ISSN:0975-9646

Improving Software Quality Assurance Using Bug Tracking System

Rajnish Kumar, Devasani Yellaiah Gattaiah, Swati Shahi, Thakur Avinash Nagendra

Department of Computer Engineering
Sir Visvesvaraya Institute of Technology, Nasik

***Abstract*—This paper entitled Improving Software Quality Assurance Using Bug Tracking System is mainly for applications developed in a company to keep track of employee skills and based on the skills assigning of the task is done to an employee and aims at creation of a Bug Tracking System for enhancing the software quality.**

This project will be accessible to all developers and its facility allows developers to focus on creating the database schema and while letting the application server define table based on the fields in JSP and relationships between them.

Improving Software Quality Assurance Using Bug Tracking System is an automated system that can be useful to employees and the managers in any functional organization. This Bug Tracking System gives the facility to define the tasks in the organization and also allows the managers to track the bugs spent by the employee for that particular task. A report generation facility is supported in BTS (Bug Tracking System) that allows the managers to analyse which those skills by employee are utilized and those which are not utilized. This tool can help managers for Bug estimation per project or application. This tool helps employees to document their Bugs and analyse.

***Keywords*— BTS, AS, ISQA**

I. INTRODUCTION

First of all, we would like to differentiate between those term which often comes in our mind such as a bug, a virus, a worm, a Trojan horse and blended threats.

Bug - A software bug is an error, mistake, failure, or fault in a computer program that prevents it from behaving as intended (e.g., producing an incorrect result). Most bugs arise from mistakes and errors made by people in either a program's source code or its design, and a few are caused by compilers producing incorrect code. According to folklore, the first computer bug was an actual bug. Discovered in 1945 at Harvard, a moth trapped between two electrical relays of the Mark II Aiken Relay Calculator caused the whole machine to shut down.

Virus - A computer virus is a computer program that can copy itself and infect a computer without the permission or knowledge of the owner and attaches itself to a program or file enabling it to spread from one computer to another, leaving infections as it travels. Like a human virus, a computer virus can range in severity: some may cause only mildly annoying effects while others can damage your hardware, software or files. Almost all viruses are attached to an executable file, which means the virus may exist on your computer but it actually cannot infect your computer unless you run or open the malicious program. It is important to note that a virus cannot be spread without a human action, (such as running an infected program) to keep it going. Because a virus is spread by human action people will unknowingly continue the spread of a computer virus by sharing infecting files or Sending emails with viruses as attachments in the email. Sometimes when you try to send a mail with an attachment of .exe file to another then it fails, this is due to a virus detection in email server.

Worm - A worm is similar to a virus by design and is considered to be a sub-class of a virus. Worms spread from computer to computer, but unlike a virus, it has the capability to travel without any human action. A worm takes advantage of file or information transport features on your system, which is what allows it to travel unaided. The biggest danger with a worm is its capability to replicate itself on your system, so rather than your computer sending out a single worm, it could send out hundreds or thousands of copies of itself, creating a huge devastating effect. One example would be for a worm to send a copy of itself to everyone listed in your e-mail address book. Then, the worm replicates and sends itself out to everyone listed in each of the receiver's address book, and the manifest continues on down the line. Due to the copying nature of a worm and its capability to travel across networks the end result in most cases is that the worm consumes too much system memory (or network bandwidth), causing Web servers, network servers and individual computers to stop responding. In recent worm attacks such as the much-talked-about Blaster Worm, the worm has been designed to tunnel into your system and allow malicious users to control your computer remotely.

Trojan horse - A Trojan horse is full of as much trickery as the mythological Trojan horse it was named after. The Trojan horse, at first glance will appear to be useful software but will actually do damage once installed or run on your computer. Those on the receiving end of a Trojan horse are usually tricked into opening them because they appear to be receiving legitimate software or files from a legitimate source. When a Trojan is activated on your computer, the results can vary. Some Trojans are designed to be more annoying than malicious (like changing your desktop, adding silly active desktop icons) or they can cause serious damage by deleting files and destroying information on your system. Trojans are also known to create a backdoor on your computer that gives malicious users access to your system, possibly allowing confidential or personal information to be compromised. Unlike viruses and worms, Trojans do not reproduce by infecting other files nor do they self-replicate.

