Software Testing

Lesson 10
Test Management – Incident Management
V1.0

Uwe Gühl

Winter 2013 / 2014

Contents

- Test Management – Incident Management
  - Terms
  - Defect reports
    - Rules
    - Attributes
  - Incident management
    - Bug life cycle
    - Tasks

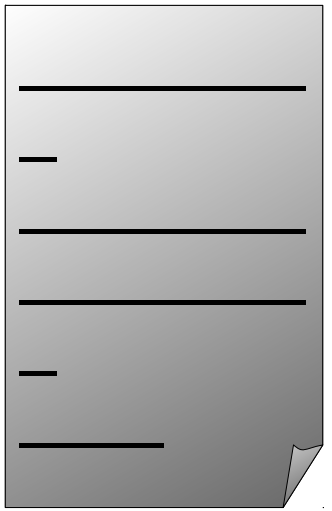
Definitions of Terms

- Incident [ISTQB-GWP12 – after IEEE 1008]
  - Synonym: Deviation
  - Any event occurring that requires investigation.

Definitions of Terms

- An incident must be investigated and may turn out to be a defect.

Hint:
In practice typically we talk about
“defects”, and
“defect management”.
ISTQB typically uses the terms
“incident”, and
“incident management”.



Defect Report

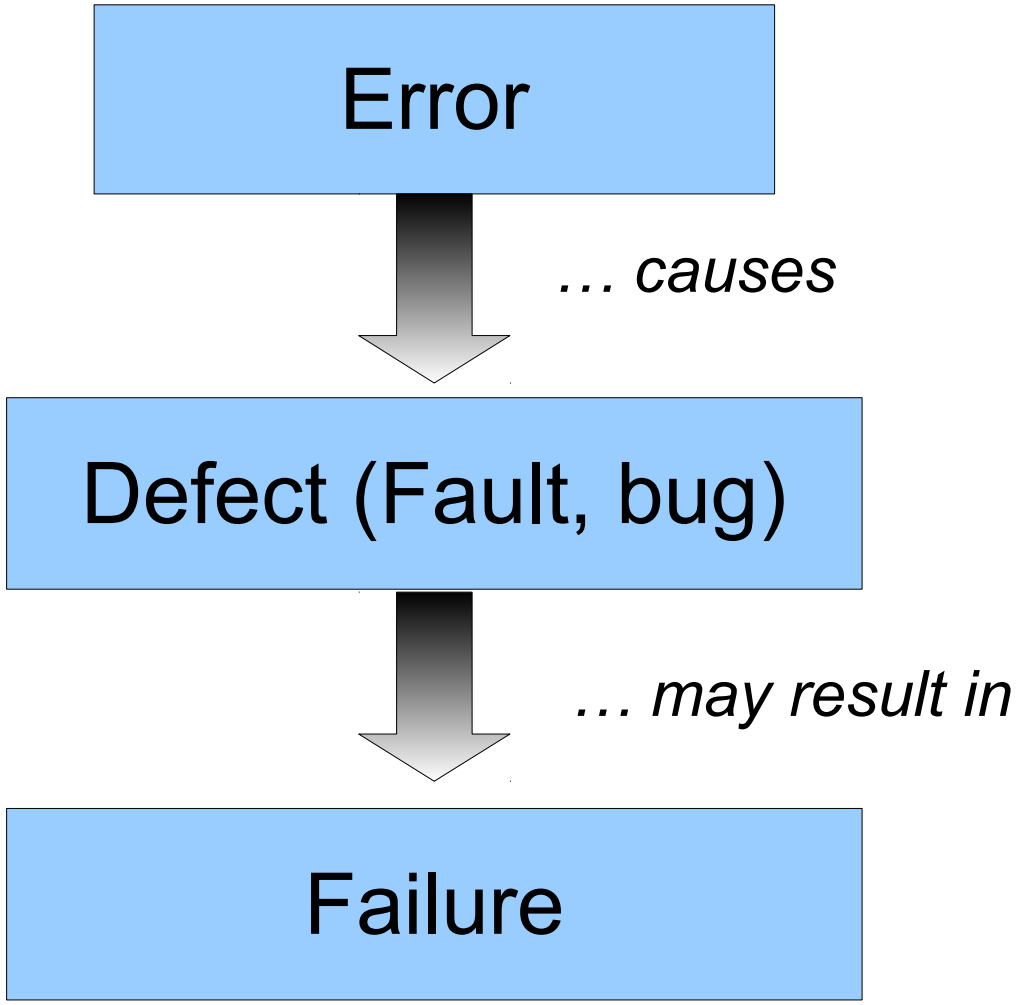
Definitions of Terms

- Defect [ISTQB-GWP12]
  - Synonyms: Bug, fault, problem.
  - A flaw in a component or system that can cause the component or system to fail to perform its required function, e.g. an incorrect statement or data definition.
    A defect, if encountered during execution, may cause a failure of the component or system.

Definitions of Terms

- Defect – More definitions:
  - Something Is Definitely Wrong With The Product [KBP01].
  - An error in construction of a product or service that renders it unusable; an error that causes a product or service to not meet requirements [IQRC14].
  - In Wikipedia "*Defect*" refers to "*Software bug*", "A failure of computer software to meet requirements.
    A "*software bug*" is the common term used to describe an error, flaw, mistake, failure, or fault in a computer program or system that produces an incorrect or unexpected result, or causes it to behave in unintended ways" [Wik14].

Definitions of Terms

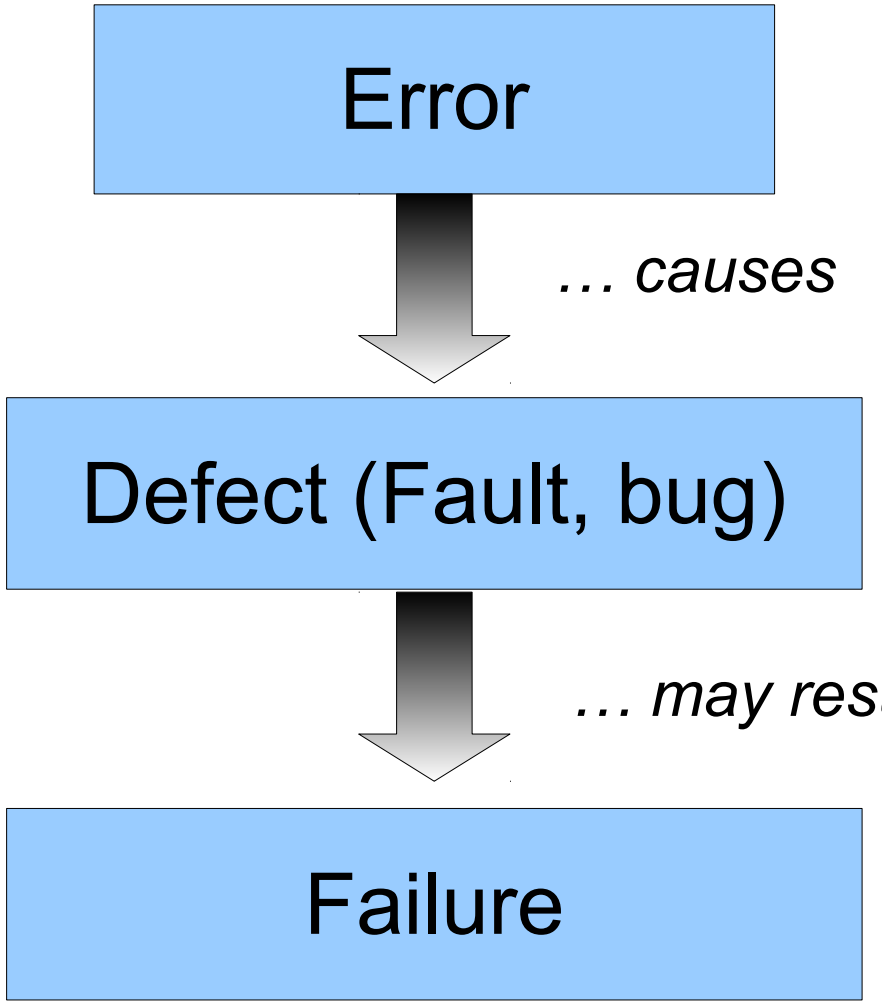


Error [ISTQB-GWP12, after IEEE 610]
A human action that produces an incorrect result.

Defect [ISTQB-GWP12]
A flaw in a component or system that can cause the component or system to fail.

Failure [ISTQB-GWP12]
Deviation of the component or system from its expected delivery, service or result

Definitions of Terms



Error [ISTQB-GWP12, after IEEE 610]
s an

Could be detected during static testing

Defect [ISTQB-GWP12]
A flaw in a component or system that can cause the component or

Could be detected only through dynamic testing

ailure [ISTQB-GWP12]
Deviation of the component or system from its expected delivery, service or result

Definitions of Terms

- Defect report [ISTQB-GWP12 after IEEE 829]
  - Synonym: Problem report, bug report.
  - A document reporting on any flaw in a component or system that can cause the component or system to fail to perform its required function.
- Incident report [ISTQB-GWP12 after IEEE 829]
  - Synonym: Deviation report.
  - A document reporting on any event that occurred, e.g. during the testing, which requires investigation.
  - The ‘Standard for Software Test Documentation’ [IEEE829] covers the structure of an incident report.

