

6.



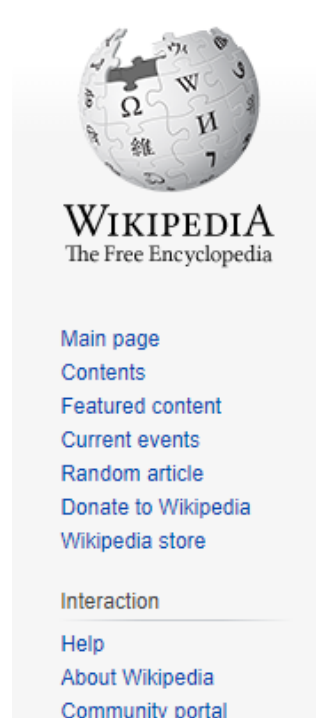
# Agenda new

- **What is a bug? Bug types.**
- Root cause analysis. Examples.
- Priority and severity of a defect
- Bug Template
- Defect's lifecycle
- Bug attachments
- Issue Management Tool
- HW6

# What is a bug?

In software engineering, *mistake metamorphism* (from Greek *meta* = "change", *morph* = "form") refers to the evolution of a defect in the final stage of software deployment. Transformation of a "mistake" committed by an analyst in the early stages of the software development lifecycle, which leads to a "defect" in the final stage of the cycle has been called 'mistake metamorphism'.

Different stages of a "mistake" in the entire cycle may be described as "mistakes", "anomalies", "faults", "failures", "errors", "exceptions", "crashes", "bugs", "defects", "incidents", or "side effects".




Article Talk Read Edit View history

## Software bug

From Wikipedia, the free encyclopedia

To report a *MediaWiki* error on Wikipedia, see *Wikipedia:Bug reports*.



This article **needs additional citations for verification**. Please help [improve this article](#) by [adding reliable sources](#). Unsourced material may be challenged and removed.  
*Find sources:* "Software bug" – news · newspapers · books · scholar · JSTOR (September 2017) (*Learn how to template message*)

A **software bug** is an error, flaw or **fault** in a **computer program** or **system** that causes it to produce an incorrect or unexpected result, or to behave in unintended ways. The process of finding and fixing bugs is termed "**debugging**" and often uses formal techniques or tools to pinpoint bugs, and since the 1950s, some computer systems have been designed to also deter, detect or auto-correct various computer bugs during operations.



# TYPES OF SOFTWARE BUGS

Based on IEEE610.12-90, the definitions of bug:

