Revisiting Bug Triage and Resolution Practices

Olga Baysal, Reid Holmes, and Michael W. Godfrey
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, ON, Canada
{*obaysal, rtholmes, migod*}*@uwaterloo.ca*

***Abstract*—Bug triaging is an error-prone, tedious and time-consuming task. However, little qualitative research has been done on the actual use of bug tracking systems, bug triage, and resolution processes. We are planning to conduct a qualitative study to understand the dynamics of bug triage and fixing process, as well as bug reassignments and reopens. We will study interviews conducted with Mozilla Core and Firefox developers to get insights into the primary obstacles developers face during the bug fixing process. Is the triage process flawed? Does bug review slow things down? Does approval takes too long? We will also categorize the main reasons for bug reassignments and reopens. We will then combine results with a quantitative study of Firefox bug reports, focusing on factors related to bug report edits and number of people involved in handling the bug.**

I. Introduction

Bug reporting and fixing is an essential part of the software development process. In large software projects such as Eclipse or Firefox, the bug tracking system is the central hub for coordination that collects informal comments about bug reports and development issues.

Hundreds of bug reports are submitted every day, and the bug triage process is often error prone, tedious to perform, and very time consuming. While there is strong anecdotal evidence that bug triage is time consuming, few qualitative studies are available to support this statement. We would like to take another look at bug triage and resolution practice in open source projects. To our knowledge, the most recent qualitative study that involved surveying real-world developers on how they handle bug report and what they think a good bug report looks like was performed in 2008 by Bettenburg [1]. We would like to revisit bug reporting and fixing practices to find out if bug tracking systems have been improved since.

Users: In our study, the users are Mozilla developers, who are the users of the Bugzilla bug tracking system. By studying real-world users and evaluating their experience with Bugzilla, we can gather insights on flaws that still exist in bug assignment system and suggest possible improvements to bug tracking systems. By studying interview data, we can also understand dynamics of bug reassignments and reopens.

II. Related work

Most work on bug fixing have focused on how particular bugs should be fixed (or who is the best person to fix it) [2] and which types of bugs get fixed [3].

Bug Triage: Bug triaging has been studied a lot by researchers of SE community. Breu et al. [4] quantitatively and qualitatively analysed the questions asked in a sample of 600 bug reports from the Mozilla and Eclipse projects. They categorized the questions and analysed response rates and times by category and project and provided recommendation on how bug tracking systems could be improved.

Categorization of bug reports: Ko et al. [5] studied bug report titles and suggested designs for more structured problem report forms. Bettenburg et al. conducted a survey among 175 developers and users from the Apache, Eclipse, and Mozilla projects to determine which information contents comprise good quality bug reports. Just et al. [6] performed a quantitative study on the responses from the same survey to suggest improvements to bug tracking systems.

Bug reassignments and reopens: As yet, little attention has been paid to studying bug reassignments and bug reopens.

Jeong et al. [7] analyzed bug report reassignments (which they called “bug tossing”) in the Mozilla and Eclipse projects. They used a graph structure and Markov chains to reduce the number of reassignments.

In an empirical study of which bugs get fixed in the Microsoft Windows codebase, Guo et al. [8] observed reassignments are not always detrimental to bug-fix likelihood; several tries might be needed to find the optimal bug fixer. While they provided a comprehensive discussion of causes for reassignments, their study was done on a large commercial software project – Microsoft Windows. They categorized five primary reasons for reassignments: finding the root cause, determining ownership, poor bug report quality, hard to determine proper fix, and workload balancing. They built a descriptive statistical model to identify the relationship between bug report features and reassignments. They also provided some recommendations for the design of more socially-aware bug tracking systems. Recently, Zimmermann et al. [9] used the same survey data in the context of Microsoft Windows operating system to study bug reopens, primary reasons of reopens and the impact of various metrics on reopening bugs ranging from the reputation of the opener to how the bug was found.

We are interested in studying bug reporting and fixing practices in *open source projects*. While we might reach similar conclusions on the main reasons for bug reassignments and reopens to the ones reported in the study on Microsoft Windows, we hope to detect differences on triage practices between open and close source software projects.

III. Revisiting Bug Triage and Resolution Practices: A Case of Mozilla Core and Firefox

We are interested in studying bug triaging and fixing practices, including bug reassignments and reopenings, in the context of the Mozilla Core and Firefox projects, which we consider to be representative examples of a large-scale open source software project. We plan to conduct qualitative and quantitative analysis of the bug assignment practices.

