Defect Life Cycle

What is Defect Life Cycle?

Defect life cycle, also known as Bug Life cycle is the journey of a defect cycle, which a defect goes through during its lifetime. It varies from organization to organization and also from project to project as it is governed by the software testing process and also depends upon the tools used.

Defect Life Cycle - Workflow:

<<<<Defect Life Cycle in Software Testing image>>>>

Defect Life Cycle States:

* New - Potential defect that is raised and yet to be validated.
* Assigned - Assigned against a development team to address it but not yet resolved.
* Active - The Defect is being addressed by the developer and investigation is under progress. At this stage there are two possible outcomes; viz - Deferred or Rejected.
* Test - The Defect is fixed and ready for testing.
* Verified - The Defect that is retested and the test has been verified by QA.
* Closed - The final state of the defect that can be closed after the QA retesting or can be closed if the defect is duplicate or considered as NOT a defect.
* Reopened - When the defect is NOT fixed, QA reopens/reactivates the defect.
* Deferred - When a defect cannot be addressed in that particular cycle it is deferred to future release.
* Rejected - A defect can be rejected for any of the 3 reasons; viz - duplicate defect, NOT a Defect, Non Reproducible.

What is Defect Logging and Tracking?

Defect logging, a process of finding defects in the application under test or product by testing or recording feedback from customers and making new versions of the product that fix the defects or the clients feedback.

Defect tracking is an important process in software engineering as Complex and business critical systems have hundreds of defects. One of the challenging factors is Managing, evaluating and prioritizing these defects. The number of defects gets multiplied over a period of time and to effectively manage them, defect tracking system is used to make the job easier.

Examples - Hp Quality Center, IBM Rational Quality Manager

Defect Tracking Parameters

Defects are tracked based on various parameters such as:

* Defect Id
* Priority
* Severity
* Created by
* Created Date
* Assigned to
* Resolved Date
* Resolved By
* Status

<<<>>>>

Bug tracking system

A bug tracking system or defect tracking system is a software application that keeps track of reported software bugs in software development projects. It may be regarded as a type of issue tracking system.

Many bug tracking systems, such as those used by most open source software projects, allow end-users to enter bug reports directly. Other systems are used only internally in a company or organization doing software development. Typically bug tracking systems are integrated with other software project management applications.

A bug tracking system is usually a necessary component of a good software development infrastructure, and consistent use of a bug or issue tracking system is considered one of the "hallmarks of a good software team".[1]

Contents

1 Components

2 Usage

3 Bug tracking systems as a part of integrated project management systems

4 Distributed bug tracking

5 Bug tracking and test management

Components

A major component of a bug tracking system is a database that records facts about known bugs. Facts may include the time a bug was reported, its severity, the erroneous program behavior, and details on how to reproduce the bug; as well as the identity of the person who reported it and any programmers who may be working on fixing it.[2]

Typical bug tracking systems support the concept of the life cycle for a bug which is tracked through status assigned to the bug. A bug tracking system should allow administrators to configure permissions based on status, move the bug to another status, or delete the bug. The system should also allow administrators to configure the bug statuses and to what extent a bug in a particular status can be moved. Some systems will e-mail interested parties, such as the submitter and assigned programmers, when new records are added or the status changes.

Usage

The main benefit of a bug-tracking system is to provide a clear centralized overview ofdevelopment requests (including both bugs and improvements, the boundary is often fuzzy), and their state. The prioritized list of pending items (often called backlog) provides valuable input when defining the product road map, or maybe just "the next release".

In a corporate environment, a bug-tracking system may be used to generate reports on the productivity of programmers at fixing bugs. However, this may sometimes yield inaccurate results because different bugs may have different levels of severity and complexity. The severity of a bug may not be directly related to the complexity of fixing the bug. There may be different opinions among the managers and architects.

A local bug tracker (LBT) is usually a computer program used by a team of application support professionals (often a help desk) to keep track of issues communicated to software developers. Using an LBT allows support professionals to track bugs in their "own language" and not the "language of the developers." In addition, an LBT allows a team of support professionals to track specific information about users who have called to complain — this information may not always be needed in the actual development queue. Thus, there are two tracking systems when an LBT is in place.

Bug tracking systems as a part of integrated project management systems

Bug and issue tracking systems are often implemented as a part of integrated project management systems. This approach allows including bug tracking and fixing in a general product development process, fixing bugs in several product versions, automatic generation of a product knowledge base and release notes.

Distributed bug tracking

Some bug trackers are designed to be used with distributed revision control software. These distributed bug trackers allow bug reports to be conveniently read, added to the database or updated while a developer is offline.[3] Fossil and Veracity both include distributed bug trackers.

Recently, commercial bug tracking systems have also begun to integrate with distributed version control. FogBugz, for example, enables this functionality via the source-control tool, Kiln.[4]

Although wikis and bug tracking systems are conventionally viewed as distinct types of software, ikiwiki can also be used as a distributed bug tracker. It can manage documents and code as well, in an integrated distributed manner. However, its query functionality is not as advanced or as user-friendly as some other, non-distributed bug trackers such as Bugzilla.[5] Similar statements can be made about org-mode, although it is not wiki software as such.

Bug tracking and test management

While traditional test management tools such as HP Quality Center and IBM Rational Quality Manager come with their own bug tracking systems, other tools integrate with popular bug tracking systems.[citation needed]