# Software Testing

An Introduction

WE ADDED A NEW PERFORMANCE TEST, BUT LEARNED THAT THE TEST ITSELF IS FLAWED.
Dilbert.com DilbertCartoonist@gmail.com
NOW OUR PRODUCT FAILS OUR OWN TESTS AND OUR CUSTOMERS ARE ASKING TO SEE THE TEST RESULTS.
8-11-10 ©2010 Scott Adams, Inc./Dist. by UFS, Inc.
DO I HAVE PERMISSION TO FAKE THE TEST DATA?
I DIDN'T EVEN KNOW DATA CAN BE REAL.

# The Triangle Program [Myers]

“A program reads three integer values. The three values are interpreted as representing the lengths of the sides of a triangle. The program prints a message that states whether the triangle is scalene, isosceles, or equilateral.”

*Write a set of test cases to test this program*

# The Triangle Program [Myers]

*Test cases for:*

1. valid scalene triangle ?
2. valid equilateral triangle ?
3. valid isosceles triangle ?
4. 3 permutations of previous ?
5. side = 0 ?
6. negative side ?
7. one side is sum of others ?
8. 3 permutations of previous ?
9. one side larger than sum of others ?
10. 3 permutations of previous ?
11. all sides = 0 ?
12. non-integer input ?
13. wrong number of values ?
14. for each test case: is expected output specified ?
15. check behaviour after output was produced ?

# The Triangle Program [Myers]

*Test cases for:*

1. is it fast enough ?
2. doesn’t it use too much memory ?
3. is it learnable ?
4. is it usable for intended users ?
5. is it secure ?
6. does it run on different platforms ?
7. is it portable ?
8. is it easily modifiable ?
9. is the availability sufficient ?
10. is it reliable ?
11. does it comply with relevant laws ?
12. doesn‘t it do harm to other applications ?
13. . . . . . .

# Paradox of Software Testing

Testing is:

- important
- much practiced
- 30-50% of project effort
- expensive
- time critical
- not constructive
  (but sadistic?)

But also:

- ad-hoc, manual, error-prone
- hardly theory / research
- no attention in curricula
- not cool :
  “if you’re a bad programmer
  you might be a tester”

*Attitude is changing:*
☞ *more awareness*
☞ *more professional*

# Testing : What

[Myers] “Testing is the process of executing a program with the intent of finding errors.”

[ISO] “Testing is a technical operation that consists of the determination of one or more characteristics of a given product, process, or service according to a specified procedure.”

[Pol] “Testing is a process of planning, preparation and measurement aimed at establishing the characteristics of an information system and demonstrating the difference between the actual and the required status.”

# Testing:  A Definition

Software testing is:

- a technical process,
- performed by executing / experimenting with a product,
- in a controlled environment, following a specified procedure,
- with the intent of measuring one or more characteristics / quality of the software product
- by demonstrating the deviation of the actual status of the product from the required status / specification.

# Testing:  A Definition

Testing is not:

- static analysis
- reviewing or inspection
- walk-through
- debugging
- bug fixing
- auditing
- quality control
- validation of the specification

But testing:

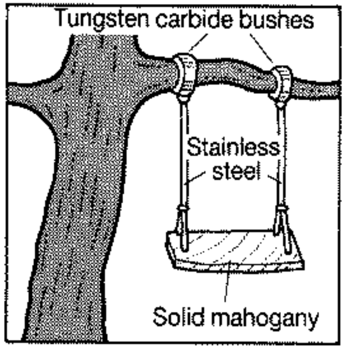
- is one of the instruments for measuring quality
- increases value of product by establishing confidence in its quality
- helps in assessing risk of putting product into operation

# Testing:

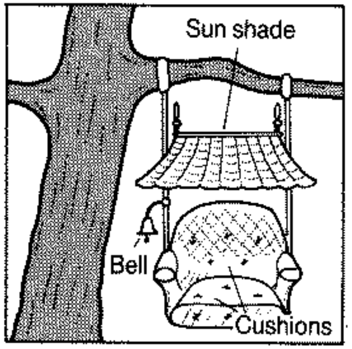
# Principles, Problems,

# and  Challenges

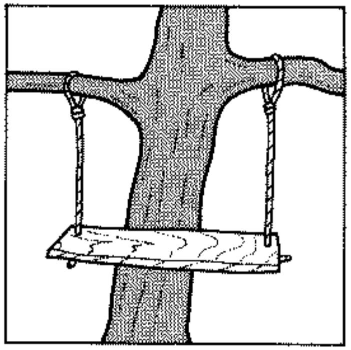
# A Testing Challenge



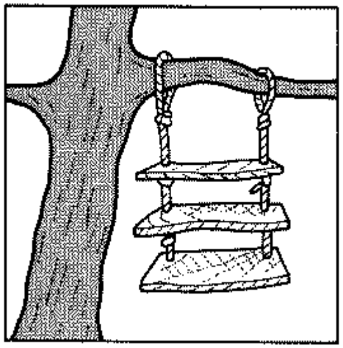
What Product Marketing specified



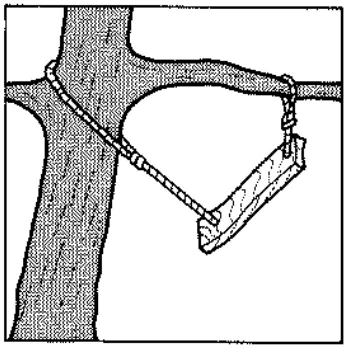
What the salesman promised



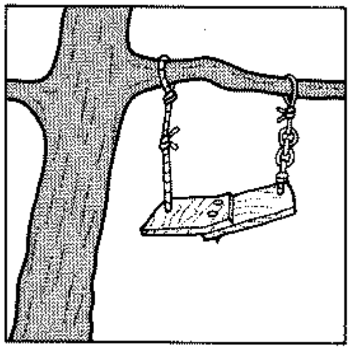
Design group's initial design



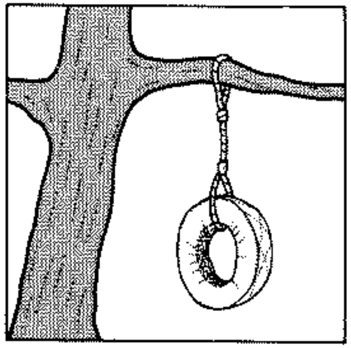
Corp. Product Architecture's modified design



Pre-release version



General release version

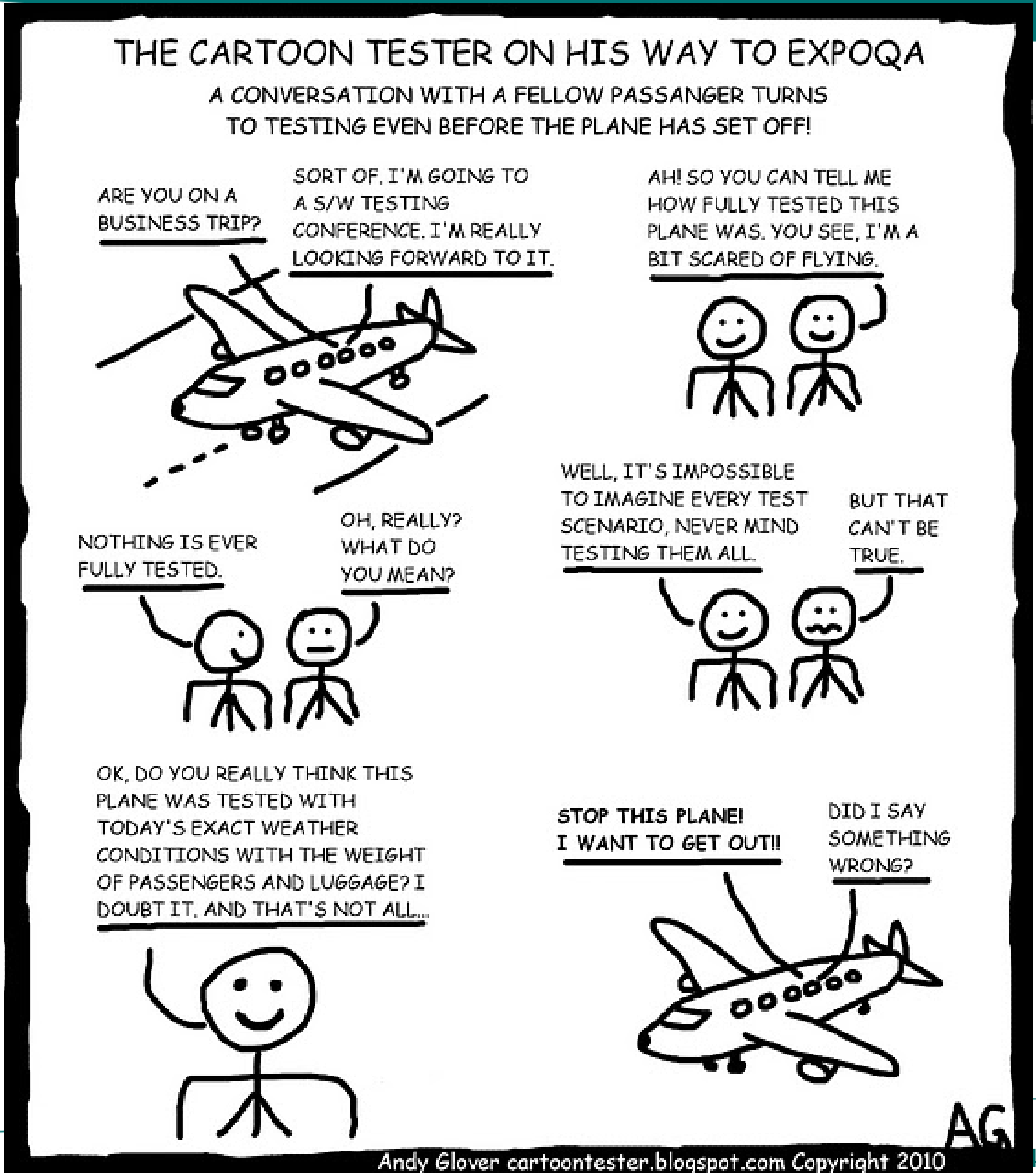


What the customer actually wanted

# Some Testing Principles

- Testing starts during the requirements phase
- The programmer shall not be the tester
- A test case specifies the test inputs and the expected outputs
- Test cases shall also cover invalid and unexpected inputs
- Test cases shall test that the program does what it should do and that it does not do what it should not do
- Test cases shall be recorded for reuse
- A test is successful when it detects an error  
  ( but the project manager thinks differently ! )
- No risk, no test