Definitions of Terms

Two important attributes of a defect in a defect report describe the failure severity, and the urgency to fix it:

- Severity [ISTQB-GWP12 – after IEEE 610]
  - The degree of impact that a defect has on the development or operation of a component or system.
- Priority [ISTQB-GWP12]
  - The level of (business) importance assigned to an item, e.g. defect.

Definitions of Terms

- Defect management [ISTQB-GWP12 – after IEEE 1044]
  - Synonym: Problem management
  - The process of recognizing, investigating, taking action and disposing of defects.
    It involves recording defects, classifying them and identifying the impact.
- Incident management [ISTQB-GWP12 – after IEEE 1044]
  - The process of recognizing, investigating, taking action and disposing of incidents.
    It involves logging incidents, classifying them and identifying the impact.

Definitions of Terms

- Defect management tool [ISTQB-GWP12]
  - Synonym: Bug tracking tool, defect tracking tool
  - A tool that facilitates the recording and status tracking of defects and changes.
    They often have workflow-oriented facilities to track and control the allocation, correction and re-testing of defects and provide reporting facilities.
- Incident management tool [ISTQB-GWP12]
  - A tool that facilitates the recording and status tracking of incidents.
    They often have workflow-oriented facilities to track and control the allocation, correction and re-testing of incidents and provide reporting facilities.

Defect report

- **Bad** bug reports [Tat99] are
  - reports that say nothing ("It doesn't work!");
  - reports that make no sense;
  - reports that don't give enough information;
  - reports that give wrong information;
  - reports of problems that turn out to be
    - user error;
    - the fault of somebody else's program;
    - network failures
- **Good:** Wonderfully clear, helpful, *informative* bug reports.

Defect report

- Defect reports have the following objectives:
  - Provide developers and other parties with feedback about the problem to enable identification, isolation and correction as necessary.
  - Provide test leaders a means of tracking the quality of the system under test and the progress of the testing.
  - Provide ideas for test process improvement.

Defect report Rules (1/3)

- Show a defect directly to the developer.
- Describe a defect so it could be reproduced.
  Best: Step by step, use screenshots, videos.
- Describe, what you expected and what you got.
  What works and what went wrong?
- Notice contents of error messages, esp. numbers.
- Report the symptoms
  - **Must:** What are actual facts
    "I was at the computer and this happened".
  - **Could:** What are speculations, your ideas as proposal
    "I think the problem might be this". [Tat99]

Defect report Rules (2/3)

- Try to work around for intermittent faults and inform about version, operating system, etc.
  - Try other machines, web browsers, screen resolution;
  - Does it depends on size of files you use, other programs you use parallel?
- Try to help that the defect could be fixed
  - Provide extra information on request like version numbers,
  - Special activities, so that developer could locate the defect. [Tat99]

Defect report Rules (3/3)

- Write clearly and as neutral as possible
  - *Be specific*. **Not:** "I selected Load"
    Better: "I clicked on Load", or "I pressed Alt-L".
  - *Be verbose.*
    If you write one sentence only, developer must ask and ask.
  - *Be careful of pronouns.*
    Not: "I started FooApp. It put up a warning window.
    I tried to close it and it crashed."
    Better: "I started FooApp, which put up a warning window.
    I tried to close the warning window, and FooApp crashed."
  - *Read what you wrote*.
    Try to reproduce a listed sequence of actions yourself.
- Don't joke

[Tat99]

Defect report Attributes

- Details of the defect report may include:
  - Author, date of issue, issuing organization.
  - Test item (configuration item).
  - Environment (Operating system, web browser, etc.)
  - Description of the defect to enable reproduction
    - Which test cases, which test steps, which test data?
    - Screenshots.
    - Logs, dumps.
    - Database, used files.
  - Expected and actual results.

Defect report Attributes

- Details of the defect report may include:
  - Severity and Priority
    In practice: Both parameter are used similar, but originally difference meanings
  - Severity – of the impact on the system
    - 1 – very high: Data loss, not usable
    -
    - ...
    - n – very low: Disfigurement

Defect report Attributes

- Details of the defect report may include:
  - Severity – example for definitions
    - Severity 1: Critical
      Total system outage; system upgrade failed (e.g. system does not boot); restore not possible.
    - Severity 2: Major
      Data migration too slow; excessive number of alarms; sporadic system re-starts; loss of synchronisation.
    - Severity 3: Minor
      Incomplete list of commands; documentation issues.
    - Severity 4: Non
      Cosmetic problems; not well structured printouts.

Defect report Attributes

- Details of the defect report may include:
  - Priority – Urgency to fix
    - 1 – very high: Fastest fixing necessary
    - ...
    - n – very low: Place back handling:
      Defect could be tolerated; possible solution: Listing in Release Notes as „open points / proposals“
    - Special status: Defect must not be fixed
  - Alternative: Control priority with “Planned fix date”.

Defect report Attributes

- Details of the defect report may include:
  - Status of the defect
    Typical: New, open (in progress), fixed, re-test (ready for re-test), closed.
  - Software or system life cycle process
    - in which the defect was observed.
    - in which a fix is expected (planned fix date).
    - in which a fix is delivered (fix date).
  - Change history, such as the sequence of actions taken by project team members with respect to the defect to isolate, repair, and confirm it as fixed.

Defect report Attributes

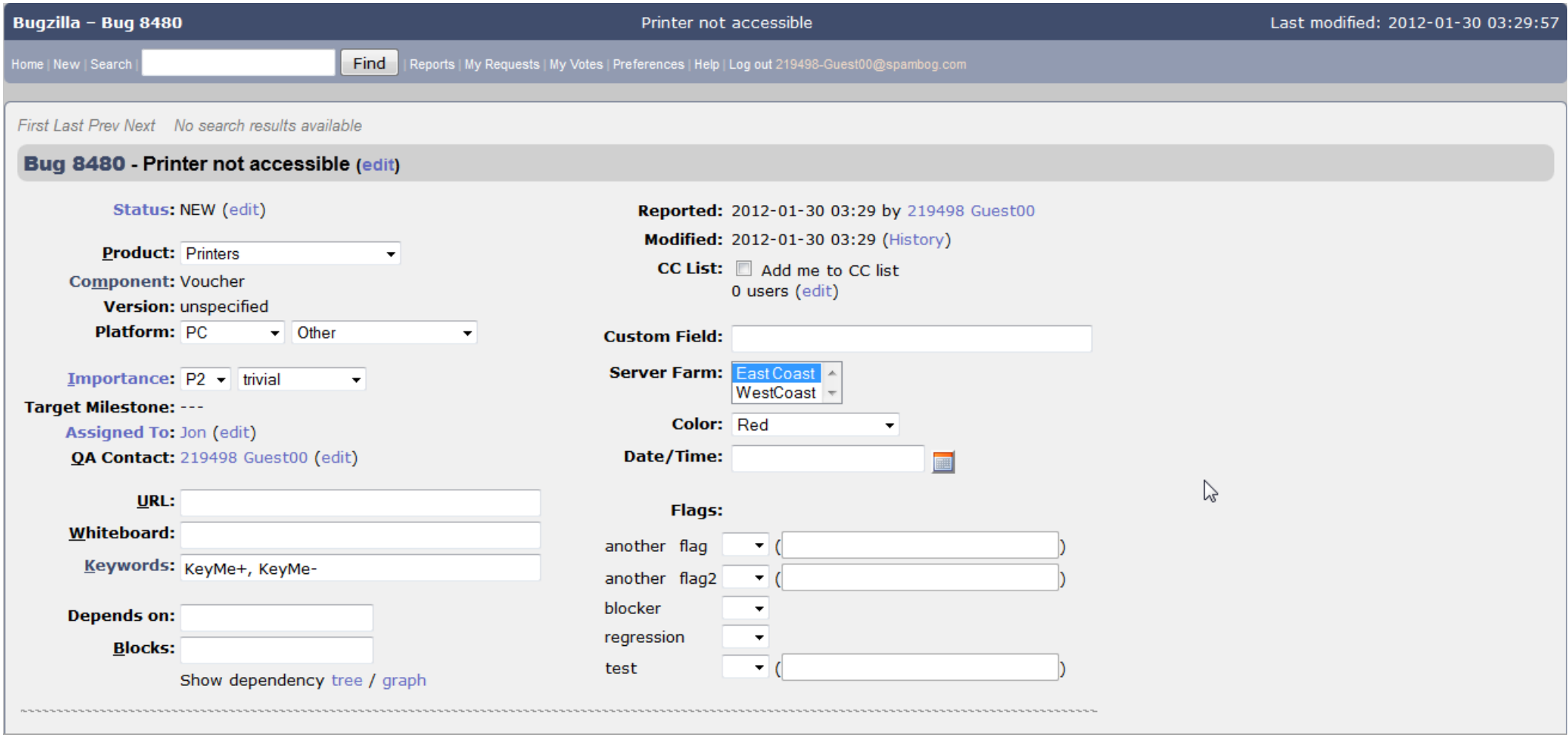
- Details of the defect report may include:
  - Conclusions, recommendations and approvals.
  - References,
    including the identity of the test case specification that revealed the problem.
  - Global issues,
    such as other areas that may be affected by a change resulting from the defect.
  - Scope or degree of impact on stakeholder(s) interests.