Blended Threats - Added into the mix, we also have what is called a blended threat. A blended threat is a more sophisticated attack that bundles some of the worst aspects of viruses, worms, Trojan horses and malicious code into one single threat. Blended threats can use server and Internet vulnerabilities to initiate, then transmit and also spread an attack. Characteristics of blended threats are that they cause harm to the infected system or network, they propagates using multiple methods, the attack can come from multiple points, and blended threats also exploit vulnerabilities. To be considered a blended thread, the attack would normally serve to transport multiple attacks in one payload. For example it wouldn't just launch a DOS attack — it would also, for example, install a backdoor and maybe even damage a local system in one shot. Additionally, blended threats are designed to use multiple modes of transport. So, while a worm may travel and spread through e-mail, a single blended threat could use multiple routes including e-mail, IRC and file-sharing sharing networks. Lastly, rather than a specific attack on predetermined .exe files, a blended thread could do multiple malicious acts, like modify your exe files, HTML files and registry keys at the same time — basically it can cause damage within several areas of your network at one time. Blended threats are considered to be the worst risk to security since the inception of viruses, as most blended threats also require no human intervention to propagate.

1.1 Purpose

The purpose of improving software quality assurance using Bug Tracking System is to provide better service to the administrator or useful for applications developed in an organization.

1.2 Scope

The Improving Software Quality Assurance Using Bug Tracking System is a web based application that can be accessed throughout the organization. This system can be used for logging bugs against an application/module, assigning bugs to team members and tracking the bugs to resolution. There are features like email notifications, user maintenance, user access control, report generators etc. in this system.

1.3 Overview

Bug tracking is the process of reporting and tracking the progress of bugs from discovery through to resolution, where a bug is defined as a deviation from requirements. Other terminology frequently used to describe this process include

1. Problem Tracking
2. Change Management
3. Fault Management
4. Trouble Tickets

Bug tracking systems are most commonly used in the coding and testing phases of the software development process. However, tracking systems can in fact be used for many other purposes such as general issue tracking, simple task lists, help desk situations or contact management, where the focus is on the tracking aspect rather than what is being tracked. Even in software development, tracking systems are quite often not limited to simply tracking bugs, but extended to track feature requests or enhancements as well as enquiries.

II. Existing System

The existing system consists of entering the details in the Microsoft Excel Sheets for the storing of the data. When a manager needs information of the employee he searches for the specified file in the file system. He opens the file and takes the information. Report Generation done manually by copying the content of the different files into another file. The Manually generated report was then printed.

2.1 Limitations:

Followings are the limitations in existing system:

1. Information retrieval is a very big process.
2. Lack of organization of the files may porn to information loss due to accidental deletion of files.
3. No security because the files are visible to the users.
4. Report generation will be a big task.

III. Proposed System

The Proposed system is a browser which is completely related to online system, which provides the centralized database. It stores bugs data and description of the particular bug data. It can also create Excel reports and PDF documents based on the information in its database.

3.1 Advantages over Existing System

Following advantages can be mentioned:

1. The performance is increased due to well-designed database.
2. Security is increased
3. Time saving in report generation
4. Easy to update the details.

IV. Project Modules

In this project, we have used 5 modules:

4.1 Admin

This module has the entire access to all other modules, admin creates the project and assigning the projects to the created manager, adding members to the managers, assigning bugs based on the priority and can update the manager, members and access to the particular project data.

4.2 Manager

Manager has the full access to the particular project assigned by the admin and controls the team member access to the bugs assigned. He has the permission to generate the report and update the information of team member and adding members to the project.

4.3 Developer

Developer can access the task or bug assigned by the manager, view assigned projects and resolving the assigned bug. Developer can view the bugs list assigned by the manager.

4.4 Tester

Tester can access to the projects or bugs assigned by the manager and can view the assigned projects and can add a new bug to the list and send the bug back to the manager. Tester can login to the system and access the assigned projects list.

4.5 Reports

Both Admin and Manager can access this module and generate the reports based on the requirements.

V. PRODUCT FEATURES

These are the following features which are very important of this project:

5.1 Clustering

Clustering can be implemented at different levels of the system, including hardware, operating systems, middleware, systems management and applications. The more layers that incorporate clustering technology to more complex the whole system is to manage. To implement a successful clustering, solution specialists in all the technologies (i.e. hardware, networking, software) are required. In this project, clustering has been used to track the bugs and enhance the software quality.

The advantages of clustering computers for scalability include increased application performance and the support of a greater number of users and also for high availability which can be seen if one of these computers fails, another computer in the cluster can then assume the workload of the failed computer.

