UNIT IV : Defect Management

Defect

- Inconsistency in behavior of s/w
- s/w doesn’t meet requirement
- Errors in coding or logic
- Expected result don’t match with actual results.
- Root causes of defects :
- --requirement are not well defined
- -Design or implementation are not proper
- --people are not trained

Defect Classification

- **Requirement Defect** : product fail to understand what is required by customer
  - Functional defect : absent of functionality
  - Interface defect : user interface problems , like problem with connectivity to others (hardware).

- **Design Defect :** Problem with design creation
  - Algorithmic defect : problem in design
  - Module Interface defect : problem with module interface
  - User Interface defect : problem with usability of product like navigation , look and feel.
  - System interface defect : problem with interaction of environment like product not able to recognise inputs coming from environment

- **Coding Defect :** when designs are implemented wrongly.
  - Variable declaration / initialization :
  - Commenting /documentation :no adequate commenting
  - Data base related defects :
- **Testing Defect :** arises due to wrong defects.
  - Test design defect : defect in test plan , test case
  - Test tool defect : difficult to find test tool defect.
  - Test environment defect : arise when test environment not proper . Like h/w , s/w , simulator , people

- **Assign severity , probability and priority**

- **Severity wise: can be decided based on how bad is the defect.**
- -- severity 1 / critical : eg : s/w crash frequently.
- -- severity 2 /major : Operational errors
- --severity 3 / medium :misspelling , UI layout
- --severity 4/ low : Suggestion
- **Probability wise : Likelihood of a user encountering the defect**
- -- High : encountered by all the users.
- --Medium : encountered by 50% of the users
- --Low : encountered by very few users.

- **Priority Wise : defined order in which the defect should be resolved**
- **--** Urgent : must be fix immediately
- -- High : Must fix before product release
- -- Medium : Should fix when time permits
- --Low : would like to fix but product can be release
-

- **Defect management process** focuses on:
- --preventing defects.
- --catching defects as early
- --minimizing impact of defects

Defect Management Process

- 1. defect prevention
- 2. deliverable base line
- 3. defect discovery
- 4. defect resolution
- 5. process improvement

- **<u>1. defect prevention:</u>**
- -- a. identify critical risks:
- -- missing key requirements
- -- H/w malfunctioning
- -- performance is poor
- -- b. estimate expected impact:
- -- E= P * I I→ impact
- P → probability of risk to become real
- -- c. minimizing expected impact:
- --eliminate risk
- -- reduce probability of risk
- -- reduce the impact : eg : disaster recovery plan for reducing impact of data loss

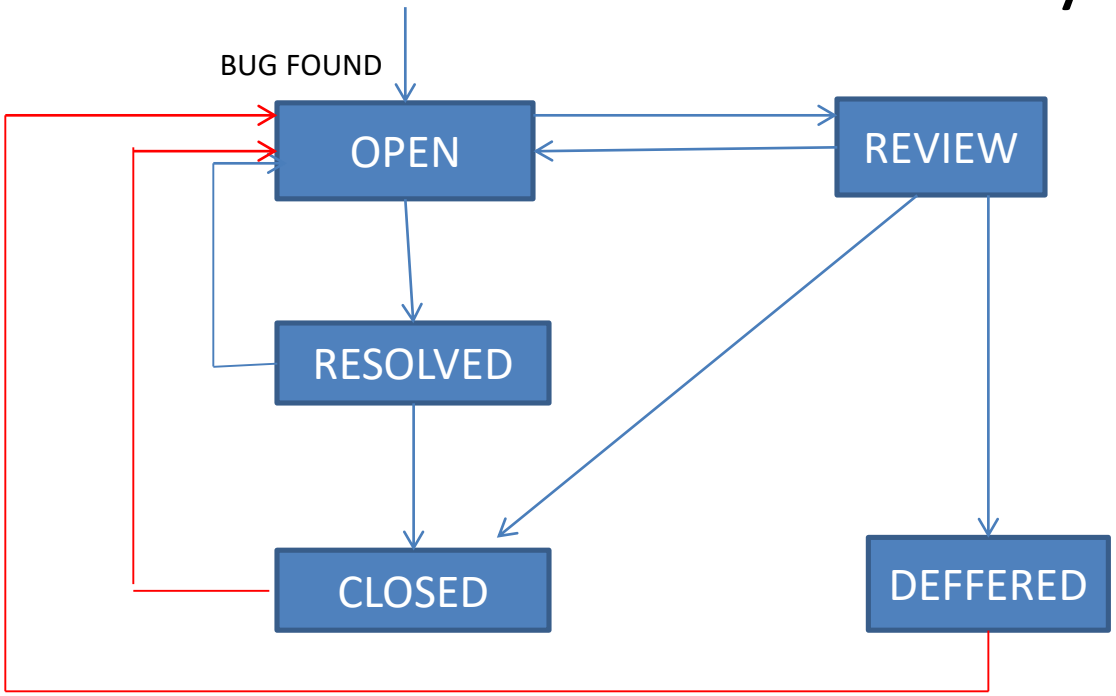
- **2. deliverable base line:**
- deliverables is a base lined when it reaches a predefined mile stone in its development
- Milestone involve transferring the product from one stage to next
- Baseline refers to a predefined benchmark which when reached determine next step to be taken.
- As work product moves from one stage to next , defect in the deliverable have large impact on rest of system and making changes become expensive.
- Error caught before deliverable baselined would not be considered defects.
- Deliverables should be baselined when defect in the deliverables have impact on deliverables other people are working on .
- If deliverables is baselined and defect found then it can not proceed further till bug is fixed