Defect reports

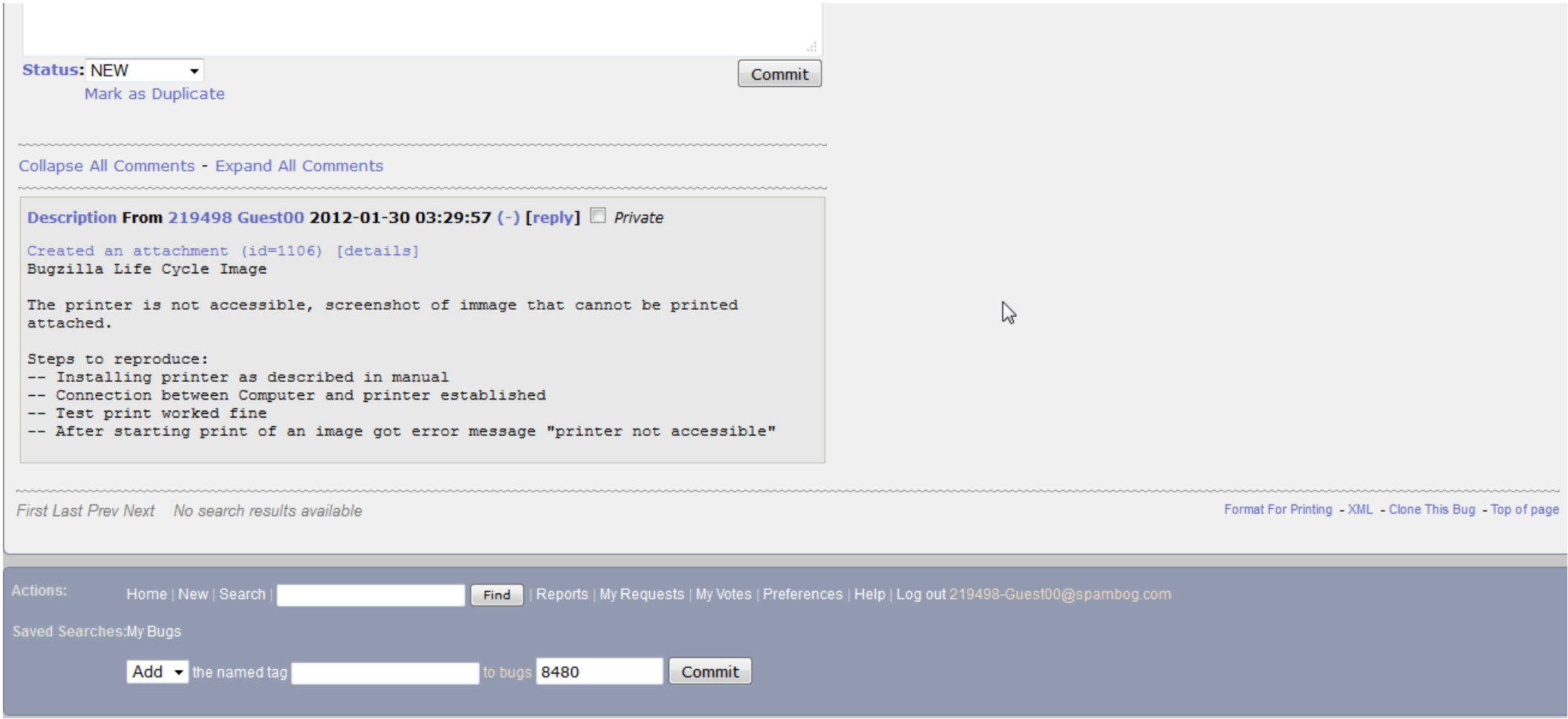
- How to write reports? Example
  - Step 10: Enter zip code ✓
  - Step 20: Do not enter city name ✓
  - Step 30: Verify data base entry zip code ✓
  - Step 40: Verify data base entry city name ✗
    Nullpointer exception: Window with unreadable message appears, but could be closed.
    Proposal: A check for city name before sending the data to the server.
  - Step 50: Error message dis

Important: Your task is to report the bug in the best way so it could be fixed – Ideas for reasons and solutions are **really only optional**

Defect reports Example (1/2)



Defect reports Example (2/2)



Incident Management

Motivation

- One goal of testing: Finding defects.
- Discrepancies between actual and expected outcomes ⇨ Logging as incidents.
  ⇨ May turn out to be a defect.
- How to handle appropriate actions?
  ⇨ Incident management / Bug life cycle.

Incident Management

- Incident management
  - Track incidents and defects
    - from discovery and classification
    - to correction
    - to confirmation of the solution.
  - Establish an incident management process.
  - Define rules for classification.

Incident Management

When?

- Incidents may be raised during
  - development,
  - review,
  - testing, or
  - use of a software product.

Incident Management

Concerning what?

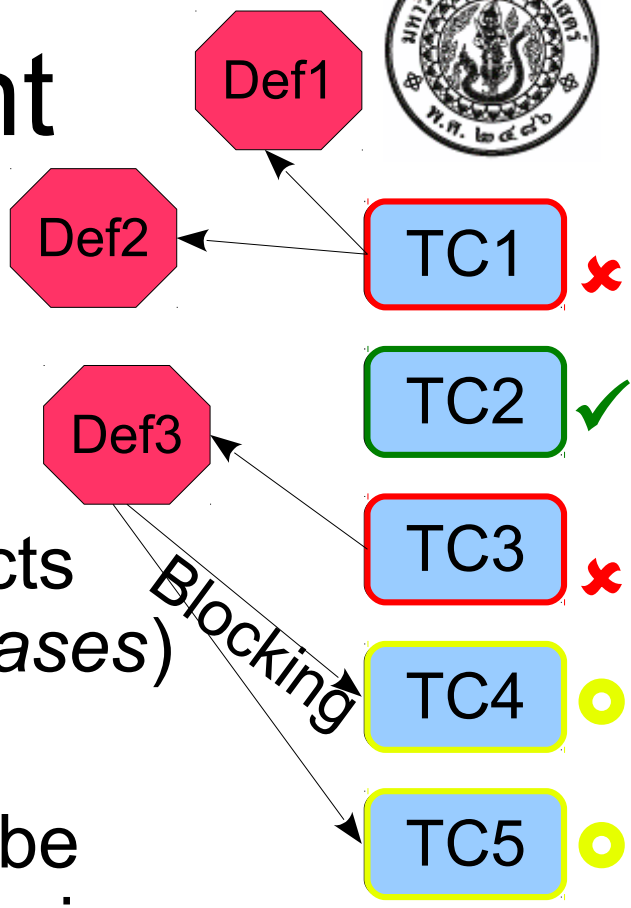
- Incidents may be raised for
  - issues in code or the working system, or
  - any type of documentation including
    - requirements,
    - development documents,
    - test documents,
    - user information such as “Help”,
    - installation guides.

Incident Management

What are possible root causes?

- Distinguish
  - specification fault like wrong requirements,
  - software defect,
  - environment failure,
  - interface defect,
  - error in the test case or test scenario,
  - error in test data.

