QUALITY APPROACH IN SOFTWARE DEVELOPMENT

Mr BRIDOUX

Agenda

16-sept Lundi 13h30-17h45
23-sept Lundi 13h30-17h45
30-sept Lundi 13h30-17h45
07-oct Lundi 13h30-17h45
14-oct Lundi 13h30-17h45
21-oct Lundi 13h30-17h45
04-nov Lundi 13h30-17h45

Part I : Introduction to software quality

Chapters 1 : Testing Fundamentals

1.1 What are tests?

1.2 Why testing is necessary ?

1.3 Seven testing Principles ?

1.4 Key terminology and Fundamental Concepts

1.5 Testing process

1.6 Psychology of testing

1.1 What are tests?

Some definitions:

Debugging : The process of finding, analyzing and removing the causes of failures in a component or system.

Functional testing : Testing performed to evaluate if a component or system satisfies functional requirements.

Non-functional testing : Testing performed to evaluate that a component or system complies with non-functional requirements.

test object : The work product to be tested.

Verification : The process of confirming that a work product fulfills its specification.

Validation : Confirmation by examination that a work product matches a stakeholder's needs.

1.1 What are tests?

- **Our world** has become a technological world
- Information Systems on the one hand and technological tools are omnipresent
- Each technological tool contains one or more electronic components operating based on microprocessors
- Each microprocessor is controlled by computer code
- There are potentially “bugs” in these lines of code

Our world has become a world of "bugs"

1.1 What are tests?

Testing and testing activities

- A common misconception about testing activities is that it only involves running the software and checking the results.
- But testing activities include:
  - Planning and control
  - Selection of test conditions
  - Test design
  - Running tests and verifying results
  - Quality assessment and test object
  - Carrying out and finalizing final closing activities

1.1 What are tests?

Testing is not just verification!

- Some consider testing to be just about checking that the system works as specified.
- But this only verifies that the coder followed the specifications, and proves NOTHING about the quality of the software!

Verify = Meet Requirements **Validate** = Meet Needs

Example a Ferrari and a tractor: Same engine but not the same need!

1.1.1 Usual test objectives

Depending on the projects, the testing objectives may vary:

- Evaluate deliverables (work products) such as requirements, user stories, code, etc.
- Prevent defects
- Find failures and defects
- Reduce the level of risks of inadequate software quality
- Increase confidence in the level of quality
- Comply with contractual, legal or regulatory requirements or standards

1.1.2 Testing and debugging

Testing (tester) is different from debugging (Developer)

- Debugging consists of:
- Detect defects
- Correct defects
- Check that the faults are corrected

The test consists of:

- Detect defects
- Create an anomaly sheet
- Check that the defects are corrected once the correction has been made and delivered

1.1 What are tests? QCM

Quel est l’objectif principal des tests ?

A - La détection de différences par rapport à des spécifications (formalisées ou non)

B - Apporter la preuve de l’absence de défauts dans le logiciel

C - La mesure des performances du logiciel en charge

D - Vérifier la qualité des livraisons des développeurs

1.1 What are tests? QCM

Quelle est la meilleure réponse concernant le terme “valider” ?

A - S’assurer que l’on a bien construit le produit final

B - Vérifier que tous les tests ont bien été exécutés

C - Un objectif de test permettant de s’assurer qu’on a construit le bon produit

D - S’assurer que l’on a testé tout le logiciel

1.1 What are tests? QCM

L’activité de déboguer consiste à :

A - Permettre rapidement le fonctionnement d’une fonctionnalité de base

B - Apporter la preuve de fonctionnement d’un programme

C - Trouver, analyser et éliminer les causes des défaillances dans un logiciel

D - Analyser le code source d’un programme sans exécuter le logiciel

1.2 Why testing is necessary ?

Définition :

Anomaly : A condition that deviates from expectation.

Root cause : A source of a defect such that if it is removed, the occurrence of the defect type is decreased or removed.

Test case: A set of preconditions, inputs, actions (where applicable), expected results and postconditions, developed based on test conditions.
Failure : An event in which a component or system does not meet its requirements within specified limits.

Error : A human action that produces an incorrect result.

Defect : An imperfection or deficiency in a work product where it does not meet its requirements or specifications or impairs its intended use.

1.2 Why testing is necessary ?

Définition :

Quality : The degree to which a work product satisfies stated and implied requirements.

Risk: A factor that could result in future negative consequences.