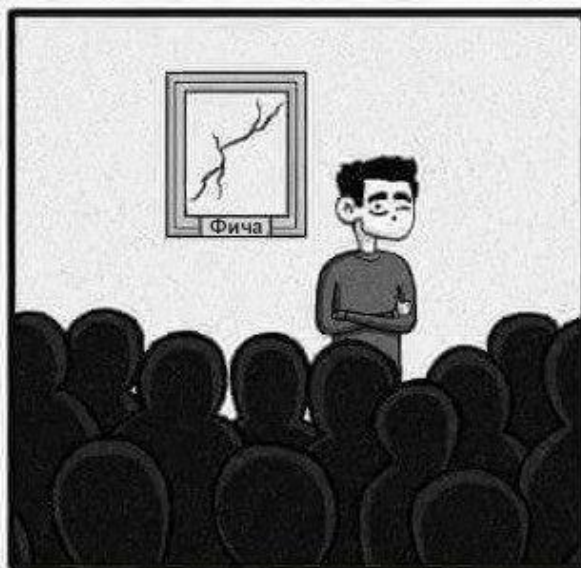
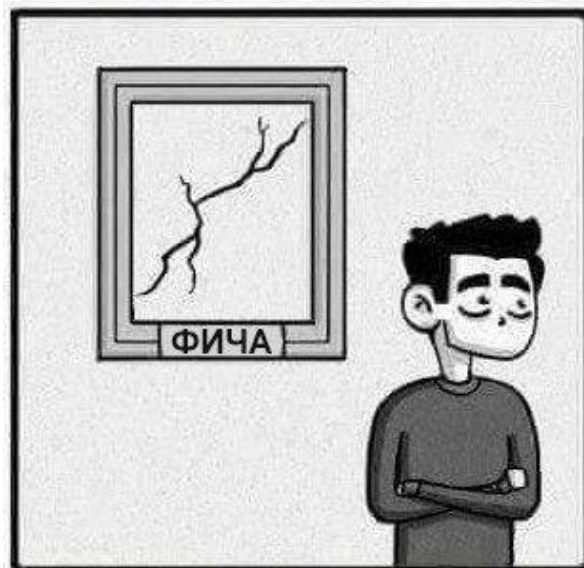
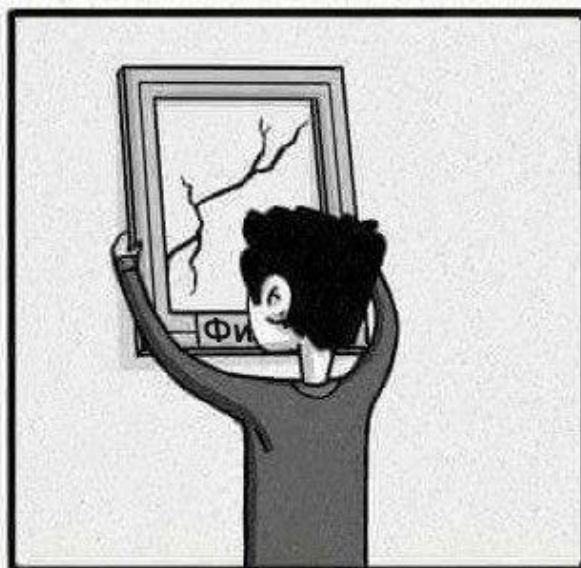
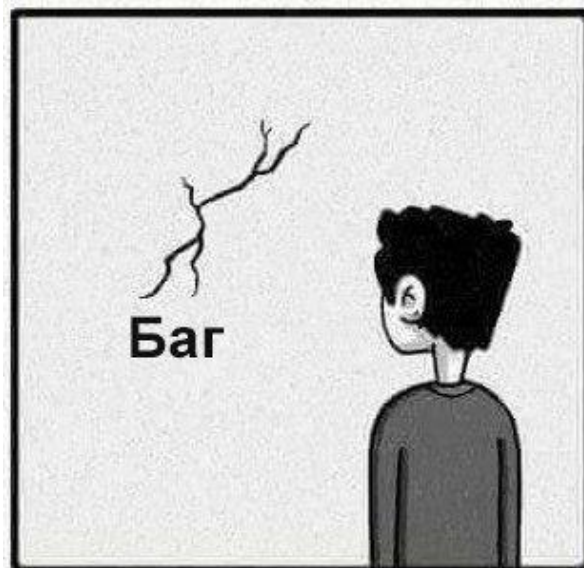
- Mistake: A human action that produces an incorrect result.
- Fault: An incorrect step, process, or data definition in a computer program
- Error: A difference...between a computed result and the correct result
- Failure: The [incorrect] result of a fault



## Types of Errors

- Incorrect calculations
- Functional errors
- Error handling errors
- Communication errors
- Syntactic errors
- Missing command errors
- Boundary related errors



