

# Test Management

Software Quality Assurance and Testing

# Outline

- Test organization
- Test planning and estimation
- Test Plan vs Test Strategy
- Importance of a Test Plan
- How to create a Test Plan
- Test progress monitoring and control
- Configuration management
- Risk and testing
- Defect management

# Test organization and independence

- Testing software and developing (building) software are not the same
  - Different tasks involved
  - Require different mindsets from testers and developers
- Testing is an assessment of quality
  - Assessments are not always positive
- Separate the testers from the developers
  - Improve defect finding by using independent testers
  - Avoid author bias --> Objective assessments

# Test organization and independence

- The effectiveness of finding defects by testing and reviews can be improved by using independent testers
- Options for independence are:



1. **No independent** testers. Developers test their own code.
2. Independent testers **within the development teams**.
3. Independent test team or group **within the organization**, reporting to project management or executive management.
4. Independent testers **from the business organization or user community**.
5. **Independent test specialists** for specific test targets such as usability testers, security testers or certification testers (who certify a software product against standards and regulations).

# Test organization and independence

- For large, complex or safety critical projects, it is usually best to have multiple levels of testing , with some or all of the levels done by independent testers
- Development staff can participate in testing, especially at the lower levels

## Advantages & disadvantages



**Independent** testers see other and **different defects**, and are **unbiased**.

An **independent** tester can **verify assumptions** people made during specification and implementation of the system.



**Isolation** from the development team (if treated as totally independent).

Independent testers **may be the bottleneck** as the **last checkpoint**.

**Developers** may **lose** a sense of **responsibility** for quality.

# Advantages & disadvantages

## Note

- Testing tasks may be done by people in a specific testing role, or may be done by someone in another role, such as:
  - project manager
  - quality manager
  - developer
  - business and domain expert
  - infrastructure or IT operations
- *(this can be both good and bad)*

# Tasks of the test leader and tester

- There is wide variation in the roles that people within the test team play. The two most common roles are
- **Test leader** = test manager / test coordinator.
  - The test leader plans, monitors and controls the *testing activities and tasks*.
- **The tester**
  - The tester reviews and contributes to test plan, analyses, designs, prepares, implements and executes tests.

# Tasks of the test leader

Coordination	of the test strategy and plan with project managers
Plan the tests	Understanding the test objectives and risks –including: <ul style="list-style-type: none"><li>• selecting test approaches</li><li>• estimating the time, effort and cost of testing</li><li>• acquiring resources</li><li>• defining test levels, cycles</li><li>• planning defect management</li></ul>
Test specifications, preparation and execution	Initiate the specification, preparation, implementation and execution of tests <ul style="list-style-type: none"><li>• monitor the test results</li><li>• check the exit criteria</li></ul>
Adapt planning	based on test results and progress and take any action to compensate for problems
Manage test configuration	Set up adequate configuration management for traceability
Introduce metrics	For measuring test progress and evaluating the quality of testing & product
Automation of tests	Decide what should be automated, to what degree, and how
Select test tools	Select tools to support testing and organize trainings for tool users
Test environment	Decide about the implementation of the test environment
Test summary reports	Write test summary reports based on the information gathered during testing



# Tasks of the tester

Test plans	Review and contribute to test plans
Requirements and specifications	Analyze review and assess user requirements, specifications and models for testability.
Test specifications	Create test specifications
Test environment	Set up the test environment (often coordinating with system administration and network management).
Test data	Prepare and acquire test data
Testing process	<ul style="list-style-type: none"><li>• Implement tests on all test levels,</li><li>• execute and log the tests,</li><li>• Evaluate the results</li><li>• and document the deviations from expected results.</li></ul>
Test tools	Use test tools (for management or monitoring) as required
Test automation	Automate tests (may be supported by a developer or a test automation expert)
Other metrics	Measure performance of components and systems (if applicable)
Help the others	Review tests developed by others

# Skills

People involved in testing need

- **Application or business domain**

- A tester must understand the intended behavior and the problem the system will solve in order to spot improper.

- **Technology**

- A tester must be aware of issues, limitations and capabilities of the chosen implementation technology, in order to locate problems and the 'likely to fail' functions and features.

- **Testing**

- A tester must know the testing topics discussed in this book in order to carry out the test tasks assigned.

# Skills

## People involved in testing

- need basic professional and social qualifications such as
  - literacy
  - the ability to prepare and deliver written and verbal reports
  - the ability to communicate effectively
  - ...

# Test planning and estimation

Who will do  
what,  
when  
and how

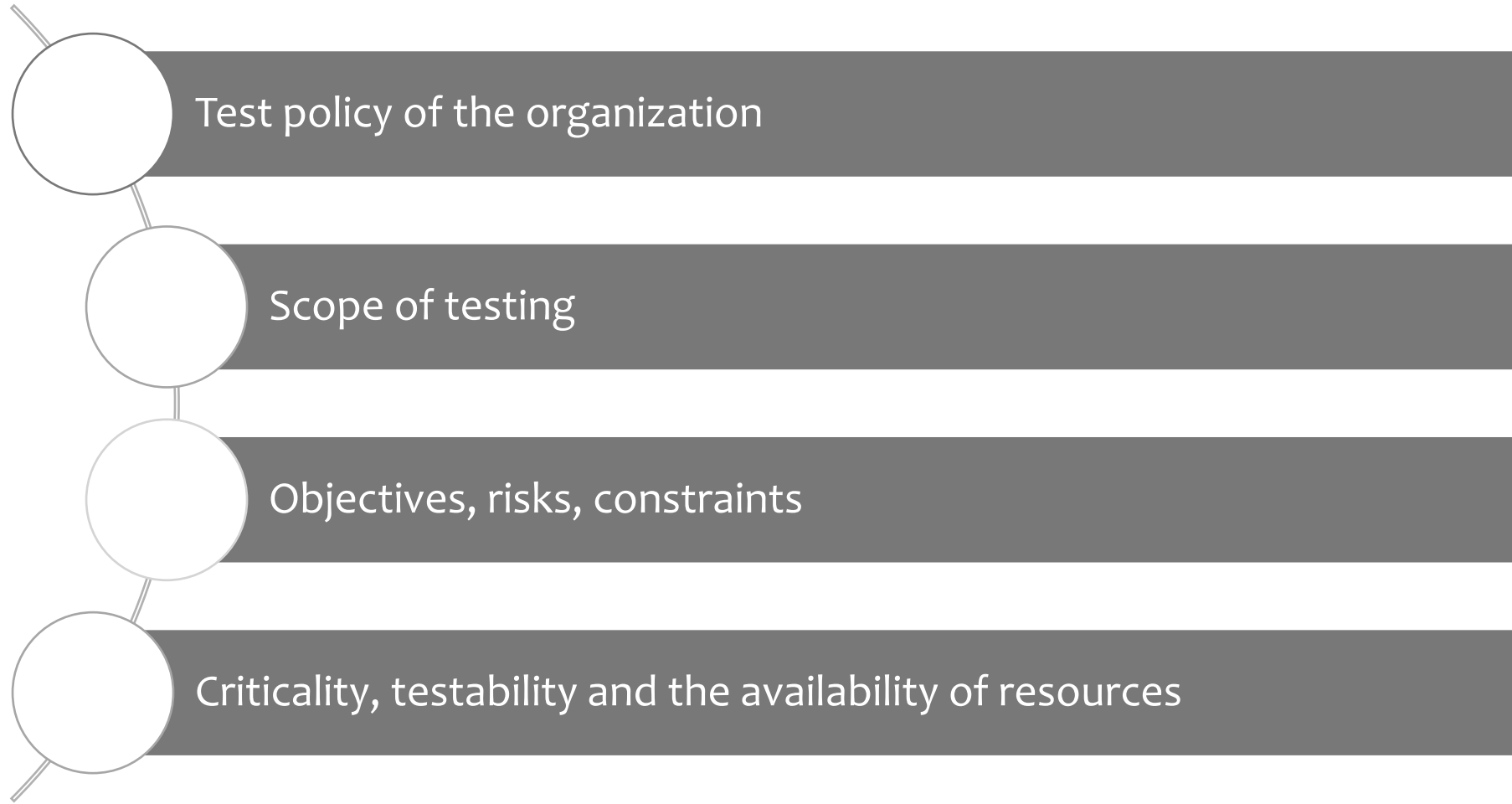
# The purpose of a test plan

- A test plan is the project plan for the testing work to be done.
- Writing a test plan forces us to confront the challenges that await us and focus our thinking on important topics.
- The test planning process and the plan itself serve as vehicles for communicating with other members of the project team, testers, peers, managers and other stakeholders.
- The test plan also helps us manage change. As we gather more information, we revise our plans.