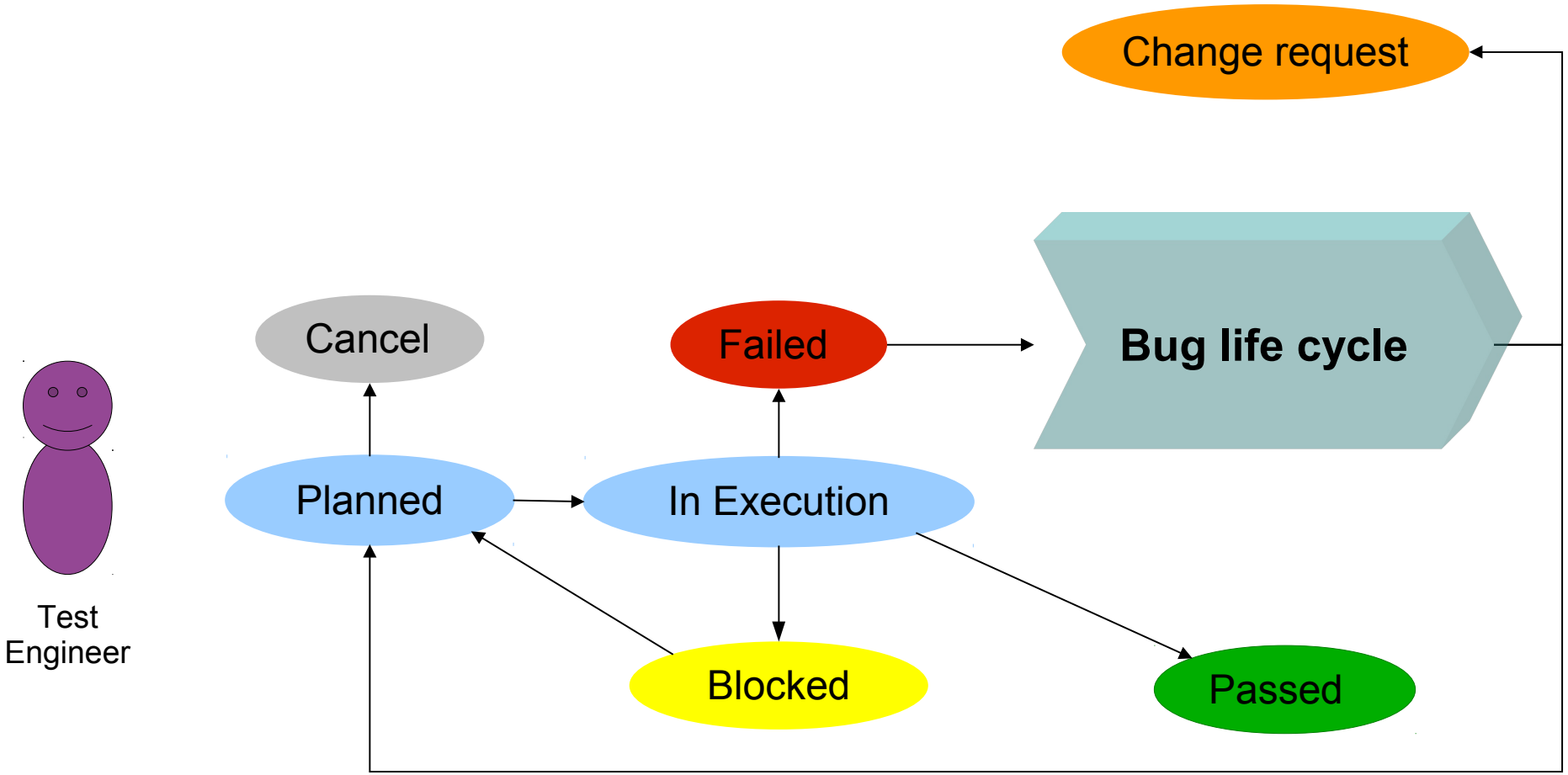
Incident Management



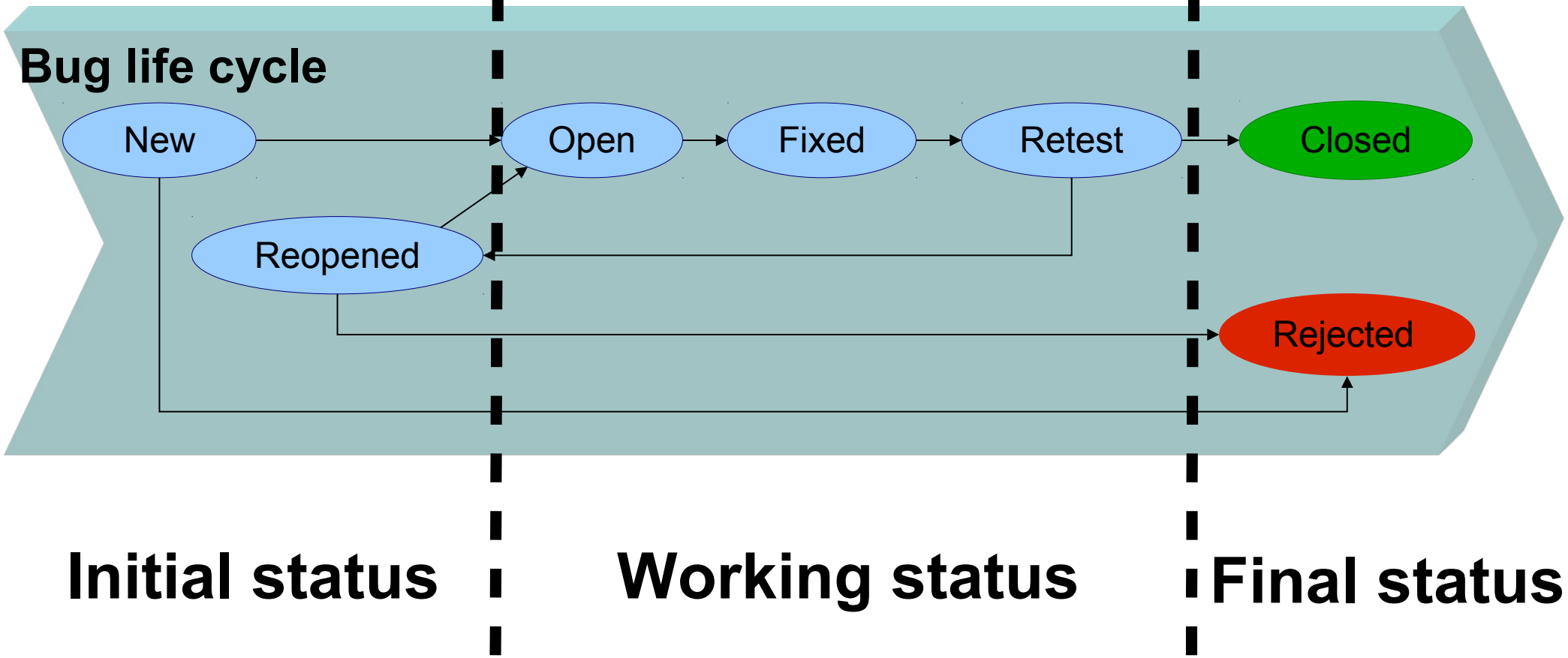
- Defects and test cases
  - Relation is `m:n`
  - A test case could have several defects (*Hint: That's why design small test cases*)
  - Hidden defects
    If an execution of a test case has to be stopped, possible defects in the following test steps could only be detected, after the defect is fixed and could be retested.
  - A defect could block other test cases
    Example: Interface tests.

Incident Management
Bug life cycle

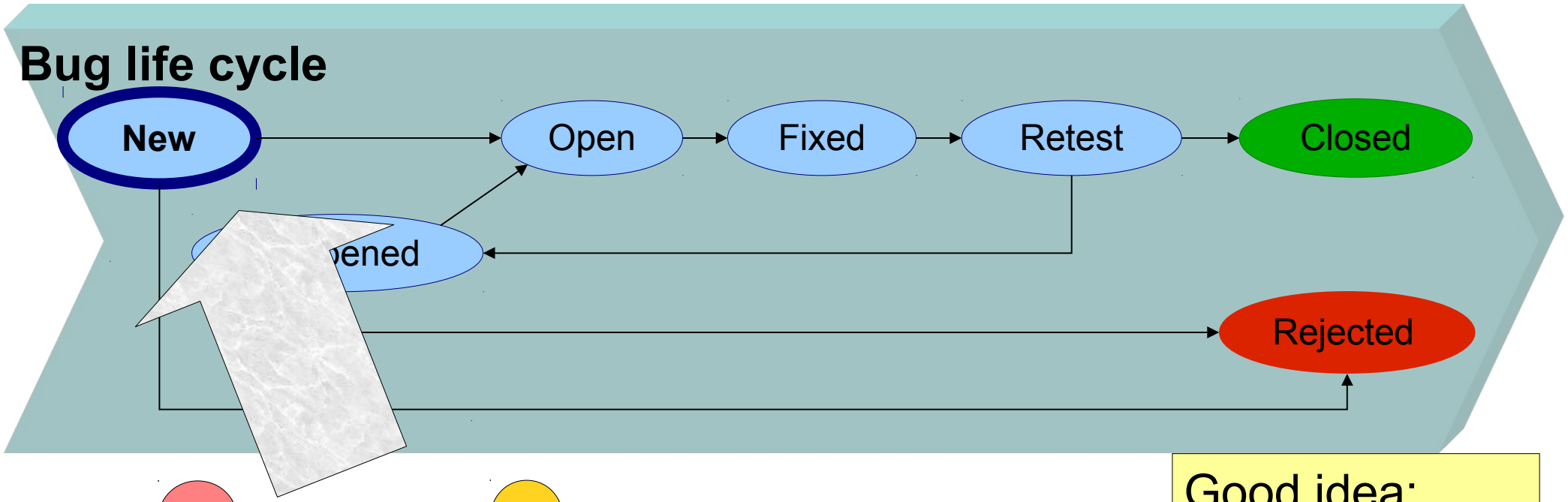
Execution of test cases and bug life cycle