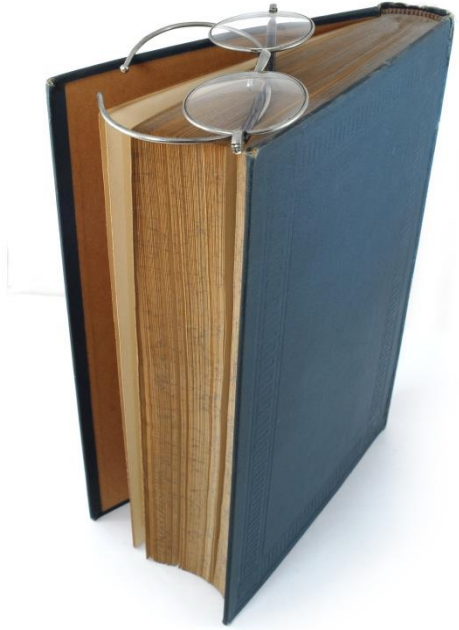


# Agenda new

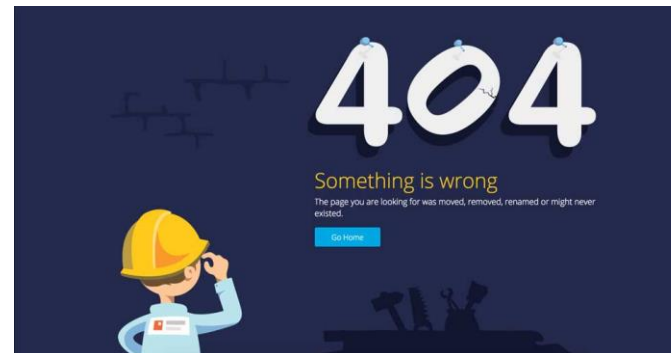
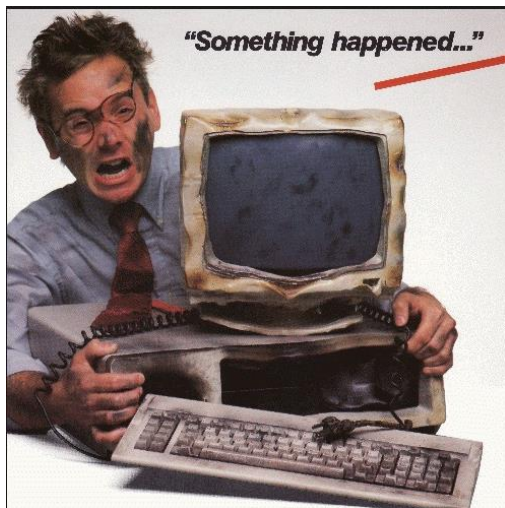
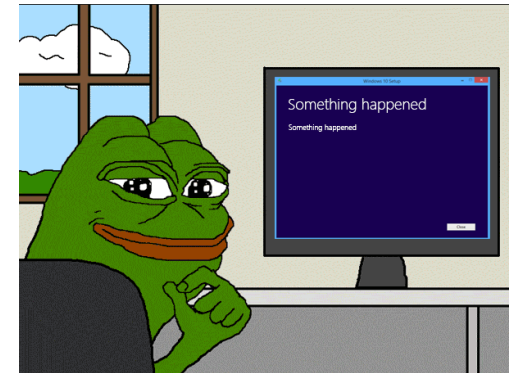
- What is a bug? Bug types.
- **Root cause analysis. Examples.**
- Priority and severity of a defect
- Bug Template
- Defect's lifecycle