# Test planning

- Planning may be documented in:
  - a project or master test plan
  - and in separate test plans for test levels, such as integration testing, system testing and acceptance testing.
- Test planning is a continuous activity and is performed in all life cycle processes and activities.
- Feedback from test activities is used to recognize changing risks so that planning can be adjusted.

# Test planning



*Outlines of test planning documents are covered by the 'Standard for Software Test Documentation' (IEEE 829).*

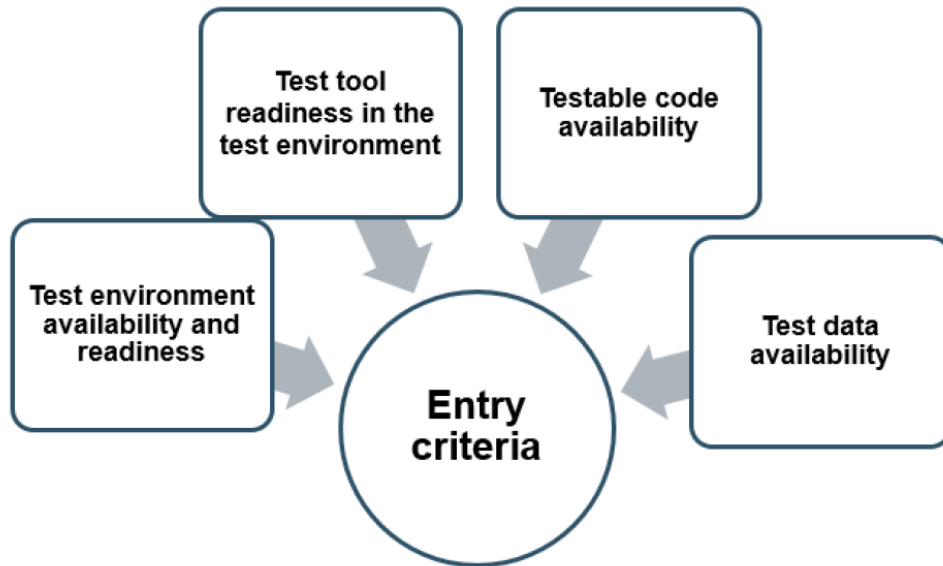
# Test planning activities

Scope and risk	Determining the scope and risks of testing
Objectives	Identifying the objectives of testing
Overall approach	Defining the overall approach of testing, including: <ul style="list-style-type: none"><li>• the definition of the test levels</li><li>• entry and exit criteria</li></ul>
Test activities	Integrating and coordinating the testing activities into the software life cycle activities: <ul style="list-style-type: none"><li>• acquisition and supply</li><li>• development</li><li>• operation</li><li>• Maintenance</li></ul>
Strategy	Making decisions about: <ul style="list-style-type: none"><li>• what to test</li><li>• what roles will perform the test activities</li><li>• how the test activities should be done</li><li>• and how the test results will be evaluated</li></ul>
Schedule	Scheduling test analysis and design activities Scheduling test implementation, execution and evaluation
Resources	Assigning resources for the different activities defined.
Metrics	Selecting metrics for monitoring and controlling test preparation and execution, defect resolution and risk issues.

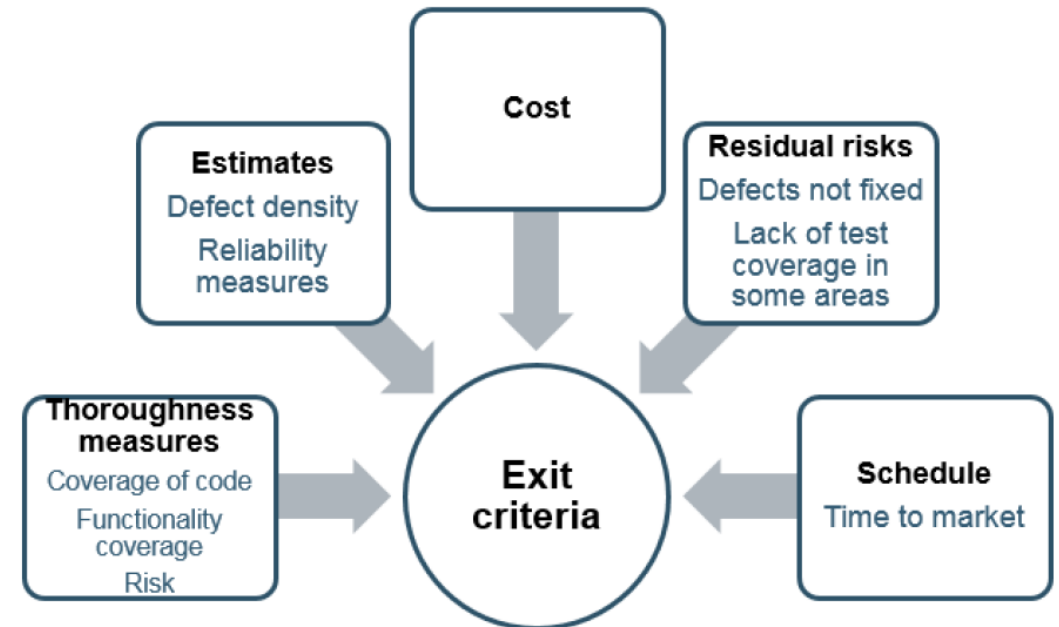


# Entry/Exit criteria

- Entry criteria defines when to start testing



- Exit criteria is to define when to stop testing, such as at the end of a test level, end of project or when a set of tests has a specific goal.



# Test estimation

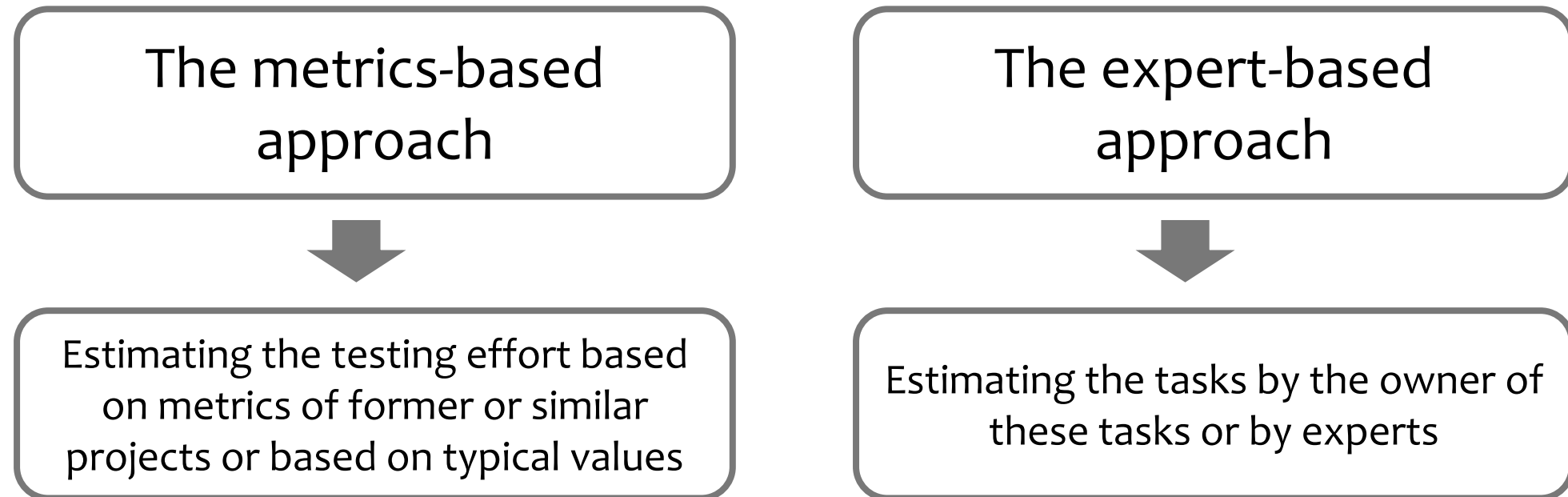
- The testing work is usually a subproject within the larger project.
- Fundamental techniques of estimation can be adapt for testing
- We can break down a testing project into phases
  - planning and control
  - analysis and design
  - implementation and execution
  - evaluating exit criteria and reporting
  - test closure
- Within each phase we identify activities and within each activity we identify tasks and perhaps subtasks .