Test : A set of one or more test cases.

regression testing : A type of change-related testing to detect whether defects have been introduced or uncovered in unchanged areas of the software.

1.2.1 Contribution of tests to success

Example of problems related to faulty software

THERAC: X-ray and radiotherapy machine from 1985 to 1987

Impact: Officially, 4 over-irradiated deaths / not measurable

June 4, 1996: The Ariane 5 launcher exploded

Impact: Complete loss of rocket and satellite / €150 million

Friday July 19, 2024: Bug in CrowdStrike software

Impact: Flight delays and cancellations / Hospitals affected / TV channels

1.2.1 Contribution of tests to success

The tests allow:

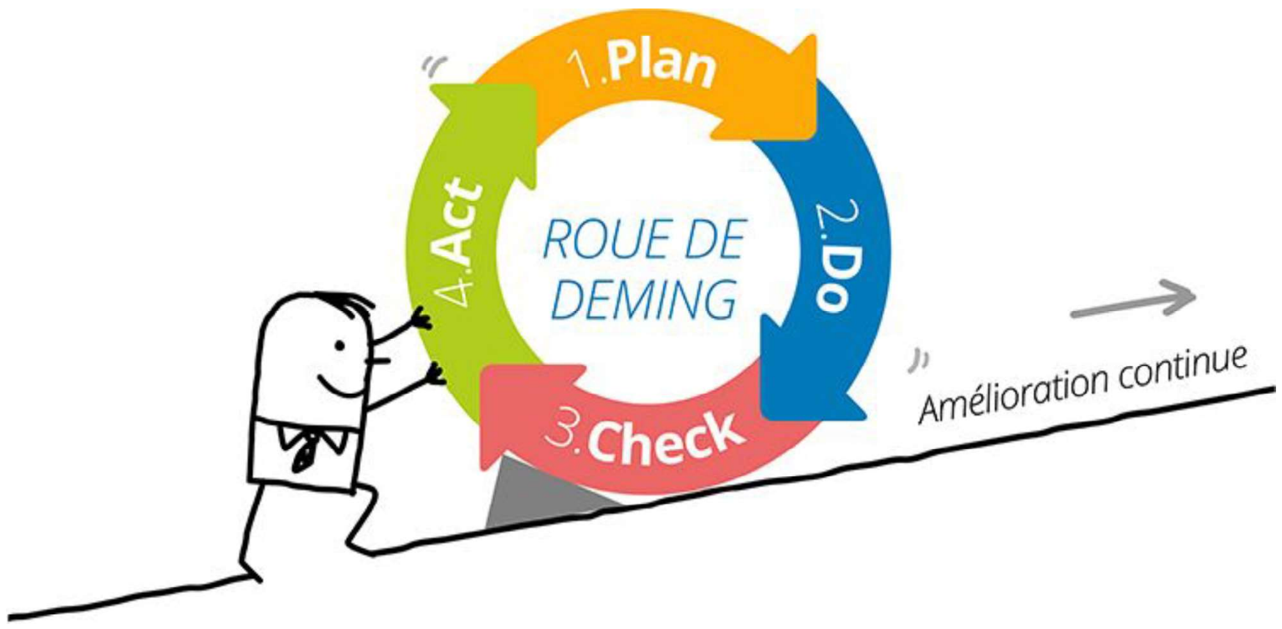
- to reduce the number of “bugs”
- To achieve a measurable level of quality in the products around us
- Reduce the risk of a product malfunctioning
- A malfunction can have benign…or fatal consequences.

1.2.2 Quality assurance and testing

The concept of quality

- A properly designed and error-free test reduces the overall risk level of product malfunction
- Testing increases the level of confidence in the quality of software if it finds few or no defects
- When testing finds defects, the quality of the software system increases when these defects are corrected.
- Feedback helps improve the level of quality.
- Testing **is a quality control activity**

1.2.2 Quality assurance and testing



1.2.2 Quality assurance and testing

Number of tests: quantity vs. quality

- Over quality is detrimental
- It is necessary to prioritize activities
- It is necessary to optimize the testing effort

Contrary to popular belief, very often too many tests are prepared and executed

1.2.3 Errors, Default and failure

Errors -> Human

Example: Poorly managed project, Team not competent enough, Stress, Functional and business complexities

Default -> Static (I execute)

Example: False or missing management rules, Missing instruction in a portion of code

Failure -> Dynamic (I execute and see)

Example: Failure is always caused by a defect

1.2 Why testing is necessary ? - QCM

Donner des exemples montrant la nécessité des tests, parmi les affirmations suivantes:

1. Vérifier la documentation associée au logiciel
2. Aider à réduire le risque de défaillances en production
3. Contribuer à la qualité du système
4. Vérifier la qualité des développeurs
5. Vérifier les exigences légales

Lesquelles sont vraies concernant la nécessité des tests ?

