DEFECT TRACKING SYSTEM

Akash B. Thengre[1], Akshay G. Badge[2], Abhijeet S.Barley[3],
Govind K. Purve[4], Sachin S. Wararkar[5]
Dept. of CT, KDK college of engineering
akash.thengre@gmail.com

Abstract*—According to the present scenario, open source software revolution is gaining momentum in information technology era. In the battlefield of software technology, the open source software cannot be ignored. The philosophy behind open source extends towards quality and efficiency. In Open Source Software development there is a shared understanding among open source developers, reporters and users that efficiently improves the product quality. Yet quality and efficiency of Open Source Software depends upon the bugs present in the product. Thus understanding and tracking of system is vital process. In Open Source Software development tracking of bug is most important step. Bug Tracking system plays an important role in tracking of bugs. Bug tracking system contains the large amount of information about the bug in open source software. Choosing a good bug tracking system for any product will increase productivity of software. So we analyzed the existing bug tracking system and find out the limitations. Later we proposed the framework for bug tracking system based on the limitations founded in existing bug tracking system.

Introduction

Open source software (OSS) is computer software with its source code that is made freely available. This software is distributed under licensing agreement. This licensing agreement allows the source code to be shared, viewed and modified by users or organization. Open source software is an alternative way to develop software. Open source software is developed by community of volunteer developers over the internet. The source code of open source software is available to everyone and it can be freely used, modified and distributed at no cost. Steve Weber described open source as an experiment in social organization around a distinctive notion of property. Property in open source is configured fundamentally around the right to distribute, not the right to exclude [1]. Open Source has been getting much attention in the last few years .Now a days Many corporations, large and small, have taken an interest in this growing software market. This software may require additional module or enhancement of existing module time to time. But no software is perfect. Some of the software or module may contain defects. These defects can be unnoticed that are left from time to time. In Open Source Software development there is a shared understanding among open source developers, reporters and users that efficiently improves the product quality. Yet quality and efficiency of Open Source Software depends upon the defects present in the product. Thus tracking of defect is important .Defect tracking system plays in important role in tracking of defect.

A defect tracking system is an application that lets one to keep track of defects for software project in database. The report of the defect stored in defect tracking system .Where assignee of the defect Fix it. In open source software environment, user of open source software often write a "defect report" when they find defect or come across a mistake. Defect tracking system allows people anywhere in the world to report and describe the defect whenever they like. Anyone can access the repository of defect. An efficient defect tracking system that can be mapped well to development and quality process is an invaluable tool. Conversely a poor defect tracking system is difficult to use and does not fully reveal the state of software [2] [3].In our study we compare different defect tracking system and proposed a framework for defect tracking system on the basis of our findings in existing defect tracking system.

II LITERATURE

Nicolás Serrano and Ismael Ciordia [4] has compared two defect tracking tool i.e. are Defectzilla and ITracker. The objective of their research was to provide a comparative study of these two defect tracking tool based on the criteria platform independence, database independence, how customizable is it, are the number of users limited and life of cost. G Abaee and D.S Guru [5] gave the best practice to test documentation and effort estimations have been investigated as well as Defect Tracking Tools. They compare the four different existing Defect Tracking Tools with each other along with their features and drawbacks. Then they proposed new one, the Dedefectger Thomas Zimmermann, et al. [6] addressed

the concerns of defect tracking systems by proposing four broad directions for enhancements .They discussed that it is important that information provided in defect reports is relevant and complete in order to help resolve defects quickly.. Poorly designed defect tracking systems are partly to blame for exchange of information being stretched over time enhancements. As a proof-of-concept, they also demonstrate a interactive defect tracking system that gathers relevant information from the user and identifies files that need to be fixed to resolve the defect. Nicholas Jalbert & westley Weimer [7] discussed that Defect tracking systems are important tools that guide the maintenance activities of software developers. The utility of this tool is affected by an excessive number of duplicate defect reports–in some projects as many as a quarter of all reports are duplicates. If Developers manually identify duplicate defect reports, this identification process is time-consuming & exacerbates the already high cost of software maintenance. So they propose a system that automatically classifies duplicate defect reports as they arrive to save developer time.