- Bug attachments
- Issue Management Tool
- HW6

# Incident

Any event occurring that  
requires investigation.



# Something happened





# How it might happen



How the customer explained it.



How the project leader understood it.



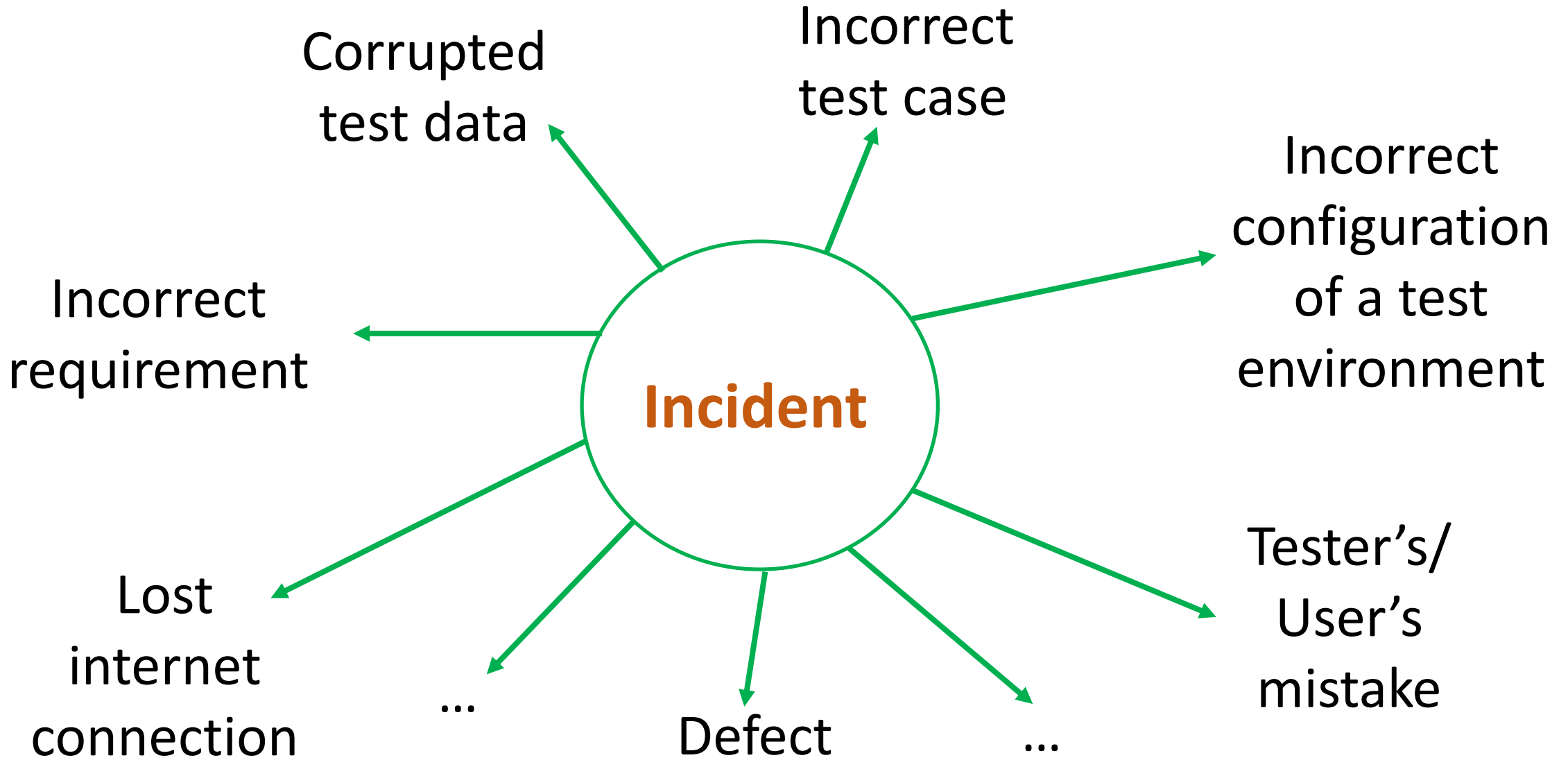
How the analyst designed it.



