ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 9, September 2015

Bug Tracking and Reporting System (BTRS) For Software Quality Assurance and Maintenance

K Poornachandra[1], G Naga Lakshmi[2]

M. Tech Student, Department of Software Engineering, Siddartha Institute of Science and Technology, JNTU, Ananthapur, , India

HOD, Department of Software Engineering, Siddartha Institute of Science and Technology, JNTU, Ananthapur, India

ABSTRACT: Maintenance of the accessible system is an important process in the Software Development Life Cycle (SDLC). Keeping track of the problems rise in the existing and who is assigned with the maintenance work. The solutions provided for the problems are always a tedious job. The Bug Tracking and Reporting System (BTRS) are developed to handle these tasks and maintaining software quality. It is a Multi User online system span over intranet/internet. It is involved in almost all the stages of SDLC such as Requirement Analysis, Design, Coding, Testing and Maintenance. Also it will maintain project teams, members of the teams, their roles (Program leader (PL), Developer, and Tester). Although BTRS is involved in all phases of SDLC, the primary phase is testing. Before submit to the client or hosting any site of the project, all the bugs in the project should be rectified. Based on this process the system will maintain a database for each project, in that it maintain project details, employees working on the project, and the bugs in the project which are reported by Testers. While the Tester is in the process of finding the new bugs, Bug tracking developers read the reported bugs and fixes them and updates the same information in the bug reports. Thus, BTRS is primarily designed to report and track bugs of different projects simultaneously to obtain software quality and deliver reliable software to the client. This system helps organization to maintain errors, bugs and defects occur at SDLC, which helps in upcoming projects.

KEYWORDS: Software Development Life Cycle, Bug Tracking, Bug, Error, Defects, Software Quality.

I. INTRODUCTION

Computer code has become a thrust. It's engine that force business judgment making .It is the supported trendy scientific investigation and engineering draw back resolution. it is a key issue that differentiates trendy product and services. It embedded in system of all kinds: a Transportation, Medical, amusement, and work product the list is type of endless. Code is type of inevitable throughout a these days. And as we've got an inclination to enter the ordinal century, it's going to become the propulsion for complete bright advanced in everything from educational activity to biotechnology.

A. *software engineering*

The purpose of a scientific, disciplined, quantitative, approach to the event, procedure and maintenance of software system is termed software engineering. The software engineering was popularized by F.L Bauer for the duration of the NATO software engineering conference in 1968.It encompasses techniques and procedures, frequently regulated by a software development practice, with the reason of improving the reliability and maintainability of software systems.

The authority of software production includes information, tools and strategies for software system needs, design, construction, testing, and maintenance tasks [1]. software system engineering is associated to the discipline of computing, engineering, management, arithmetic, project management, software system bioengineering, and systems engineering [2][3].

B. Common issues in computer code development:

a) *Poor requirements:* If requirements aren't clear, incomplete, too broad or not testable, there'll be issues.

DOI: 10.15680/IJIRCCE.2015. 0309141

b) *Unrealistic schedule*: If an excessive amount of of work is crammed in insufficient time, issues are usual.
c) *Inadequate testing*: nobody can acknowledge whether or not or not the program is any smart quality till the client complaints or systems collapse.
d) *Futurities*: requirements to stack on new options once development is current, extraordinarily common.
e) *Miscommunications:* If developers don't recognize what's required or customers have inaccurate expectations, issues are bonded.

C. *Testing to improve software quality:*

To maintain software quality testing is a major phase in SDLC.Testing meets three objectives:[4][5]

a) *Identification of Errors*: These are obvious anomalies that show up within the behavior of program or a entity or a module. Such behavior because the following is taken into account miscalculation. Wrong total, arrangement, messages that say wrong issue, action that don't execute as promised: the delete button doesn't delete, the update menu doesn't update properly.etc...

b) *Conformance to requirements:* These errors are the results of testing the functions within the software system against demand the need Definition Document to confirm that each requirement, functional or non functional is within the system which it works properly. Typically this can be often referred to as associate Operational Qualification (OQ).However .note that although a number of the requirements don't appear to be "Operational", this is often associate operational check. For elements .if the message aren't show, then there's an absence of conformity and therefore the system doesn't operate properly.

c) *Performance Qualification:* These aren't "errors "as such however failure to evolve to Performance Qualification (PQ) became a regular methodology of testing for past reasons. Some systems can perform otherwise below completely different masses and condition. for instance, a subject search application may have to control among specific time reply for a load of up to three hundred queries an hour. The software perform is also designed properly.ie, might pass the operational qualification, however might fail to fulfill the required masses attributable to poor programming or too several information calls.

