Software Testing

L03. The testing process

Christian Millán

Autumn 2020

Introduction

Issues of testing implementation:

- Effectiveness
- Efficiency

Introduction

Reduction of the percentage of undetected errors

Vs

Performance of those tests with fewer resources

Introduction

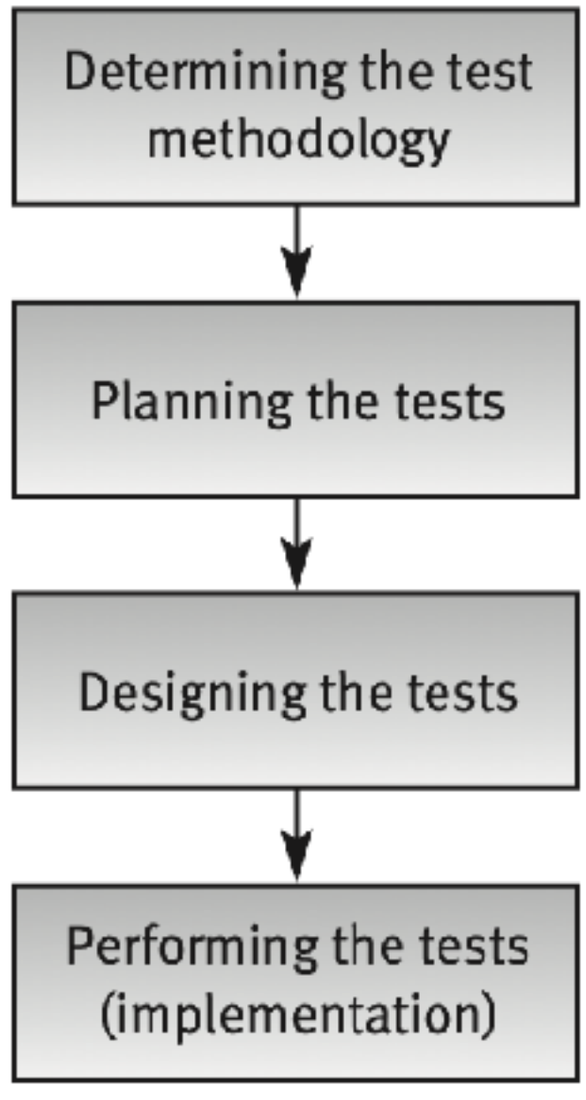
How to… ?

Improve test **efficiency** by reducing the total number of required test cases and, at the same time, increasing test **effectiveness**.

Realizing the potential of automated computerized testing.

The testing process

Planning, design and performance of testing are carried out throughout the software development process.



The testing process

1. Determining the test methodology phase

The main issues that testing methodology has to contend with are:

- The appropriate required software quality standard
- The software testing strategy.

The testing process

1.1. Determining the appropriate software quality standard

- Example 1: A software package for a hospital patient bed monitor requires the **highest software quality standard** considering the possibly severe consequences of software failure.

The testing process

1.1. Determining the appropriate software quality standard

- Example 2: A package developed for handling feedback information for an organization's internal employee training program could make do with a **medium-level software quality standard**, assuming that the cost of failure is relatively low (or much lower than that of Example 1).

The testing process

1.1. Determining the appropriate software quality standard

- Example 3: A software package has been developed for sale to a broad range of organizations. The sales prospects justify **higher quality standards** than would a custom-made software package having similar characteristics yet developed to serve a single customer.

The testing process

1.1. Determining the software testing strategy

- The testing strategy: should a big bang or incremental testing strategy be adopted? If **incremental testing** is preferable, should testing be performed bottom-up or top-down?

- Which parts of the testing plan should be performed according to the **white box testing model**?

- Which parts of the testing plan should be performed according to the **automated testing model**?

3.1. Planing the test

The tests to be planned include:

- Unit tests
- Integration tests
- System tests.

3.1. Planing the test

Before initiating a specific test plan:

- What to test?
- Which sources to use for test cases?
- Who is to perform the tests?
- Where to perform the tests?
- When to terminate the tests?

3.1. Planing the test

- **What to test?**

A full and comprehensive software test plan that requires performing:

- **unit tests** for ~~all~~ the individual units,

- **integration tests** for ~~all~~ the unit integrations,

- and a **system test** to test the software package as a whole.

