Fuzzy Systems and Soft Computing
ISSN : 1819-4362

A Comparative Analysis of Multiple Defect Tracking Systems

Rajesh Kumar [1] ,Prasant Kumar Pani[2]
Raajdhani Engineering College, Bhubaneswar
[1]rajeshkumarsahoo@rec.ac.in
[2]prasantkumarpani@rec.ac.in

Abstract

Bug tracking systems, a cornerstone in many software projects, facilitate communication between users and developers by reporting issues and requesting new features. Despite their importance, these systems are not flawless, and no single system can be deemed the ultimate solution. Each has its own strengths and weaknesses, with varying features that contribute to their overall utility. While some may offer additional features compared to others, they generally share a common set of functionalities. This paper conducts a comparative analysis of different defect tracking tools currently accessible.

Keywords*: Defects, Defect Tracking Systems, Bugzilla.*

I. Introduction

Today, the using bug tracking system for tracking bugs and other issues is well spread. Bug tracking systems are using for organizing and monitoring bugs. Without these systems, monitoring a large amount of bugs will be impossible or very hard. Because of that, today there exist a lot of bug tracking systems. They very differ by quality, security, costs, and functionality they offer to the use. In this paper, we give short overview of various aspects of some Defect Tracking System with their advantages & Disadvantages.

II. Various Defect Tracking Systems

a) BUGZILLA:

Bugzilla is a very popular, actively maintained and free bug tracking system, used and developed together with Mozilla, giving it considerable credibility. It is based on Perl and once it is set up, it seems to make its users pretty happy. It's not highly customizable. Bugzilla installations tend to look pretty much the same wherever they are found, which means many developers are already accustomed to its interface and will feel they are in familiar territory. Bugzilla has a very advanced reporting system and you can create different types of charts including line graph, bar graph or pie chart.

Bugzilla UI is strictly functional. There is nothing very nice about it, it provides plenty of functions within a small space, and in the beginning, the user can feel quite uncomfortable and lost; however, after discovering it, the user will find out that it is not very complicated and working with it is straightforward.

Advantages & Disadvantages:

- Bugzilla notifies users of any new or updated bugs by e-mail.
- Bugzilla supports basic time tracking.
- Bugzilla also supports a system of votes, in which users can vote for issues or features they wish to see implemented.
- Bugzilla is particularly complicated to install and maintain,
- It supports large Projects.
- It doesn't have user-friendly interface.

b) MANTIS:

Mantis is a free web-based bug tracking system. It is written in the PHP scripting language and works with MySQL, MS SQL, and PostgreSQL databases and a web server. Mantis can be installed on Windows, Linux, Mac OS and OS/2. Almost any web browser should be able to function as a client. The main complaint is its interface which doesn't meet modern

standards. On the other hand, is easy to navigate, even for inexperienced users. There not exist some advanced features such as charts and reports. In short, the whole system is sloppily done; there are plenty of bugs and very little functionality.

c) BUGTRACKER.NET:

BugTracker.NET is a free, open-source, web-based bug tracker or customer support issue tracker written using ASP.NET, C#, and Microsoft SQL Server Express. BugTracker.NET is easy to install and learn how to use. When you first install it, it is very simple to setup and you can start using it right away. Later, you can change its configuration to handle your needs. It has a very intuitive interface for generating lists of bugs. It has two very useful features. First of them is a screen capture utility that enables you to capture the screen, add annotations and post it as bug in just a few clicks. The second feature is the fact that it can integrate with your Subversion repository so that you can associate file revision check-ins with bugs.

d) BUG-TRACK:

Bug-Track is web-based defect and bug tracking software allows you to document manage and assign all of your bugs and tasks and empowers you to organize your bugs, defects or issues into distinct projects. It can run on virtually any web-server like Microsoft, Linux, Unix, etc... Since it is a commercial application it is expected that it is better than other free products. But it isn't true. It has nothing new and better than other free bug tracking systems. One better thing is fact that it has moreintuitive interface then others and that is his onlybenefit.

e) REDMINE:

Redmine is a flexible web-based project management web application. Written using Ruby on Rails framework, it is cross-platform and cross-database. Redmine is open source and released under the terms of the GNU General Public License. Redmine is flexible issue tracking system. We can define our own statuses and issue types. It supports multiple projects and subprojects. Each user can have a different role on each project. Interface is very simple, intuitive and easy to navigate. Redmine is a very good recommended defect tracking system.

f) BUGZERO:

Bugzero is a web-based bug, defect, issue and incident tracking software. Its single code base supports both Windows and Unix and supports database systems including Access, MySQL, SQLServer, Oracle, and etc. Bugzero can be customized for software bug tracking, hardware defect tracking, and help desk customer support issue and incident tracking. Bugzero have intuitive interface but it lacks form features. The main drawback is the fact that Bugzero is an commercial product and there are much better product for free.

III. Factors to Classify Defect Tracking System

These are the factors which are useful for average user & they give an ease to use bug tracking system. These factors also used for decision making for choosing best Defect Tracking System.