5.2 Failover including sessions

We can get Failover by deploying identical ColdFusion applications and configurations to multiple server instances and adding the instances to a cluster. Each instance must have the same applications deployed and the same resources configured (such as data sources, Verity collections, and mappings). The web server connector optimizes performance and stability by automatically balancing load and by switching requests to another server instance when a server instance stops running.

5.3 Load Balancing

Load balancing is an important feature of this project in which the application server automatically alternates requests among the server instances in a cluster. Clustering also enables application servers to route requests to a running server instance when the original server instance goes down.

5.4 Distributed caching(using JBoss caching)

In this project, JBoss Cache has been used for clustering solutions that aims to provide high availability and dramatically improve performance by caching frequently accessed Java objects.

5.5 Distributed Deployment

This project has also the feature of distributed deployment, in which the layers of the application reside on separate physical tiers. Distributed deployment allows to separate the layers of an application on different physical tiers. We can see from the below figure:

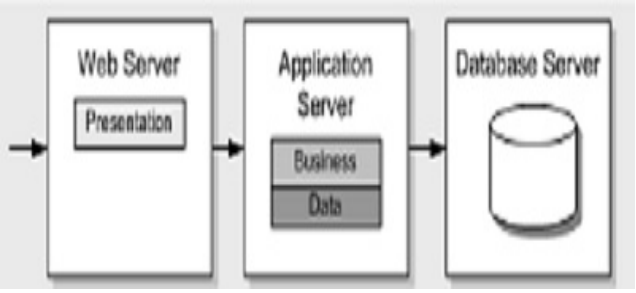

Fig. 5.5 Distributed Deployment

VI. FUNCTIONAL REQUIREMENTS

These are the following functional requirements:

6.1 Increase visibility of development process

Improve customer satisfaction by increasing communication and allowing them to monitor the progress of development.

6.2 Traceability of bugs and their resolutions

Maintain audit trails to ensure all changes are accounted for.

6.3 Release planning

Manage the bugs and enhancements that are to be resolved for the next product release.

6.4 Resource scheduling

Manage the bugs that are assigned to each team member.

6.5 Prioritization

Assign priorities to the bugs to ensure critical errors are addressed before minor issues, such as the wording of an error message.

6.6 Improved control of a project

Monitor the status and progress of bugs, to follow the improvement in stability of a product or to ensure early detection of failing the project.

6.7 Information consolidation

Capture all bugs, feature requests, ideas and etc. in one place to promote the sharing of information project wide.

6.8 Improve the quality of your software by increasing productivity

Notification of bug creation and status change to team members increases awareness and responsiveness.

VII. SPECIFIC FUNCTIONAL DESCRIPTIONS

7.1 Authentication

a. Login to the system through the first page of the application.
b. Change the password after login to the application.
c. See his/her details and change it.
d. Help from the system.

7.2 User Maintenance:

Creating, Granting & Revoking access and deleting users from pplication.

7.3 Component Maintenance:

Creating a component (application being developed / enhanced), Granting & Revoking access on components to Users and Marking a component as “Active” or “Closed”.

7.4 Bug Tracking:

Creating, Assigning Bugs to users, Modifying and Closing a Bug. A Bug screen should at least have following details

1. Bug Number and Title
2. Bug priority
3. Date created

7.5 Find User:

A search screen to find users and display results.

7.6 Find component:

A search screen to find components and display results.

7.7 Find Bug:

A search screen to find Bugs and display results.

7.8 Report:

Generate reports on Bugs.

VIII. NONFUNCTIONAL REQUIREMENTS

These are the nonfunctional requirements:

8.1 Performance Requirements:

The proposed system that we are going to develop will be used as the major performance system for providing help to the developer related to their problems. Therefore, it is expected that the system would perform functionally all the requirements that are specified.

1. The system should be easy to handle.
2. System should give expected performance results.
3. Bug tracking will be excellent.

8.2 Security Requirements:

We are going to develop an improved bug tracking system. There are different categories of users namely Administrator, Developer who will use some specific information from the system.

8.3 Safety Requirements:

The system may get crashed at any certain time due to virus or operating system failure. Therefore, it is required to take the backup.

8.4 Software Quality Attributes:

1. The application is easy to interact and communicate with user.
2. This application provides better user interface for ease of working.

IX. External Interface Requirements

9.1 User Interface:

This system is purely user friendly for administrator and developer.