- **3. defect discovery:**
- --defect is said to be discovered when documented and acknowledged as valid defect from developer side
- --find defect
- --report defect
- --acknowledge defect
- Organization should predefine defects by category.
  - identify defect and then get an agreement that they are defects.
  - The objective is to minimize conflicts over the validity of defects.

- **4. defect resolution:**
- --done by developer
- -- prioritize defect : critical , major , minor
- --schedule to fix: based on priority, defect fix should be scheduled
- --fix defect: correcting deliverables .
- --report resolution: developer response back to tester
- **5. process improvement:**
- --identify and analysis of process in which a defect originated
- -- improve process to prevent future occurrence of similar defects.

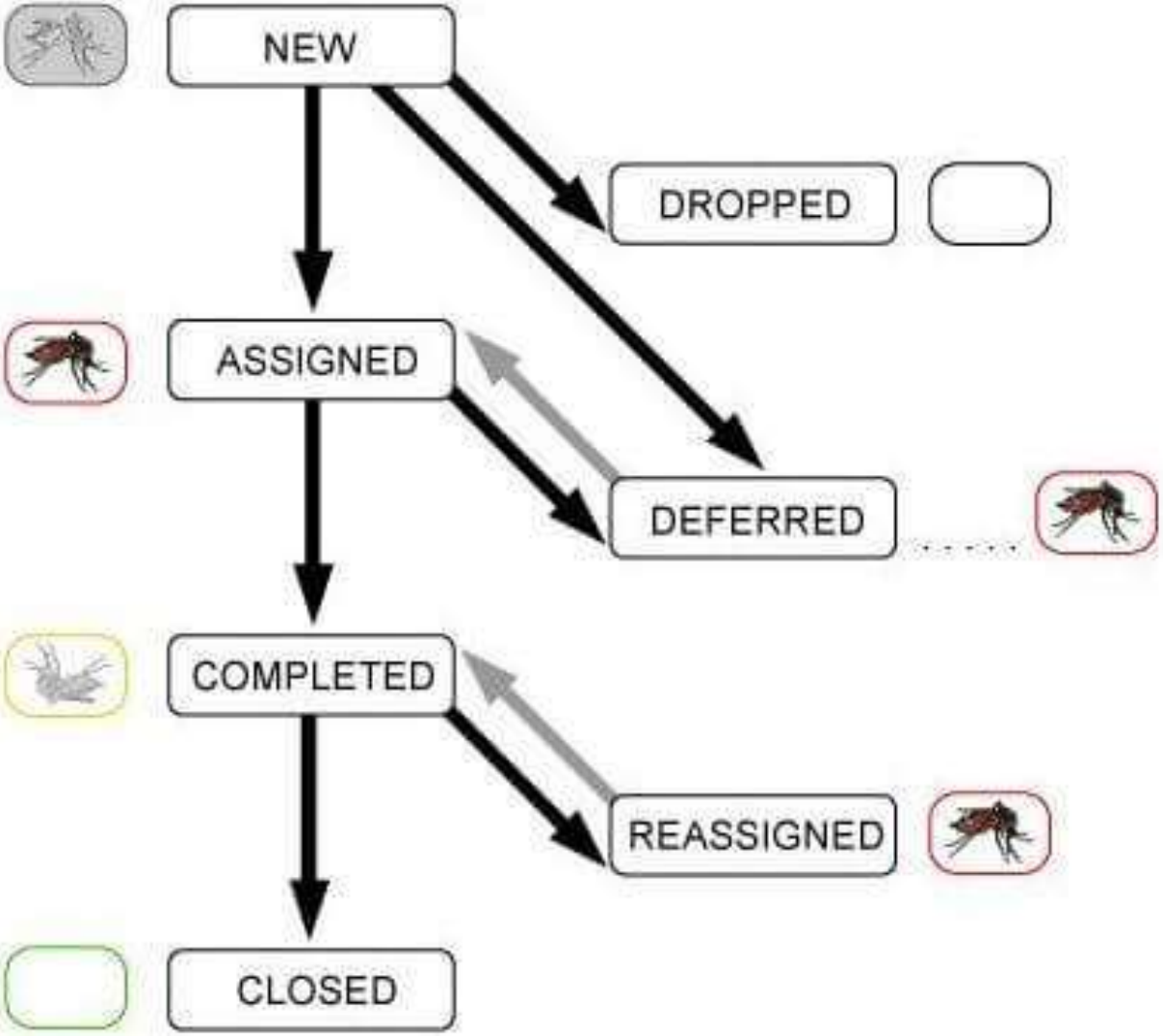
bug life cycle

- It start when defect is found and end when a defect is closed.
- Bug has different states in the life cycle.