How the programmer wrote it.



What the customer really wanted.



# Root Cause Analysis (RCA)

Root cause analysis (RCA) is defined as a collective term that describes a wide range of [approaches, tools, and techniques](#) used to uncover causes of problems. Some RCA approaches are geared more toward identifying true root causes than others, some are more general problem-solving techniques, and others simply offer support for the core activity of root cause analysis.

Root Cause Analysis (RCA) is a popular and often-used technique that helps people answer the question of why the problem occurred in the first place. It seeks to identify the origin of a problem using a specific set of steps, with associated tools, to find the primary cause of the problem, so that you can:

- Determine what happened.
- Determine why it happened.
- Figure out what to do to reduce the likelihood that it will happen again.

You'll usually find three basic types of causes:

1. **Physical causes** – Tangible, material items failed in some way (for example, a car's brakes stopped working).
2. **Human causes** – People did something wrong, or did not do something that was needed. Human causes typically lead to physical causes (for example, no one filled the brake fluid, which led to the brakes failing).
3. **Organizational causes** – A system, process, or policy that people use to make decisions or do their work is faulty (for example, no one person was responsible for vehicle maintenance, and everyone assumed someone else had filled the brake fluid).

# Simple Steps To Analyze The Root Cause

## BUG DESCRIPTION

First, we need to collect the first information you received about the bug. The focus is on the problem analysis. Avoid thinking about technical solutions.

## DEFINE THE PROBLEM SCOPE

Clarify the following points:

- How many users are affected?
- How long was the bug undetected?
- Is there a feasible workaround for the users?

If required, you can define short-term measures to reduce the impact of the bug. For example, inform your users or stakeholders about the bug.

## “5 WHYS” METHOD

After creating a clear picture of the problem, we can start to analyze the **root cause** of the software bug using the 5 Whys method. Start with your problem description and ask multiple times “Why?” to figure out the cause for each problem. The basic theory behind this is that every cause has an effect. Here, the visible effect is the bug you want to fix. For difficult bugs, the root cause is likely hidden behind another cause. There will be a sequence of causes and their effects that finally produced the bug.

With the 5 Whys method, we want to track this sequence backward to find the **root cause** of the software bug.

## “IS / IS NOT” METHOD

One more method to analyze and/or describe the bug. The process deconstructs the Problem into 4 logical components, What the problem is, Where the problem occurs, When it occurs and the Extent to which it occurred.