3.1. Planing the test

- **What to test?**

Rating units, integrations and applications

Factor A: **Damage severity level**. The severity of results in case the module or application fails.

Factor B: **Software risk level**. The level of risk represents the probability of failure.

See Table.

3.1. Planing the test

- **What to test?**

Generating a combined rating based on the two factors

$C = A + B$

$C = k \times A + m \times B$

$C = A \times B$

3.1. Planing the test

- **Who perform the test?**

- **Integration tests**, but especially **unit tests**, are generally performed by the software **development team**.
- **System tests** are usually performed by an **independent testing team**
- In cases of **large software systems**, **more than one testing team** can be employed to carry out the system tests.

3.1. Planing the test

- **Who perform the test?**

In small software development organizations

- Testing by **another development team.**
- **Outsourcing** of testing responsibilities.

3.1. Planing the test

- **Where to perform the test?**

- The software developer’s site
- The customer’s site.
- The consultan’s site.

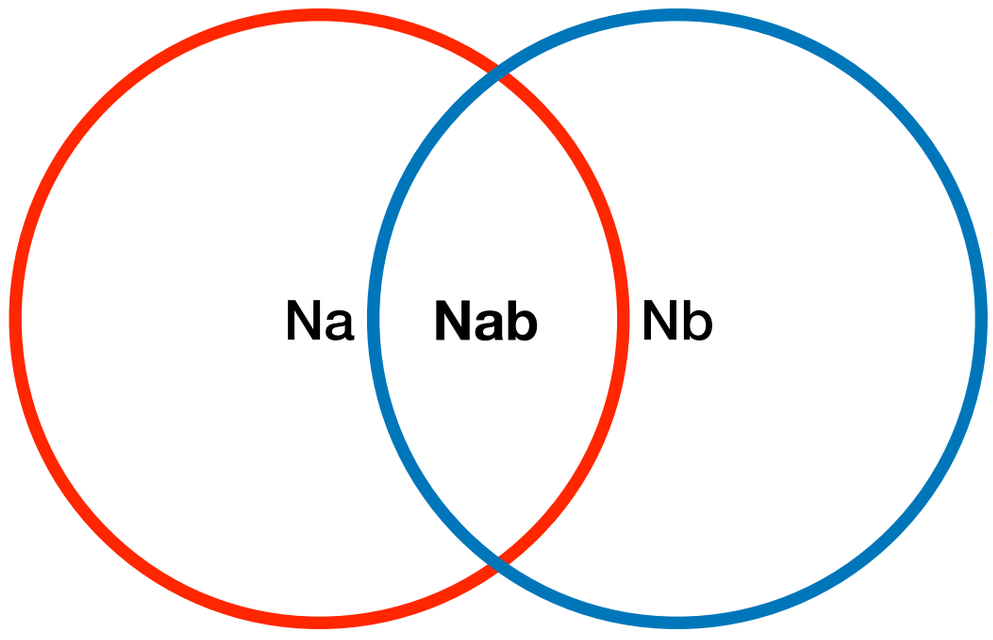
3.1. Planing the test

- **When are tests terminated?**

- The completed implementation route.
- The mathematical models application route.
- The error seeding route.
- The dual independent testing teams route.
- Termination after resources have petered out.