# Test estimation

- To identify the activities and tasks, we work both
  - forward
  - Backward
- To ensure accuracy of the estimate, make sure that you subdivide the work into tasks that are short in duration, say one to three days.
- If they are much longer say two weeks there will be a risk that long and complex sub tasks are 'hidden' within the larger task.

# Test estimation techniques

- Two approaches for the estimation of test effort are covered:



# Test estimation

- A good solution is to combine the two strategies:
  - **First:** create the work-breakdown structure and a detailed bottom-up estimate.
  - **Second:** We then apply models and rules of thumb to check and adjust the estimate bottom-up and top-down using past history.
- This approach tends to create an estimate that is both more accurate and more defensible than either technique by itself.

# Test estimation - an example

- Performance tests need to be run in a special test environment that look like the production or field environment, due to the response time and resource utilization
- Performance testing involves
  - special tools to generate load and check response
  - environment acquisition and configuration
  - time to identify and hire a performance test professional , if required
  - training activities, if required
- A detailed test plan for performance testing should be written in the test planning phase, where time is required to draft, review and finalize a performance test plan.

# Factors affecting the test effort

- The testing effort may depend on a number of factors, including:

<b>Product factors</b>	<ul style="list-style-type: none"><li>• the quality of the specification (the test basis)</li><li>• the size of the product</li><li>• the complexity of the problem domain</li><li>• the importance of non functional quality e.g. usability, performance, security etc.</li></ul>
<b>Process factors</b>	<ul style="list-style-type: none"><li>• the development model</li><li>• availability of test tools (e.g. test executing tools)</li><li>• skills of the people involved</li><li>• time pressure</li></ul>
<b>The outcome of testing</b>	<ul style="list-style-type: none"><li>• the number of defects</li><li>• the amount of rework required</li></ul>

# Test approaches and strategies

- The test approach is the implementation of the test strategy for a specific project.
- Since the test approach is specific to a project , the approach should be documented in the test plan
- One way to classify test approaches or strategies is based on the point in time at which the bulk of the test design work is begun:

Preventative approaches: Tests are designed as early as possible

Reactive approaches: Test design comes after the software or system has been produced



# Test approaches and strategies

<b>Analytical approaches</b>	e.g. risk-based testing -testing is directed to areas of greatest risk, requirement based testing
<b>Model based approaches</b>	e.g. testing using statistical information about failure rates (such as reliability growth models)
<b>Methodical approaches</b>	e.g. failure based (including error guessing and fault attacks), experienced based, check list based, and quality characteristic based
<b>Process/standard compliant approaches</b>	e.g. specified by industry specific standards or the various agile methodologies
<b>Dynamic and heuristic approaches</b>	e.g. exploratory testing, execution & evaluation are concurrent tasks.
<b>Consultative approaches</b>	e.g. test coverage is evaluated by domain experts outside the test team.
<b>Regression-averse approaches</b>	include reuse of existing test material, extensive automation of functional regression tests

# Test approaches and strategies

- The choice of test approaches and strategies is one powerful factor in the success of the effort and the accuracy of the test plans and estimates.
- When we choose test strategies there are many factors to consider
  - Risks
  - Skills
  - Objectives
  - Regulations
  - Product
  - Business

# Test Plan and Test Strategy

- **Test Plan** reflects test focus and project scope are defined.
  - It deals with test **coverage**, **scheduling**, **features** to be tested, features not to be tested, estimation and **resource management**.
- **Test Strategy** is a **guideline** to be followed to achieve the test objective and execution of test types mentioned in the testing plan.
  - It deals with test objective, test environment, test approach, automation tools and strategy, contingency plan, and risk analysis

# Test Plan

- A **test plan** is a detailed document that outlines the **test strategy**, **testing objectives**, **resources** (manpower, software, hardware) required for testing, test **schedule**, test **estimation** and test **deliverables**.
- The test plan serves as a **blueprint** to conduct software **testing** activities as a defined process which is minutely monitored and controlled by the test manager.

# Importance of a Test Plan

- Test Plan helps us determine the **effort** needed to validate the quality of the application under test (AUT).
- Help people outside the test team such as developers, business managers, customers **understand** the details of testing.
- Test Plan **guides** our thinking. It is like a rule book, which needs to be followed.
- Important aspects like test estimation, test scope, Test Strategy are **documented** in Test Plan, so it can be reviewed by Management Team and re-used for other projects.

# Test Plans

- The most common question about testing is
  - “ How do I write a test plan? ”
- This question usually comes up when the focus is on the document, not the contents
- It's the contents that are important, not the structure
  - Good testing is more important than proper documentation
  - However – documentation of testing can be very helpful
- Most organizations have a list of topics, outlines, or templates

# Test Plan

## Objectives

- To create a set of testing tasks
- Assign resources to each testing task
- Estimate completion time for each testing task
- Document testing standards

# Good Test Plans

- Developed and Reviewed early
- Clear, Complete and Specific
- Specifies tangible deliverables that can be inspected
- Staff knows what to expect and when to expect it
- Realistic quality levels for goals
- Includes time for planning
- Can be monitored and updated
- Includes user responsibilities
- Based on past experience
- Recognizes learning curves



# Standard Test Plan

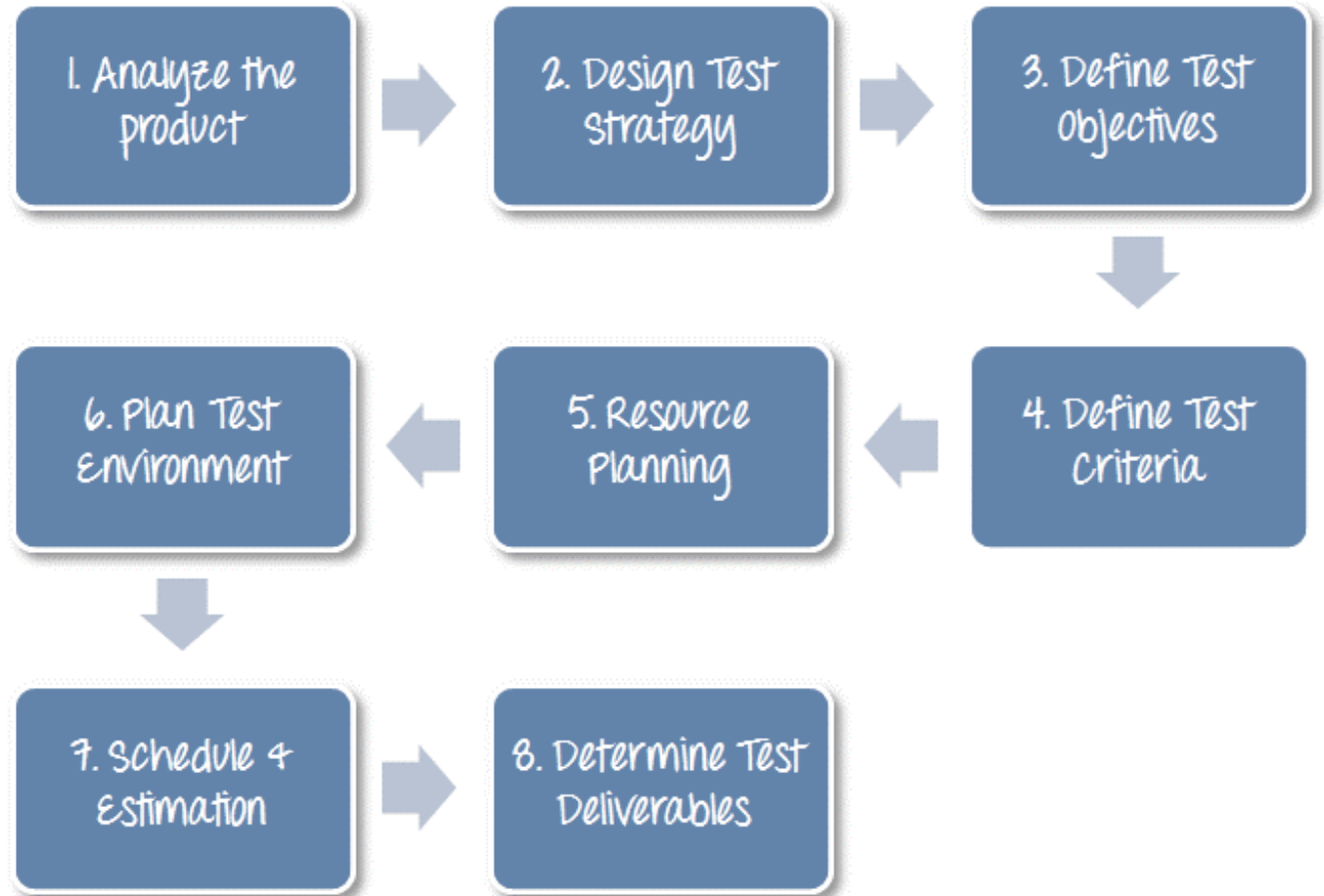
- IEEE Standard 829-2008 is old but still used:

## Test Plan

- A document describing the scope, approach, resources, and schedule of intended testing activities. It identifies test items, the features to be tested, the testing tasks, who will do each task, and any risks requiring contingency planning
- Many organizations are required to adhere to this standard
- Unfortunately, this standard emphasizes documentation, not actual testing – often resulting in a well documented vacuum

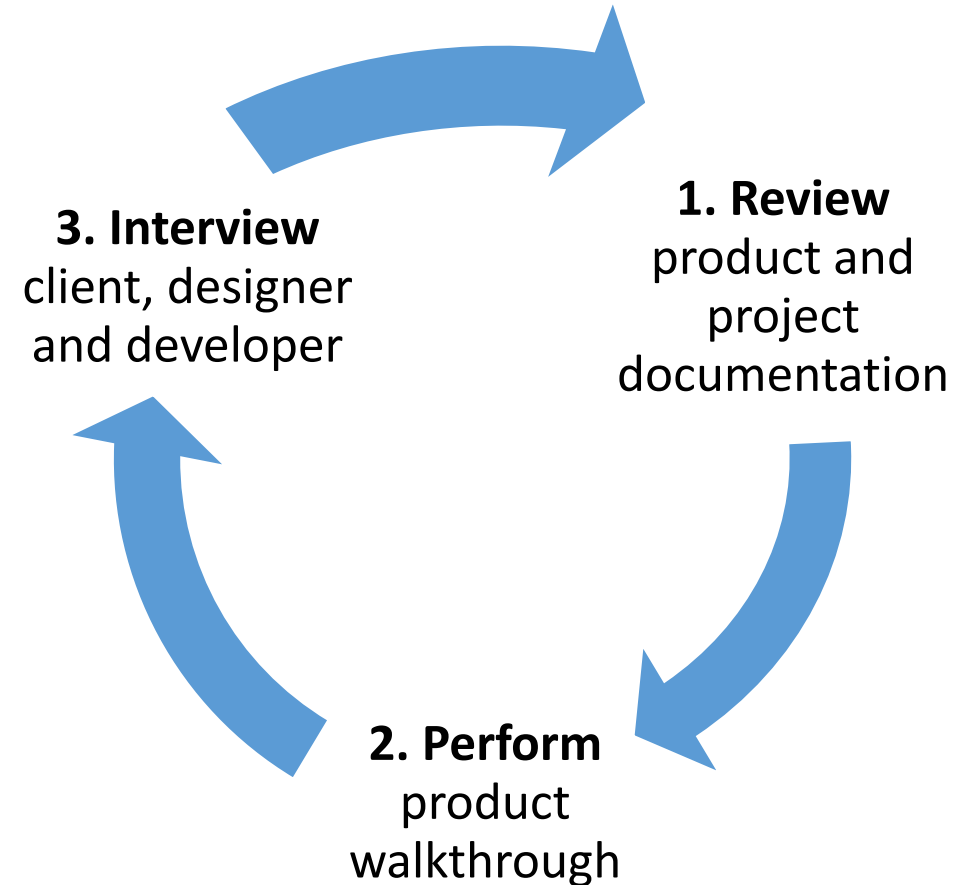
# How to write a Test Plan

- Creating a **Test Plan** is the most important task of Test Management Process.



# 1. Analyze the Project

- Can you test a product **without** any information about it?
- No... You must learn a product **thoroughly** before testing it.
- You can use the following approach to **analyze** the product:



## 2. Develop Test Strategy

- Test Strategy is a **critical step** in making a Test Plan. This document defines:
  - The project's **testing objectives** and the means to achieve them
  - Determines testing **effort** and **costs**
- To develop Test Strategy for testing a product, you should follow steps below:
  - 2.1 Define Scope of Testing
  - 2.2 Identify Testing Type
  - 2.3 Document Risks and Issues
  - 2.4 Create Test Logistics

## 2. Develop Test Strategy

### Step 2.1) Define Scope of Testing

- Scope of the testing should be known.
  - **“in scope”**: The components of the system to be tested (hardware, software, middleware, etc.)
  - **“out of scope”**: The components of the system that will not be tested
- To determine scope, you must consider:
  - Customer requirement
  - Project Budget
  - Product Specification
  - Skills & talent of your test team

## 2. Develop Test Strategy

### Step 2.2) Identify Testing Type

- A **Testing Type** is a standard test procedure that gives an expected test outcome. (many are available..)
- The **commonly used** testing types are:
  - **Unit Test**: test the smallest piece of verifiable software in the application.
  - **API Test**: test the API's created for the application
  - **Integration Test**: individual software modules combined and tested as group
  - **System Test**: conducted on a complete, integrated system to evaluate system's compliance with its specified requirements
  - **And more ...**

## 2. Develop Test Strategy

### Step 2.3) Document Risk & Issues

- Risk is future's **uncertain event** with a probability of **occurrence** and a **potential** for loss. When the risk actually happens, it becomes the 'issue'.
- You should document those risks in the Test Plan.

#### (Example)

Risk	Mitigation
Team member lack the required skills for website testing.	Plan training course to skill up your members
The project schedule is too tight; it's hard to complete this project on time	Set Test Priority for each of the test activity
Wrong budget estimate and cost overruns	Establish the scope before beginning work, pay a lot of attention to project planning and constantly track and measure the progress

## 2. Develop Test Strategy

### Step 2.4) Create Test Logistics

- In Test Logistics, the Test Manager should answer the following questions:
  - **Who** will test?
  - **When** will the test occur?



# 3. Define Test Objective

- Test Objective is the overall goal and achievement of the test execution
- A sample of the Objectives for an online bank website:  
*“The test objectives are to **verify** the functionality of the website of the bank, the project should focus on testing the **banking operation** such as Account Management, Withdrawal, and Balance... etc. to **guarantee** all these operations can work **normally** in real business environment.”*

# 4. Define Test Criteria

- Test Criteria is a standard or rule on which a test procedure or test judgment can be based. There're 2 types of test criteria as following:
  - **Suspension Criteria:** Specify the critical suspension criteria for a test. If the suspension criteria are met during testing, the active test cycle will be **suspended** until the criteria are **resolved**.

## Example:

“If testing team report that there are **40%** of test cases failed, you should **suspend** testing until the development team fixes all the failed cases.”

## 4. Define Test Criteria

- Test Criteria is a standard or rule on which a test procedure or test judgment can be based. There're 2 types of test criteria as following:
  - **Exit Criteria:** It specifies the criteria that denote a **successful** completion of a test phase. The exit criteria are the targeted results of the test and are necessary before proceeding to the next phase of development.

### Example:

“95% of all critical test cases must pass.”