# A Testing Challenge


THE CARTOON TESTER ON HIS WAY TO EXPOQA
A CONVERSATION WITH A FELLOW PASSANGER TURNS TO TESTING EVEN BEFORE THE PLANE HAS SET OFF!
ARE YOU ON A BUSINESS TRIP?
SORT OF. I'M GOING TO A S/W TESTING CONFERENCE. I'M REALLY LOOKING FORWARD TO IT.
AH! SO YOU CAN TELL ME HOW FULLY TESTED THIS PLANE WAS. YOU SEE, I'M A BIT SCARED OF FLYING.
NOTHING IS EVER FULLY TESTED.
OH, REALLY? WHAT DO YOU MEAN?
WELL, IT'S IMPOSSIBLE TO IMAGINE EVERY TEST SCENARIO, NEVER MIND TESTING THEM ALL.
BUT THAT CAN'T BE TRUE.
OK, DO YOU REALLY THINK THIS PLANE WAS TESTED WITH TODAY'S EXACT WEATHER CONDITIONS WITH THE WEIGHT OF PASSENGERS AND LUGGAGE? I DOUBT IT. AND THAT'S NOT ALL...
STOP THIS PLANE! I WANT TO GET OUT!!
DID I SAY SOMETHING WRONG?
AG

Andy Glover cartoontester.blogspot.com Copyright 2010

# Challenges of Testing

# Problems of Testing

- Infinity of testing:
  - too many possible input combinations -- infinite breadth
  - too many possible input sequences -- infinite depth
  - too many invalid and unexpected inputs
- Exhaustive testing never possible:
  - when to stop testing ?
  - how to invent effective and efficient test cases with high probability of detecting errors ?
- Optimization problem of testing yield and invested effort
  - usually stop when time is over ......
- *In theory of (model-based) testing we can deal with infinity*

# Challenges of Testing

- No realistic reliability models for software

- How to measure the quality of a test suite ?

- Test automation
  - test activities can be boring and require precision
  - in particular regression testing
  - maintainability

- Bad specification or no specification at all
  - what do you test ?

- Many operating environments and contexts

# Challenges of Testing : The Oracle

- Oracle : expected output of the system under test

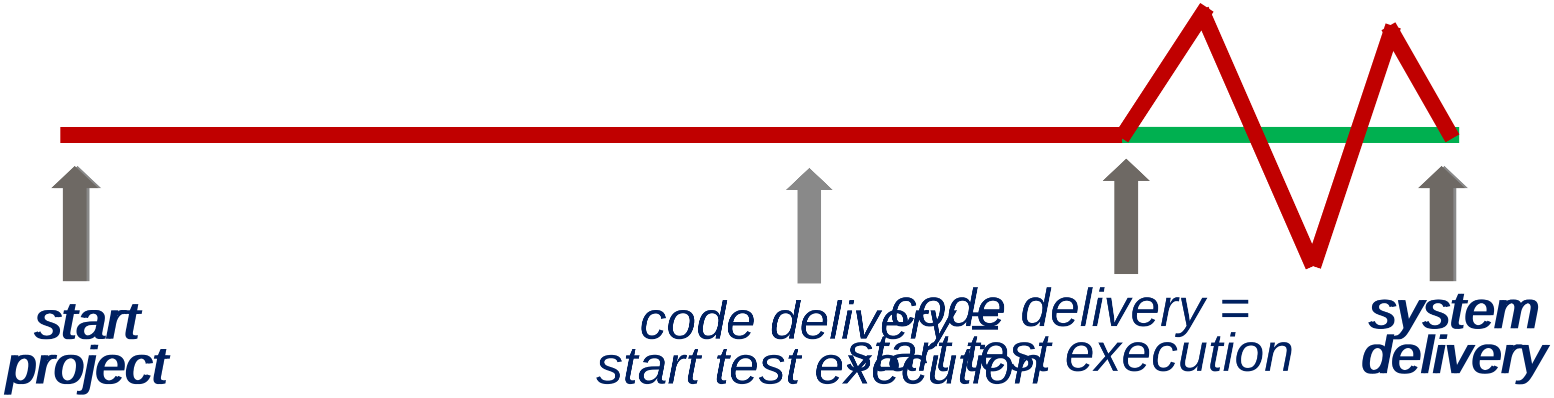
*A test case specifies the test inputs*
*and the expected outputs*

- Preferably automatic oracle ( effective oracle )
  - needs specification
    - executable
    - property
- Not always easy
  - $\pi$, pretty printer, display, search engine, autonomous car, ……