Following screens has been provided:

1. A login screen for entering the username and password, so that the authorized user can have an access without any problems.
2. There will be a screen which will be displaying the major tasks that the system will be performing i.e. organization, bugs, reports, security.
3. Each major tasks has some tasks inside that as for example; in organization, view projects and view members are present, in bugs; view projects and view bugs are present.
4. All the major tasks mentioned above will have their separate forms and will perform the desired actions.

9.2 Software Interface

1. Operating System : Windows XP or Linux or Solaris
2. Programming Language: Java
3. User Interface: Html/CSS
4. Client Side Scripting: Java Script
5. Database: Oracle 10g
6. IDE: My Eclipse 8.0
7. Web Applications: Jdbc, Jndi, Servlet and Jsp.
8. Server Deployment: RetHat JBoss AS(Application Server)

9.3 Hardware Interface

1. Intel Pentium IV or Higher processor
2. 1.80 GHz
3. 40 GB of Hard Disk
4. 256 MB of RAM

X. Architecture

This project's architecture can be understood from following diagram:

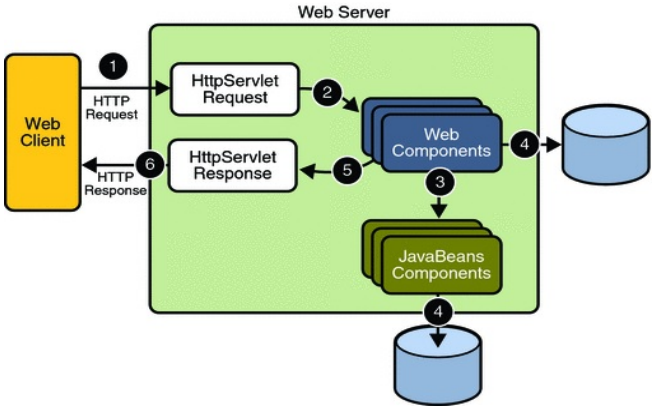

Fig. 10.1 Architecture of ISQA Using BTS

XI. Some Screenshots

Following are some screenshots of this project:

11.1 Login Page

Fig. 11.1 Login Page

11.2 Admin Home Page

Fig. 11.2 Admin Home Page

11.3 Organization Page

Fig. 11.3 Organization Page

11.4 Bugs Page

Fig. 11.4 Bugs Page

11.5 View Priority Page

Fig. 11.5 View Priority Page

11.6 Change Password Page

Fig. 11.6 Change Password Page

11.7 Developer Page

Fig. 11.7 Developer Home

11.8 Tester Page

Fig. 11.8 Tester Page

XII. LIMITATIONS

Following are the limitations:

1. Only the permanent employees can access the system.
2. System works with windows'98 and its compatible environments.
3. Advanced techniques are not used to check the authorization.
4. Once the employee is registered to a course cannot drop, without completing.

XIII. FUTURE ENHANCEMENTS

It is not possible to develop a system that makes all the requirements of the user. User requirements keep changing as the system is being used. Some of the future enhancements that can be done to this system are:

1. As the technology emerges, it is possible to upgrade the system and can be adaptable to desired environment.
2. Because it is based on object-oriented design, any further changes can be easily adaptable.
3. Based on the future security issues, security can be improved using emerging technologies.
4. Sub admin module can be added.

XIV. CONCLUSIONS

Following conclusions can be concluded:

1. Instant access.
2. Improved productivity.
3. Optimum utilization of resources.
4. Efficient management of records.
5. Simplification of the operations.
6. Less processing time and getting required information.
7. User friendly.

ACKNOWLEDGMENT

We would like to thank Prof. S. M. Rokade, Head of Computer Engineering Department, Sir Visvesvaraya Institute of Technology, Nasik and our project guide Mr. Y Ragava, Sr. Software Engineer at Chaithanya IT Solutions, Hyderabad, for his valuable guidance and support.

REFERENCES

[1] "Core Java™ 2 Volume I – Fundamentals 7th Edition Pearson Education – Sun Microsystems" by Cay S. Hortsman, Gary Cornell
[2] "Core Java™ 2 Volume II – Advanced Pearson Education – Sun Microsystems" by Cay S. Hortsman, Gary Cornell
[3] "Head First Servlets & JSP O'Reilly – SPD" by Eric Freeman , Elisabeth Freeman
[4] "The Book of JavaScript 2nd Edition - SPD" by thau
[5] "Effective Java – Programming Language Guide, Pe" by Shadab Siddiqui
[6] "JAVA server pagesarson Education – Sun Microsystems" by Joshua Bloch