We are interested in providing insights into several areas:

- triage practices, review and approval processes;
- root cause analysis of bug reassignments and reopens in open source software projects; and
- recommendations for improvements/redesign of bug tracking systems.

Data: Our primary data source is available interview transcripts [10], with questions about various aspects of the bug triaging, fixing process, and use of a bug tracking system. One-hour interviews were conducted by Mozilla product manager Martin Best and free-response statements are transcribed to permit analysis. Interview questions are focused on developers' experience with the Bugzilla bug tracking system.

Approach: Since the length of the responses varies from phrases to paragraphs, we will print the responses and split them into individual statements written on index cards. We plan to perform a card sort to organize interview responses into hierarchies to deduce a higher level of abstraction and identify common themes in the participants' feedback.

Card sorting is an inexpensive and user-centered sorting technique that is widely used in information architecture to create mental models and derive taxonomies from input [11]. There are three phases within a card sort: 1) preparation, in which participants and contents of the card sort are selected; 2) execution, where the indexed cards are sorted into meaningful groups with a descriptive title; and 3) analysis, in which the cards are sorted to form more abstract hierarchies that are used to deduce themes.

During the card sort execution process, we plan to categorize participants' feedback according to a number of topics including triage, review, reassignments, reopens. We hope to detect more emerging themes, such as bottlenecks in bug fixing process, development workflow, usability issues, report quality, new feature suggestions, etc.

Two of the authors will independently perform an open card sort and then merge the results into a single taxonomy; this will be used to inform improvements to Bugzilla to help it better meet the needs of Mozilla developers.

IV. Workshop goals

We hope to use the USER workshop as a venue to learn about user evaluation methodologies and to seek feedback on the execution and analysis of the proposed qualitative study. We hope to receive guidelines on what methods are the best for analysing the kind of data we have; how to physically organize the data; how to make sure our findings are consistent with and reflective of data; how to establish reliability, validity and triangulation in qualitative studies, etc.

References

[1] N. Bettenburg, S. Just, A. Schröter, C. Weiss, R. Premraj, and T. Zimmermann, "What makes a good bug report?" in *Proc. of the 16th ACM SIGSOFT Int. Symp. on Foundations of Soft. Eng.* ACM, 2008, pp. 308–318.

[2] J. Anvik, L. Hiew, and G. C. Murphy, "Who should fix this bug?" in *Proc. of the 28th Int. Conf. on Soft. Eng.*, 2006, pp. 361–370.

[3] S. Zaman, B. Adams, and A. E. Hassan, "Security versus performance bugs: a case study on Firefox," in *Proc. of the 8th Working Conf. on Mining Soft. Rep.*, 2011, pp. 93–102.

[4] S. Breu, R. Premraj, J. Sillito, and T. Zimmermann, "Information needs in bug reports: improving cooperation between developers and users," in *Proc. of the 2010 ACM Conf. on Computer Supported Cooperative Work*, 2010, pp. 301–310.

[5] A. J. Ko, B. A. Myers, and D. H. Chau, "A linguistic analysis of how people describe software problems," in *Proc. of the Visual Lang. and Human-Centric Comp.*, 2006, pp. 127–134.

[6] S. Just, R. Premraj, and T. Zimmermann, "Towards the next generation of bug tracking systems," in *Proc. of the 2008 IEEE Symp. on Visual Lang. and Human-Centric Comp.*, 2008.

[7] G. Jeong, S. Kim, and T. Zimmermann, "Improving bug triage with bug tossing graphs," in *Proc. of the the 7th joint meeting of the European Soft. Eng. Conf. and the ACM SIGSOFT Symp. on The Foundations of Soft. Eng.*, 2009, pp. 111–120.

[8] P. J. Guo, T. Zimmermann, N. Nagappan, and B. Murphy, ""Not my bug!" and other reasons for software bug report reassignment," in *Proc. of the ACM Conf. on Computer Supported Cooperative Work*, March 2011.

[9] T. Zimmermann, N. Nagappan, P. J. Guo, and B. Murphy, "Characterizing and predicting which bugs get reopened," in *Proc. of the 34th Int. Conf. on Soft. Eng.*, June 2012.

[10] M. Best, "The Bugzilla Anthropology." [Online]. Available: https://wiki.mozilla.org/Bugzilla_Anthropology

[11] Wikipedia, "Card sorting — Wikipedia, the free encyclopedia," February 2012. [Online]. Available: http://en.wikipedia.org/wiki/Card_sorting