# Challenges of Testing :  Deadlines

- Moving implementation deadlines

  ......  but fixed delivery deadlines



**start project**

code delivery = start test execution

code delivery = start test execution

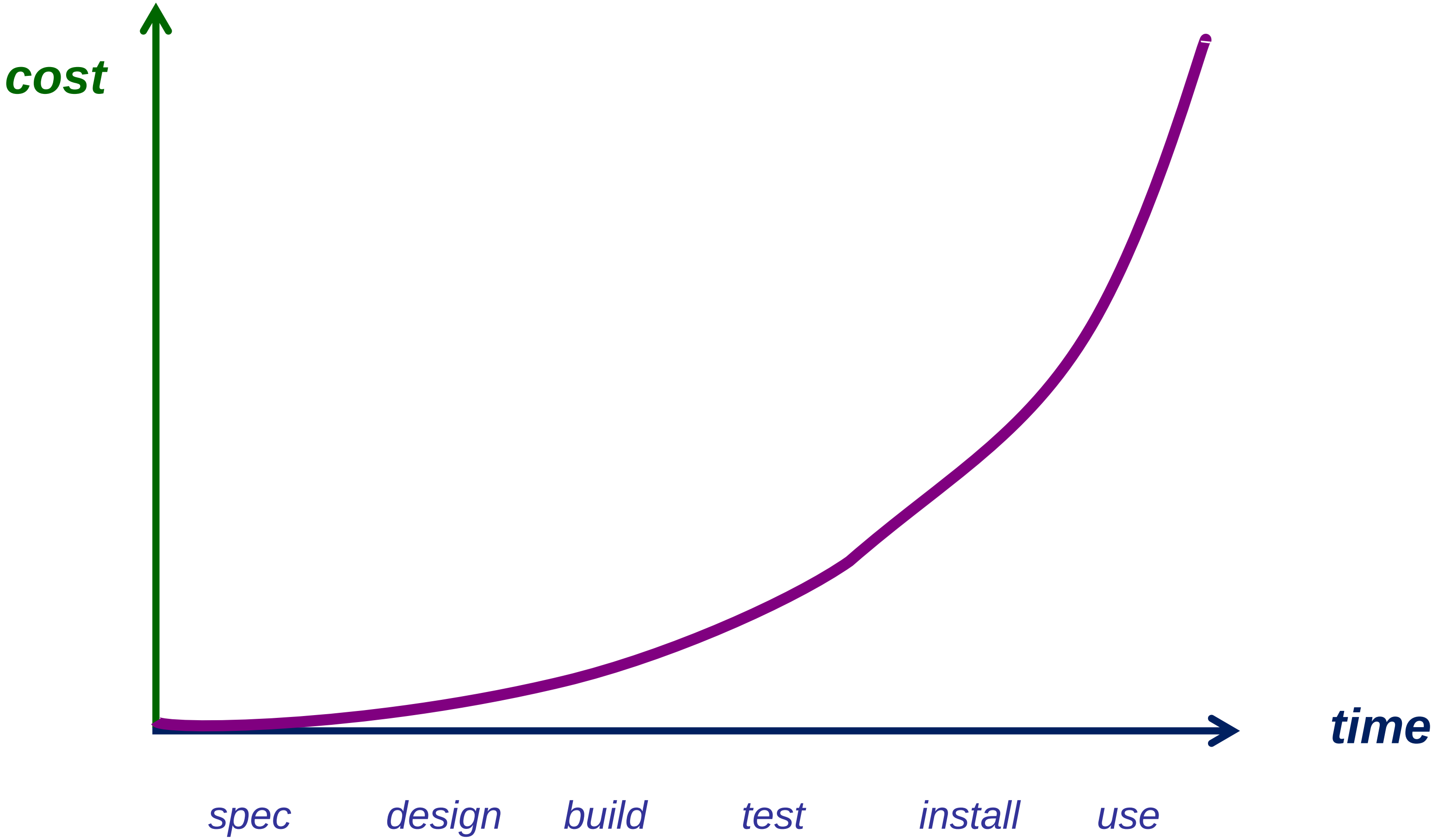
**system delivery**

# 7 Fundamental Principles of Testing *(ISTQB)*

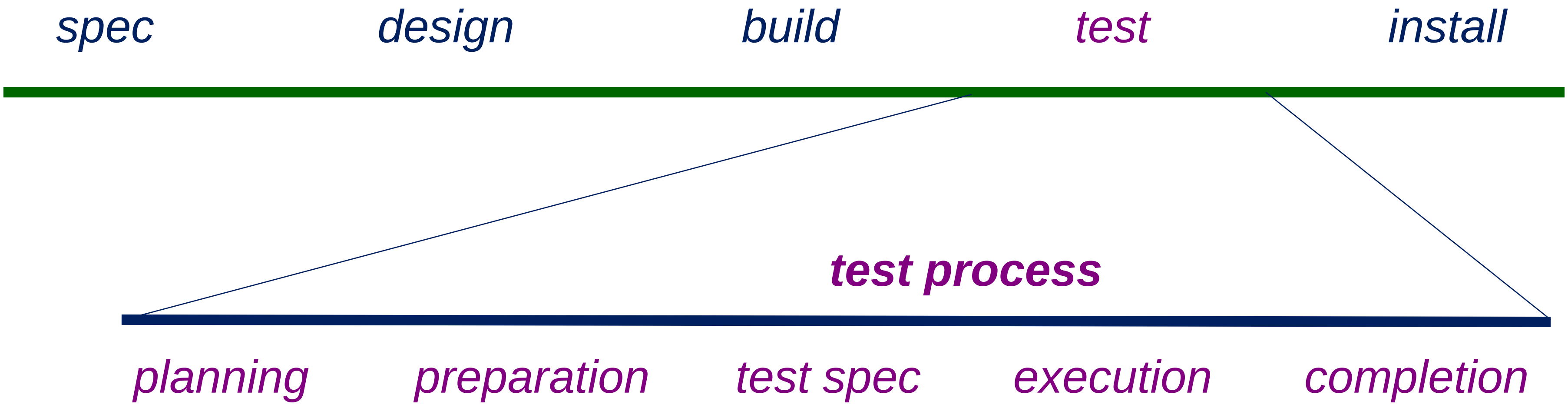
1. *Testing shows presence of defects*
2. *Exhaustive testing is impossible*
3. *Early testing saves*
4. *Defects cluster together*
5. *Pesticide paradox*
6. *Testing is context dependent*
7. *Absence-of-errors fallacy*

# Testing Process

# Software Development Trajectory
# Cost of Defects



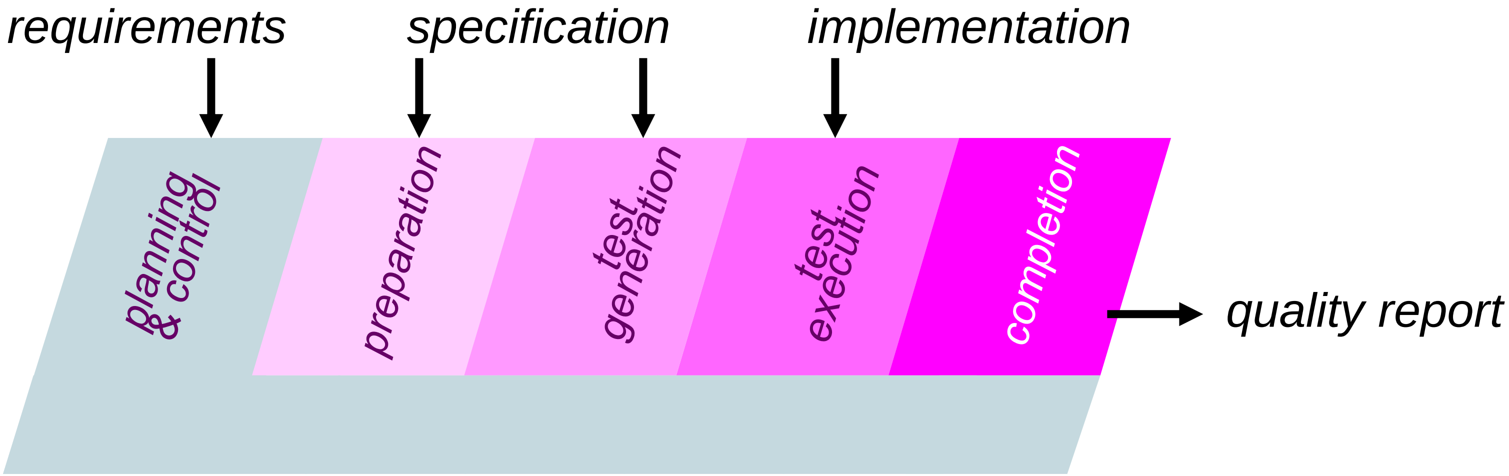
# Software Development Trajectory



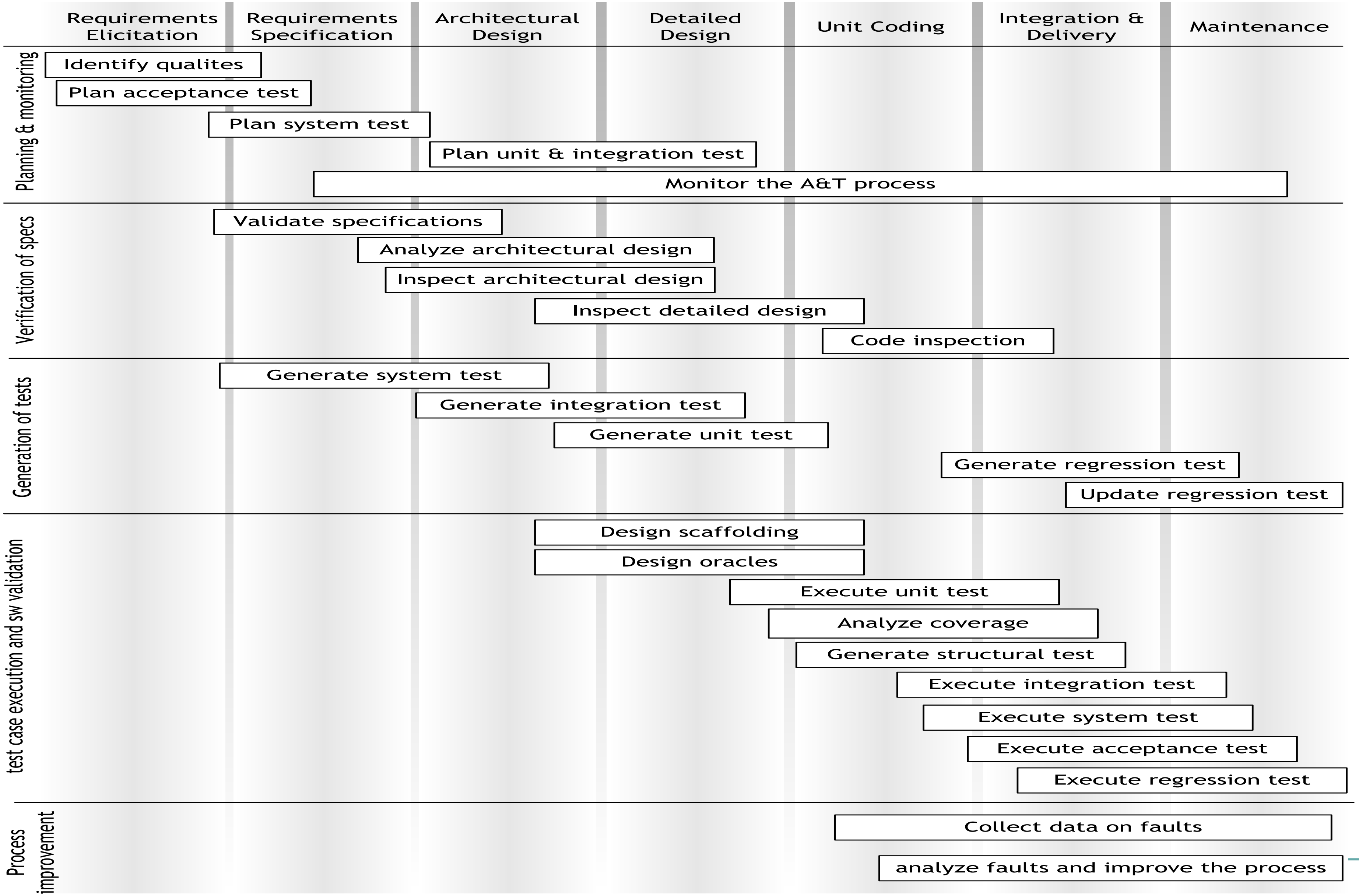
# Life Cycle :  Testing as a Process

*TMAP*  Testing as a process itself

- with its own phases
- in parallel with development process



# Testing Activities



Requirements Elicitation

Requirements Specification

Architectural Design

Detailed Design

Unit Coding

Integration & Delivery

Maintenance

Planning & monitoring

Identify qualites

Plan acceptance test

Plan system test

Plan unit & integration test

Monitor the A&T process

Verification of specs

Validate specifications

Analyze architectural design

Inspect architectural design

Inspect detailed design

Code inspection

Generation of tests

Generate system test

Generate integration test

Generate unit test

Generate regression test

Update regression test

test case execution and sw validation

Design scaffolding

Design oracles

Execute unit test

Analyze coverage

Generate structural test

Execute integration test

Execute system test

Execute acceptance test

Execute regression test

Process improvement

Collect data on faults

analyze faults and improve the process

# Some Points

- The quality process has different goals:
  1. improving a software product
  2. assessing the quality of the software product
  3. improving the quality process

- Testing depends on organization and application domain
- Cost-effectiveness depends on the extent to which techniques can be re-applied as the product evolves
- Planning and monitoring are essential to evaluate and refine the quality process.