1.2 Why testing is necessary ? - QCM

Expliquer la relation entre les tests et l'assurance qualité et donner des exemples montrant comment les tests contribuent à une amélioration de la qualité

Pourquoi les tests permettent d'augmenter la confiance en la qualité des logiciels ?

A. L'absence de défaut trouvé par les tests garantit la qualité du logiciel
B. La confiance en la qualité du logiciel est augmentée par la complétude de la couverture des tests
C. Les tests permettent de trouver et corriger des anomalies, ce qui améliore le sentiment de confiance que l'on peut avoir dans la qualité du logiciel
D. S'ils sont bien conçus et exécutés, les tests permettent de trouver tous les défauts du logiciels

1.2 Why testing is necessary ? - QCM

Faire la différence entre erreur, défaut et défaillance

Quelle est la définition d’un défaut ?

A. A condition that deviates from expectation.
B. An imperfection or deficiency in a work product where it does not meet its requirements or specifications or impairs its intended use.
C. A human action that produces an incorrect result.

1.3 Seven testing Principles

General principles

- Tests show the presence of defects
- Exhaustive testing is impossible.
- Test as soon as possible
- Grouping of defects.
- The pesticide paradox.
- Tests depend on context
- The illusion of absence of error

1.3 Seven testing Principles

Principle 1 - Tests show the presence of defects

Tests can prove the presence of defects, but cannot prove their absence

1.3 Seven testing Principles

Principle 2 - Exhaustive testing is impossible.

- Testing everything is not feasible except for trivial cases
- Rather than exhaustive, we use risk and priority analysis to focus efforts.
- 20% of the cost of testing will detect 80% of defects

1.3 Seven testing Principles

Principle 3 – Test early

Testing activities should start as early as possible in the software development cycle and should be focused towards defined objectives

Testing early helps identify defects early and therefore correct them with less effort, which helps deliver software with the expected quality on time.

1.3 Seven testing Principles

Principle 4 - Grouping of defects

- The test effort must be set proportionally to the density of defects expected and observed in the different modules.

80% of defects are present in 20% of the code

The test method must make it possible to detect these defects as a priority

1.3 Seven testing Principles

Principle 5 – Pesticide paradox

- If the same tests are repeated many times, it will happen that the same test no longer finds a new fault.

The “pesticide paradox” when applying the same pesticide too much, it becomes ineffective. Too much insecticide creates a mutation in the parasites which become resistant, which completely cancels the effect of the insecticide.

-> Regularly update test cases

1.3 Seven testing Principles

Principle 6 – Testing is context dependent

- Tests are performed differently in different contexts. For example, critical security software is tested differently than that of an ecommerce site.
- The test (test effort) must be adapted to the future context of use of the product
  - The consequences of not detecting a defect are not the same...

1.3 Seven testing Principles

Principle 7 - The illusion of absence of defect

- Finding and correcting defects does not help if the designed system is unusable and does not meet the needs and expectations of users
- During a first test campaign on a new application, “zero defects found” means, as desired and not exclusively, that:
    - Tests are irrelevant
    - The test team has not started running tests
    - The testing team lacks rigor

- **In all cases, “zero defects found” is:**
    - **Nor a guarantee of quality**
    - **Nor a guarantee that the product will “please” the user**

1.3 Seven testing Principles - QCM

Après une analyse de risques, il est décidé d’ajouter des tests sur les parties du système ou les premiers tests ont trouvés plus de défauts que la moyenne.

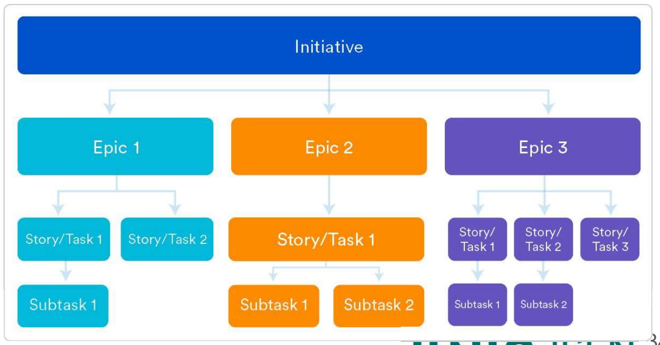
Quel est le principe qui a été appliqué ?