## ROOT CAUSE DESCRIPTION

After we found the root cause, we need to write it down. It is very simple but it will help you to check again if there are any unclear points. Your colleagues need to understand it as well. In addition, you should think about why the bug was not detected until now and how to avoid it in the future.



# Agenda new

- What is a bug? Bug types.
- Root cause analysis. Examples.
- **Priority and severity of a defect**
- Bug Template
- Defect's lifecycle
- Bug attachments
- Issue Management Tool
- HW6

# Severity and priority of a defect

## Severity

Severity determines  
The defect's effect on the application.

How bad the defect is

Severity is given by QA testers

## Priority

Priority determines  
The defect urgency of repair.

How soon we need to fix

Priority is given by Test lead or  
project manager.

# Severity and priority levels

**Blocker:** The defect totally blocks the testing. It cannot be continued.

**Critical:** The defect affects critical functionality or critical data. It does not have a workaround. Example: Unsuccessful installation, complete failure of a feature.

**Major:** The defect affects major functionality or major data. It has a workaround but is not obvious and is difficult. Example: A feature is not functional from one module but the task is doable if 10 complicated indirect steps are followed in another module/s.

**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.

**Critical:** Must be fixed immediately / 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.

# Priority vs Severity

		SEVERITY	
		HIGH	LOW
PRIORITY	HIGH	Key features failed and no workaround <b>E.g.</b> Login button is not working	Basic feature failed but it has a huge impact on customer's business <b>E.g.</b> Misspelled Company logo
	LOW	Key features failed but there is no impact on customer's business <b>E.g.</b> Calculation fault in yearly report which end user won't use regularly	Cosmetic issues <b>E.g.</b> Font family mismatch in a report



# Agenda new

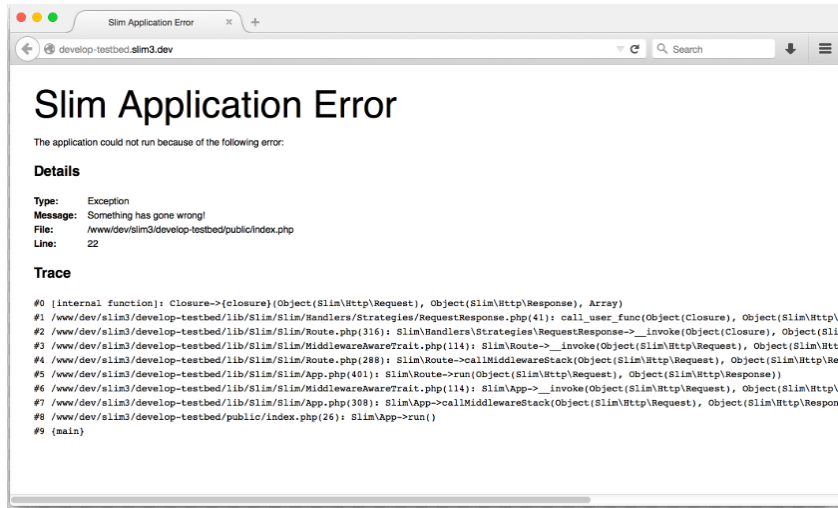
- What is a bug? Bug types.
- Root cause analysis. Examples.
- Priority and severity of a defect
- **Bug Template**
- Defect's lifecycle
- Bug attachments
- Issue Management Tool
- HW6

# Efficient Defect Report

Before reporting a defect always reproduce and localize it first!

- **summary** is meaningful & self-descriptive enough to tell about the defect
- **summary** is short & properly conveys the defect description without further reading
- add proper, sequential and clear **steps to reproduce**
- add preconditions, environment and test data information, if relevant
- don't miss anything, which makes a defect non-reproducible
- no complicated words and unwanted information – only **relevant content**
- be specific: **description** is to the point – no essay writing about the problem
- **single defect** should cover only **single problem** or similar problems
- **description** is clear and precise
- meaningful sentences and simple words
- add **references** to specifications or other related defect id
- **don't take business decisions**, only pass the Information: be objective and add the actual result & the expected result if the requirements are specified, otherwise ask responsible persons to clarify the behavior (for example, business analysts)
- think carefully of **priority and severity** of a defect
- add **attachments** if can help
- avoid mistakes in the report