# Independence of the Tester

Several levels of **independence**:

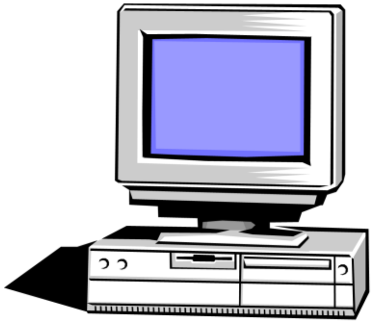
- tests designed by the developers
- tests designed by another person , e.g. from development team
- tests designed by a person from a different organizational group
- tests designed by a person from another organization or company

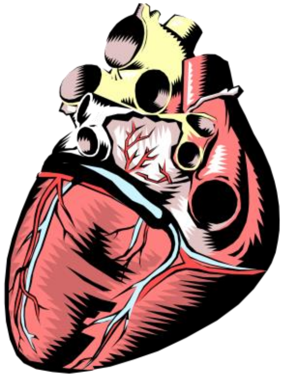
# Independence of the Tester

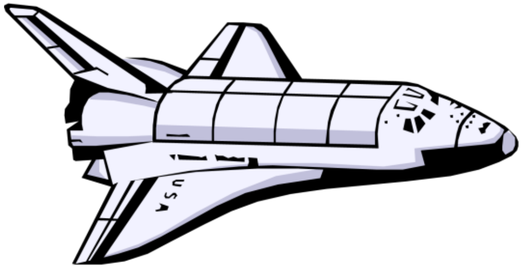
- Finding failures may be perceived as criticism against the product and against the author
- Testing can be destructive !
- The objective of testing must be clear
- If errors, defects or failures are communicated in a constructive way, bad feelings between the testers and the analysts, designers and developers can be avoided

# Testing and Quality

# Software is Everywhere!
# But what is its Quality?







# Quality : What

| | |
|---|---|
| *popular* | “how ‘good’ the product is” |
| [Crosby] | “Conformance to the requirements” |
| [Juran] | “Fitness for use” |
| [Weinberg] | “Value to someone” |
| [IS 9126] | “The totality of characteristics of an entity<br>that bear on its ability<br>to satisfy stated and implied needs” |
| [IS 9001] | “Manufactured according to well-defined processes” |

# Testing and Quality

Testing: *measuring the quality of a product;*

*obtaining confidence in the quality of product*

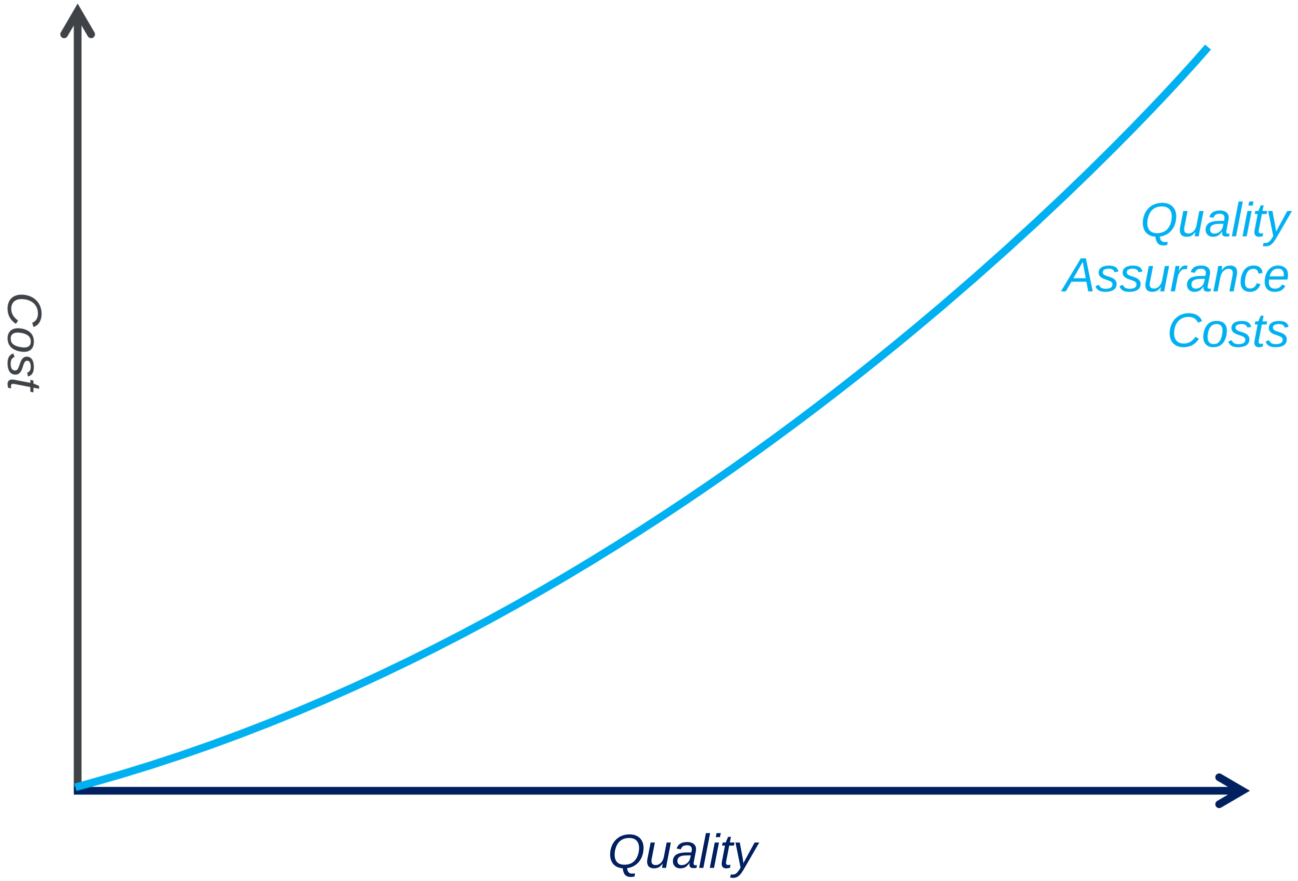
- How to specify quality ?
  - explicit / implicit / legal requirements
- How to classify quality ?
- How to quantify quality ?
- How to measure quality ?
- How to obtain good tests ?