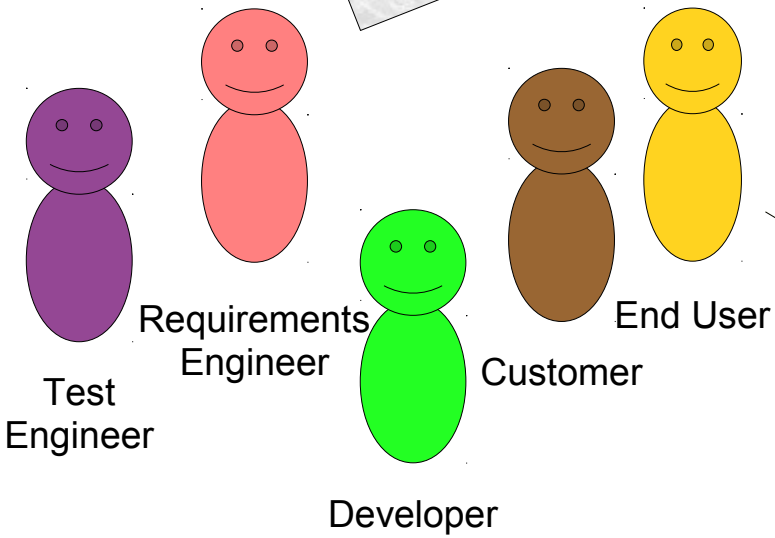


Incident Management
Bug life cycle – Example 1



Incident Management
Bug life cycle – Example 1

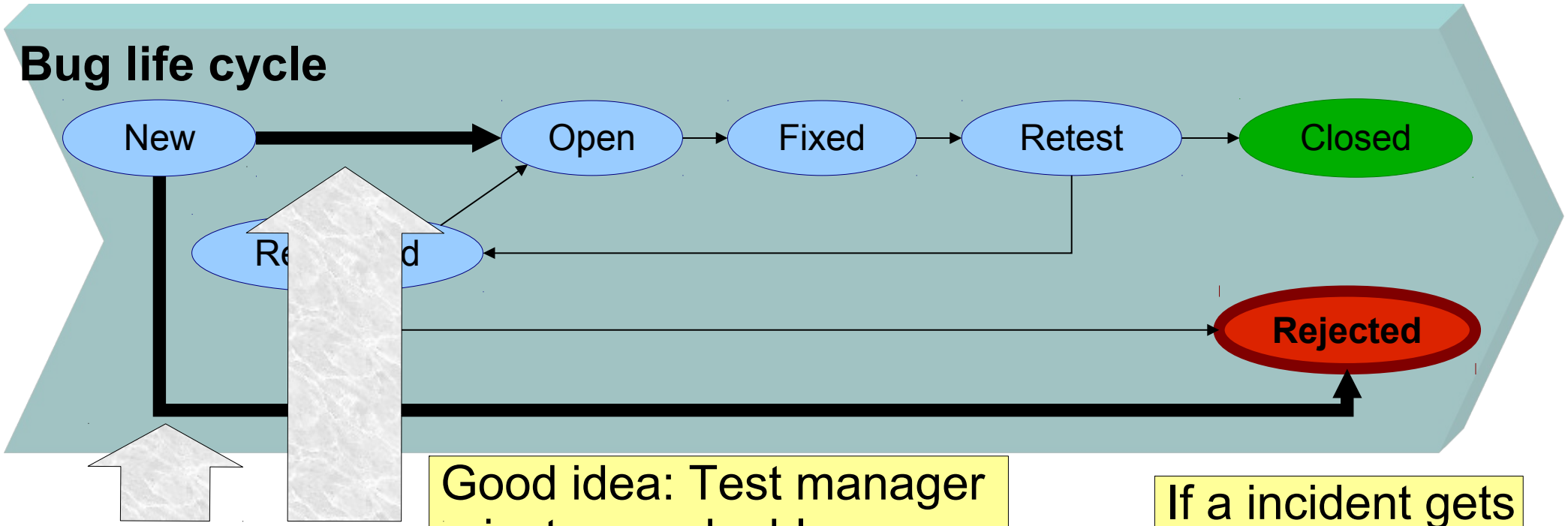




Everyone who detects an incident, may report it

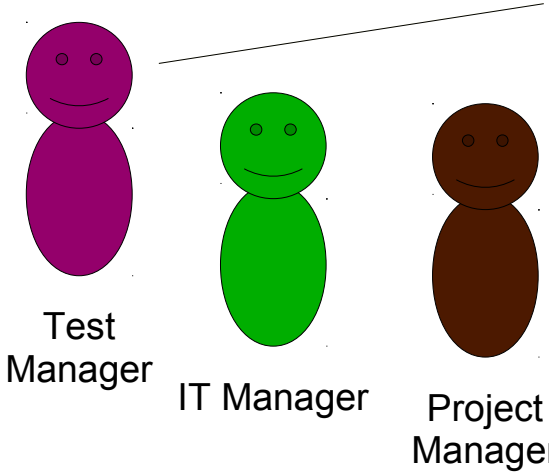
Good idea: Show / discuss it first with a developer before entering

Incident Management
Bug life cycle – Example 1



Good idea: Test manager rejects e.g. doubles

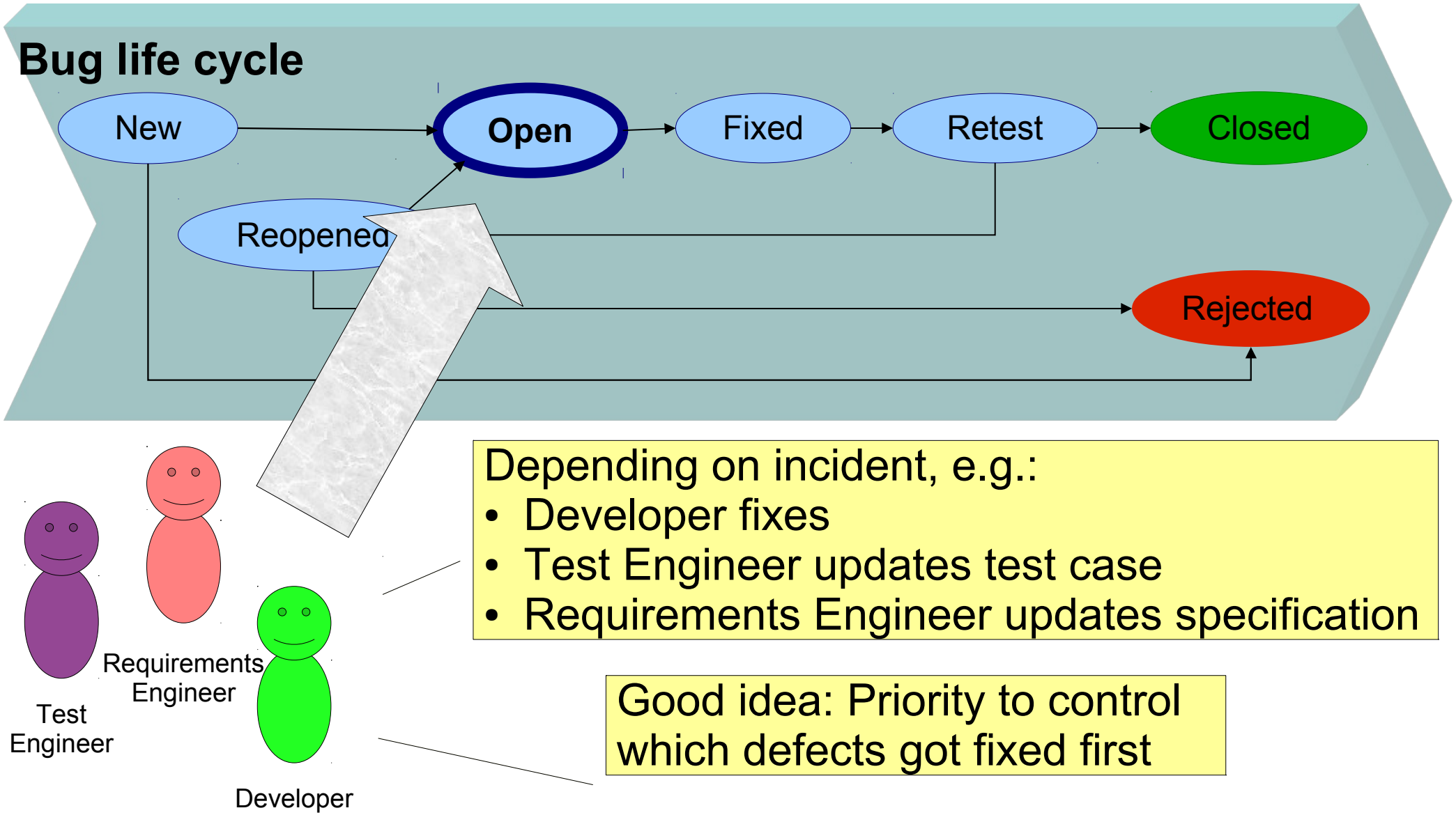
If a incident gets rejected, it is in a final state