A. Les tests montrent la présence des défauts
B. Regroupement des défauts
C. Paradoxe du pesticide
D. Les tests dépendent du contexte

1.4 Key terminology and Fundamental Concepts

Requirement / User Story (US)

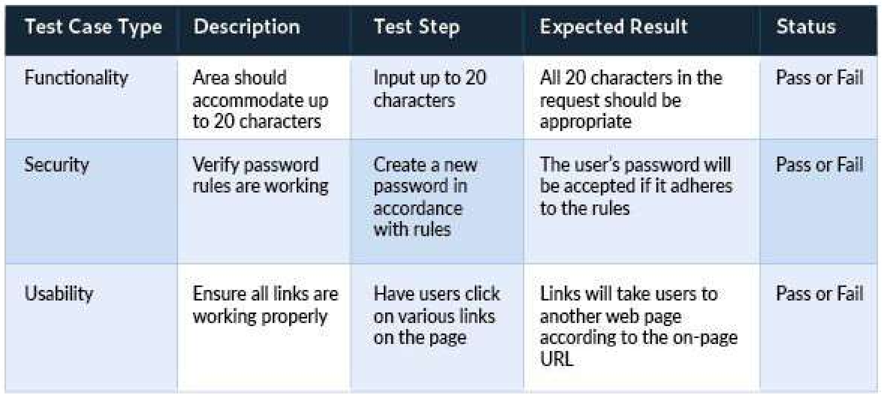
- A user story is a non-formal, general explanation of a software feature written from the end user's perspective. The goal of a user story is to define how a job will add certain value to the customer.
  - User stories are often expressed in a simple sentence, structured as follows:
    - “As a [persona], I [wish that] [in order to].”



1.4 Key terminology and Fundamental Concepts

Test case

- Test case, is a set of prerequisites, input data, actions and expected results, developed with the aim of verifying the proper functioning of a development or product.
- **Test cases** make it possible to concretize a **use case**, on concrete, real data.

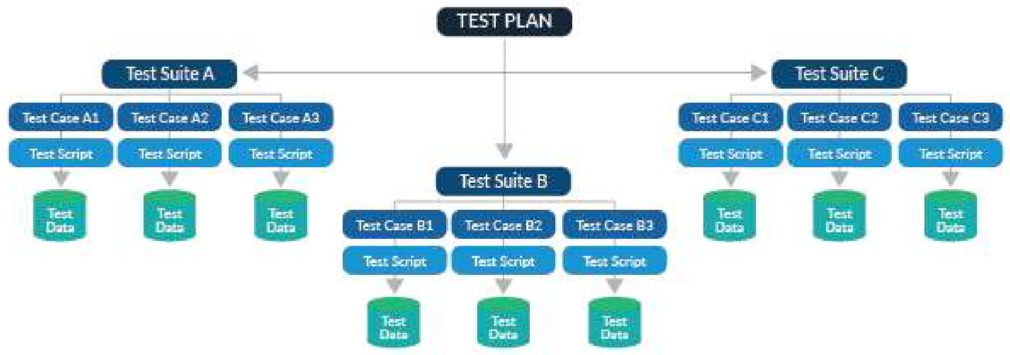


Test Case Type	Description	Test Step	Expected Result	Status
Functionality	Area should accommodate up to 20 characters	Input up to 20 characters	All 20 characters in the request should be appropriate	Pass or Fail
Security	Verify password rules are working	Create a new password in accordance with rules	The user's password will be accepted if it adheres to the rules	Pass or Fail
Usability	Ensure all links are working properly	Have users click on various links on the page	Links will take users to another web page according to the on-page URL	Pass or Fail

1.4 Key terminology and Fundamental Concepts

Test suite

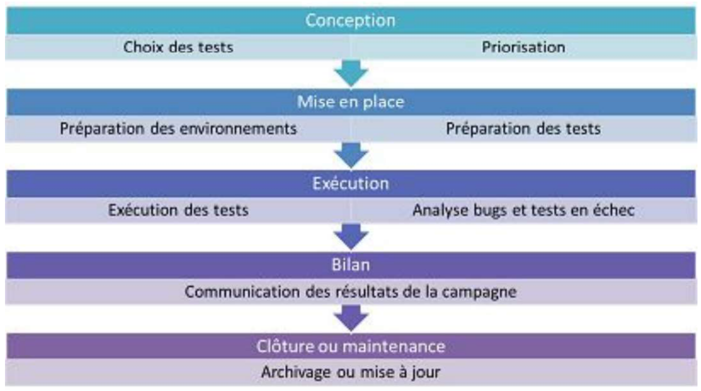
- A complete test suite refers to a set of test cases that cover various aspects of a software application.