# Classification into Quality Characteristics

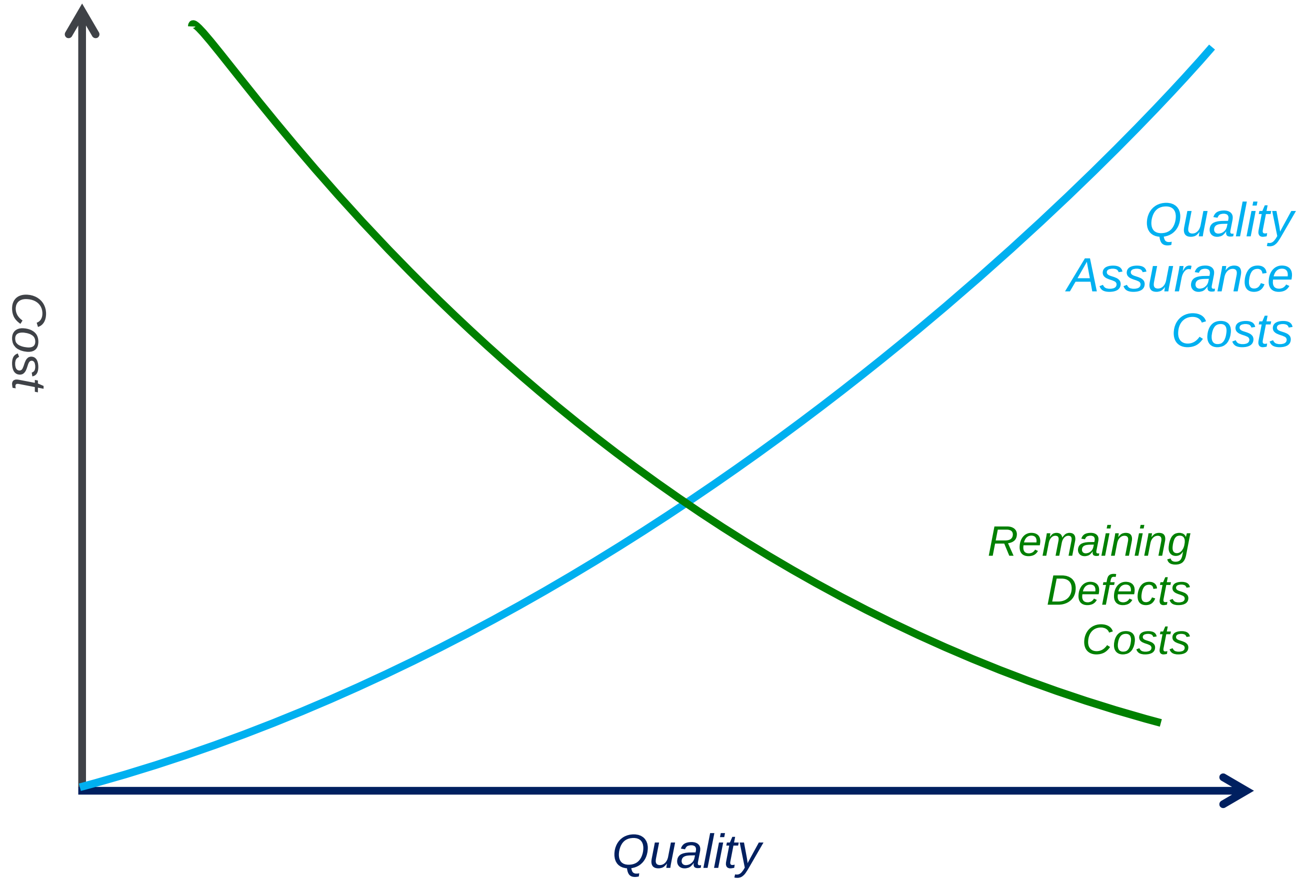
Characteristics according to IS 9126  ( IS 25010 )

- Functionality
  - suitability accuracy, security, compliance, interoperability
- Reliability
  - maturity, fault tolerance, recoverability
- Usability
  - understandability, learnability, operability
- Efficiency
  - time behaviour, resource utilization
- Maintainability
  - Analysability, changeability, stability, testability
- Portability
  - Adaptability, intallability, conformance, replaceability

