IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 6, May 2022
Impact Factor: 6.252

Bug Tracking System

Harshit Paliwal, Akash Tomar, Ajay Verma, Swati Tyagi
Department of Computer Science & Engineering
Dronacharya College of Engineering, Greater Noida, India

Abstract: *Online Bug Tracking System on the Web is a web-based code bug tracking system which runs on the web portal. It is a open source package which is written in the Django scripting language. It can be installed on the different operating system such as Windows and Linux. Almost all the web browsers will be able to work as a client for tracking system. The Bug Tracking system generates the guick notifications to the ADMIN. With this bug tracking system members can easily assign the bug to the project admin in online user interface on the web. It is open source and easier for developers /tester to work on it. It is an intelligent platform where any organization can easily communicate to the members, and members are also free to directly contact with Admin, Developer and Tester. Admin can store the complete data of that Bug or the member who reported that in the special feature "Admin dashboard".*

Keywords: Bug Tracking System

I. INTRODUCTION

A bug tracking system is an application that lets you keep track of bugs (and often suggestions) for your software project in a database. An efficient bug tracking system that can be mapped well to your development and quality processes is an invaluable tool. Conversely, a poor bug tracking system that is difficult to use and does not fully reveal the state of your software can be an albatross around the neck of your project. Most bug tracking systems support the triage of incoming bugs, that is setting the priority (and perhaps severity) of a bug and assigning it to a particular developer. It's also standard practice to be able to define filters for the bugs in the system so that targeted bug lists can be created, such as a list of open crashing bugs or a list of bugs assigned to a particular developer. Related to this, some bug tracking systems will also provide report generation functions, often with the ability to display graphs and pie charts. This can be indispensible for generating guality metrics about your software, which together with code coverage results can be used to direct further testing efforts more efficiently.

II. SYSTEM REQUIREMENTS

Operating System :	Any Desktop OS
Environment:	Local or Remote host
Language :	**Python, HTML, CSS, Django**
Web Browser:	Supports all browsers
Connectivity :	Minimum 128 Kbps

Develop And Debug Software On SIM ICS

Bug Transportation • One of the hardest problems in debugging is to correctly and reliably reproduce a bug found by someone else. Typically, test departments and other users of software create long and brittle instructions to reproduce errors in bug-tracking systems. Accompanying each bug report is a list of relevant facts such as the version of the software, the OS version on the machine, the nature of the hardware, any attached external hardware, and anything else deemed relevant. With Simics checkpoints, all of this can be captured in a checkpoint and sent on from the bug reporter to the developer responsible. Using checkpoints in this way is called bug transportation (Engblom, 2010). • Thanks to the portability of Simics checkpoints and the determinism of the simulator, a checkpoint saved in one office by one user can be opened in any other office by any other user; and he or she will see exactly the same system setup and exactly the same system behavior

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 6, May 2022
Impact Factor: 6.252

when executing from the checkpoint. To ensure that the behavior seen repeats itself, it might be necessary to accompany the checkpoint with a series of actions taken on the target system.

Bug Report Severity Level Prediction In Open Source Software : K Survey And Research Opportunities

• Bug severity • The terminology of bug severity can be expressed by various terms depending on of the BTS used. Meaning of bug severity from three popular BTS are described next: • Bug Severity in Bugzilla indicates how severe the problem is -from blocker ("application unusable") to trivial ("minor cosmetic issue"). Also, this field can be used to indicate whether a bug is an enhancement reguest. On the other hand, the priority defines how urgent a bug needs to be fixed. In Bugzilla, the combination between priority and severity defines the importance of a Bug. • Bug Severity in Jira is referred to as "priority" and indicates how important the bug - from blocker ("highest priority") to minor ("low priority") is in relation to others bugs. • Bug Severity in Google Issue Tracker is also referred to as "priority" and indicates how priority the bug is - from PO ("needs to be addressed immediately") to P4 ("needs to be addressed immediately eventually") in relation to others bugs.

Better Bug Tracking System:

Having complete information in the initial bug report (or as soon as possible) helps developers to quickly resolve the bug. The focus of our work is on improving bug tracking systems with the goal of increasing the completeness of bug reports.Specifically, we are working on improving bug tracking systems in four ways.

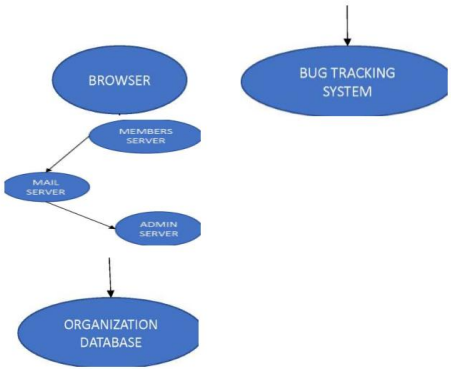
• Tool-centric: Tool-centric enhancements are made to the features provided by bug tracking systems. They can help to reduce the burden of information collection and provision. For example, bug tracking systems can be configured to automatically locate the relevant stack trace and add it to a bug report. Also, providing steps to reproduce can be automated by using capture/replay tools or macro-recorders; observed behaviour can be easily shown by simplifying screen capture; test case scan be automatically generated [3]. All of the above examples aim to help with the collection of information needed by developers to fix bugs.

• Information-centric: These improvements focus directly on the information being provided by the reporter. Bug tracking systems can be embedded with tools such as CUEZILLA [4] that provide real-time feedback on the quality of the information provided and what can be added to increase value. With helpful reminders from the system, reporters may be motivated to go the extra mile and collect more information. Systems can be further modified to perform validity checks such as verifying whether the reported stack trace is consistent and complete, submitted patches are valid, and the like

Process-centric: Process-centric improvements to bug tracking systems focus on administration of activities related to bug fixing. For example, bug triaging, i.e. determining which developer should resolve the bug can be automated to speed up the task [1, 5]. Other examples are better awareness of the progress made on bug reports (so that reporters are aware of the actions taken as a response for their efforts) or to provide users with an early estimate for when their bug will be fixed.

• User-centric: This includes both reporters and developers. Reporters can be educated on what information to pro-vide and how to collect it. Developers too can benefit from similar training on what information to expect in bug reports and how this information can be used to resolve bugs

Work Flow



IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
Volume 2, Issue 6, May 2022
Impact Factor: 6.252

CONCLUSIONS AND CONSEQUENCS

Current bug tracking systems do not effectively elicit all of the information needed by developers. Without this information developers cannot resolve bugs in a timely fashion and so we believe that improvement to the way bug tracking systems collect information are needed. This paper proposes four broad areas for improvements . While implementing a range of improvements from the seareas may be ideal, bug tracking systems may instead prefer to specialize, thus providing a rich set of choices. This would be a healthy change to the current situation where they all provide identical functionality. As an example of the kind of improvements we envision, we have described an interactive system for collecting information from reporters and leveraging that information locate the source of the bug. To demonstrate the potential of this idea we have conducted an initial study in which we simulated an interactive bug tracking system. The system asks the user context-sensitive questions to extract relevant information about the bug early on to suggest candidate files that will need to be fixed. This is likely to speed up the process of resolving bugs. In the future, we will move from the current prototype of the interactive system to a full-scale system that can deal with a variety of information to gather, os commonly observed in the real world.