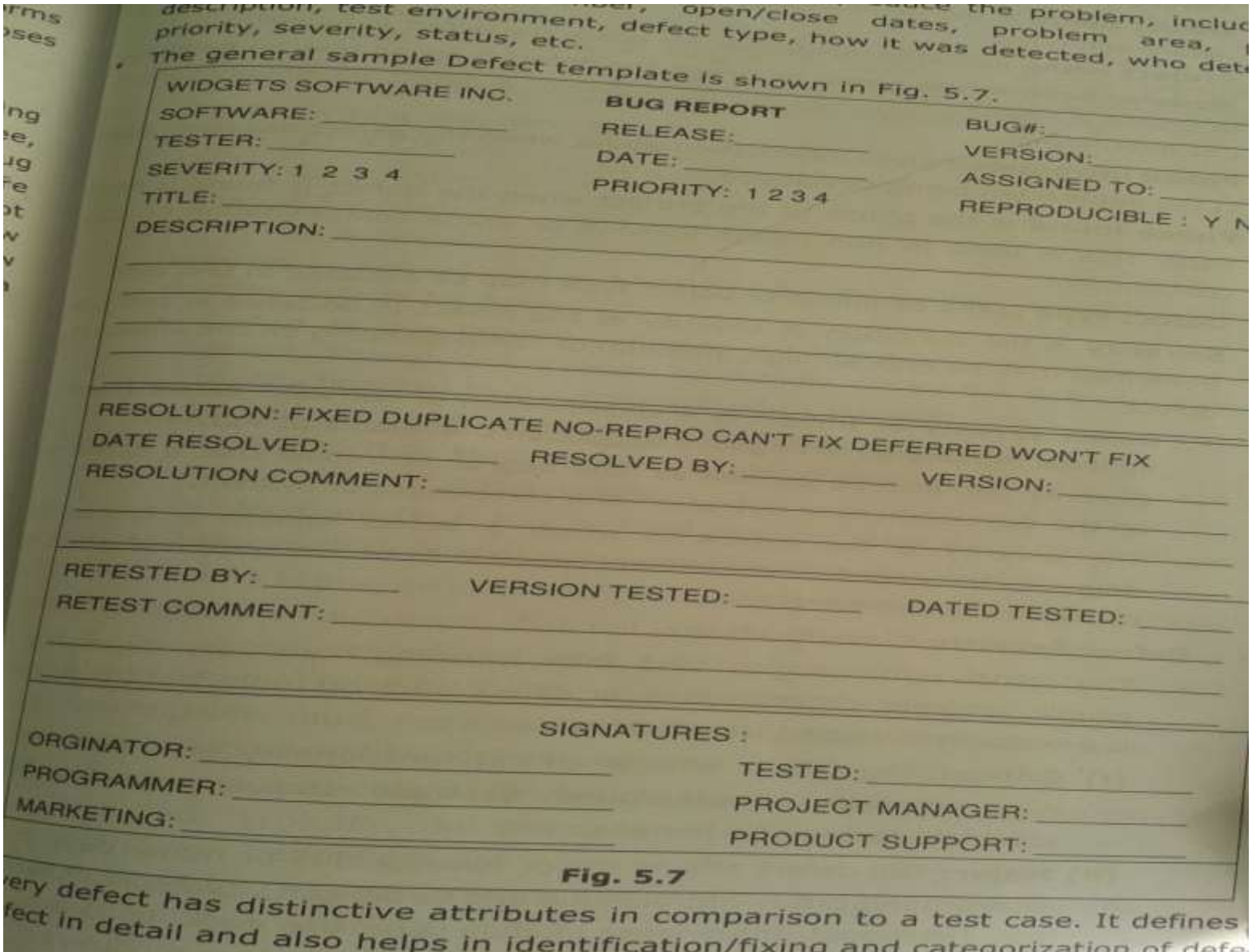


- OPEN: when bug found by tester
- OPEN-RESOLVED : once programmer fixes the code
- RESOLVED-CLOSED : tester close the bug after verifying
- OPEN-REVIEW : CCB decide whether the bug should be fixed or not
- REVIEW- CLOSED : CCB decide that bug is not a real problem
- REVIEW-DEFERRED: CCB decide that bug should be fix in future , but not for this release of the s/w
- REIEW-OPEN: CCB decide that bug should be fix.
- RESOLVED – OPEN : tester find bug is not fix
- CLOSED – OPEN : bug may occur in future
- DEFERRED – OPEN: bug has to be fix in future.

BUG LIFE CYCLE



Defect template



priority, severity, status, etc.

The general sample Defect template is shown in Fig. 5.7.

WIDGETS SOFTWARE INC. **BUG REPORT**

SOFTWARE: _____ RELEASE: _____ BUG#: _____

TESTER: _____ DATE: _____ VERSION: _____

SEVERITY: 1 2 3 4 PRIORITY: 1 2 3 4 ASSIGNED TO: _____

TITLE: _____ REPRODUCIBLE : Y N

DESCRIPTION: _____

RESOLUTION: FIXED DUPLICATE NO-REPRO CAN'T FIX DEFERRED WON'T FIX

DATE RESOLVED: _____ RESOLVED BY: _____ VERSION: _____

RESOLUTION COMMENT: _____

RETESTED BY: _____ VERSION TESTED: _____ DATED TESTED: _____

RETEST COMMENT: _____

SIGNATURES :

ORGINATOR: _____ TESTED: _____

PROGRAMMER: _____ PROJECT MANAGER: _____

MARKETING: _____ PRODUCT SUPPORT: _____

Fig. 5.7

ery defect has distinctive attributes in comparison to a test case. It defines
fect in detail and also helps in identification/fixing and categorization of defe

Defect Template

- **4 sections:**
- **1.section : tester report**
- -Defect id , Defect name , Date , Tester
- -Severity (1,2,3,4), Priority (1,2,3,4),
- -Description
- **2.section: Developer report**
- -resolution(defect state)
- -resolved date , version , resolved by , comment
- **3.section : retesting report**
- - retested by , version tested , comment
- **4.section: Signature**

Severity : 1.Critical 2.Major 3.medium 4.low
Priority : 1. Critical 2.high 3.medium 4.low

- Example:
- 1.s/w crashes as soon as you start
- Severity -1 priority – 1

- 2. tester thing that button should be moved little down on the page.
- Severity – 4 Priority – 4

- 3.misspelling in the setup instruction.
- Severity – 3 Priority - 2

Estimate Expected impact of a defect