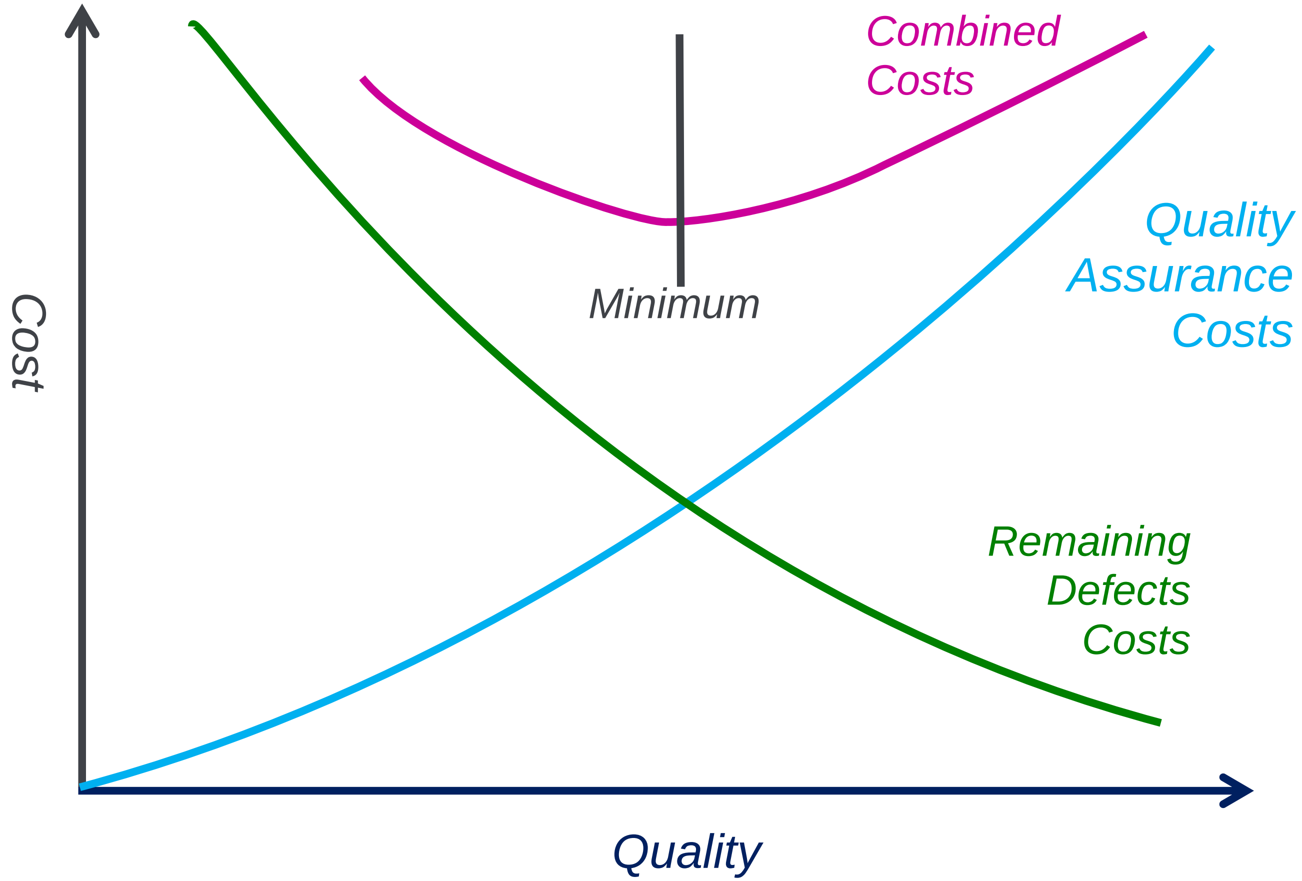
# Investing in Quality and Testing



# Investing in Quality and Testing

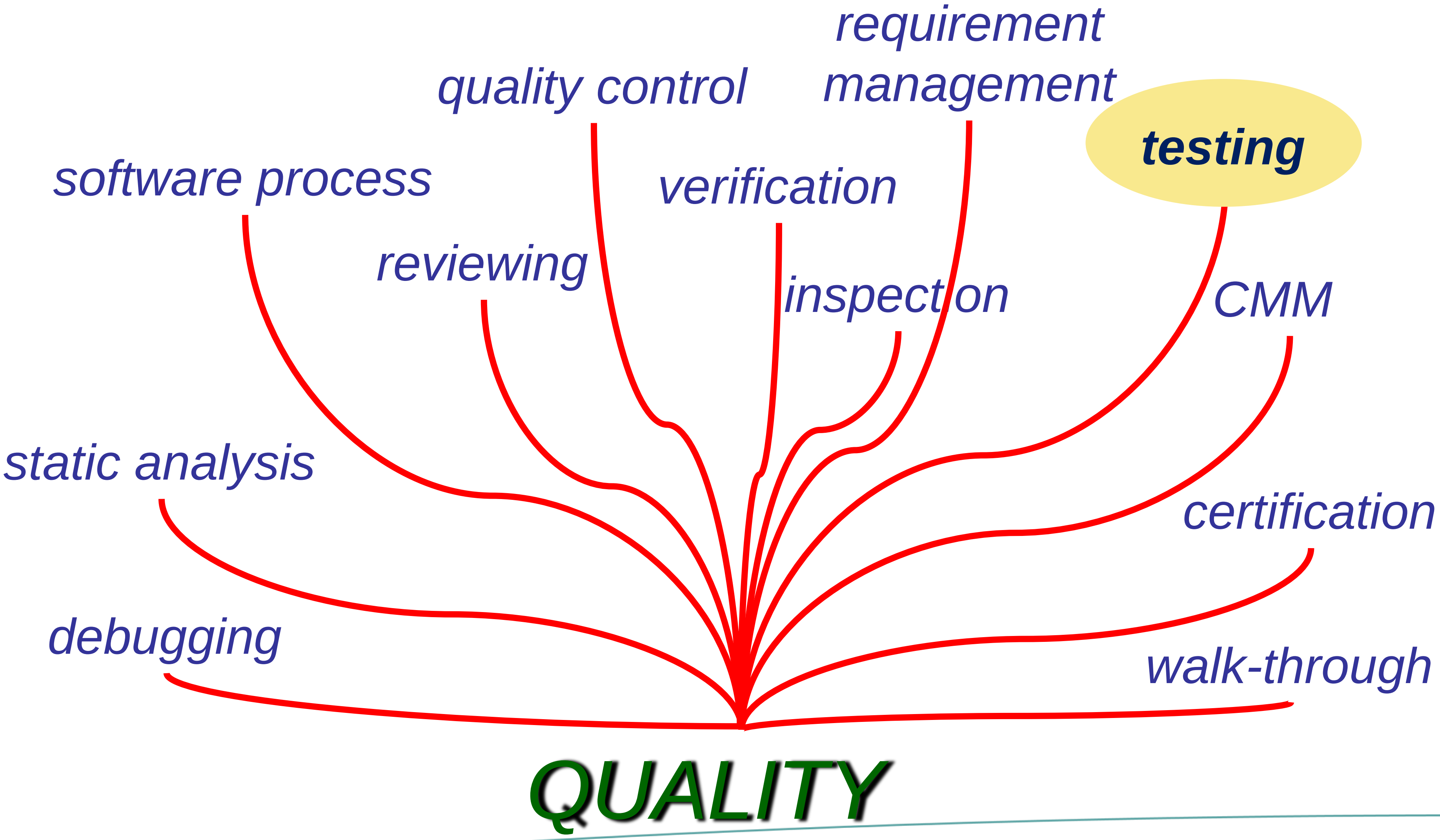


# Investing in Quality and Testing



# Sorts of Testing

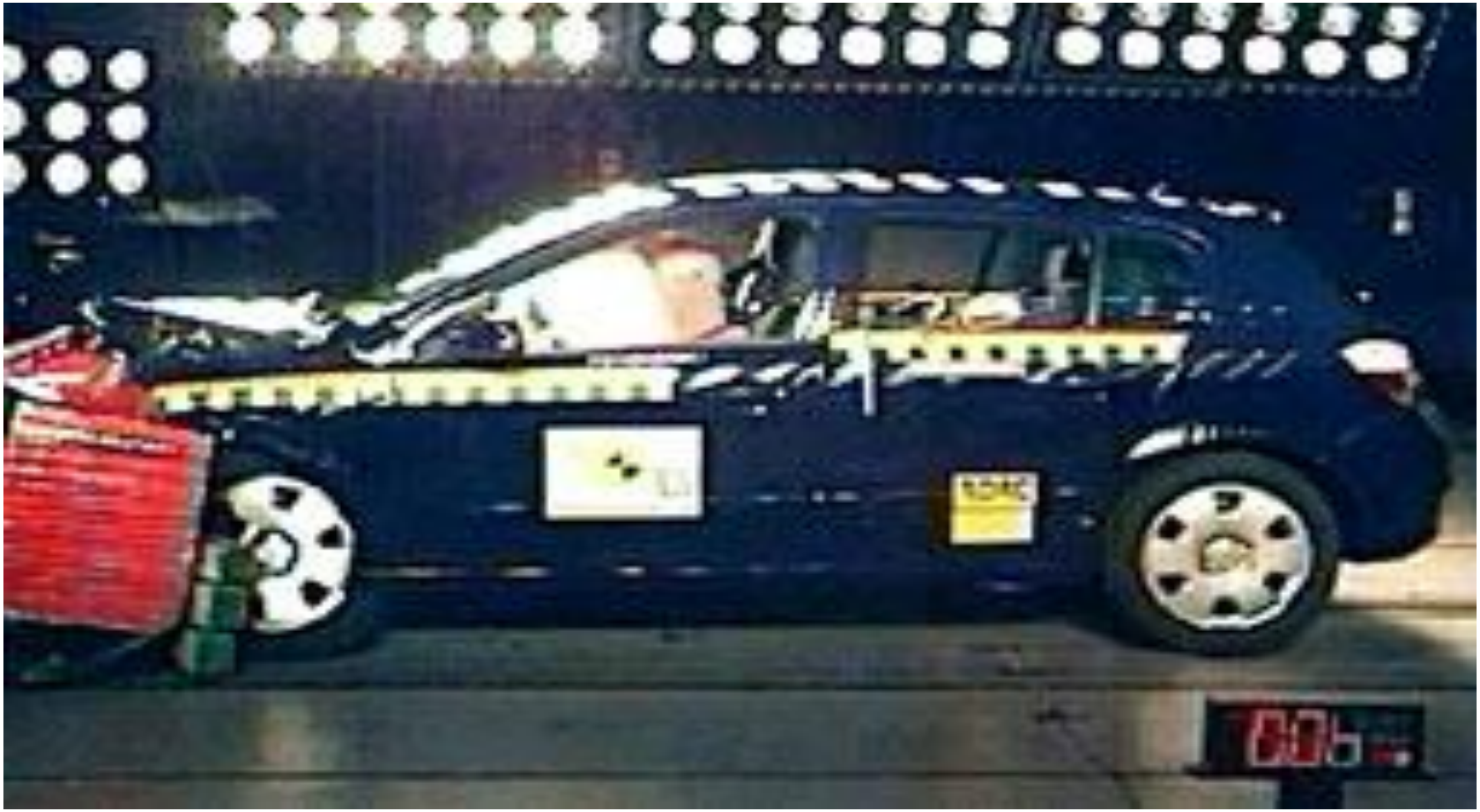
# Quality : There is more than Testing



# Errors, Faults, Failures

| | |
|---|---|
| Failure | incorrect behaviour of the IUT |
| Fault | problem in the program causing the failure |
| Defect | detected failure |
| Issue | politically correct term for failure<br>( before cause is determined ) |
| Error | can only be made by a human being |
| Bug | fault |

# Sorts of Testing



# Sorts of Testing

Many different types and sorts of testing:

- functional testing,  acceptance testing, duration testing,
- performance testing,  interoperability testing,  unit testing,
- black-box testing,  white-box testing,
- regression testing,  reliability testing,  usability testing,
- portability testing,  security testing,  compliance testing,
- recovery testing,  integration testing,  factory test,
- robustness testing,  stress testing,  conformance testing,
- developer testing, acceptance,  production testing,
- module testing, system testing,  alpha test,  beta test
- third-party testing,  specification-based testing,  ………

# Sorts of Testing : Classification

- Quality characteristics:
  - functional, security, compliance, interoperability
  - reliability, robustness, usability, learnability
  - performance, resource, stress, portability, conformance
- Who:
  - developer, tester, user, QA, third party, certifier, support dept, . . .
- Phase
  - programming, integration, acceptance, regression, . . .
- Unit under test
  - unit, module, component, subsystem, system, system-of-systems
  - documentation, system-in-context
- Goal of testing
  - bug finding, confidence, certification, . . .
- Black / white-box
- . . . . .

# Classification into Quality  Characteristics

Characteristics according to ISO 9126 / 25010 :

- Functionality ⇒ *functional testing*
  - suitability accuracy, security, compliance, interoperability
- Reliability ⇒ *reliability testing*
  - maturity, fault tolerance, recoverability
- Usability ⇒ *usability testing*
  - understandability, learnability, operability
- Efficiency ⇒ *performance testing*
  - time behaviour, resource utilization
- Maintainability ⇒ *maintainability testing*
  - Analysability, changeability, stability, testability
- Portability ⇒ *portability testing*
  - Adaptability, intallability, conformance, replaceability

# Classification into Quality  Characteristics

Characteristics according to ISO 9126 / 25010 :

- Functionality ⇒ *functional testing*
  - suitability accuracy, security, compliance, interoperability
- Reliability ⇒ *reliability testing*
  - maturity, fault tolerance, recoverability
- Usability ⇒ *usability testing*
  - understandability, learnability, operability
- Efficiency ⇒ *performance testing*
  - time behaviour, resource utilization
- Maintainability ⇒ *maintainability testing*
  - Analysability, changeability, stability, testability
- Portability ⇒ *portability testing*
  - Adaptability, intallability, conformance, replaceability

# Non-Functional Testing

- In the old days . . . . .
  when computers were operated by experts,
  running in batch mode, over-night, stand-alone,
  testing involved :