Fischer et al. [8] discussed that Version control and defect tracking systems contain large amounts of historical information that can give deep insight into the evolution of a software project. Unfortunately, these systems provide only insufficient support for a detailed analysis of software evolution aspects. They addressed this problem and introduced an approach for populating a release history database that combines version data with defect tracking data and adds missing data not covered by version control systems such as merge points.

S. Just et al. [9] concluded that Developers typically rely on the information submitted by end-users to resolve defects. They conducted a survey on information needs and commonly faced problems with defect reporting among several hundred developers and users of the ECLIPSE, APACHE and MOZILLA projects.

M.P. Francisco et al. [10] have developed a tool to extract and to store information from Debian's BTS (Defect Tracking System) in a relational database. In this paper they showed that there is a strong dependence between three variables which can be used to analyze the activity of a project through its defects: communications between users and developers, defect notifications and people involved. They explained that defects are an essential part of software projects because they lead its evolution. Without defect notifications developers cannot know if their software is accomplishing its tasks properly.

A. Hora et al. [11] discussed that to harness the complexity of big legacy software; software engineering tools need more and more information on these systems. This information may come from analysis of study of execution traces, the source code, computing of metrics, etc. One source of information received less attention than source code: the defects on the system. Little is known about the evolutionary behavior, lifetime, distribution, and stability of defects. In this paper, they proposed to consider defects as first class entities and a useful source of information that can answer such topics.

Stephen Blair in his paper [12] provided tips and guidelines for evaluating features, and explains how these features fit into a defect tracking process. He discussed that evaluating a defect tracking system requires that you understand how specific features, such as configurable workflow and customizable fields, relate to your requirements and your current defect tracking process. He explained before you start evaluating defect tracking systems; make sure you identify your requirements for the system.

III NEED AND SCOPE OF STUDY

With the increase in the use of open source software the information technology era has given birth to new revolution. Wide range of Open Source Software is available in any usage area. Wide range of software products namely Operating System, Webservers, Word Processor, Databases, Defect Tracking System, Antivirus, Data Mining Software etc. are available

The various Defect Tracking Tool are available in Open source domain. Defect Tracking system gives the complete information about the defects which help the developer to keep the track of defects in the software product. The software product now a day are becoming more complex and it is becoming more difficult to keep the track of huge amount of defects in software having the complete and accurate information about the defects helps the developers to resolve it. So choosing a good tracking system for any of product will increase the productivity of software, improve the communication between the developers, produce the reliable and secure software and it will raise the customer satisfaction.

Keeping the importance of defect tracking system in mind a comparison of six different open source defect tracking system chosen for study are : Defect Genie, Mantis, Defect Tracker, Defectzilla ,ITracker, WebIssues**.**

IV.Objectives

The broad objective of the study is to present a comparison of six, different Defect Tracking System. The specific objectives of the study are:

(i) To perform the comparative study of Defect Tracking system.

(ii) To identify the limitations of Defect Tracking System under study.

(iii) To propose a framework for Defect Tracking tool.

V.Research methodology

In order to meet the objective theoretical approach has been used .The theoretical approach concentrate on describing Open source software, software defects, defect life cycle, and defect tracking system. The theoretical approach is based on review of secondary data acquired from literature survey, articles, books, research paper and internet.

VI.Research work

There are many Defect tracking System in software market. Choosing a good defect tracking system helps in increasing productivity, customer satisfaction and also improves communication between developers. We selected six defect tracking tool and their analysis is done on the basis of following criteria platform, user interaction, size and usage, functionality

All these categories are further divided into sub-categories. Tool considered for analysis are Defectzilla, Defect Genie,
Mantis, Defect tracker, iTracker, WebIssues

A. Analysis on the basis of User Interaction

The comparison based on User interaction basis shown in a table I. The criteria for User interaction is based on User interface, Language in which available, E mail notification etc.

