

Customization of an Enterprise request Management System

Ashna Shah¹, Chinmay Balutkar², Bhargavee Singh³, Rajesh. B. Singh⁴

Student, Computer Department, Sinhgad Institute Of technology, Lonavala, India^{1,2,3}

Associate Professor, Computer Department, Sinhgad Institute Of technology, Lonavala, India⁴

Abstract: Information provided in issue reports are relevant and complete in order to help resolve issues quickly. However, often such information trickles to developers after several iterations of communication between End user and reporters. This paper addresses the concerns of Customization of an Enterprise management system by proposing for handling of the issues such as bugs, query and enhancements. As a proof-of-concept, we also demonstrate a prototype interactive enterprise request management system that gathers relevant information from the user and identifies files that need to be fixed to resolve the issues. The main contribution of this application is in the domain of business as we are developing Enterprise request Management System.

Keywords: Bugs, Issues, query, enhancement.

I. INTRODUCTION

The use of Enterprise Request Management Systems as a tool to organize maintenance activities is widespread. The systems serve as a central repository for monitoring the progress of issue reports, requesting additional information from reporters, and discussing potential solutions for fixing the Issue. Developers use the information provided in Issue reports to identify the cause of the defect, and narrow down plausible files that need fixing. A survey conducted amongst developers from the APACHE, ECLIPSE, and MOZILLA projects found out which information items are considered useful to help resolve bugs [4]. Items such as stack traces, steps to reproduce, observed and expected behaviour, test cases, and screenshots ranked high on the list of preferred information by developers. Developers are then forced to actively solicit information from reporters and, depending on their responsiveness, this may stall development. The effect of this delay is that bugs take longer to be fixed and more and more unresolved bugs accumulate in the project's bug tracking system. We believe that one reason for this problem is that current bug tracking systems are merely interfaces to relational databases that store the reported bugs. They provide little or no support to reporters to help them provide the information that developers need. Our on-going work is directed towards rectifying the situation by proposing ideas that can fundamentally transform bug tracking systems to enhance their usability such that relevant information is easier to gather and report by a majority of users.

A. FUNDAMENTAL CONCEPTS ON (DOMAIN):

Bug tracking systems is not that much developed it consist of many errors. So we are making the improvement in our enterprise request management system. In our system we will handle the issues that comes up in managing the projects. User will notify the issues related to the project management to the reporter. Then reporter will handle the issues if he gets the issue is valid then he will forward that issue to the developer, else he will send it back to the user for clarification that is the user will add more information

to the issue and again will report the issue to the reporter. Developer then will handle the issues and will fix them. This system will help to manage the issues in the business domain by fixing them. The issues might be a bug, query or the enhancement.

B. CONTRIBUTIONS:

In the last decade, Bug tracking is a system that is indispensable for any system that has to perform well. It is a tool that facilitates fixing of bugs faster and ensures the quality of software being developed or being used. These systems are widely used and they are treated as essential repositories that help in finding status of bugs and quickly resolving them thus the progress of the project can be ascertained. The potentiality and soundness of a good bug tracking system has no parallel in helping the systems to enhance the quality to meet the expectations of clients. Software engineers frequently use bug reports and try to fix them if there is enough information required. The information items that can be found in the bug reports include screenshots, test cases, expected behaviour observed behaviour, stack traces and steps to reproduce etc.

II. LITERATURE SURVEY

Bug and issue tracking systems are often implemented as a part of integrated project management system. This approach allows including bug tracking and fixing in a general product development process, fixing bugs in several product versions, automatic generation of a product knowledge base and release notes. Some bug trackers are designed to be used with distributed revision control software. These distributed bug trackers allow bug reports to be conveniently read, added to the database or updated while a developer is offline. Distributed bug trackers include fossil. Recently, commercial bug tracking systems have also begun to integrate with distributed version control. Fog bugz, for example, enables this functionality via the source-control tool, kiln. Although wikis and bug tracking systems are conventionally viewed

as distinct types of software, ikiwiki can also be used as a distributed bug tracker. It can manage documents and code as well, in an integrated distributed manner. However, its query functionality is not as advanced or as user-friendly as some other, non-distributed bug trackers such as bugzilla. Similar statements can be made about org-mode, although it is not wiki software as such.

A. Existing System

The problem in the older system can be defined as the whole project maintenance, user's maintenance and their assignment has to be maintained manually. The Software development companies have to face a lot of problems while maintaining manually all the maintenance of the projects, their bugs and their status. This type of problem makes the whole system an inefficient one and thus making a poor and unorganized working. In order to remove this type of problem, So that the paper is planned to develop. Bug tracking software is a "Defect Tracking System" or a set of scripts which maintain a database of problem reports. Bug tracking software allows individuals or groups of developers to keep track of outstanding bugs in the product effectively. Bug tracking software can track bugs and changes, communicate with members, submit and review patches, and manage quality assurance. This web-based business application is a great tool for assigning and tracking issues and tasks during software development and any other projects that involve teams of two or more people.

B. Problems

These are the Problems that Appear in the system thoroughly.

1) Conflict:

Bug resolution is ripe with conflicts. Such conflicts are fuelled by the uncertainty of failures, the conflicts of interests between developers and users, power disputes among developers, and the discussion medium itself [7], [8], [6]. In much of these cases, well-known developers often ignore or disregard the contributions of less well-known developers, sending them large amounts of unfair critique. As this can discourage participation from beginners [5], [6], projects seeking to grow its community need to manage such conflicts.

