DEFECT MANAGEMENT SYSTEM

NANDAKUMAR M
Assistant Professor, Department of Computer Applications, AJK College of Arts & Science, Coimbatore, Tamil Nadu, India

ABSTRACT
While there is no harm in using an excel sheet to record/track and emails to report/alert/communicate – as the magnitude of the projects, the number of test cycles, the sum of the people involved grows – it becomes absolutely important that we need a much stronger mechanism that will make the organisation of these issues simpler and consistent so we can concentrate harder on actually finding more issues in the AUT than managing the ones previously found. o enable the same, the QA market has seen the emergence of various defect management tools over the years.

Keywords: Defect, Tracking

INTRODUCTION
Defect cycle or defect life cycle is ride of a defect from discovering defect to closure of defect. Defects management in defect cycle is significant to ensure the software quality. Preventing, identifying, rectifying defect is important to improve the quality. Defect is managed and tracked easily throughout the defect cycle with the use of defect tracking tools like JIRA, Mantis, Team Service, Bugzilla, and Redmine etc. In a project identifying defects in early stage and fixing will take less cost when compare with identifying and fixing defects in later stage of the development. A tester need to identify valid defect and raise the defect with all the detail and make sure the defect gets closed. We also need to make sure the closed defect is verified and in the close stage in future releases also. If it gets reopen that should be resolved, verified and closed again. Defect has several stages from identification to closure. Managing entirely the defects on particular software project is important to complete the project successfully.

DEFECT MANAGEMENT PROCESS

Identifying defect
To identify defect, the tester should know exactly what the expected behaviour of the system is. While doing the test execution tester will compare with the expected behaviour and if it varies, there is a defect discovered. Tester should think is non-functional areas such as
usability, user friendly also and if there is any problem in that that also need to be raised as a defect or communicate as an improvement. Always tester with the defect when report the defect. Sample should be should make sure the exact expected result is mentioned along concern and raise the valid faults. Defects that hold the state New, Assigned and Reopened are falls under this stage. After this stage defect shall be resolved.

**Defining Defect**

Tester and developer should rectify the defect without the negative encouragement to other areas. While resolving the defect, the expected behaviour should work correctly and defect should not be there any longer. Defects that hold the state fixed duplicated, cannot reproduce, and cannot resolve and not a defect are under this stage. After this stage the defect is ready to retest.

**Retesting Defect**

After developer rectifies the defect, tester will retest the particular defect. Tester will test whether the defect is fixed and the expected behaviour of the system is applied. Defects that hold the state fixed, duplicated, cannot reproduce, cannot resolve and not a defect will be verified in this stage. Once Defect’s journey is end when the defect is closed. But the defect should be verified in later releases also to check the retested the defect transfer either to reopen or closed state expected. behaviour is working and defect is no more occurring.

**DEFECT LIFE CYCLE STEPS**

**New**

When a defect is encountered in the software project, the tester is supposed to raise it. At the very first time when the defect is raised, the stage of the defect is ‘New’. Tester should always analyze and make sure that it is a valid defect. Tester should know what is the expected result whenever encounter a defect.

**Allotted**

Once the defect is raised it will be assigned to the developer to resolve the defect. Tester should know to which developer the exact defect should be assigned. Once assigned the defect is transfer to the developer hand.

**Immovable**

Developer will analyze, check and fix the defect. While fixing the defect designer should make sure there is no other negative influence to other areas in the software
Delayed
There are some instance the defect is valid but it will be fixed. As a good practice when deferred the defect, the decision needs to be taken after discussing with the team members. In the future releases, then the defect transfer to deferred state.

Replacement
There are some cases the defect is valid but it’s been already reported either by another tester or in another form. At this time developer will mark the defect as duplicate defect.

Not a defect
If the defect is not a valid defect then the developer will state the defect as ‘Not a defect’.

Cannot Reproduce
There are some instance the developer can’t reproduce the defect, then the developer will mark as cannot reproduce. It’s better to check with the tester and try to re-create the defect. Some defects are data specific or situation specific. At this time it’s good to check with tester and find the root cause and resolve the fault.

Closed
While tester retest the defect if the defect is fixed then the defect goes to ‘Closed’ state. Tester should verify the closed defect in future and mark sure the defect is not happening any more in the software.

Defect life cycle
Below diagram explain the flow of the defect throughout the defect life cycle.

Conclusion
Defect Management system, when used right – as a tester, you understand your ecosystem better and as a team, it will improve the overall efficiency. Therefore, if you are still using the primitive spreadsheet method for bug tracking, it’s time to change. The choices for bug tracking tools are many. If using a test management tool you are going to have access to defect tracking as well. Some companies create in-house bug tracking tools. They are similar to the commercial ones existing. So, they do the job just fine. Commercial, yet affordable tools. Finally, if all your team wishes is a tool for defect tracking and if the entire testing is still maintained manually, your best option is to go with an open source defect management system.
REFERENCES

1. A comparative study of automated software. Testing tools Nazia Islam. Nazia Islam. isna1301@stcloudstate.edu


3. A study on Various software Automation Testing Tools Neha bhateja, Department of computer science, Amity university Gurgaon, Haryana, India.