# 5. Resource Planning

- Resource plan is a **detailed summary** of all types of resources required to complete project task.
- Resource could be **human, equipment** and **materials** needed to complete a project
- The resource planning is important factor of the test planning because it helps in **determining** the **number** of resources (employee, equipment... ) to be used for the project. Therefore, the Test Manager can make the correct schedule & estimation for the project. (check the Sample Test Plan on Moodle for an example)

# 6. Plan Test Environment

- How do you plan for setting up a **test environment** for a product?
- To finish this task, you need a **strong cooperation** between **Test Team** and **Development Team**
- Test team should ask the development team all required details about the product to run to be able to **clearly** understand the product under test.
- **Sample of questions:**
  - What is the maximum user connections which this product can handle at the same time?
  - What are hardware/software requirements to install this product?
  - Does the user's computer need any particular setting to browse the website?
  - Other questions ...

# 7. Schedule & Estimation

- To do the Schedule and time estimation it is better to break down the whole project into small tasks and add the estimation for each task separately. An example is provided below:

How can we estimate required effort?

Task	Members	Estimate effort
Create the test specification	Test Designer	170 man-hour
Perform Test Execution	Tester, Test Administrator	80 man-hour
Test Report	Tester	10 man-hour
Test Delivery		20 man-hour
Total		280 man-hour

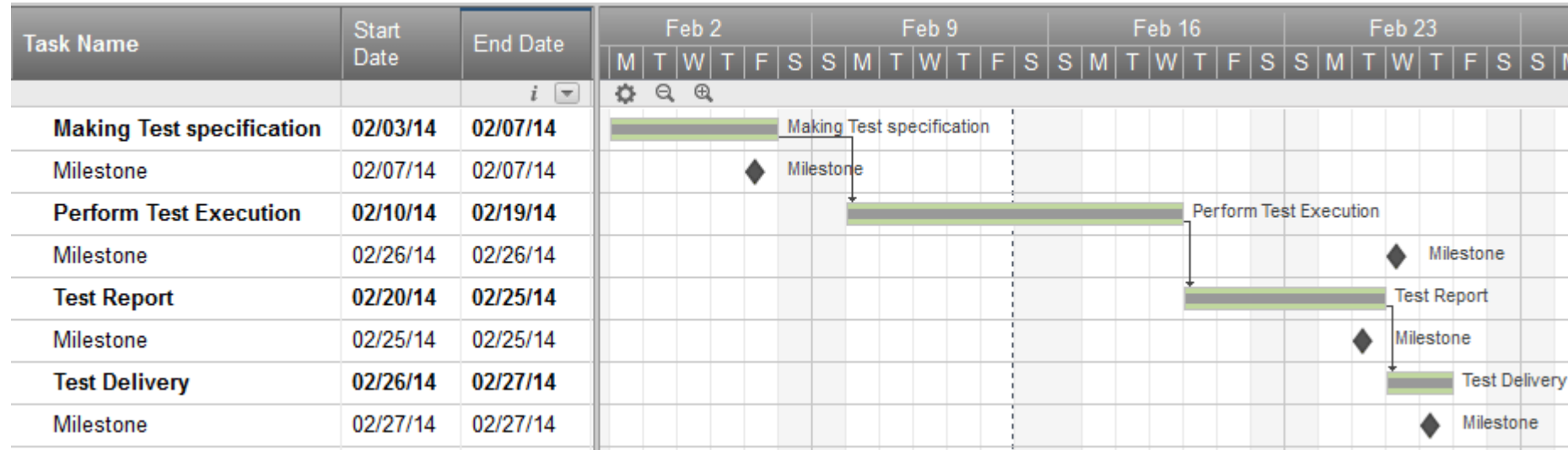
# 7. Schedule & Estimation

- Then you create the **schedule** to complete these tasks.
- To create the project schedule, the Test Manager needs several types of input as below:
  - **Employee and project deadline:** The working days, the project deadline, resource availability are the factors which affected to the schedule
  - **Project estimation:** Base on the estimation, the Test Manager knows how long it takes to complete the project. So he/she can make the appropriate project schedule
  - **Project Risk :** Understanding the risk helps Test Manager add enough extra time to the project schedule to deal with the risks

# 7. Schedule & Estimation

- **Example:**

- Suppose that you need to complete testing of your project in one month, you already estimated the effort for each tasks in Test Estimation. You can create a schedule similar to the one below:





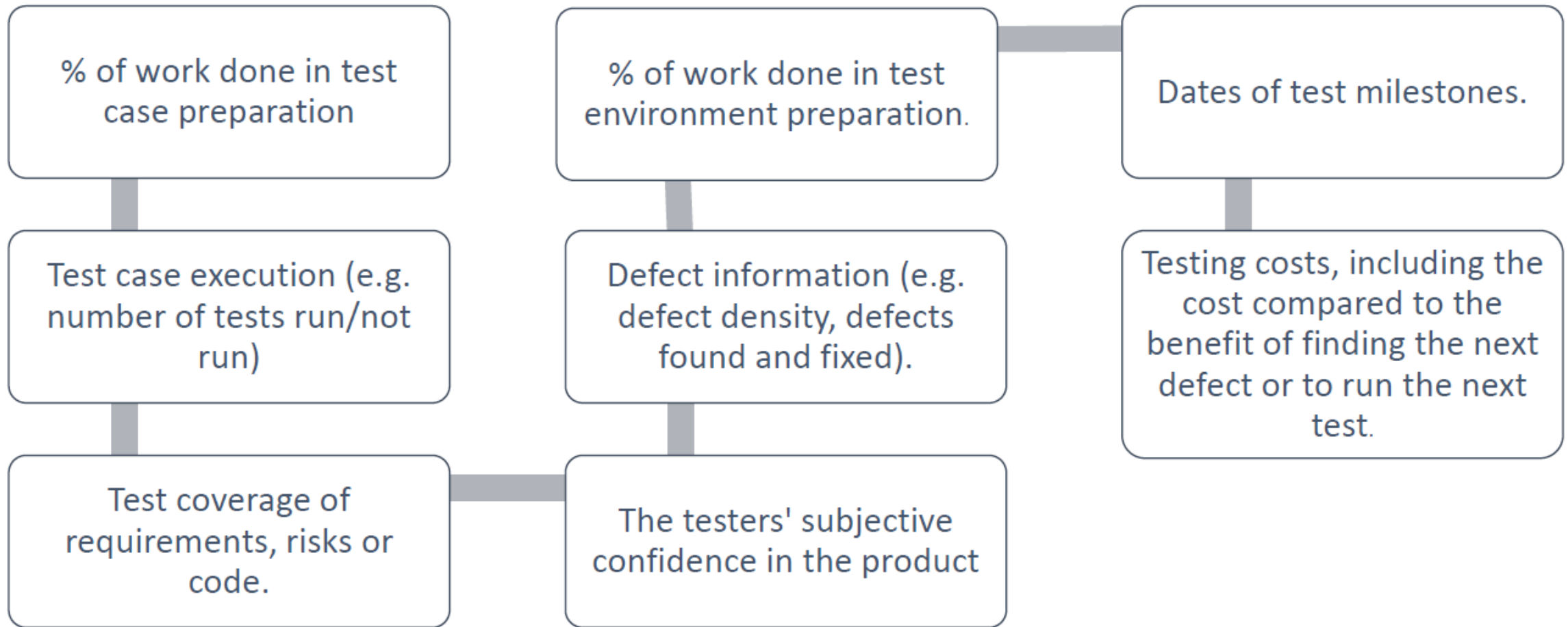
# 8. Test Deliverables

- Test Deliverables is a list of all the **documents, tools** and other **components** that has to be developed and maintained in support of the testing effort.
- There are different test deliverables at every phase of the software development lifecycle:
  - **Before Testing** (e.g. test plans, test cases, test design specifications)
  - **During Testing** (e.g. test scripts, test data, test traceability matrix, error logs)
  - **After Testing** (e.g. test results and reports, defect report, installation guidelines, release notes)

# Test progress monitoring

- The purpose of test monitoring is to give feedback and visibility about test activities .
- Information to be monitored may be collected manually or automatically and may be used to measure exit criteria , such as coverage.
- Metrics may also be used to assess progress against the planned schedule and budget.