- What loss will be suffered by a user in case of occurrence of risk.
- Estimation my be done by different methods to find probability of risk occurrence. send impact when it become real.
- Some organization categories risk impact as high , medium and low or calculate on the basis of money value of the loss

1. Way to handle risk

- **1.accept the risk as it is :**
- Some risk may not have any solution like natural disasters.
- These type of risk may be accepted by organization
- They make fallback arrangement , but may not define ways to eliminate risks.
- This make team well prepared to accept the risk
- Customer are given information about probable failures and effect of such failure. (called accident prone zone)

- **2. by passing the risk:**
- If approach is very risky to the users , mgmt decide to by pass.

  By pass risk by avoiding the particular approach.
- It required when the risk faced by user cannot be accepted or no action can be taken to reduce probability or impact.

2. Minimize risk impact

- **1.eliminate risk** :taking steps to remove risk from root cause.
- 2. **mitigation of risk: action** taken by organization to minimize possible damage .
- **3.detection ability improvement**: people are aware of the risk , they can be well prepared to handle
- **4.contingency planning**: actions initiated by an organization , when preventive and corrective actions fails and risk actually occurs

Techniques of finding defects

- **1. static techniques:**
- Testing is done without physically executing a program.
- Eg:reviews
- **2.dynamic techniques:** physical execution
- Eg : test cases execution
- **3.operational techniques:** defect is found as a result of a failure.
- While using final software , customer found that software is not working or fail.

Defect reporting

- Discovered defect must be brought to the developers attention.
- Guidelines while reporting :
- 1.be specific:
- --don’t say anything which add confusion.
- --in case of multiple paths , mention exact path
- --be detailed : provide more information

- Be objective: stick to fact and avoid emotions
- Reproduce the defect:
- -- don’t be impatient .
- --replicate it at least once more
- -- state exact test condition
- Review the report