# Test Plan

* Identification of what needs to be tested
* Specification of how the testing will be performed
* Identification of needed resources
* Testing activity breakdown and scheduling
* Risk analysis

# Test Case Development

1. Identify the purpose of the test
2. Identify the system under test (SUT)
3. Identify the environment that needs to be set up for running the test case
4. Identify the input data to be used for the test case
5. Identify the steps to be followed to execute the test
6. Identify the expected results that are considered to be correct.
7. Identify a step to compare the actual results to the expected results
8. Identify any relationships between this test and other tests

# Traceability Matrix

A traceability matrix is a document, usually in the form of a table, which correlates baselined items that require a many-to-many relationship to determine the completeness of the relationship.

|  |
| --- |
|  |

# To Automate or not to Automate

Criteria

* How repetitive are the tests?
* How much effort is involved in automating?
* How much manual intervention is needed for the test?
* How much does the automation tool cost?

# Writing Test Cases

Automated

* Writing of the scripts in the automation language

Manual

* Writing detailed step-by-step instructions for executing the test and validating the results

Should also have change history documentation

1. What was the change?
2. Why was the change necessary?
3. Who made the change?
4. When was the change made?
5. How was the change implemented?
6. Other files affected by the change

# Test Case Execution

Updating of the defect repository

* Defect Repository:
  + The primary vehicle of communication between test team and development team
  + Contains information about all defects

# Collecting and Analyzing Metrics

**Project Metrics**: indicate how the project is planned and executed

* Effort Variance
* Schedule Variance
* Effort distribution across phases

**Progress Metrics**: tracks how different activities of the project are progressing

* Defect find rate
* Defect fix rate
* Outstanding defects rate
* Priority outstanding rate
* Defect trend
* Defect classification trend
* Weighted defects trend
* Defect cause trend
* Component-wise defect distribution
* Defect density and defect removal rate
* Age analysis of outstanding defects
* Introduced and reopened defects trend

**Productivity Metrics**: helps in planning and estimating test activities

* Defects per 100 hours of testing
* Test cases executed per 100 hours of testing
* Test cases developed per 100 hours of testing
* Defects per 100 test cases
* Defects per 100 failed test cases
* Test phase effectiveness
* Closed defect distribution

**Release Metrics**

* A consideration of all previous metrics

# Test Reporting

Test Incident Report

* An entry in the defect repository

Test Cycle Report

* Summary of activities during this cycle
* Defects uncovered
* Defects fixed since last cycle
* Outstanding defects

Test Summary Report

* Summary of activities during this cycle
* Variance from planned activities
* Summary of results
* Comprehensive assessment and recommendation for release

# Test Case Database

Test case

* Test case ID
* Test case name
* Test case owner
* Associated files for the test case
* Module ID

Test case run history

* Test case ID
* Run date and time
* Time taken
* Run status (success/failure)

Test case defect report

* Test case ID
* Defect ID

Defect Details

* Defect ID
* Defect status (open/closed)
* Defect classification
* Defect description
* Affected artifact(s)
* Any relevant version information
* Environmental information
* Who encountered the problem
* Date and time of defect occurrence

Defect test details

* Defect ID
* Test case ID

Fix Details

* Defect ID
* Fix details

Communication

* Test case ID
* Defect ID
* Communication details

# Defect Classification

Extreme

* Product crashes or unstable
* Needs to be fixed immediately

Critical

* Basic functionality of the product not working
* Needs to be fixed before next test cycle starts

Important

* Extended functionality of the product not working
* Does not affect the progress of testing
* Fix it before the release

Minor

* Product behaves differently
* No impact on the test team or customers
* Fix it when time permits

Cosmetic

* Minor irritant
* Need not be fixed for this release

# Defect Priority

1. [Highest] Fix the defect on highest priority; fix it before the next build
2. [High] Fix the defect on high priority before next test cycle
3. [Moderate] Fix the defect on moderate priority when time permits, before the release
4. [Low] Postpone this defect for the next release
5. [Lowest] Live with this defect

# Testing Categories

Static Analyses

* Inspections
* Walkthroughs
* Buddy checks

Dynamic Analyses

* Unit testing
* Integration testing
* System testing
* Acceptance testing
* Regression testing
* Stress testing

## Inspections

* Planned structured meeting
* All individuals prepare for it
* Team of 3 – 6 people, led by impartial moderator
* Presenter is the reader not producer

## Walkthroughs

* Planned meeting where only the presenter must prepare
* 2 – 7 people, led by producer/author
* Presenter is the producer

## Buddy Check

* Having someone else review the artifact

## Unit Testing

Unit testing is a software development process in which the smallest testable parts of an application, called units, are individually verified. Unit testing is often automated but it can also be done manually.

## Integration Testing

* Testing the interaction of the various modules
* Testing the interaction with other systems

## System Testing

* Testing the complete integrated system
* Testing to the specified requirements

## Acceptance Testing

* Testing from the point of view the clients and uses of the system

## Regression Testing

* Testing done during maintenance
* Ensures that enhancements or fixes does not introduce defects to existing functionality

## Stress Testing

* Testing the system beyond the limits of specified requirements or resources

# Bibliography

Collofello, J. S. (1988). Introduction to Software Verification and Validation. *SEI Curriculum Module SEI-CM-13-1.1*.

Desikan, S., & Ramesh, G. (2006). *Software Testing Principles and Practices.* Pearson Education.

Kaik, K., & Tripathy, P. (2008). *Software Testing and Quality Assurance Theory and Practice.* Wiley.