II. INTERRELATED WORK

In software engineering industry in order to report a bug which is generated in the project, while testing it tester should send a report manually to the administrator, then administrator will send the report the corresponding project leader to the related project. Next the project leader find out to which module does it belongs to and forwards the bug to the corresponding developer who develops corresponding module [fig-1]. Then the developer will tracks or rejects the bug and submit the report to the project leader and project leader will forward it to tester. All these reporting will be done manually and consumes a lot of time. Time is main factor in software industry. In my work, it reduces the time in reporting and tracking the bugs. It also increases the effective communication among developer, tester and others who involves in bug reporting. Using my work they can report the bugs from any branch of the organization.

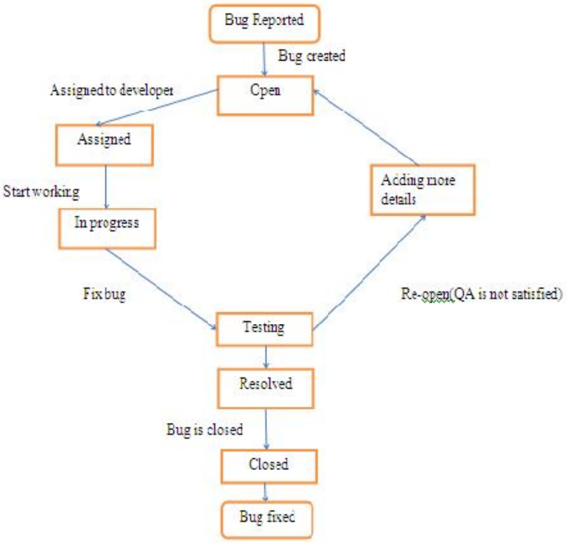

Fig-1: Bug Life Cycle.

DOI: 10.15680/IJIRCCE.2015. 0309141

Inside any computer code improvement bugs are to be anticipated. Let it's in any quite object bugs arise at any a part of development. One should take a wonderful care among the right maintenance and determination of the bugs. inside the current system the bugs are not properly maintained and they are simply relied on shared lists and email to watch the bugs [fig-2].In this reasonably system it becomes difficult to trace a bug if a bug is over looked then it's reaching to cause tremendous error at intervals subsequent half and may improve the worth of project despite necessary effort spent on the bug maintenance won't be worthy. so bug history should be maintained properly. And there's no economical search technique.

In this form of system it becomes robust to trace a bug if a bug is over looked then it's progressing to root unthinkable errors at intervals succeeding section and should improve the worth of project notwithstanding necessary effort spent on the bug maintenance may not be worthy. so bug history possesses to be maintained properly. And there is no economical search technique. One should search the whole data for the tiny print of scrupulous bug that might have occurred sometime earlier. it's every time overwhelming and error prone. And it's very difficult to share the bug among many users as there isn't any correct maintenance of the bugs. in order to possess academic degree economical product bugs ought to be maintained properly and will be resolved in time ever to cut back time and money spent on system development.

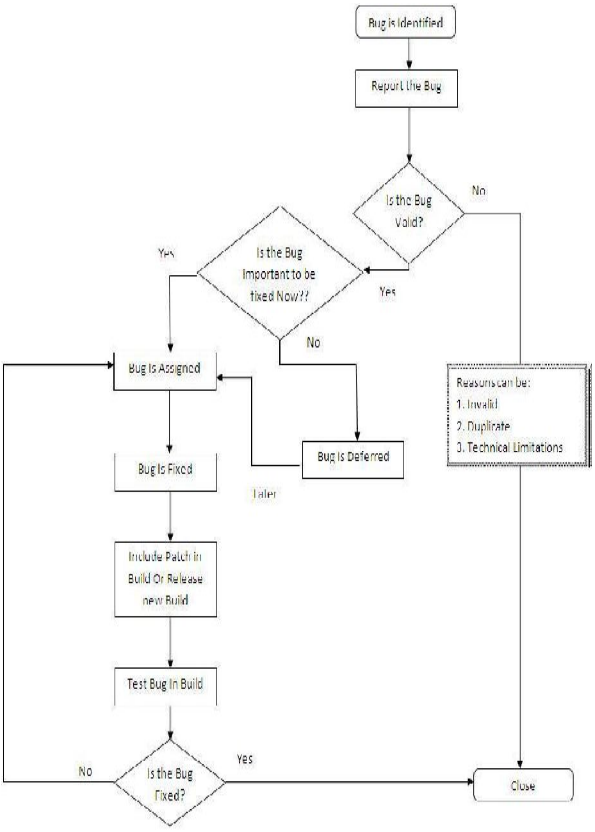

Fig 2: Basic Bug Reporting System.