Possible reason for “rejected”: Not reproducible

E.g. in a daily meeting it will be decided how to proceed

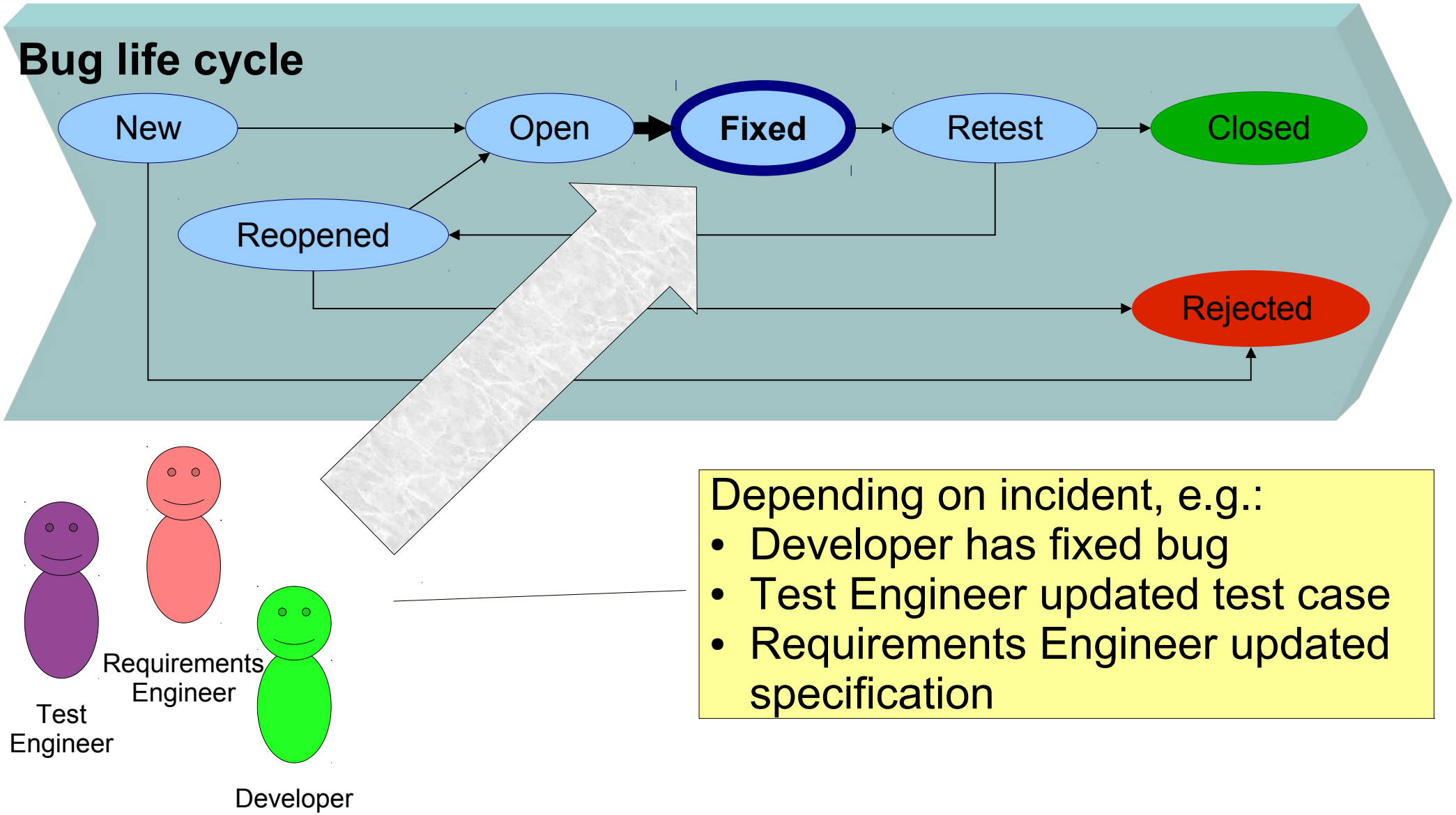
Incident Management
Bug life cycle – Example 1



Depending on incident, e.g.:
- Developer fixes
- Test Engineer updates test case
- Requirements Engineer updates specification

Good idea: Priority to control which defects got fixed first

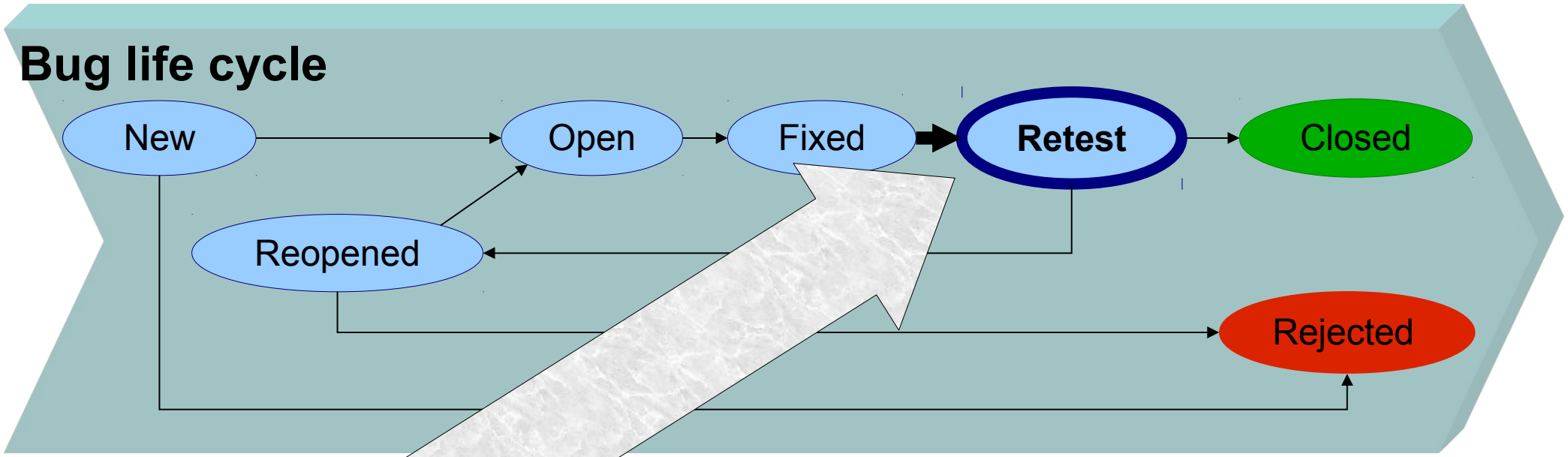
Incident Management
Bug life cycle – Example 1

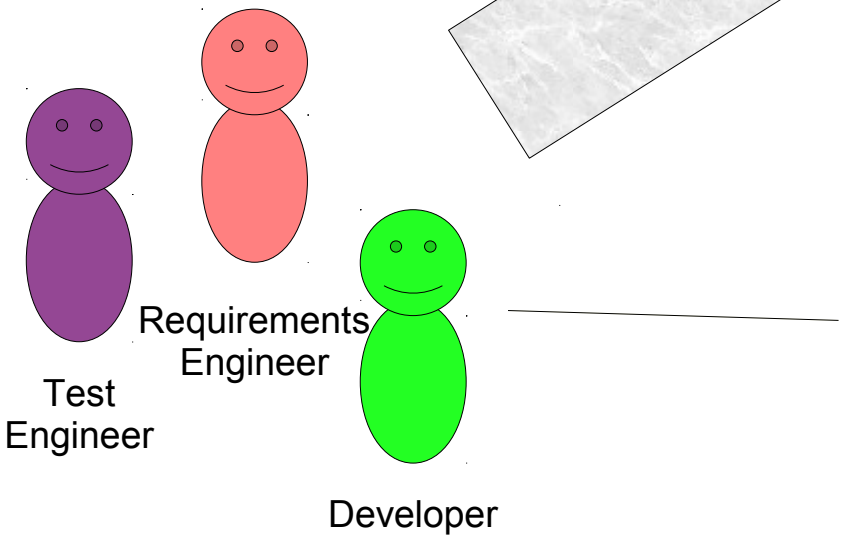


Depending on incident, e.g.:

- Developer has fixed bug
- Test Engineer updated test case
- Requirements Engineer updated specification