3.1. Planing the test

- **When are tests terminated?** The dual independent testing teams route.
-



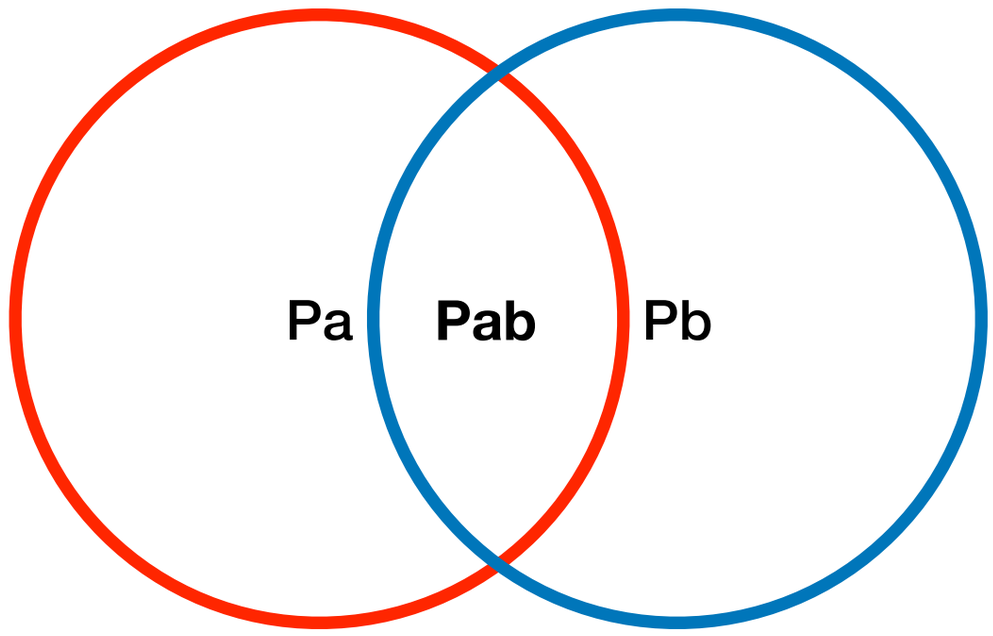
Na = number of errors detected by Team A

Nb = number of errors detected by Team B

Nab = number of errors detected by both Team A and Tema B

3.1. Planing the test

- **When are tests terminated?** The dual independent testing teams route.



Pa = proportion of errors detected by Team A

Pb = proportion of errors detected by Team B

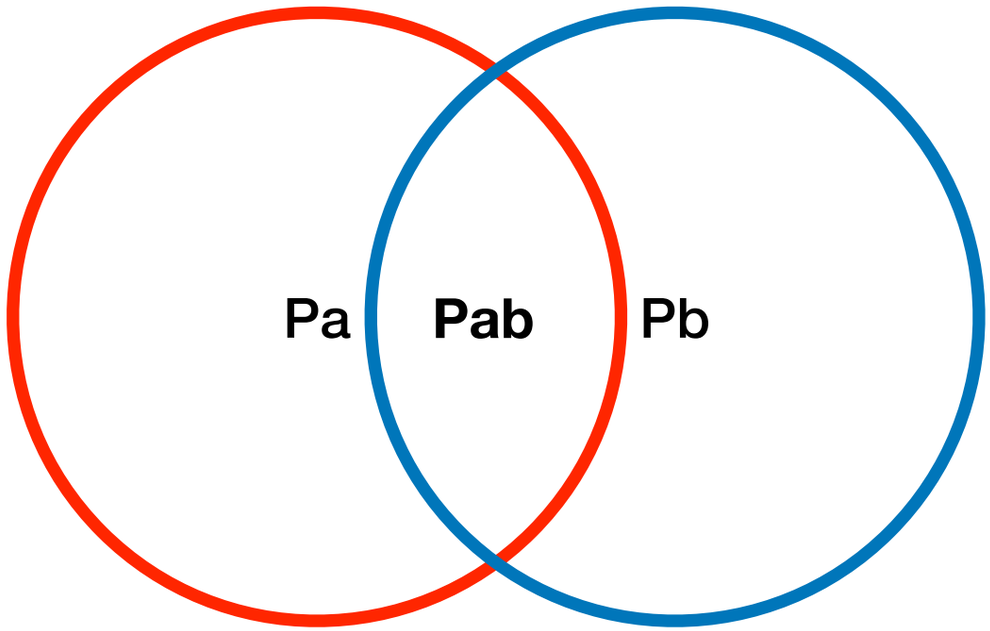
Pab = proportion of errors detected by both Team A and Tema B

P(a)(b) = proportion of errors undetected by both teams

N(a)(b) = number of errors undetected by both teams
N = total number of errors in the software package/program.

3.1. Planing the test

- **When are tests terminated?**



(1) Pab=Pa×Pb=Nab/N
(2) Pa = Na/N
(3) Pb = Nb/N
(4) P(a)(b)=(1–Pa)×(1–Pb)

(5) **N**=Na×Nb/Nab
(6) **Pa**=Nab/Nb
(7) **Pb**=Nab/Na
(8) **P(a)(b)** =(1–Pa)×(1–Pb)=
= (Na – Nab) × (Nb – Nab) / (Na × Nb)
(9) N(a)(b) =(Na–Nab)×(Nb–Nab)/Nab

Pa = proportion of errors detected by Team A

Pb = proportion of errors detected by Team B

Pab = proportion of errors detected by both Team A and Tema B

P(a)(b) = proportion of errors undetected by both teams

N(a)(b) = number of errors undetected by both teams
N = total number of errors in the software package/program.