1.4 Key terminology and Fundamental Concepts

Test campaign

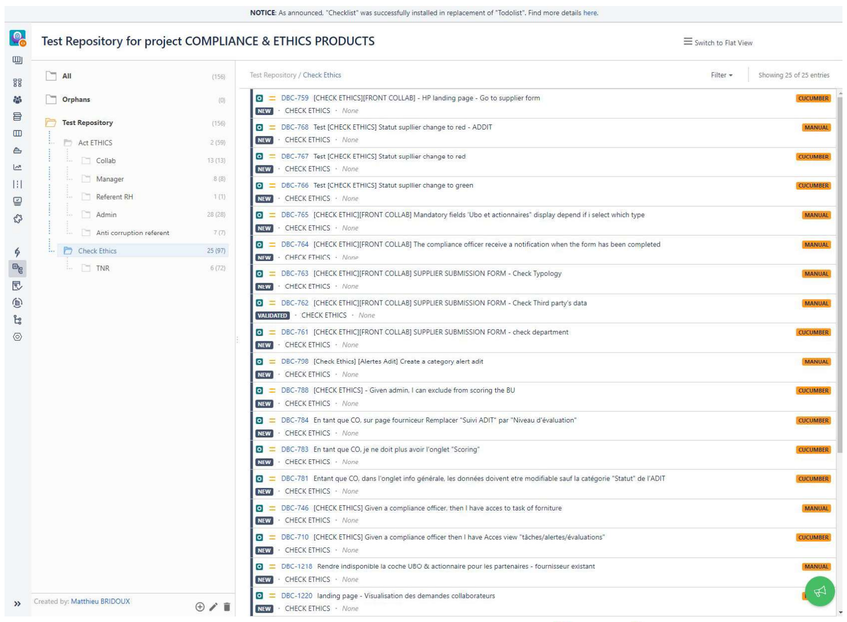
- A test campaign is much more than executing test cases. A test campaign is a process that must be thought out, followed and closed. As is often the case in the testing profession, you have to know how to anticipate, choose and define priorities in order to obtain the best possible visibility on the quality of the application.



1.4 Key terminology and Fundamental Concepts

Test repository

- **Definition :** A central database of all artifacts related to the testing process, including test cases, test documents, bug reports.



NOTICE: As announced, "Checklist" was successfully installed in replacement of "Todolist". Find more details here.

Test Repository for project COMPLIANCE & ETHICS PRODUCTS

Switch to Flat View

All	(156)
Orphans	(0)
Test Repository	(156)
Act ETHICS	2 (59)
Collab	13 (13)
Manager	8 (8)
Referent RH	1 (1)
Admin	28 (28)
Anti corruption referent	7 (7)
Check Ethics	25 (97)
TNR	6 (72)

Created by: Matthieu BRIDOUX

Test Repository / Check Ethics — Filter — Showing 25 of 25 entries

DBC-759 [CHECK ETHICS][FRONT COLLAB] - HP landing page - Go to supplier form — CUCUMBER
NEW · CHECK ETHICS · None

DBC-768 Test [CHECK ETHICS] Statut supllier change to red - ADDIT — MANUAL
NEW · CHECK ETHICS · None

DBC-767 Test [CHECK ETHICS] Statut supllier change to red — CUCUMBER
NEW · CHECK ETHICS · None

DBC-766 Test [CHECK ETHICS] Statut supllier change to green — CUCUMBER
NEW · CHECK ETHICS · None

DBC-765 [CHECK ETHIC][FRONT COLLAB] Mandatory fields 'Ubo et actionnaires" display depend if i select which type — MANUAL
NEW · CHECK ETHICS · None

DBC-764 [CHECK ETHIC][FRONT COLLAB] The compliance officer receive a notification when the form has been completed — MANUAL
NEW · CHECK ETHICS · None

