**Bug Tracking Tool**

Tracking of defects plays an important role in the testing environment of a software project.

Testing is a crucial part of the software life cycle, and recent trends in software engineering evidence the importance of this activity all along the development process.

Defect tracking is a system that is applied for any system software so that system performs well. The information items that can be found in the defect reports include screenshots, test cases, expected behaviour observed behaviour, and steps to reproduce etc. Generally this is the minimum preferred information needed by engineers to fix defects easily.

A defect report should clearly specify the characteristics‟ like having clearly specified defect number, reproducible and be specific. Generally, developers expect descriptions beyond the information found in the present defect reports. Depending upon the defect priority and the severity the testing of the defect is done.

Priority is generally set as p1 to p4. Severity says the impact of the defect.

**RELATED WORK**

Earlier, people use to use a simple way for defect tracking by sending mail to a related technical department. This is difficult to keep track the defect as the emails are scattered. So, then testers started using manual testing (manual defect tracking) via email or Excel.

Defect tracking is the process of finding defects in a product, (by inspection, testing, or recording feedback from customers), and making new versions of the product that fix the defects. Defect tracking is important in software engineering as complex software systems typically have tens or hundreds of thousands of defects: managing, evaluating and prioritizing these defects is a difficult task. Defect tracking systems are computer database systems that store defects and help people to manage them.

Using Defect tracking tool the following process is followed:

a. .Logging into the tool

b. .Defect Life Cycle

c. Creating a defect

d. Changing status of defects

e. Generating metrics and reports

*Defect Life Cycle* (Defect Life cycle) is the journey of a defect from its identification to its closure. The Life Cycle varies from organization to organization and is governed by the software testing process the organization or project follows and/or the Defect tracking tool being used.

*Defect Status Explanation*

*NEW*: Tester finds a defect and posts it with the status NEW

*OPEN*: Test / Development / Project lead studies the NEW defect and if it is found to be valid it is assigned to a member of the Development Team. The assigned Developers responsibility is now to fix the defect and have it COMPLETED.

*DEFERRED*: If a valid NEW or ASSIGNED defect is decided to be fixed in upcoming releases instead of the current release it is DEFERRED. This defect is ASSIGNED when the time comes.

*REJECTED*: Test / Development/ Project lead studies the NEW defect and if it is found to be invalid, it is REJECTED. Note that the specific reason for this action needs to be given.

*FIXED*: Developer fixes the defect that is ASSIGNED to him or her. Now, the fixed defect needs to be verified by the Test Team and the Development Team assigns the defect back to the Test Team. A FIXED defect is either CLOSED, if fine, or REASSIGNED, if still not fine.

If a Developer cannot fix a defect, some organizations may offer the following status:

1. *Can’t Reproduce*: The Developer is unable to reproduce the defect.
2. *Need More Information*: The Developer needs more information on the defect from the Tester.

*REOPENED*: If the Tester finds that the fixed defect is in fact not fixed or only partially fixed, it is reassigned to the Developer who fixed it. A REOPENED defect needs to be COMPLETED again.

*CLOSED*: If the Tester / Test Lead finds that the defect is indeed fixed and is no more of any concern, it is CLOSED. This is the happy ending.

*DEFECT SEVERITY CLASSIFICATION*

The actual terminologies, and their meaning, can vary depending on people, projects, organizations, or defect tracking tools, but the following is a normally accepted classification.

*Critical:* The defect affects critical functionality or critical data. It does not have a workaround.

*Major:* The defect affects major functionality or major data. It has a workaround but is not obvious and is difficult.

*Minor:* The defect affects minor functionality or non-critical data. It has an easy workaround. Example: A minor feature that is not functional in one module but the same task is easily doable from another module.

Trivial: The defect does not affect functionality or data. It does not even need a workaround. It does not impact productivity or efficiency. It is merely an inconvenience. Example: Petty layout discrepancies, spelling/grammatical errors.

Severity is also denoted as:

S1 = Critical

S2 = Major

S3 = Minor

S4 = Trivial

*Defect Priority* indicates the importance or urgency of fixing a defect. Though priority may be initially set by the Software Tester, it is usually finalized by the Project/Product Manager.

Priority can be categorized into the following levels:

*Urgent:* Must be fixed in the next build.

*High:* Must be fixed in any of the upcoming builds but should be included in the release.

*Medium:* May be fixed after the release / in the next release.

*Low:* May or may not be fixed at all.

Severity is also denoted as:

P1 = Critical

P2 = Major

P3 = Minor

P4 = Trivial