Search is very useful criteria and it is present in all selected products. Email notifications gives user opportunity to be noticed about happenings in the current bug tracking system. The user does not need to check frequently bug tracking system for new changes. All he need is an email account. Reports give user a brief and concise overview about past happenings in our system. Charts give clear graphical view of selected criteria which is very intuitive to the human being. Time tracking is an feature that give information about happenings of some specific bug trough time. Like an email RSS/Atom feed gives user opportunity to be noticed about happenings in the current bug tracking system. Configurable system is capable to be configured to meet certain user needs, so the system should be configurable as much aspossible, because that will help to satisfy the larger population of customers. At the end, the much important issue for choosing the right bug tracking

system is the fact is it free or not and how much he cost.

IV. Conclusion

Comparative study of some defect tracking system has been done and it is seen that current Defect tracking systems do not effectively collect all of the information needed by developers. Without this information developers cannot resolve defects timely and so it has been seen that improvements to the way issue tracking systems collect information are needed. We have summarized factors that are used in modern bug tracking systems. Such factors often don't give appropriate results in describing defect. Some additional features must be added to the existing Defect Tracking tools to enhance usability and functionality.

References

[1] Aaen, I. (2003). Software Process Improvement: Blueprints versus Recipes. IEEE Software 20(5), 86–93.

[2] Adeel, K., (Aug. 2005). Defect prevention techniques and its usage in requirementgathering – industry practices , Proceedings of Engineering Sciences and Technology, ISBN 978-0-7803-9442-1, SCONEST, IEEE Computer Society Publisher, pp 1-5.

[3] Boehm, B. and V. Basili (2001). Software Defect Reduction Top 10 List. Computer 34(1), 135–138.

[4] Bassin, K. and P. Santhanam (2001). Managing the maintenance of ported, outsourced, and legacy software via orthogonal defect classification. In Proceedings of the IEEE International Conference on Software Maintenance, Washington, DC, pp. 726. IEEE Computer Society.

[5] Boehm, B. , Basili,V., (2001) Software Defect Reduction Top 10 List Journal Computer archive Volume 34 Issue 1, IEEE Computer Society,, CA, USA

[6] Boehm, B. and V. Basili (2001). Software Defect Reduction Top 10 List. Computer 34(1), 135–138.

[7] Børretzen, J.A. , Conradi, R., (2003). "Results and Experiences From an Empirical Study of Fault Reports in Industrial Projects", Proceedings of the 7th International Conference on Product Focused Software Process Improvement, Amsterdam, pp. 389-394

[8] Børretzen, J.A. , (Sept. 2009) .Investigating the Software Fault Profile of Industrial Projects to Determine Process Improvement Areas: An Empirical Study", Proceedings of the 14th European Systems & Software Process Improvement and Innovation Conference, Potsdam, Germany, pp. 212-223.

[9] Card, D. N. (2002). Managing software quality with defects. In Proceedings of the 26th International Computer Software and Applications Conference on Prolonging Software Life: Development and Re-development pp.472–474.

[10] Card, D.N., (October 2006). Myths and Strategies of Defect Casual Analysis, Proceedings of twenty-fourth Annual Pacific Northwest Software Quality Conference, Portland, Oregon, pp 469-474.

[11] Chillarege, R. and K. Prasad (2002). Test and Development Process Retrospective -A Case Study Using ODC Triggers. In Proceedings ofthe International Conference on DependableSystems and Networks, Florence, Italy, pp. 669– 678. IEEE Computer Society Press.

[12] Drago, J., (2011). "Role of Defect Management Software in Software Development Life cycle", Available at -http://ezinearticles.com/?Role-of-Defect-Management-Software-in-The-Software-Development-Life-Cycle&id=6205930.

[13] Fredericks, M., Basili, V.,(1998). "Using defect tracking and analysis to improve software quality". Report DACS-SOAR-98-2, Experimental Software Engineering Group, University of Maryland, College Park, MD, USA

[14] Fairley, R. E. and M. J. Willshire (2005). Iterative rework: The good, the bad, and the ugly. Computer 38(9), 34–41.

[15] Howles, T. and S. Daniels (2003). Widespread Effects of Defects. Quality Progress 36(8), 58–62.

[16] Humphrey, W. (2002). Winning with Software. Addison-Wesley Longman Publishing Co., Inc., Boston, MA.

[17] IEEE (2004). IEEE Standard for Verification and Validation/IEEE Std. 1012-2004.

[18] Jones, C.(2009)," Software Engineering Best Practices: Lessons from Successful Projects in the Top Companies". McGraw-Hill, New York, NY, USA, 2009. ISBN 007162161X.

[19] Kaner, C. (2002). Bug Advokacy. Retrieved from Cam Kaner Home Page: http://www.kaner.com/pdfs/BugAdvocacy.pdf

[20] Khilman, A., (2005). “Defect Management Process in Software Testing”, Thesis submitted in Faculty of Information Technology, MERA Technical University.

[21] Lars Ola Damm (2007) "Early and Cost-effective Fault Detection", Dissertation submitted to Blekinge Institute of Technology, Sweden

[22] Leszak, M., D. Perry, and D. Stoll (2000). ACase Study in Root Cause Defect Analysis. In Proceedings of the 22nd Int. Conference on Software Engineering, Limerick, Ireland, pp. 428–437. ACM Press.

[23] Leszak, M. (2002). Classification and evaluation of defects in a project retrospective. Journal of Systems and Software 61(3), 173–187.

[24] Lyu, M., (2007) “Software reliability engineering: A roadmap”. In FOSE ’07: 2007 Future of Software Engineering, pages 153–170, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-2829-5.