DBC-763 [CHECK ETHIC][FRONT COLLAB] SUPPLIER SUBMISSION FORM - Check Typology — MANUAL
NEW · CHECK ETHICS · None

DBC-762 [CHECK ETHIC][FRONT COLLAB] SUPPLIER SUBMISSION FORM - Check Third party's data — MANUAL
VALIDATED · CHECK ETHICS · None

DBC-761 [CHECK ETHIC][FRONT COLLAB] SUPPLIER SUBMISSION FORM - check department — CUCUMBER
NEW · CHECK ETHICS · None

DBC-798 [Check Ethics] [Alertes Adit] Create a category alert adit — MANUAL
NEW · CHECK ETHICS · None

DBC-788 [CHECK ETHICS] - Given admin, I can exclude from scoring the BU — CUCUMBER
NEW · CHECK ETHICS · None

DBC-784 En tant que CO, sur page fourniceur Remplacer "Suivi ADIT" par "Niveau d'évaluation" — CUCUMBER
NEW · CHECK ETHICS · None

DBC-783 En tant que CO, je ne doit plus avoir l'onglet "Scoring" — CUCUMBER
NEW · CHECK ETHICS · None

DBC-781 Entant que CO, dans l'onglet info générale, les données doivent etre modifiable sauf la catégorie "Statut" de l'ADIT — CUCUMBER
NEW · CHECK ETHICS · None

DBC-746 [CHECK ETHICS] Given a compliance officer, then I have acces to task of forniture — MANUAL
NEW · CHECK ETHICS · None

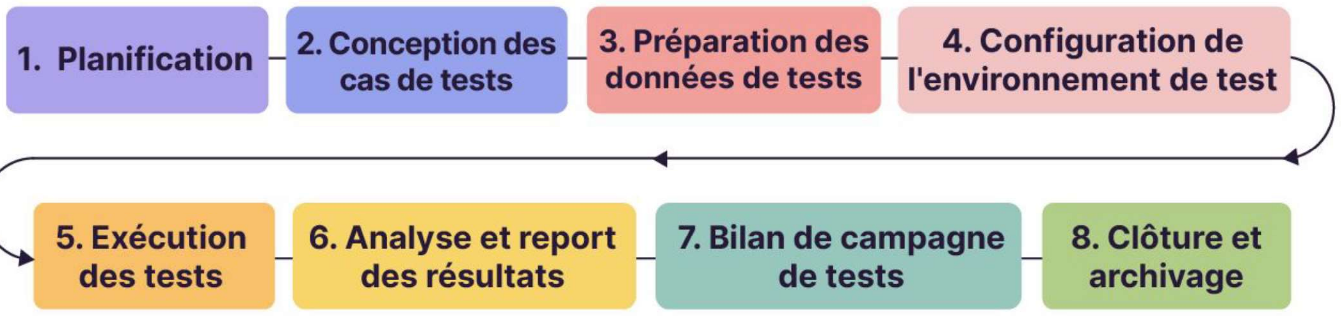
DBC-710 [CHECK ETHICS] Given a compliance officer then I have Acces view "tâches/alertes/évaluations" — CUCUMBER
NEW · CHECK ETHICS · None

DBC-1218 Rendre indisponible la coche UBO & actionnaire pour les partenaires - fournisseur existant — MANUAL
NEW · CHECK ETHICS · None

DBC-1220 landing page - Visualisation des demandes collaborateurs
NEW · CHECK ETHICS · None

1.5 Testing process

Testing activities


1. Planification
2. Conception des cas de tests
3. Préparation des données de tests
4. Configuration de l'environnement de test
5. Exécution des tests
6. Analyse et report des résultats
7. Bilan de campagne de tests
8. Clôture et archivage

1.5.2 Activities and test task

Running the tests:

- Run tests manually or using test execution tools
- Compare the results obtained and the expected results
- Analyze incidents to establish their causes
- Report anomalies based on observed failures
- Save the test execution result
- Repeat activities for confirmation and regression testing

1.4.2 Activities and test task

Test closure:

- Objective ? Consolidate the experience, provide information to stakeholders
- When ? At the end of the test cycle

Activities :

- Verification of the closure of all defects, or the entry of modification requests for unresolved defects
- Creation of a test report to communicate to stakeholders
- Finalize and archive tests, data for future reuse
- Use the information collected to improve the maturity of the testing process

1.6 The psychology of testing

Recommendation

- The detection of a defect/failure is neither a judgment nor a criticism of the work carried out, and even less of the quality of a person.
- The anomaly sheet must be factual, objective and clear
- The objective of an anomaly sheet must allow the correction of a defect, therefore improving the final product
- It’s about creating a collaborative dynamic between teams to improve the final product.