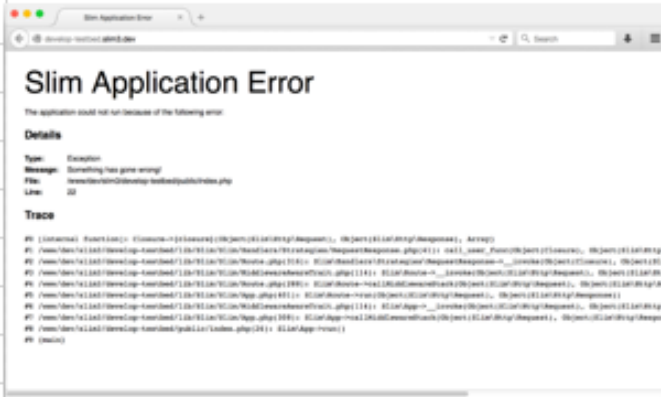
# Defect Report example



You want to create a new user with user information. For that you need to **login into the application** and **navigate to USERS menu > New User**, then **enter all the details** in the ‘User form’ like, First Name, Last Name, Age, Address, Phone etc. Once you enter all these information, you need to **click on [SAVE]** button in order to save the user. Now you can see a **success message** saying, “New User has been created successfully”.

But when you **entered into** your application by logging in and **navigated to USERS menu > New user**, **entered** all the required **information** to create the new user and **clicked on [SAVE]** button. **BANG!** The **application crashed** and you got one **error page** on the screen (corresponding screenshot with error message is captured and saved).

<b>Summary</b>	Application crashes on clicking the [Save] button while creating a new user.
<b>Bug ID</b>	<usually automatically created by the Issue Management Tool>
<b>Priority</b>	Critical
<b>Severity</b>	Major
<b>Affect Version</b>	1.0
<b>Assignee</b>	Peter
<b>Environment</b>	DEV1
<b>Description</b>	<p>Steps To Reproduce:</p> <ol style="list-style-type: none"> <li>1) Login into the application</li> <li>2) Navigate to the Users Menu &gt; New User</li> <li>3) Fill in all user information fields</li> <li>4) Click on [Save] button</li> </ol> <p>OBSERVED: Application crashes and the error page is displayed (please refer to the screenshot attached)</p> <p>EXPECTED: Save is successful and the confirmation message "New User has been created successfully" is displayed.</p>

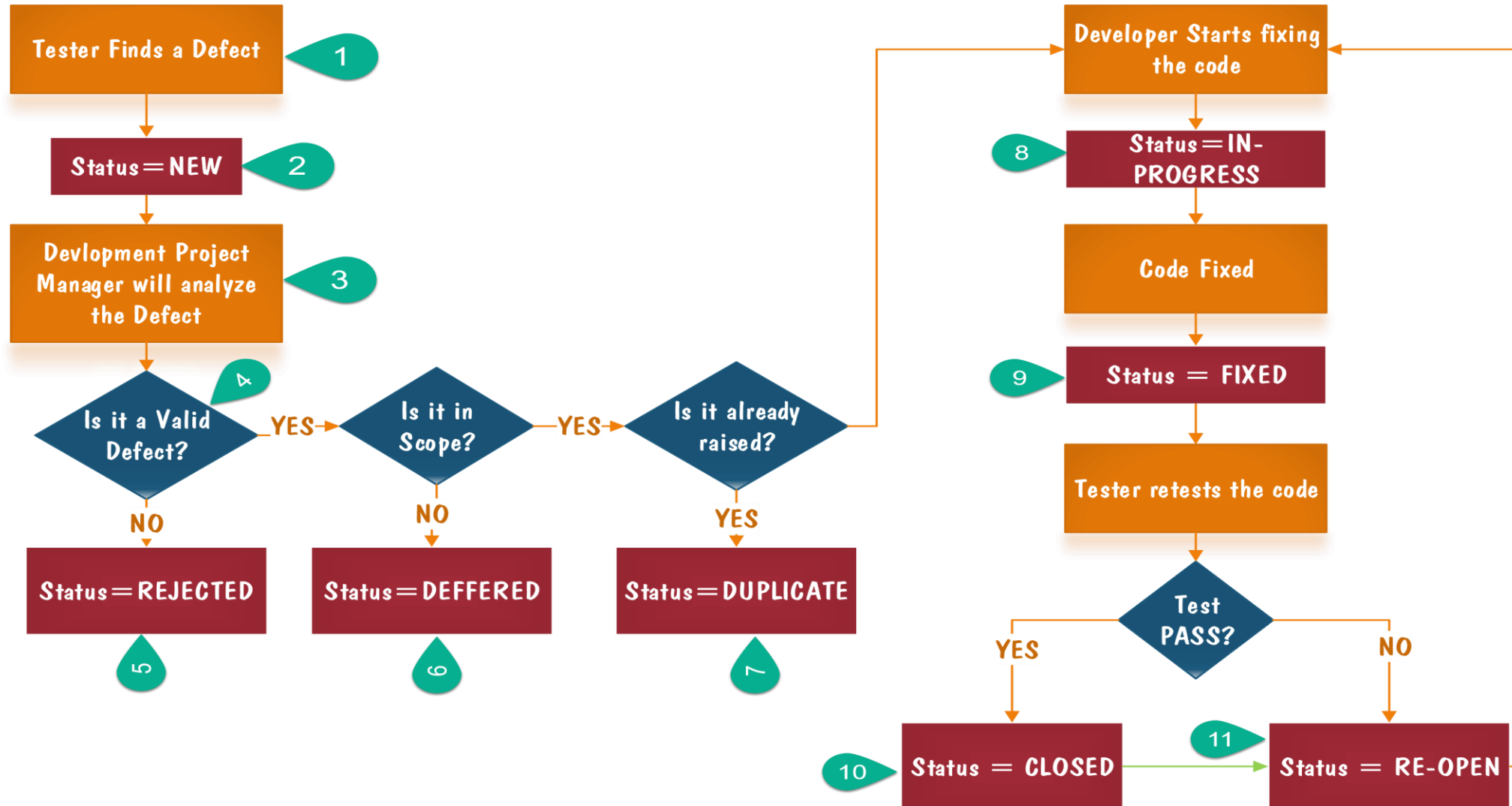
<b>Attachment</b>	
-------------------	--