# Test progress monitoring



# Test log template

---

## IEEE 829 STANDARD: TEST LOG TEMPLATE

---

Test log identifier

Description (items being tested,  
environment in which the testing is  
conducted)

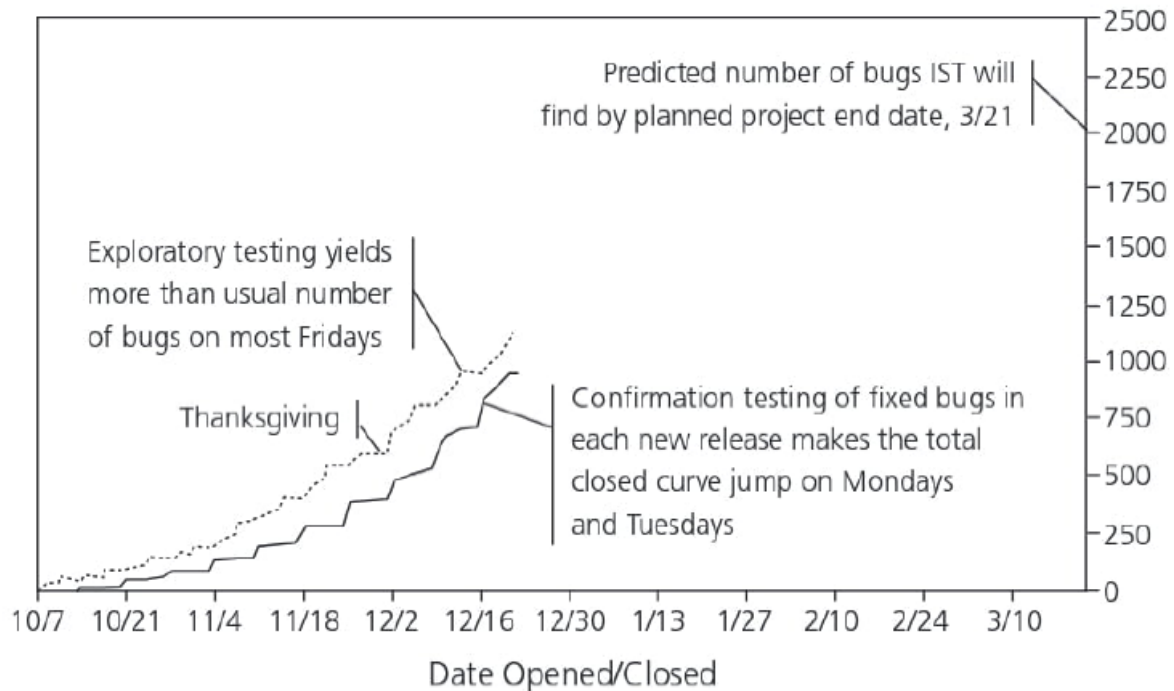
Activity and event entries (execution  
description, procedure results,  
environmental information,  
anomalous events, incident report  
identifiers)

# Test case summary - common metrics

	A	B	C	D	E	F	G	H	I	J	K	L	M
1		System Test Case Summary											
2		Cycle One											
3													
4	Test			System	Bug	Bug	Run	Plan	Act	Plan	Actual	Test	
5	ID	Test Suite/Case	Status	Config	ID	RPN	By	Date	Date	Effort	Effort	Duration	Comment
6													
7	1.000	Functionality											
8	1.001	File	Fail	A	701	1	LTW	1/8	1/8	4	6	6	
9	1.002	Edit	Fail	A	709	1	LTW	1/9	1/10	4	8	8	
10					710	5							
11					718	3							
12					722	4							
13	1.003	Font	Pass	B			IHB	1/10	1/10	4	4	4	
14	1.004	Tables	Warn	B	708	15	IHB	1/8	1/9	4	5	5	
15	1.005	Printing	Skip					1/10		4			Out of runway
16		Suite Summary						1/10	1/10	20	23	23	
17													
18	2.000	Performance/Stress											
19	2.001	Solaris Server	Warn	A,B,C	701	1	EM	1/10	1/13	4	8	24	Replan 1/11
20	2.002	NT Server	Fail	A,B,C	724	2	EM	1/11	1/14	4	4	24	Replan 1/12
21					713	2							
22					725	1							
23	2.003	Linux Server	Skip					1/12		4			Out of runway
24		Suite Summary						1/12	1/14	12	12	48	
25													
26	3.000	Error Handling/Recovery											
27	3.001	Corrupt File	Fail	A	701	1	LTW	1/8	1/9	4	8	8	
28					706	2							
29					707	4							
30					709	1							
31					710	5							
32					713	2							
33	3.002	Server Crash	Fail	A	712	6	LTW	1/9	1/10	4	6	6	
34					713	2							
35					717	1							
36		Suite Summary						1/9	1/10	8	14	14	
37													
38	4.000	Localization											
39	4.001	Spanish	Skip										
40	4.002	French	Skip										
41	4.003	Japanese	Skip										
42	4.004	Chinese	Skip										
43		Suite Summary						1/0	1/0	0	0	0	

# Total defects opened and closed chart

Integration and System Test Execution  
System Quality Problems Analysis



## Key Milestones

Integration Test Entry 10/7

System Test Entry 10/21

Integration Test Exit 2/14

System Test Exit 3/21

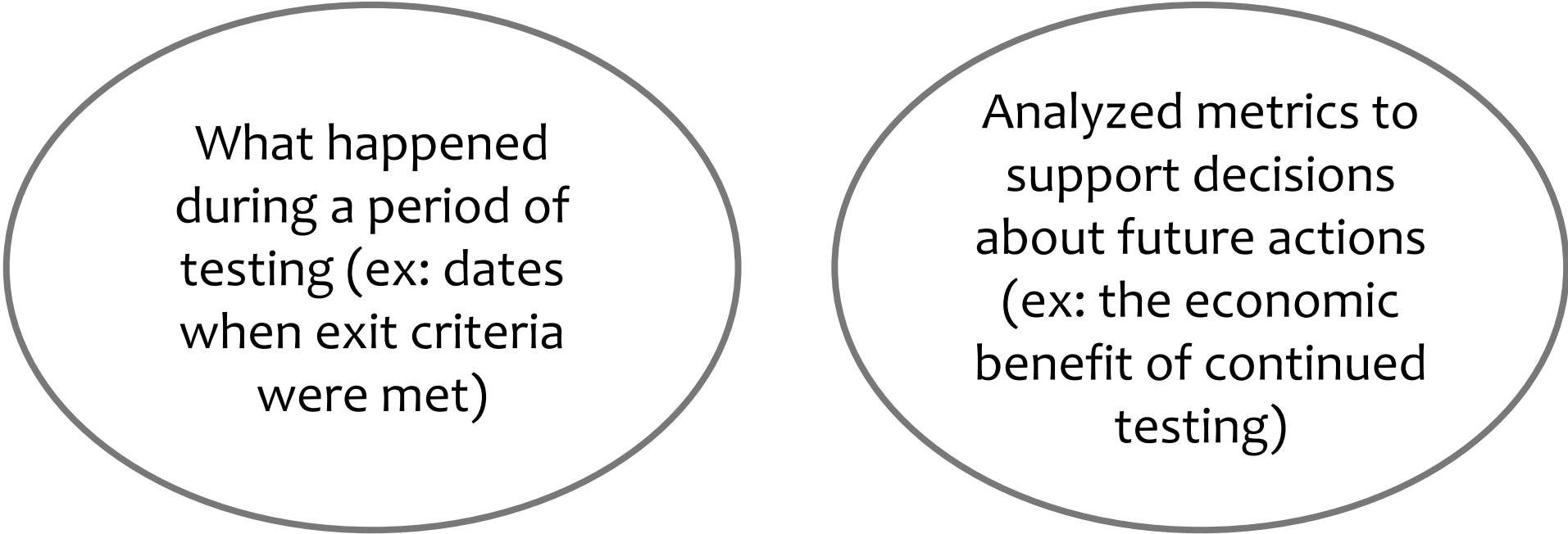
Total Opened .....