1.5 La psychologie des tests - QCM

Les relations entre les testeurs et les développeurs sont basées sur…

A. Des faits, donc l'aspect relationnel n'est pas important
B. La capacité des testeurs à trouver des défauts, donc l'aspect relationnel est négligeable
C. Un travail d'équipe ou les développeurs doivent corriger les anomalies trouvées donc l'aspect relationnel est important
D. Un travail d'équipe en collaboration, donc l'aspect relationnel est important pour éviter que les testeurs ne soient que des porteurs de mauvaises nouvelles

1.5 La psychologie des tests - QCM

La raison PRINCIPALE pouvant empêcher les développeurs de tester leur propre code est…

A. Le manque de documentation technique
B. Le manque d’objectivité
C. Le manque de formations
D. L’absence d’outils de test spécifiques pour les développeurs

2. Test during the software development lifecycle

2. Test during the software development lifecycle

2.1 Software development models

2.2 Place the tester in a team

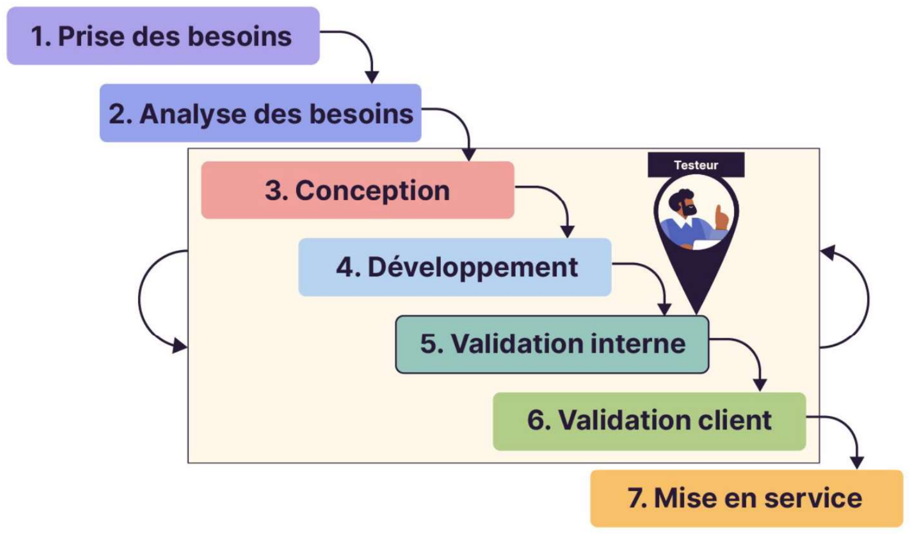
2.3 Identify the relationship between the tester and team members