3.1. Planing the test

- **When are tests terminated?** The dual independent testing teams route.

The developers of *Super Magic*, an electronic game for children aged 4–7, have decided to employ the dual test method. They determined their testing termination level to be residual undetected errors of 2.5%. As a complementary "tool" for identifying the undetected errors, they planned wide application of *beta site testing* before beginning the marketing of Super Magic.

3.1. Planing the test

- **When are tests terminated?** The dual independent testing teams route.

The testing teams summarized their achievements after eight weeks of testing and regression testing as follows:

Team A detected 160 errors (Na = 160)

Team B detected 180 errors (Nb = 180)

Nab = 144

3.1. Planing the test

- **When are tests terminated?** The dual independent testing teams route.

The testing teams summarized their achievements after eight weeks of testing and regression testing as follows:

Team A detected 160 errors (Na = 160)

Team B detected 180 errors (Nb = 180)

Nab = 144

P(a)(b) = 16 × 36 / (160 × 180) = 0.02

N(a)(b) = 16 × 36 / 144 = 4

N = 160 × 180 / 144 = 200

(5) **N**=Na×Nb/Nab
(6) **Pa**=Nab/Nb
(7) **Pb**=Nab/Na
(8) **P(a)(b)** =(1–Pa)×(1–Pb)= (Na – Nab) × (Nb – Nab) / (Na × Nb)
(9) **N(a)(b)** =(Na–Nab)×(Nb–Nab)/Nab

3.1. Planing the test

The tests planning documentation

The software test plan (STP)

1. **Scope of the tests**

1.1 The software package to be tested (name, version and revision)

1.2 The documents that provide the basis for the planned tests (name and version for each document)

2. **Testing environment**

2.1 Testing sites

2.2 Required hardware and firmware configuration

2.3 Participating organizations

2.4 Manpower requirements

2.5 Preparation and training required of the test team

3. **Test details (for each test)**

3.1 Test identification

3.2 Test objective

3.3 Cross-reference to the relevant design document and the requirement document

3.4 **Test class**

3.5 Test level (unit, integration or system tests)

3.6 Test case requirements

3.7 Special requirements (e.g., measurements of response times, security requirements)

3.8 Data to be recorded

4. **Test schedule** (for each test or test group) including time estimates for the following:

4.1 Preparation

4.2 Testing

4.3 Error correction

4.4 Regression tests

3.2. Test Design

The products of the test design stage are:

- Detail design and procedures for each test
- Test case database/file

3.2. Test Design

Software Test Description (STD)

The **test procedures** and the **test case** database/file may be documented in a “software test procedure” document and “test case file” document or in a single document called the “software test description” (STD).

3.2. Test Design

Software Test Descriptions (STD) – template

1. Scope of the tests

1.1 The software package to be tested (name, version and revision)

1.2 The documents providing the basis for the designed tests (name and version for each document)

2. Test environment (for each test)

2.1 Test identification (the test details are documented in the STP)

2.2 Detailed description of the operating system and hardware configuration and the required switch settings for the tests