Incident Management
Bug life cycle – Example 1

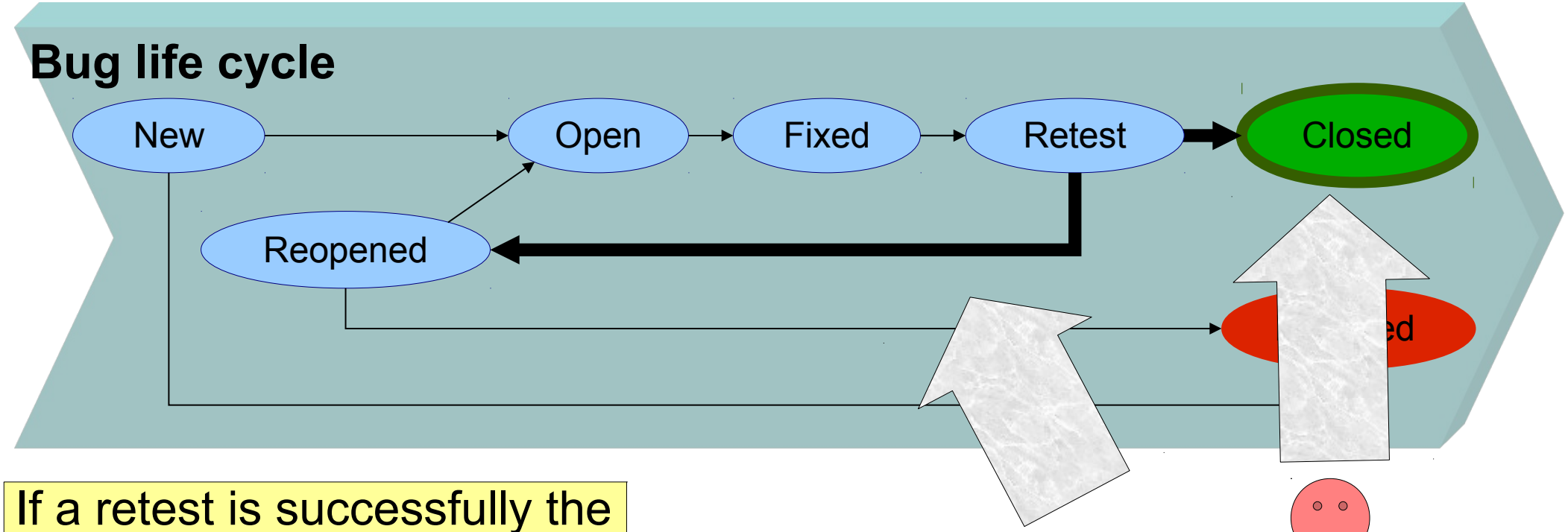




Depending on incident, e.g.:
- Fix was delivered
- Test case ready for review
- Updated specification ready for review

Retest typically done by testers
→ Who reports a defect, should retest!

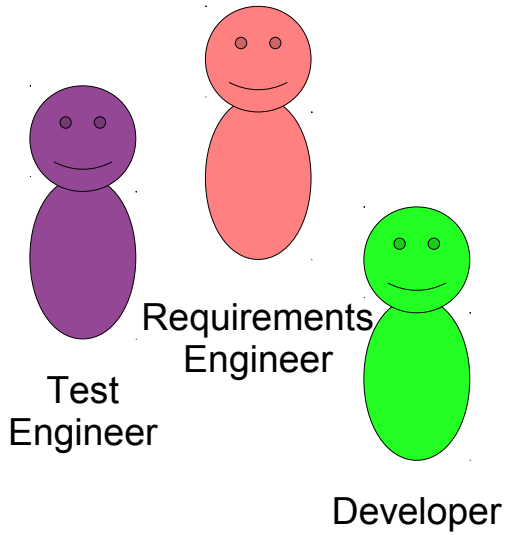
Incident Management
Bug life cycle – Example 1



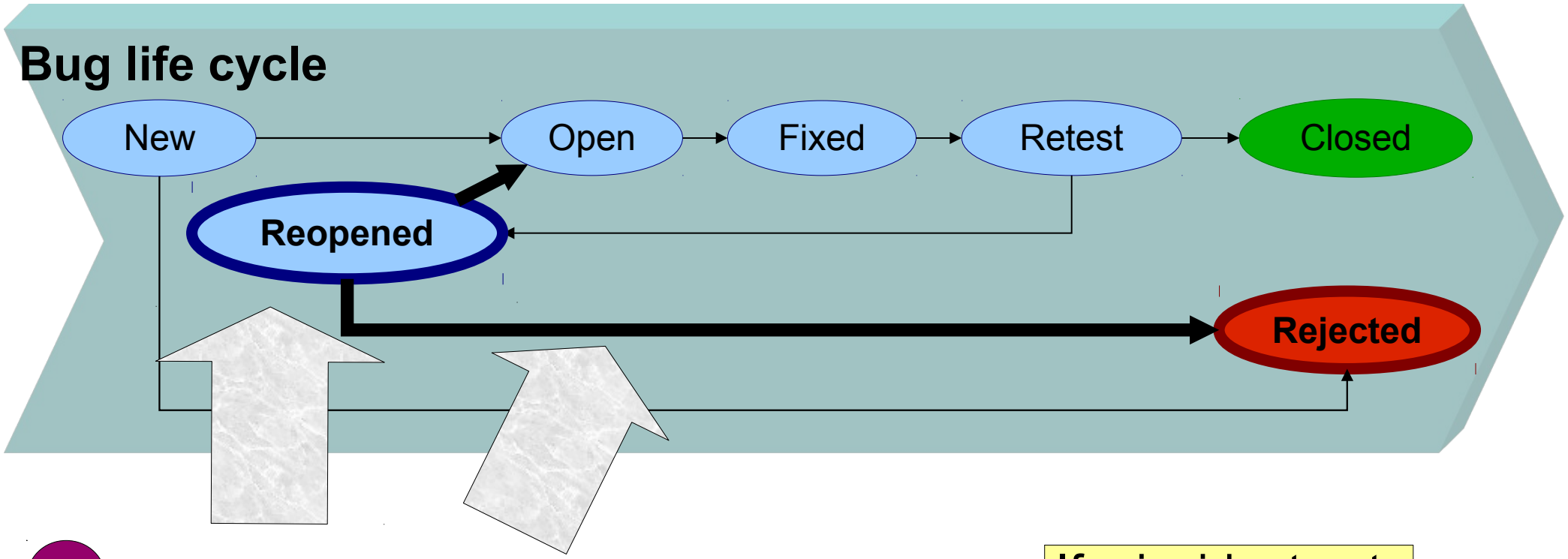
If a retest is successfully the incident could get closed, otherwise it gets reopened.

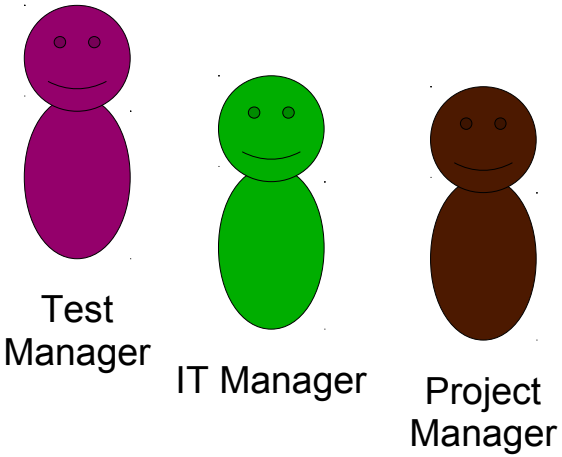
Closed is a final state

Rule: Who opens an incident, is closing it as well



Incident Management
Bug life cycle – Example 1





If a incident gets rejected, it is in a final state

"Reopened" typically handled like "new" incidents

Incident Management
Bug life cycle – Example 2

- Example 2
  Bug life cycle
  of Bugzilla
  [Wik14a]