  *checking whether the program*
  *does what is should do* ⇒ functional testing

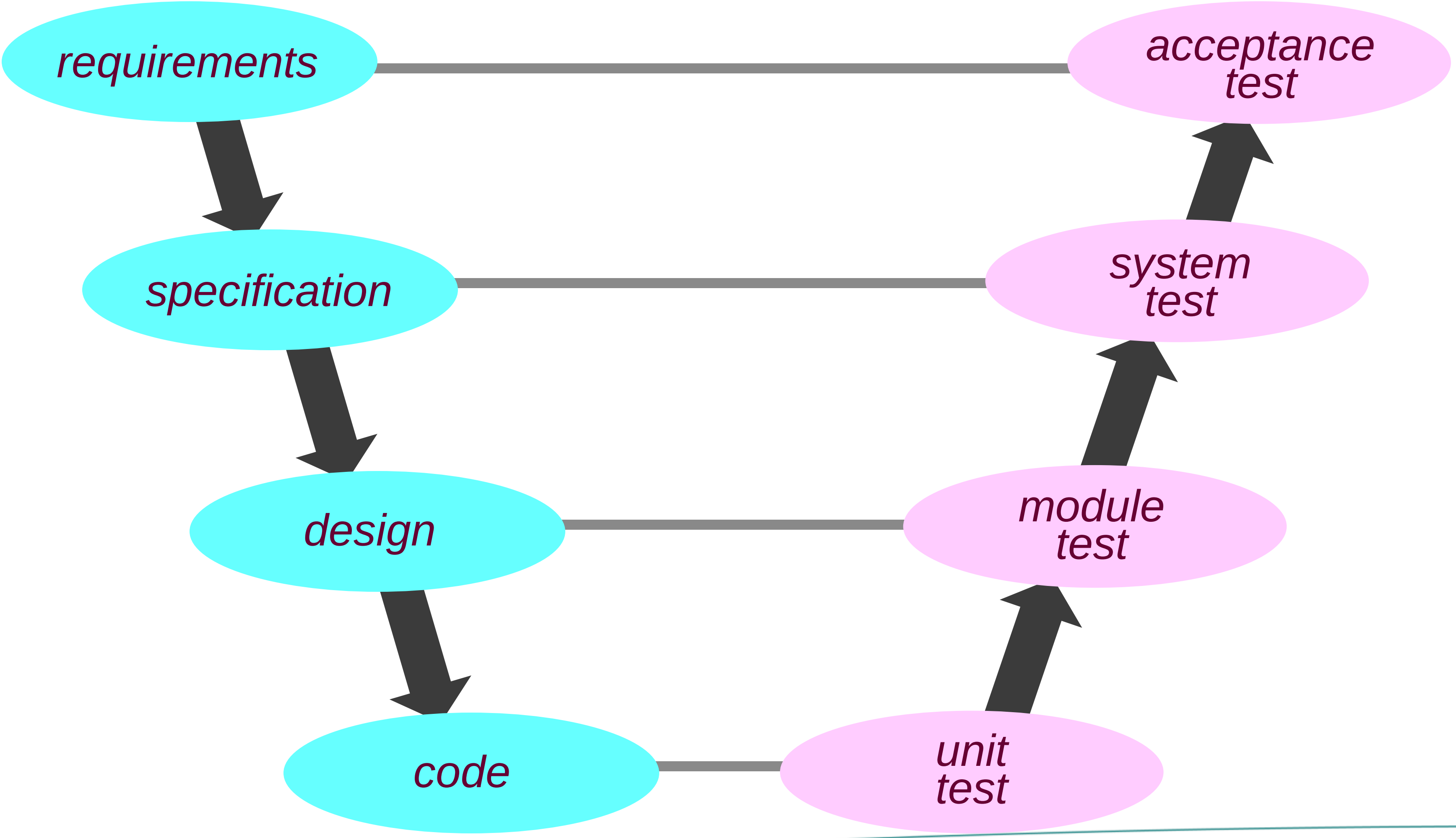
- But in these days . . . .
  now that everybody uses computers, interactively, on-line,
  connected to the world, we want more :

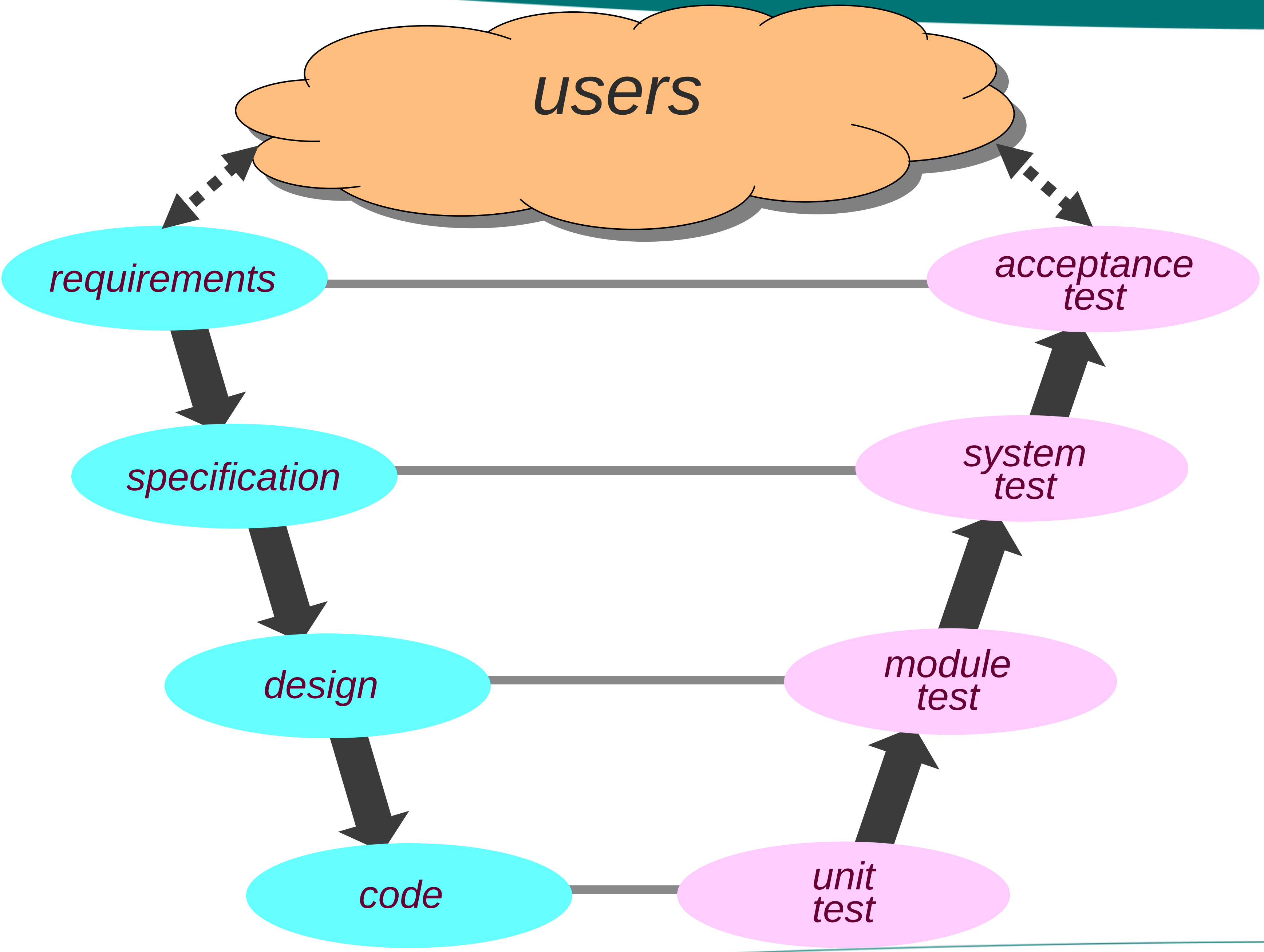
  *user-friendliness, speed,*
  *robustness, security, ……* ⇒ non-functional testing
  extra-functional testing
  qualities testing
  -illities testing

# Sorts of Testing : Who Does it

- Software developer
  - own code
  - someone else code
- Separate test team
- Separate test (SQA) department
- Third party testing
- User
- Operator
- Certification authority
- Alpha testing
- Beta testing
- ……