Total Closed —

N.B. Closed counts include deferred bugs

# Test reporting

- Test reporting is concerned with summarizing information about the testing endeavor, including:



What happened during a period of testing (ex: dates when exit criteria were met)

Analyzed metrics to support decisions about future actions (ex: the economic benefit of continued testing)

# Test reporting

- Metrics are collected at the end of a test level in order to assess:
  - The adequacy of the test objectives for that test level
  - The adequacy of the test approaches with respect to its objectives
  - The effectiveness of the testing with respect to its objectives



# Test reporting

	Unresolved defects		Test cases to be run		
Product risk areas	Number	%	Planned	Actual	%
<i>Performance, load, reliability</i>	304	28	3,843	1,512	39
<i>Robustness, operations, security</i>	234	21	1,032	432	42
<i>Functionality, data, dates</i>	224	20	4,744	2,043	43
<i>Use cases, user interfaces, localization</i>	160	15	498	318	64
<i>Interfaces</i>	93	8	193	153	79
<i>Compatibility</i>	71	6	1,787	939	53
<i>Other</i>	21	2	760	306	40
	<i>1,107</i>	<i>100</i>	<i>12,857</i>	<i>5,703</i>	<i>44</i>

# Test summary report - template

**IEEE 829 STANDARD:  
TEST SUMMARY REPORT TEMPLATE**

---

Test summary report identifier	Evaluation
Summary	Summary of activities
Variances	Approvals
Comprehensive assessment	
Summary of results	

---

# Test control

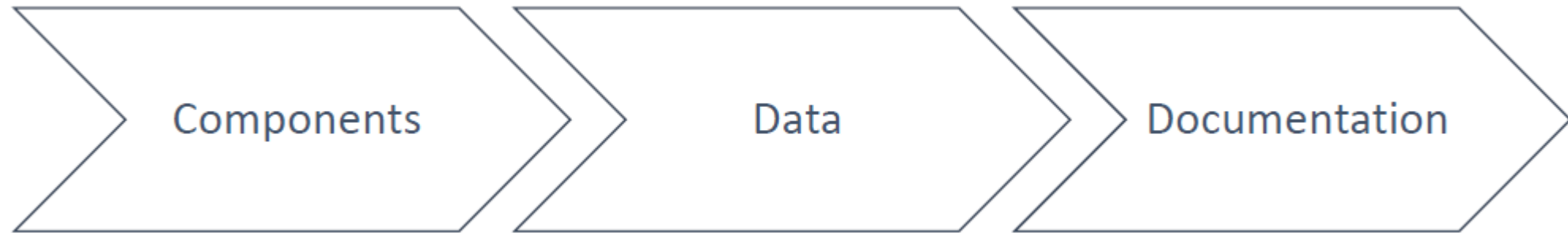
- Projects do not always go according to plan.
- Different factors can lead to deviations:
  - New risks
  - New needs
  - Test findings
  - External events
  - Test environment problems
- Test control describes any guiding or corrective actions taken as a result of information and metrics gathered and reported.

# Test control

- Test control describes any guiding or corrective actions taken as a result of information and metrics gathered and reported.
- Examples of test control actions are:
  - Making decisions based on information from test monitoring
  - Re-prioritize tests when an identified risk occurs
  - Change the test schedule due to availability of a test environment
  - Set an entry criteria requiring fixes to have been tested (confirmation tested) by a developer before accepting them into a build

# Configuration management

- The purpose of configuration management is to establish and maintain the integrity of the software and related products through the project and product life cycle



# Configuration management

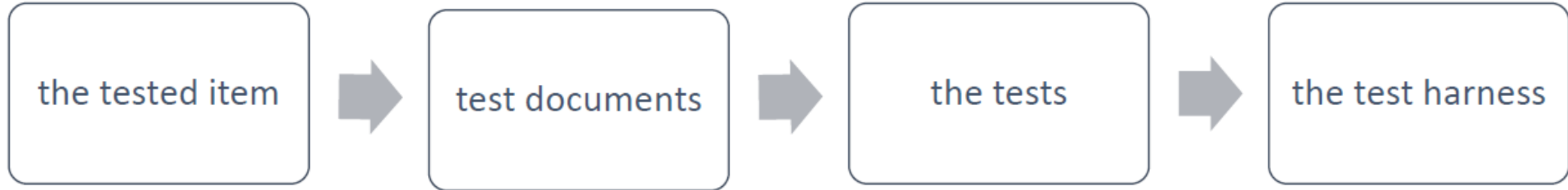
- The configuration management shall ensure that all items are
  - identified
  - version controlled
  - tracked for changes

so that traceability can be maintained throughout the test process

- All identified documents and software items should be referenced unambiguously in test documentation.

# Configuration management

- Configuration management helps to uniquely identify (and to reproduce)



- Configuration management procedures and tools should be selected during the project planning stage

# Configuration management

- When testers receive an organized, version controlled test release from a source code repository, it should be accompanied by a test item release note :

---

## **IEEE 829 STANDARD: TEST ITEM TRANSMITTAL REPORT TEMPLATE**

Transmittal report identifier

Transmitted items

Location

Status

Approvals

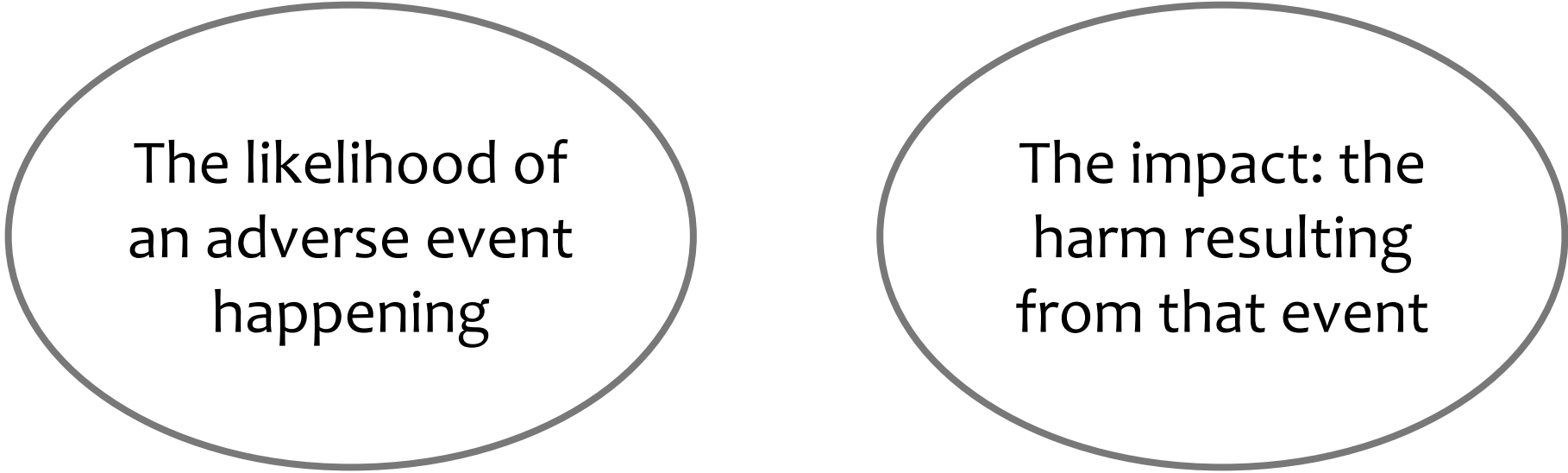


# Risk and testing

- Risk is the possibility of a negative or undesirable outcome, the possible problems that might endanger the objectives of the project stakeholders.
- Risks are related to the
  - product
  - project
- Risk analysis and risk management can help us plot a course for solid testing.

# Risk and testing

- The level of risk is determined by:



The likelihood of  
an adverse event  
happening

The impact: the  
harm resulting  
from that event

# Risk and testing

- For any risks you have four possibilities
  - Mitigate
  - Contingency
  - Transfer
  - Ignore
- When analyzing, managing and mitigating these risks, the test manager should follow well established project management principles

# Project risks

- Typical project risks
  - Logistics or product quality problems that block tests
  - Test items that won't install in the test environment
  - Excessive change to the product that invalidates test results or requires updates to test cases, expected results and environments
  - Insufficient or unrealistic test environments that yield misleading results

# Project risks

**Project risks** = the risks that surround the project's capability to deliver its objectives, such as:

## Organizational Factors

- Skill/staff shortage
- Personal/training issues
- Testers communicating their needs and test results
- Improper attitude toward testing

## Technical Issues

- Problems in defining the right requirements
- The extent that requirements can be met given existing constraints
- The quality of design, code, and tests

## Supplier Issues

- Failure of a third part
- Contractual issues

# Project risks

- **Organizational issues**

- such as shortages of people, skills or training, problems with communicating and responding to test results, bad expectations of what testing can achieve and complexity of the project team or organization.