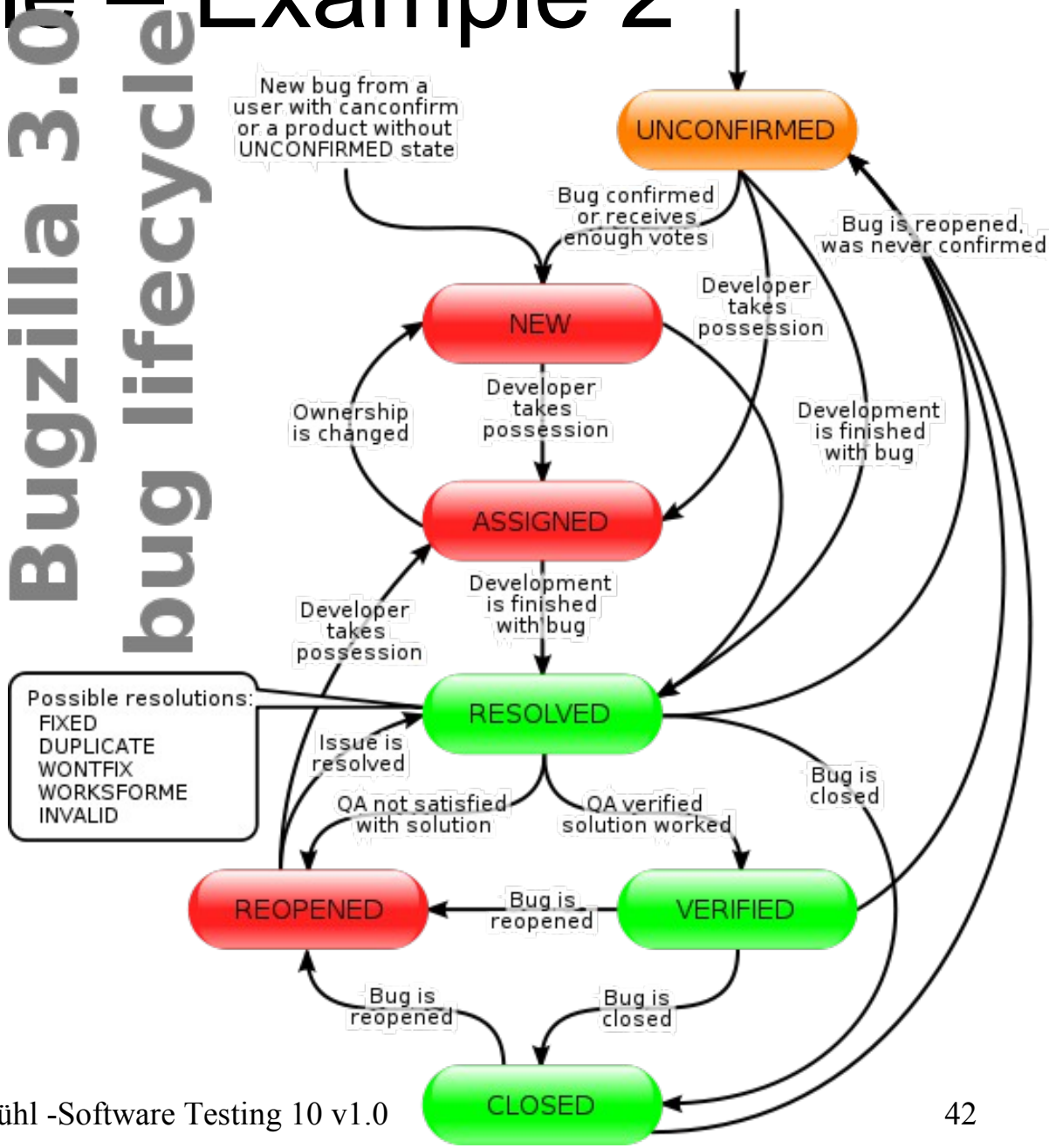


Image source: http://en.wikipedia.org/wiki/File:Bugzilla_Lifecycle_color-aqua.svg

Incident Management
Bug life cycle – Example 3

- Example 3: Bug Life Cycle [QAT14]

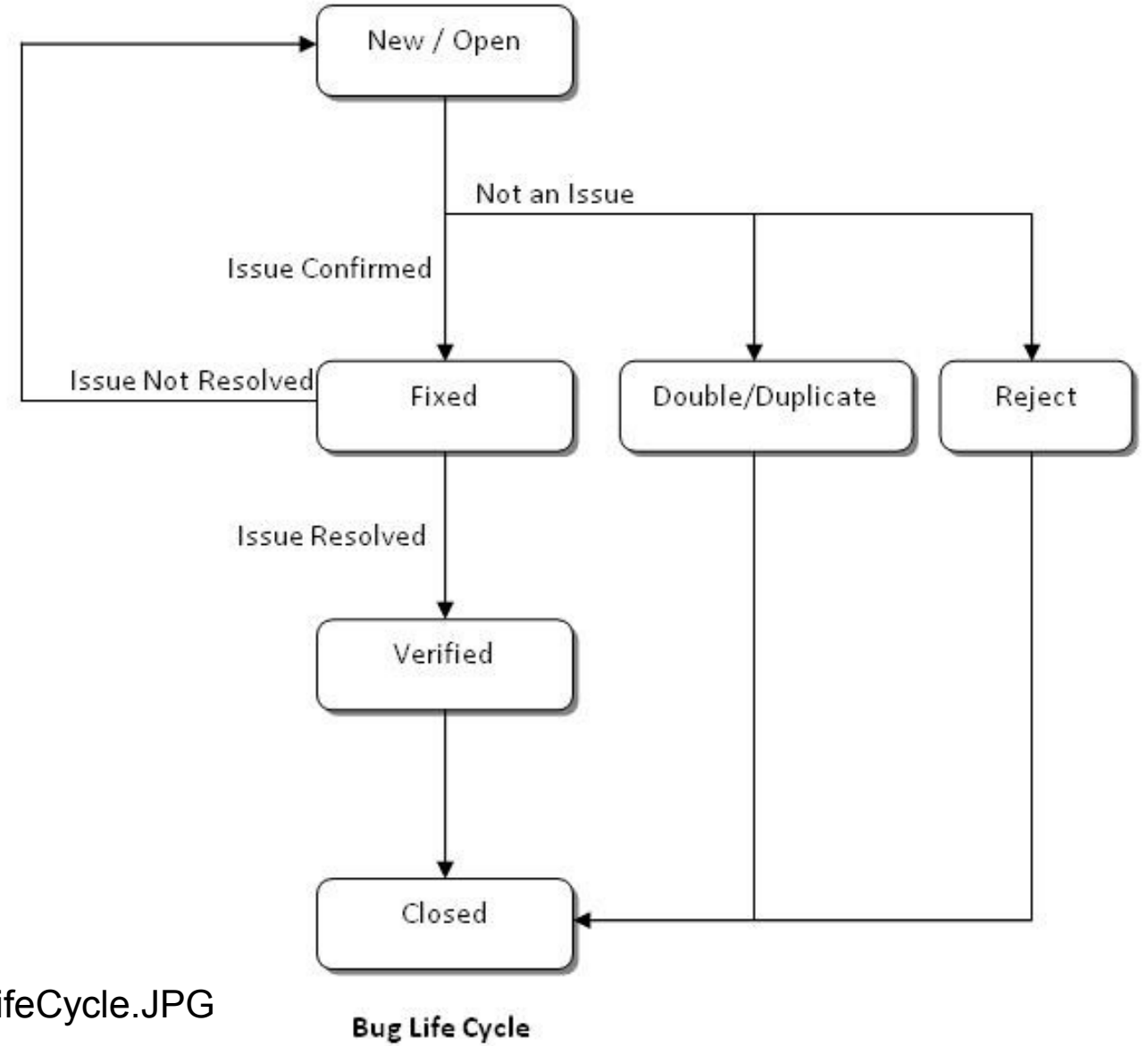


Image source:
http://www.qatutorial.com/pics/BugLifeCycle.JPG

Incident Management Tasks (1/2)

- Daily communication
  - Discussion of new defects
  - Proceeding concerning special defects
    - Defects with high severity
    - Defects with no activities for certain time
- Coordination with tester, customers, and software vendor (developers)
  - Collection and administration of defects
  - Assigning of severity and priority levels
  - Clarification of responsibilities

Incident Management Tasks (2/2)

- Monitoring of defect fixing
  - Monitoring releases:
    Which defects were fixed and delivered?
  - Organize re-testing

Sources (1/2)

- [Fen91] Fenton: Software Metrics: a Rigorous Approach, Chapman & Hall, 1991
- [IEEE610] IEEE 610.12:1990. Standard Glossary of Software Engineering Terminology.
- [IEEE829] IEEE Std 829™ (1998) IEEE Standard for Software Test Documentation.
- [IEEE1008] IEEE 1008:1993. Standard for Software Unit Testing.
- [IEEE1044] IEEE 1044:1993. Standard Classification for Software Anomalies.
- [IQRC14] International Quality and Reliability Consultants, Quality Dictionary, http://www.iqrc.com/qd_d.htm, 2014
- [ISTQB-CTFLS11] International Software Testing Qualifications Board: Certified Tester Foundation Level Syllabus, Released Version 2011, http://www.istqb.org/downloads/syllabi/foundation-level-syllabus.html

Sources (2/2)

- [ISTQB-GWP12] Glossary Working Party of International Software Testing Qualifications Board: Standard glossary of terms used in Software Testing, Version 2.2, 2012, http://www.istqb.org/downloads/glossary.html
- [KBP01] Cem Kaner, James Bach, Bret Pettichord: Lessons Learned in Software Testing: A Context-Driven Approach, John Wiley & Sons, Inc., New York, 2001
- [QAT14] QA Tutorial - QA and Testing Tutorial: Bug Life Cycle, 2014, http://www.qatutorial.com/?q=Bug_Life_Cycle
- [Tat99] Simon Tatham: How to Report Bugs Effectively, 1999, http://www.chiark.greenend.org.uk/~sgtatham/bugs.html
- [Wik14] Wikipedia: Software bug, 2014, http://en.wikipedia.org/wiki/Software_bug
- [Wik14a] Wikipedia: Bugzilla, 2014 http://en.wikipedia.org/wiki/Bugzilla