# Development Phases : *V-Model*





users

requirements

specification

design

code

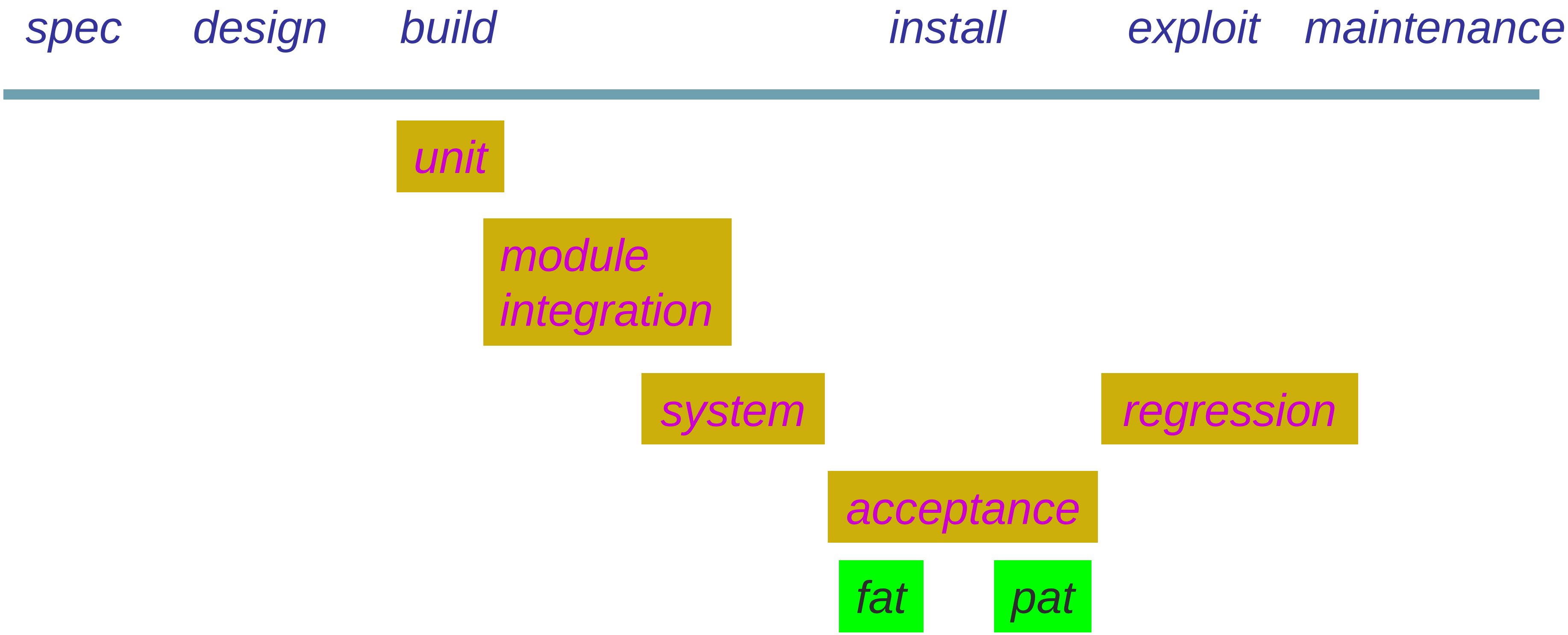
unit test

module test

system test

acceptance test

# Testing in the Development Trajectory



*spec* *design* *build* *install* *exploit* *maintenance*

*unit*

*module integration*

*system*

*regression*

*acceptance*

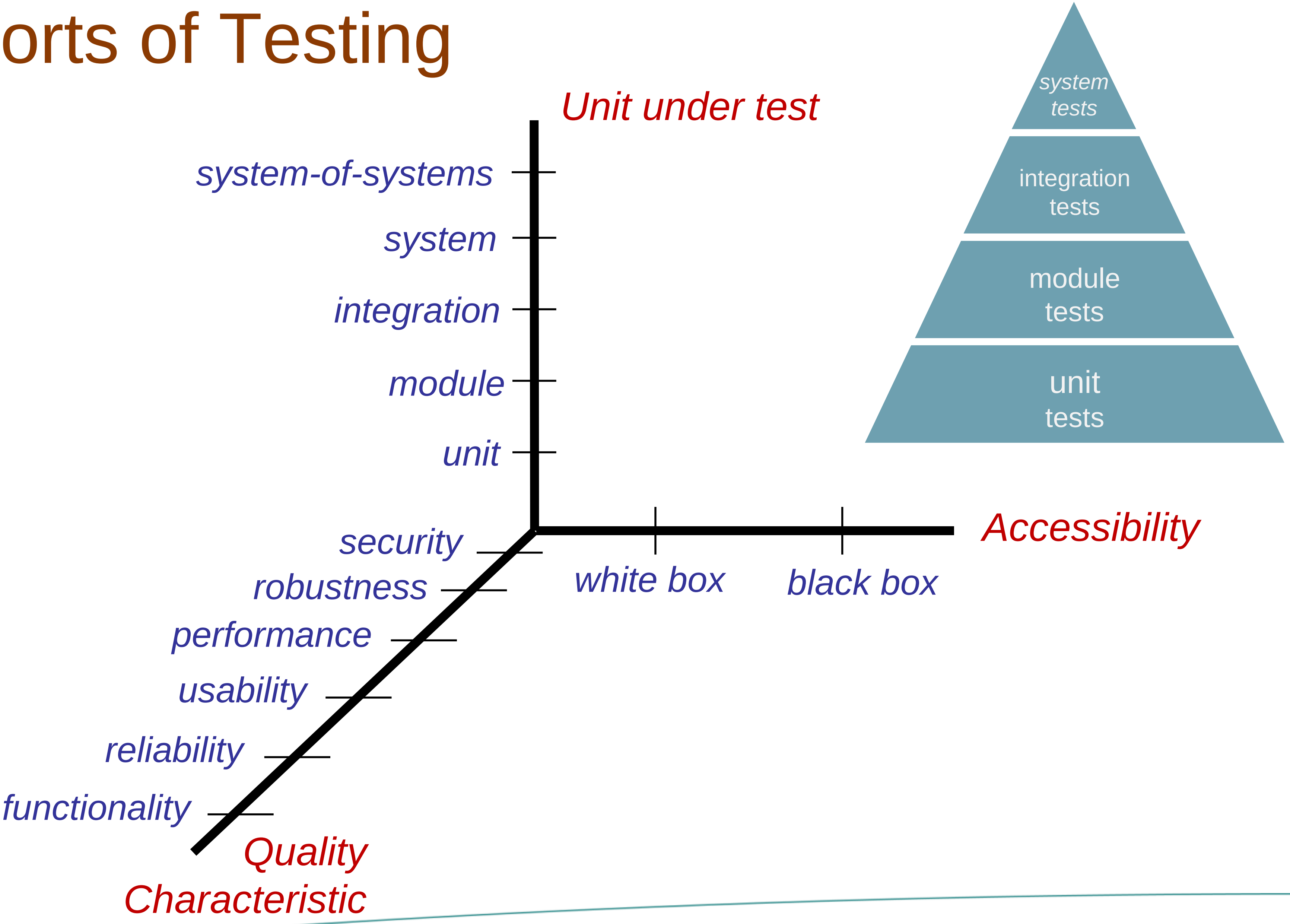
*fat*

*pat*

# Sorts of Testing : Accessibility

- Black box ( specification based -, functional - )
  - no internal details of software considered
  - based on externally observable behaviour\
- White box ( glass box -, structural - )
  - based on structure of the code
  - “oracle” required
- Grey box testing
  - anything between black and white ……
  - e.g. component based testing

# Sorts of Testing



Unit under test

system-of-systems

system

integration

module

unit

Accessibility

white box

black box

Quality Characteristic

security

robustness

performance

usability

reliability

functionality

system tests

integration tests

module tests

unit tests

# Verification & Validation

- Verification :

  building the product right

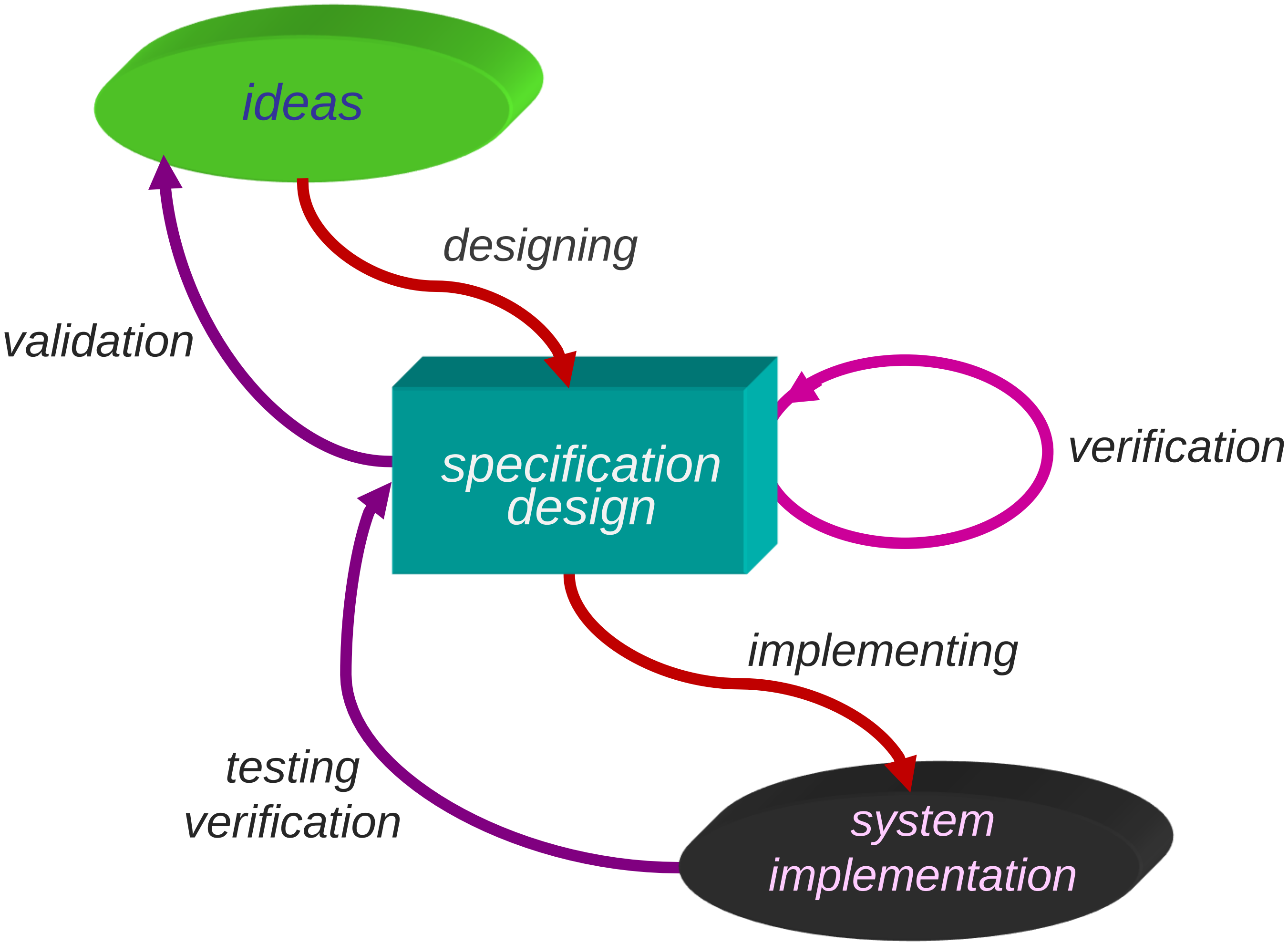
  - concerns development process
  - intermediate products
  - e.g. checking the testability of the specification

- Validation :

  building the right product

  - whether final product meets requirements and needs
  - e.g. by reviewing the specification

# Verification & Validation & Testing


ideas
designing
validation
specification
design
verification
implementing
testing
verification
system
implementation