, "Towards Effective Bug Triage with Software Data Reduction Techniques," IEEE Transactions, Volume 27, NO. 1, JANUARY 2015.

[2] Anjali, Sandeep Kumar Singh, "Bug Triaging: Profile Oriented Developer Recommendation", International Journal of Innovative Research in Advanced Engineering (IJIRAE) ISSN: 2349-2163, Volume 2, Issue 1, January 2015.

[3] D. Matter, A. Kuhn, and O. Nierstrasz, "Assigning bug reports using a vocabulary- based expertise model of developers," in Mining Software Repositories, 2009. MSR '09. 6th IEEE International Working Conference on, May2009, pp. 131 –140.

[4] Hao Hu, Hongyu Zhang, Jifeng Xuan, Weigang Sun, "Bug Triage based on Historical Bug-Fix Information". ISSRE - The 25th IEEE International Symposium on Software Reliability Engineering, 2014, Naples, Italy, Nov 2014.

[5] C. Sun, D. Lo, S. C. Khoo, and J. Jiang, "Towards more accurate retrieval of duplicate bug reports," in Proc. 26th IEEE/ACM Int. Conf. Automated Softw. Eng., pp. 253–262, 2011.

[6] ZHANG Jie, WANG XiaoYin , HAO Dan1, XIE Bing, ZHANG Lu and MEI Hong, "A survey on bug-report analysis", Science China, Information Sciences, Vol. 58 021101:2, February 2015.

[7] Henrique Rocha, Guilherme de Oliveira, Humberto Marques-Neto, and Marco Tulio Valente, "NextBug: a Bugzilla extension for recommending similar bugs", Rocha et al. Journal of Software Engineering Research and Development , 2015.

[8] Pankaj Gakare, Yogita Dhole,Sara Anjum, "Bug Triage with Bug Data Reduction", International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056,Volume 02 Issue 04, July 2015.

[9] Mamdouh Alenezi, Kenneth Magel, and Shadi Banitaan, "Efficient Bug Triaging Using Text Mining", ACADEMY PUBLISHER, 2013.

[10] H. Hadi, N. Raymond , and G. Michael, " A market based bug allocation mechanism using predictive bug lifetimes", in 16th European Conference on Software maintenance and Reengineering, pp. 149-15, 2012.

[11] J. Anvik and G. C. Murphy, "Reducing the effort of bug report triage: Recommenders for development-oriented decisions," ACM Trans. Softw. Eng. Methodol., vol. 20, no. 3, pp. 10:1–10:35, Aug. 2011.

[12] Philip J. Guo, Thomas Zimmermann, Nachiappan Nagappan, and Brendan Murphy, " "Not My Bug!" and Other Reasons for Software Bug Report Reassignments", ACM, 2011.

[13] J. Anvik and G. C. Murphy, "Reducing the effort of bug report triage: Recommenders for development-oriented decisions," ACM Trans. Soft. Eng. Methodology., vol. 20, no. 3, article 10, Aug. 2011.

[14] Jin-woo Park, Mu-Woong Lee, Jinhan Kim, Seung-won, and Hwang Sunghun Kim, "COSTRIAGE: A Cost-Aware Triage Algorithm for Bug Reporting Systems", in Twenty-Fifth AAAI Conference Artificial Intelligence, year 2011.

[15] Bhattacharya P, Neamtiu I., "Fine-grained incremental learning and multi-feature tossing graphs to improve bug triaging", In: Proceedings of the IEEE International Conference on Software Maintenance, Timisoara, 2010.

[16] G. Jeong S. Kim, and T. Zimmermann, "Improving bug triage with bug tossing graphs", in Proceedings of Seventh joint meeting of European Software Engineering Conference & ACM SIGSOFT symposium on Foundations of software engineering, ser. ESEC/FSE '09. New York, NY, USA: ACM, pp. 111–120, 2009.

[17] R. S. Pressman, "Software Engineering: A Practitioner's Approach", 7th ed. New York, NY, USA: McGraw-Hill, 2010.