2.3 Instructions for software loading

3. Testing process

3.1 Instructions for input, detailing every step of the input process

3.2 Data to be recorded during the tests

4. Test cases (for each case)

4.1 Test case identification details

4.2 Input data and system settings

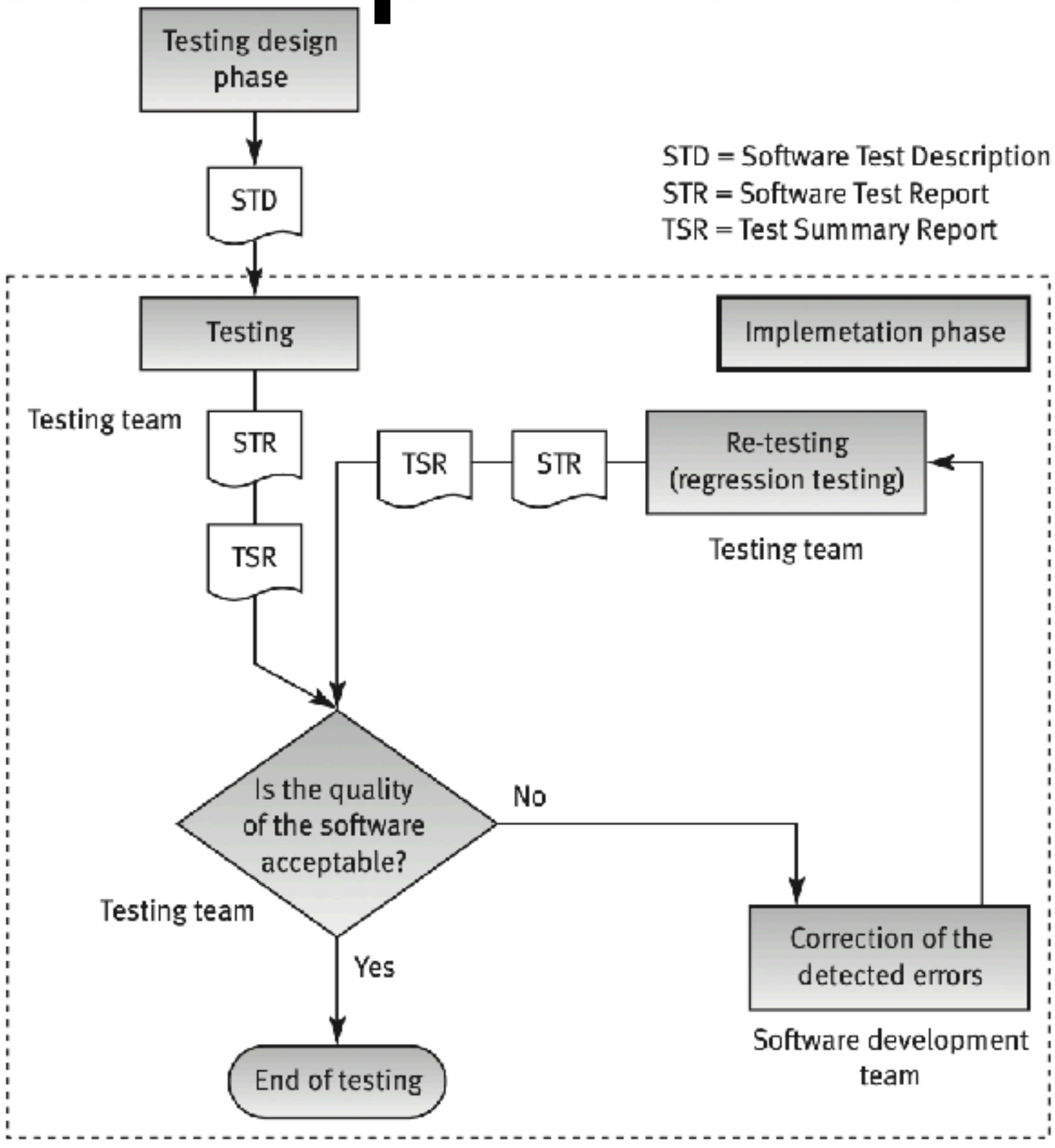
4.3 Expected intermediate results (if applicable)

4.4 Expected results (numerical, message, activation of equipment, etc.)

5. Actions to be taken in case of program failure/cessation

6. Procedures to be applied according to the test results summary

3.3. Test implementation



3.3. Test Implementation

Software Test Report (STR) – template

1 Test identification, site, schedule and participation

1.1 The tested software identification (name, version and revision)

1.2 The documents providing the basis for the tests (name and version for each document)

1.3 Test site

1.4 Initiation and concluding times for each testing session

1.5 Test team members

1.6 Other participants

1.7 Hours invested in performing the tests

2 Test environment

2.1 Hardware and firmware configurations

2.2 Preparations and training prior to testing

3 Test results

3.1 Test identification

3.2 Test case results (for each test case individually)

3.2.1 Test case identification

3.2.2 Tester identification

3.2.3 Results: OK / failed

3.2.4 If failed: detailed description of the results/ problems

4 Summary tables for total number of errors, their distribution and types

4.1 Summary of current tests

4.2 Comparison with previous results (for regression test summaries)

5 Special events and testers' proposals

5.1 Special events and unpredicted responses of the software during testing

5.2 Problems encountered during testing

5.3 Proposals for changes in the test environment, including test preparations

5.4 Proposals for changes or corrections in test procedures and test case files