TABLE I
ANALYSIS ON THE BASIS OF USER INTERACTION

User Interaction Tools	**User Interface**	**Available in Language**	**Email Notification**	**Intended Audience**
Defect Genie	web interface	English, French, German, Swedish, Norwegian, Spanish,	Yes	Customer Service, Developers, Information Technology, Quality Engineers, System Administrators Manufacturing,
Mantis	Web interface	English	Yes	Developers, System Administrators
Defect tracker	Web interface	Chinese,Czech,English, Polish,Portuguue	Yes	Developers
Defectzilla	web interface	Multiple languages	Yes	Customer Service, Developers, Other Audience, Quality Engineers, , End Users/Desktop
iTracker	Web interface	Catalan,Chinese,English,French, German,Italian,Portuguese , Turkish	Yes	Customer Service, , End Users/Desktop, Information Technology, Developers, System Administrators , quality Engineers

All the five defect tracker provides email notification and search facility for user interaction. Intended audience for defect genie are Customer Service, Developers, Information Technology, Manufacturing, Quality Engineers, System Administrators whereas for Mantis Developers and System Administrators. Developers are only audience for Defect tracker and for Defectzilla Customer Service, Developers, End Users/Desktop, Other Audience, Quality Engineers are entertained.

B. Analysis on the basis of Size and Usage

ANALYSIS ON THE BASIS OF SIZE AND USAGE

Usage and size Tool	First registered	First Release	Last updated	Total no of Version Released
Defect Genie	2003-09-05	2003-09-05	2013-03-04	67
Mantis	2000-11-18	2000-12-01	2013-04-13	107
Defect tracker	2005-03-30	2005-05-30	2011-06-19	32
Defectzilla	1988	1998-09-19	2013-02-19	116
iTracker	2002-05-22	2002-06-12	2012-12-13	22
WebIssues	2007-02-12	2007-02-12	2013-03-12	15

C. Analysis on the basis of Platform

Platform Tool	Program- ming language	Operating system	Database	License	Web Server (if Any)	Client
Defect Genie	Java Script, PHP	Cross-platform	MySQL >= 5.0 or Postgre SQL >= 8.2	Mozilla Public License 1.1 (MPL 1.1)	most web server which supports rewrite rules eg. window server, Linux server	Any web Browser
Mantis	PHP	Cross-platform	MySQL, MS SQL, and Postgre SQL	GNU General Public License v2	Apache and MS- IIS	Any web Browser
Defect- tracker	PHP	Window 2000	Postgre SQL and MySQL	GNU General Public License v2	IIS 5.0 -	Any Web Browser

VIII. Limitations

There are some limitations to current defect tracking system. During our analysis of various defect tracking system we find some issue which need to be included in current defect system. A good defect tracking system should have a good reporting design system, comfortable work environment for developer and experts and complete information for anonymous access.

1.) For good design of the reporting system a defect tracker should have user Friendly interface. There should be facility to reporter to submit report according to his

usefulness. There should not be repeated question in every defect report. Asking question according to problem defined will give a better defect report. And it should be web browser based that user can report a defect from anywhere anytime. And it should be available in multi-language so that user from all around the world can report. It should have automatic duplicate defect detection system so reporter of defect find out that defect has been reported earlier. And what is the status of defect.

2) In existing defect tracking system there is no filtration of defect report in initial phase. Existing defect tracking system allows fake defects to be submitted as defect report which is assigned to expert and entered in a defect list. The defect should not be submitted without complete description or valid description. And should not be entered in list of defect. This will reduce the effort of assignee to check the fake defect report and no fake defect is entered into the database of defect tracking system .To give the comfortable environment to developer and expert a defect tracking system should have automatic defect assignment system so that when a new defect reported into the system it is automatically assign to an expert of that area to which defect belong. For this defect tracking system should have advance capabilities. Such that there should be automated expert system in the system which will study a defect report and find a close match between defect report and expert of that field. And defect is automatically assigned to expert. But there should not over burden on one expert. Complete information should be given to expert to resolve the defect. A good defect report make easier to track the defect in defectgy code. And it will reduce the effort and time spend of expert or developer

3) Some of the users are anonymous user. They never Report a defect and neither contribute for defect fixing. They only give anonymous access to defect tracking system's defect repository. So a good defect tracking system should provide complete information to anonymous user who wants to collect the information from defect tracker's repository for their use. All defects should have complete information and history of defect .All the defect should be categories according to their priority and status. To our best knowledge no defect system provide total no of defect submitters in particular version of software. User has to calculate defect manually. There should be a feature of calculating total no of defect in particular version and there should also be record of the defects that are present in more than one version.