III. BUG TRACKING AND REPORTING SYSTEM TO IMPROVE SOFTWARE QUALITY

In my work, it reduces the time in reporting and tracking the bugs. It also increases the effective communication among developer, tester and others who involves in bug reporting. Using my work they can report the bugs from any branch of the organization. The Bug Tracking and Reporting System(BTRS)will reduce the time in bug clearing process in software engineering .BTRS is also useful in effective communication between the software developer, administrator, software tester and project leader of that particular project. BTRS will store the bugs of the projects in the database which can be useful for upcoming projects .Totally it is useful to produce preferable and efficient project by viewing bugs by the members of the organization [7].

BTRS will be used by administrator, software developer, administrator, software tester, project leader and the user of that particular project. BTRS will give a common platform to the actors by placing bug information in the database whenever they are occurred and whenever they are cleared. Security features will be well defined and implemented to make sure that the users will be able to meet the challenges and be able to do what they need to do to better serve the needy their by fulfilling the planned purpose of the organization. My aim of this project was to develop System that developer with minimal knowledge, time and minimum work can easily check correctness of the code easily without any errors. Before submitting to the user or hosting every site of the project, it should maintain testing tools. Here this developing system is based on this process only. BTRS was designed to report and track bugs of different projects while the project is requirements gathering, designed, coded, tested and also in the maintenance [fig-3].

BTRS involves reaching out to potential bugs by maintaining the database of bugs occurred in every project. Additionally, it will provide the recent and mostly occurring bugs in the projects by highlighting them for the effective removal of the bugs [7].

This system maintains the merchandise, Bugs and bug trailing. Its advantage of maintaining bug history it stores all the small print from bug origin to bug resolution. Each object can have versions for straightforward maintenance of the result and each one the user of the merchandise is hold on at intervals the knowledge. It provides the advantage of maintaining users to the bugs and resolutions provided by them. This method provides the searching based n standing, priority, and software system. It provides with user and bug hierarchy, which could be helpful in knowing the relation between bugs and users allotted to the bug. It's provided with a very genuine system with identification cryptography. And has the ability for storing attachments for a bug. One will keep a track of the bug in a passing product with plenty of lower price and energy. The foremost advantage of this method is maintaining log records that are useful in knowing any errors or misuse of the system by completely different users [fig-3].

Compensation:

a) This system Reducing paperwork in organization,
b) Efficient utilization of time in reporting the bugs in that project.
c) Interaction between the employees in the organization.
d) Focus on the bugs which affect the project and remotely providing solution to the bug raised in project.