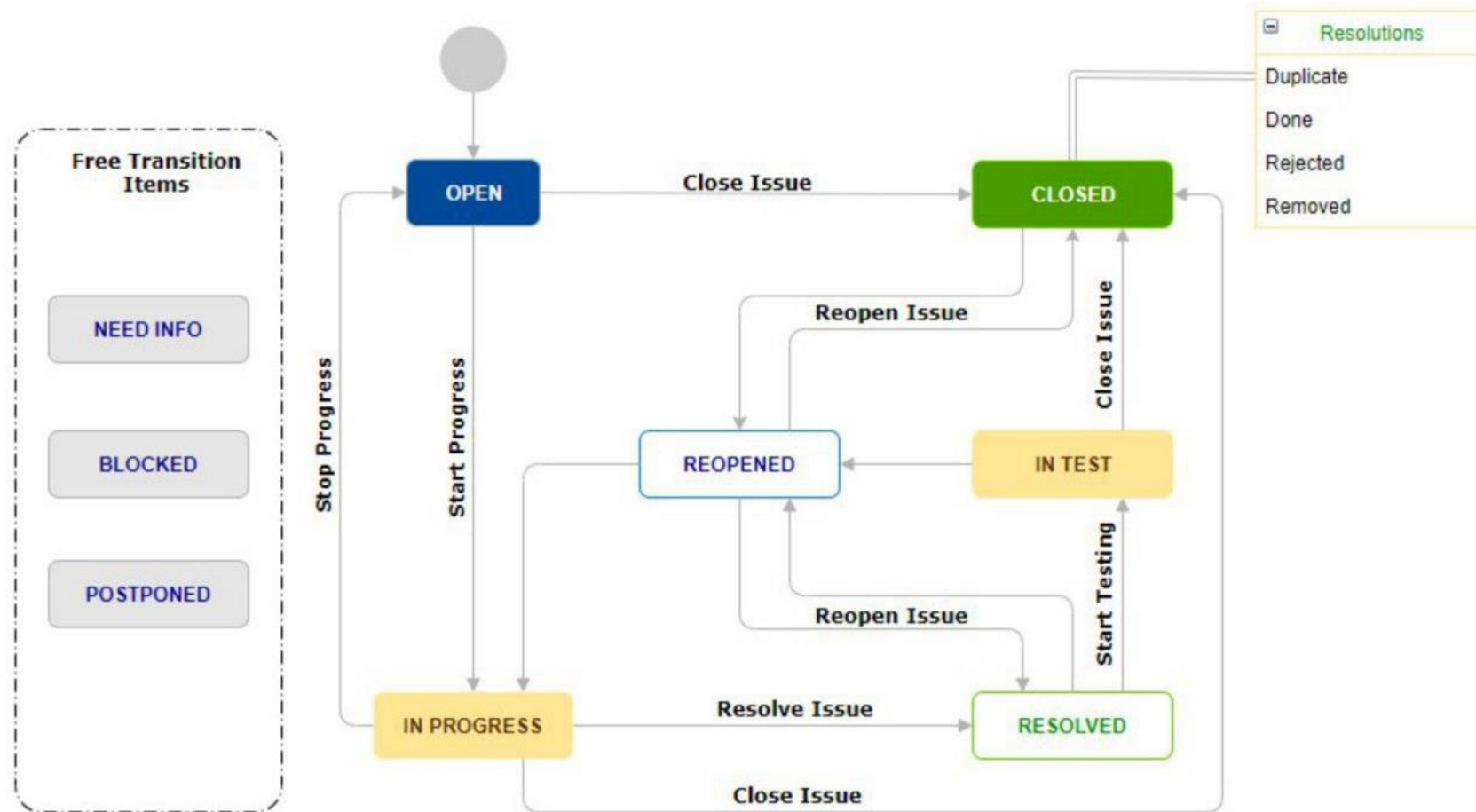


# Agenda new

- What is a bug? Bug types.
- Root cause analysis. Examples.
- Priority and severity of a defect
- Bug Template
- **Defect's lifecycle**
- Bug attachments
- Issue Management Tool
- HW6

# Defect lifecycle





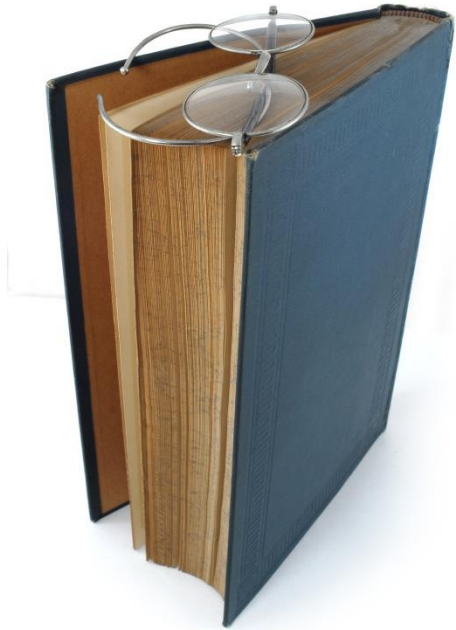
# Defect Workflow

# Agenda new

- What is a bug? Bug types.
- Root cause analysis. Examples.
- Priority and severity of a defect
- Bug Template
- Defect's lifecycle
- **Bug attachments**
- Issue Management Tool
- HW6



# Issue Management Tool

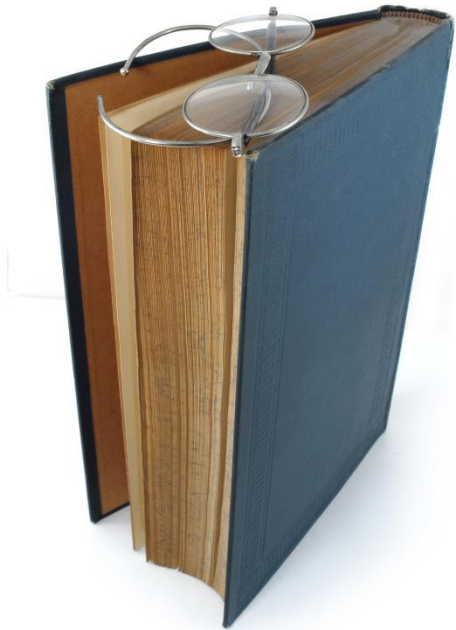


**Issue Management Tool or Issue Tracking System** is a software, which allows to raise, to manage and to maintain issues.

# Issue Management Tool's examples



# Logs



A **Log file** is a file that records either events that occur in an operating system or other software runs, or messages between different users of a communication software.

**Logging** is the act of keeping a log. In the simplest case, messages are written to a single log file.

# Agenda

- What are an error, a mistake, a defect, a bug, a fault?
- What is a failure?
- What is an incident?
- Priority and severity of a defect
- Issue Management Tool
- Defect's lifecycle
- Efficient Defect Report
- Log and Log Management Tool
- **HW6**