- **Technical issues**

- such as problems related to ambiguous, conflicting or not prioritized requirements

- **Supplier issues**

- problems with underlying platforms or hardware
- failure to consider testing issues in the contract
- failure to properly respond to the issues when they arise

# Product risks

- Product risk is a risk directly related to the test object
- Product risk is the possibility that the system or software might fail to satisfy some reasonable customer, user, or stakeholder expectation

# Product risks

- Product risks = Potential failure areas in the software
- They are a risk to the quality of the product, i.e.:
  - Failure-prone software delivered
  - Software/hardware could cause harm to an individual or company
  - Poor software characteristics (e.g. functionality, reliability, usability and performance).
  - Software that does not perform its intended functions.
- Risks are used to decide where to start testing and where to test more.



# Product risks

- Testing is used to
  - reduce the risk of an adverse effect occurring ,
  - Reduce the impact of an adverse effect
- In a risk based approach the risks identified may be used to:
  - Determine the test techniques to be employed.
  - Determine the extent of testing to be carried out.
  - Prioritize testing in an attempt to find the critical defects as early as possible
  - Determine whether any non testing activities could be employed to reduce risk e.g. providing training to inexperienced designers).

# Risk analysis

- Identifying the risk items
- Determine the likelihood and impact for each item
- Use a rating scale (1 10) classify the level of risk for each item
- Priority the risk items according to their rating values

# Risk analysis

- Risk analyses are educated guesses! Make sure that you follow up and revisit the risk analysis at key project milestones
- If you're following a V model, you might perform the analysis during
  - the requirements phase
  - at the end of the design phase
  - at the end implementation phase
  - prior to starting unit test, integration test, and system test
  - during testing
- You might find you have discovered new risks or found that some risks weren't as risky as you thought and increased your confidence in the risk analysis .

# Risk analysis

- **Risk-based testing** also involves measuring how well we are doing at finding and removing defects in critical areas, as was shown in the table:

Product risk areas	Unresolved defects		Test cases to be run		
	Number	%	Planned	Actual	%
<i>Performance, load, reliability</i>	304	28	3,843	1,512	39
<i>Robustness, operations, security</i>	234	21	1,032	432	42
<i>Functionality, data, dates</i>	224	20	4,744	2,043	43
<i>Use cases, user interfaces, localization</i>	160	15	498	318	64
<i>Interfaces</i>	93	8	193	153	79
<i>Compatibility</i>	71	6	1,787	939	53
<i>Other</i>	21	2	760	306	40
	1,107	100	12,857	5,703	44

# Risk analysis

- Analyze risks early in the project.
- You should manage risks appropriately, based on likelihood and impact, but do not confuse impact with likelihood or vice versa
- The goal of risk based testing should not be cannot practically be a risk free project.
- Best practices in risk management to achieve a project outcome that balances risks with quality, features, budget and schedule.

# Defect management

- Defect
  - Discrepancies between actual and expected test outcomes .
- Defect management
  - The process of recognizing, investigating , taken action and disposing of incidents.
- Defect report
  - A report document reporting on any event that occurred, e.g. during testing, which requires investigation .

# Defect management

- What is the objectives of a Defect report?
  - 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 reports

- To write a good Defect report you must keep in mind the following questions:
  - What is the objective of the report?
  - What is the purpose?
  - Who is the reader(s)?
  - What goes into it?



# Defect reports

- What goes in a Defect report?
  - A description of some situation, behavior or event that occurred.
  - One or two screens with information gathered by a defect tracking tool.
  - A description of the steps done to reproduce and isolate the incident.
  - The impact of the problem.
  - Classification information i.e. the scope , severity and priority of the defect).

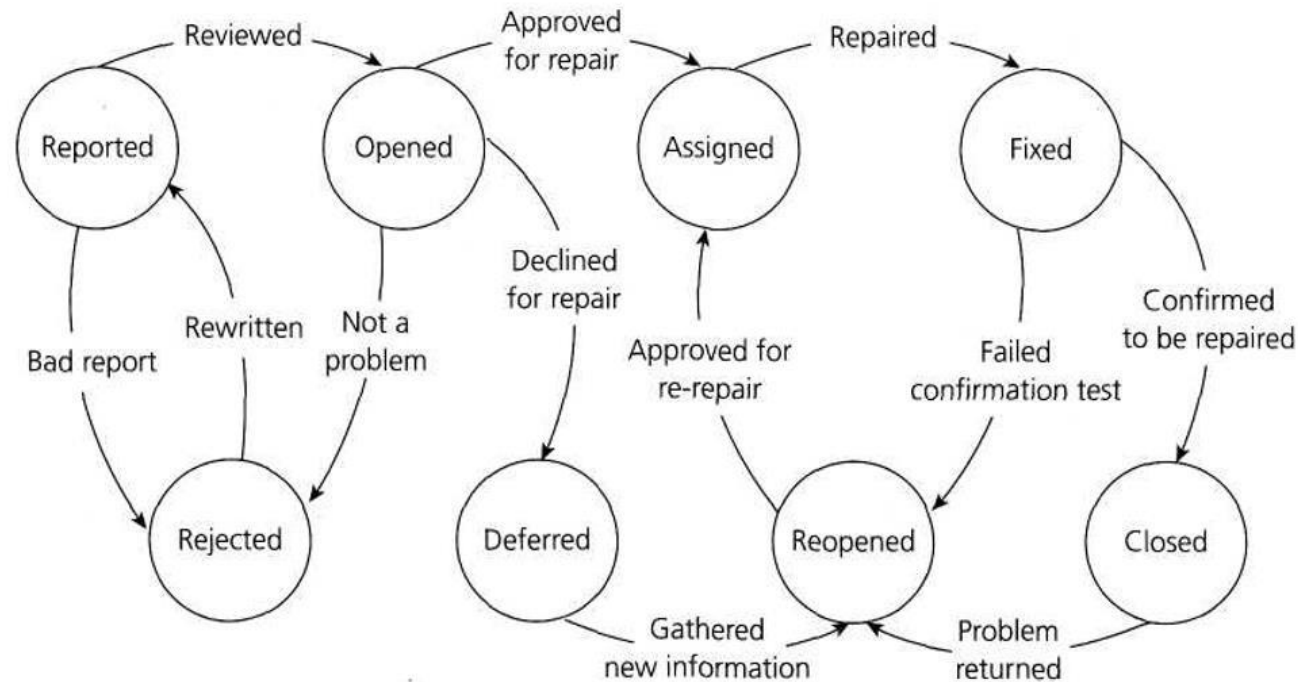
# Defect reports

- A level of priority , assigned by the test managers
- The risks, costs, opportunities and benefits associated with fixing or not fixing the defect assigned by the project team or a committee.
- The root cause , captured by the programmer,
  - the phase of introduction
  - the phase of removal
- Conclusions and recommendations captured by the managers, programmers or others
- Throughout the life cycle of the Defect report, the defect tracking system should allow each person who works on the Defect report to enter status and history information.

# Defect reports

- When to raise incidents?
  - During development, review, testing or use of a software product.

Statuses of Defect reports:



# Defect reports

Details of the Defect report may include (cf. IEEE 829):

Date: \_\_\_\_\_

Project: \_\_\_\_\_

Programmer: \_\_\_\_\_ Tester: \_\_\_\_\_

Program/Module: \_\_\_\_\_ Build/Revision/Release: \_\_\_\_\_

Software Environment: \_\_\_\_\_ Hardware Environment: \_\_\_\_\_

Status of the incident \_\_\_\_\_

Number of Occurrences: \_\_\_\_\_ Severity: \_\_\_\_\_ Impact \_\_\_\_\_ Priority \_\_\_\_\_

Detailed Description: \_\_\_\_\_ (logs, databases, screenshots)

Expected result / Actual result: \_\_\_\_\_

Change history \_\_\_\_\_

References (including the identity of the test case specification that revealed the problem) \_\_\_\_\_

Assigned To: \_\_\_\_\_

Defect Resolution: \_\_\_\_\_