We proposed a defect tracking system that eliminates some limitations of existing defect tracking system.We propose that there should be facility to reporter to add fields according to his usefulness during the reporting of defect. There should be an automatic process that ask relevant question to reporter according to problem defined. There should be filtration of fake defects in initial steps .The filtration will describe defect as Defect or Not A Defect. We proposed that there should be automatic identification of defect report which will identify between actual defect and NotADefect to reduce the manual work. We also propose an automatic defect assignment /reassignment process to expert. For this there should be an expert system which will find a close match between defect report and expert of that field and automatically assigned to expert. After resolving the defect

Quality assurance team will verify the defect after the verification of defect if quality assurance team will not satisfy the defect can be reopened and if satisfy the defect will closed. Reopened defect assigned to expert again. Closed defect can be Fix, Wontfix and Incomplete. Fixed defect should be commit in version control .Fig 1. shows the proposed defect tracking system in oss domain.

X.Conclusions

Defect Tracking System is important software typically have tens or hundreds or thousands of defects. Defect tracking system is use to manage, fix and prioritize these defects. Defect tracking system is computer database system that store defect and help people to manage them. The objective of our study is to present comparison of different defect tracking system and to identify limitation of current defect tracking system. We concluded that current defect tracking system have some of limitation. They do not effectively collect all the information needed by developer, reporter and anonymous user. We have done analysis of defect tracking system on the basis of some criteria. But such criterion often doesn't give desired result.

So we propose a new framework of defect tracking system .the propose framework will give a improved level of satisfaction for current defect tracking system

References

[1] Steve Weber, "*The success of open source",* Harvard University Press, 2009.

[2] K Sowe Sulayman, G. Stamelos Ioannis, "*Emerging free and open source software practices*", Idea Group Inc (IGI), 2008.

[3] Martin Reddy, "*API Design for C++*", Elsevier, 2011.

[4] Nicolas Serrano, Ismael Ciordia, "Defectzilla, ITracker, and Other Defect Trackers", *IEEE software*, Vol 22, pp. 11-13,

[5] G Abaee, D.S. Guru, "Enhancement of Defect Tracking Tools; the Dedefectger", *Software Technology and Engineering (ICSTE), 2nd International Conference*, Vol 1, 2010.

[6] Thomas Zimmermann, Rahul Premraj, Jonathan Sillito, and Silvia Breu, "Improving Defect Tracking Systems*", 31th International Conference on Software Engineering , Vancouver, BC, Canada*, 2009.

[7] Nicholas Jalbert, Westley Weimer "Automated Duplicate Detection for Defect Tracking Systems" *International Conference on Dependable Systems & Networks*: *Anchorage, Alaska, IEEE*, 2008.

[8] M. Pinzer Fischer, H. Gall "Populating a Release History Database from version control and defect tracking systems" *Software Maintenance, IEEE*, 2003.

[9] S, Just, R. Premraj and T. Zimmermann. "Towards the next generation of defect tracking systems", *Visual Languages and Human-Centric Computing, IEEE*, 2008.

[10] M.P Francisco, P.B. Perez and G. Robles "Correlation between defect notifications, messages and participants in Debian's defect tracking system" *Empirical Software Engineering and Measurement, First International Symposium*, 2007.

[11] A. Hora, N. Anquetil, S. Ducasse, M. Bhatti, C. Couto, M.T. Valente and J. Martins, "Defect Maps: A Tool for the Visual Exploration and Analysis of Defects" *Software Maintenance and Reengineering (CSMR), 16th European Conference*, 2012.

[12] Stephen Blair "*A Guide to Evaluating a Defect Tracking System*, White paper, 2004.