IV. BTRS Structural Design

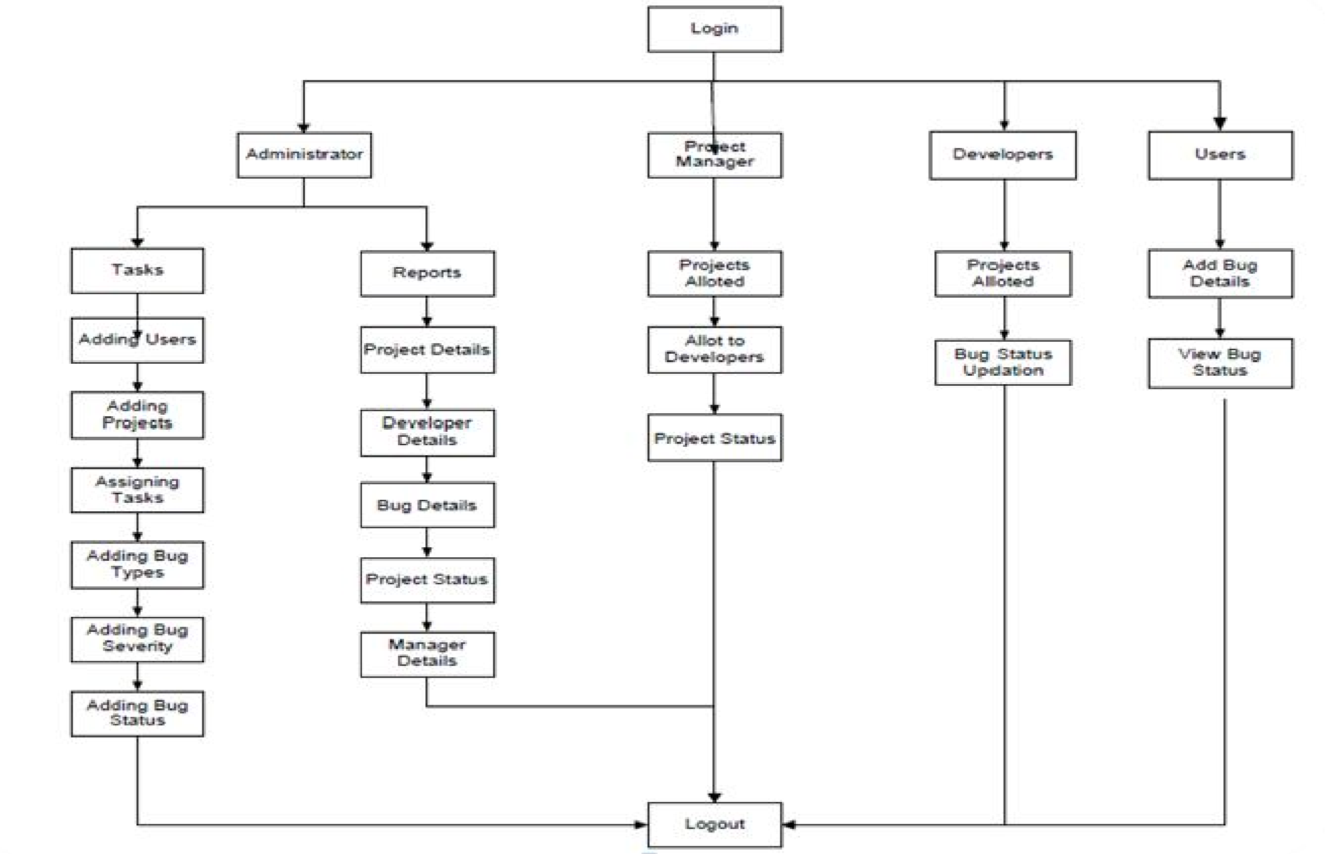

Fig 3: BTRS structural design

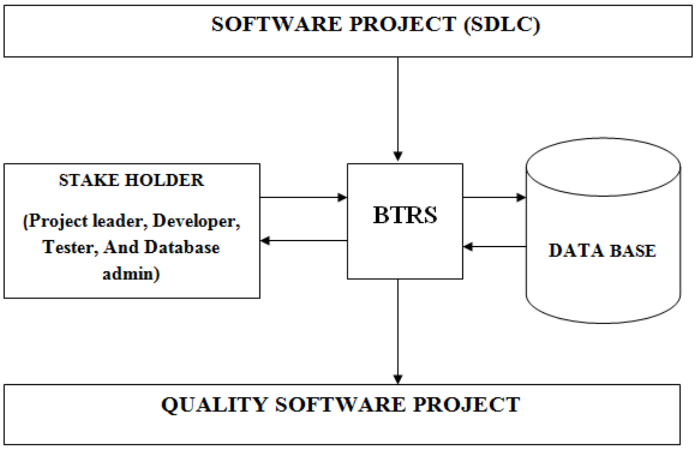

Fig 3: BTRS ARCHITECTURE

V. CONCLUSION

This system as explained in the entire documentation has the most reliable database as on and user-friendly interfaces with useful tool tips provided for the convenience of the user whenever they are required. We have taken optimum care in the design of the system and reviewing the same so that any additional feature, if at all the client wants to include, can be added very easily in future. This system reducing paperwork in organization, efficient utilization of time in reporting the bugs in that project. Interaction between the employees in the organization. Focus on the bugs which affect the project and remotely providing solution to the bug raised in project.

VI. FUTURE WORK

As software system is employed, by the customer/user can acknowledge extra functions which will provide benefit. Perceptive maintenance extends the software system on the far side its original performs requirements. By adding the report generation practical to the current system. We can enhance it by placing the communication between user and developer, tester in order to find required bugs in the projects.

REFERENCES

[1].http://en.wikipedia.org/wiki/Bug_tracking_system

[2]. Bug Tracking and Reliability Assessment System (BTRAS) International Journal of Software Engineering and Its Applications Vol. 5 No. 4, October, 2011.

[3]. Change and Bug Tracking System: Anjung Penchala Sdn. Bhd. International Journal of Computer Applications (0975 – 8887) Volume 10– No.3, November 2010.

[4]. Defect Tracking System International Journal of Electrical, Electronics ISSN No. (Online): 2277-2626 and Computer Engineering **3**(1): 212-217(2014).

[5]. Identification and Listing of Factors Affecting Software Quality Assurance, International Journal of Internet Computing ISSN No: 2231 – 6965, VOL- 1, ISS- 3 2012.

[6]. How to Improve Software Quality Assurance in Developing Countries, Advanced Computing: An International Journal (ACIJ), Vol.3, No.2, March 2012

[7]. Bug Tracking and Reporting System, International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-3, Issue-1, March 2013.

[8]. The Bug Life Cycle, 8 CROSSTALK The Journal of Defense Software Engineering September 2003.