2) Misleading Bug Reports:

As opposed to useful bug reports, misleading reports are those that after inspection, do not result in changes to the target system—are not fixed—and are often considered diverters of valuable contributor attention.

III. PROPOSED SYSTEM

- We are using enterprise management system in which we will be handling the issues according to issue type.
- We handle the issues fast with the help of the issues status.
- We will handle the issues according to their priorities.
- With the help of game mechanism we will solve the issues fastly and arrange them according to the issue resolution.

IV. RELATED WORK

Many researchers have investigated what factors aid or indicate the quick resolution of bugs. Bug reports with more comments get fixed sooner. They also noted that bug reports that are easier to read also have shorter life times. Communication that takes place between developers outside the bug tracking system. Asking the right questions, is a crucial part of debugging and several techniques such as algorithmic debugging and the WhyLine tool support developers in doing so. In contrast, we propose to move this process from the developer side to the user side. Instead of recorded feedback by many users as it happens in Collaborative Bug Isolation, our proposed approach requires explicit feedback by only a single user.

V. IMPORTANT COMPONENTS :

A. ISSUES

Issues could represent a software bug, a project task, a helpdesk ticket, a leave request form etc.

B. Issue type

This system is used to track many different types of issues. The default types are listed below

Bug- A problem which impairs or prevents the functions of the project.

Improvement- An enhancement to an existing feature.

New Features- A new feature of the product.

Task- A task that needs to be done.

Custom Issue- A custom issue type, as defined by your organization if required.

C. Issue Priority

An Issue's priority indicates its relative importance. The default priorities are below

Blocked- Highest priority indicates that this issue takes precedence over all others.

Critical- Indicates that this issue is causing a problem and requires urgent attention.

Major- Indicates that this issue has a significant impact.

Minor- Indicates that this issue has a relatively minor impact.

Trivial- Lowest priority.

D. Issue Status

Each issue has a status, which indicates where the issue currently is in its lifecycle. An issue starts being 'Open', then generally progresses to 'Resolved' and then 'closed' or depending on circumstances, it may progress to other status.

Open — This issue is in the initial 'Open' state, ready for the assignee to start work on it.

In Progress — This issue is being actively worked on at the moment by the assignee.

Resolved — A Resolution has been identified or implemented, and this issue is awaiting verification by the reporter. From here, issues are either 'Reopened' or are 'Closed'.

Reopened — This issue was once 'Resolved' or 'Closed', but is now being re-examined.

(For example, an issue with a Resolution of 'Cannot Reproduce' is Re-opened when more information becomes available and the issue becomes reproducible). From here, issues are either marked In Progress, Resolved or Closed.

Closed — This issue is complete.

E. Issue's Solution

An issue can be solved in many ways, only one of them being 'Solved'. A solution is usually set when the status is changed.

Solved – A solution for this issue has been implemented.

Won't Solved -This issue will not be solved.

Conflict – This issue is a conflict of another issue.

Not Clarified- his issue does not have enough information to be solved.

VI. DISCUSSION AND FUTURE WORK:

In this project we are going to take the issues from the end user and then with the help of the information provided by the end user about the issue it will be forwarded to the reporter.

Reporter will check the issue if the issue is valid then it will be forwarded to the developer to solve the issue. However, if the reporter finds that the provided information about the issue is less than the reporter will forward it to the end user for the clarification. Then the end user will add the more information to the report and the will forward it to reporter. With the help of issue report the developer will solve the issue. And then forward it to the reporter. And the issue will be closed.

In this project, Customization of Enterprise management system will be using game mechanism, word conflict for improving the solution of the issues. However, the project will consist of issues, bugs enhancements, etc which will be searched and sorted according to issue status, issue priority, etc. With the help of Sorting and Searching the work will be progressed fast.

VII. CONCLUSIONS

This paper Customization of enterprise management request management system will help an software concern to detect and manage the issues in their products effectively and efficiently. This software will troubleshoot the errors for testing and for development processes. With the ability to provide comprehensive reports about the issues, documentation, searching capabilities, sending the screen shots of the issues will help a lot in software development.

Some of the enhancements that will be made in the system are as follows:

- 1) Changing issue bucket when logged.
- 2) Project wise admin required.
- 3) MIS report giving month wise summary of tickets buckets wise project wise.
- 4) Bulk issue fields editing screen.
- 5) Only the assigned user-role group of people should see option to edit the issue rest shouldn't see drop down to change status and write comments

REFERENCES

- [1] "Using JIRA and Redmine in Requirement Development for Agile Methodology" 978-1-4577-1531-0/11/\$26.00 ©2011 IEEE
- [2] "Towards Improving Bug Tracking Systems with Game Mechanisms" 978-1-4673-1761-0/12/\$31.00 ©2012 IEEE
- [3] "Bug Tracking and Reporting System" International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-3, Issue-1, March 2013
- [4] N. Battenburg, S. Just, A. Schröter, C. Weiss, R. Premraj, and T. Zimmermann. What makes a good bug report? In FSE'08: Proceedings of the 16th International Symposium on Foundations of Software Engineering, pages 308–318, November 2008.
- [5] K. Fogel, Producing Open Source Software – How to Run a Successful Free Software Project. O'Reilly Media, 2010.
- [6] M. Bergquist and J. Ljungberg, "The power of gifts: organizing social relationships in open source communities," ISJ, 2001.
- [7] A. Ko and P. Chilana, "Design, discussion, and dissent in open bug reports," iConference, 2011.
- [8] S. Hiltz and K. Johnson, "Experiments in Group Decision Making: Communication Process and Outcome in Face-to-Face Versus Computerized Conferences," Human Communication, 1986.