2.4 Test levels

2.5 Types of testing

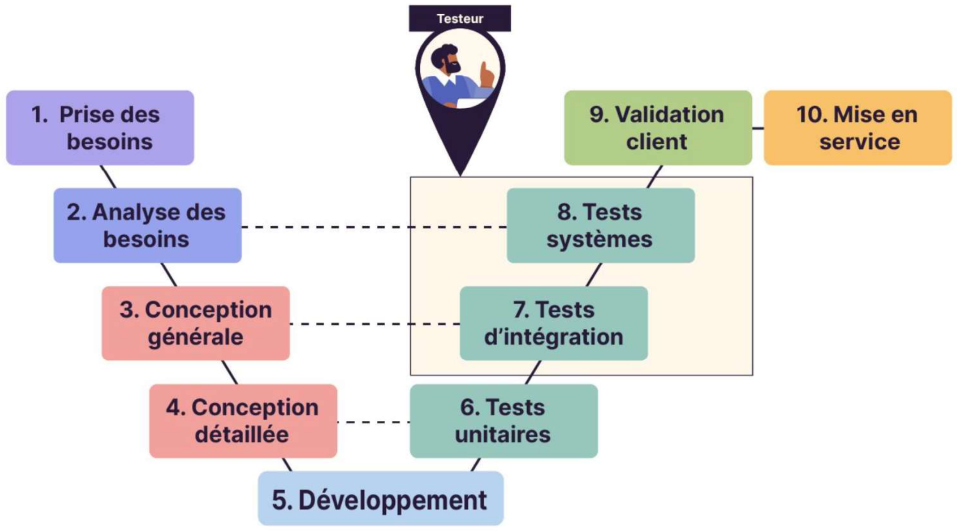
2.1 Software development models

Le modèle en cascade



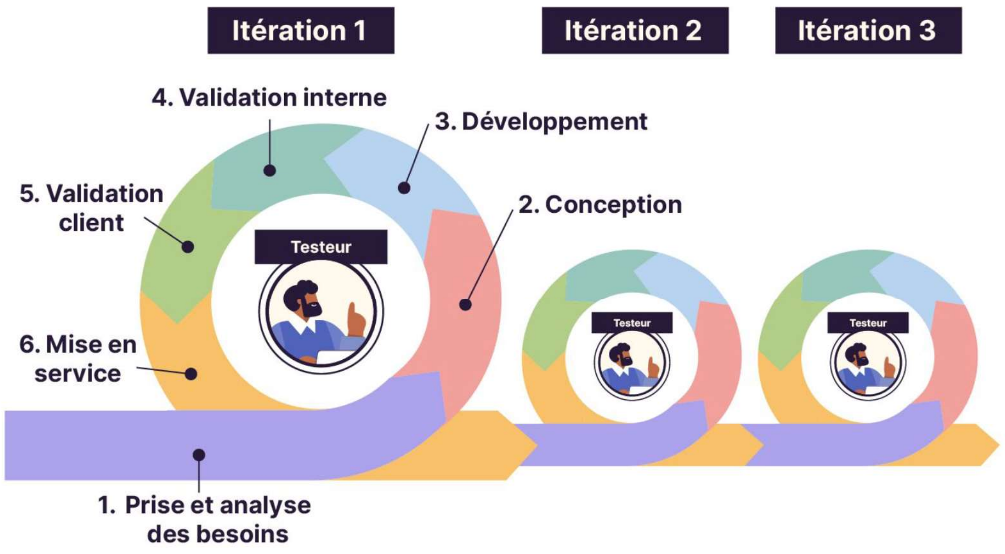
2.1 Software development models

Le modèle en cycle en V :



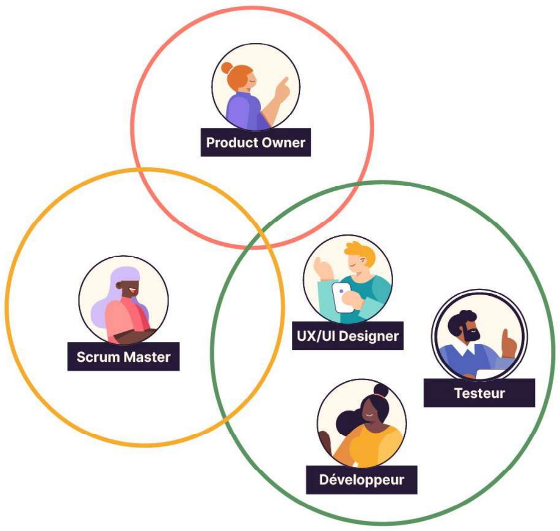
2.1 Software development models

La méthode agile : Scrum



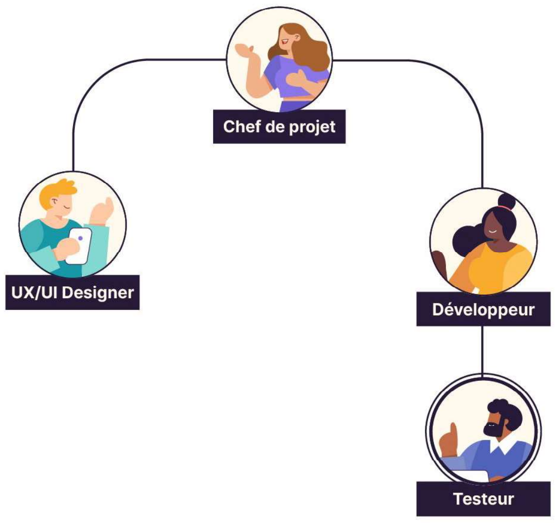
2.2 Place the tester in a team

Dans une équipe structurée en **méthode itérative** (**agile**), on retrouve :



2.2 Place the tester in a team

méthode séquentielle (dite cycle en V ou en cascade)



2.3.1 Identify the relationship between the tester and team members

- Les testeurs et les développeurs travaillent souvent ensemble. Ils sont amenés à échanger régulièrement
- Dans les premiers projets de développement, le testeur n’existait pas.
- C’était le développeur qui vérifiait son code
- Mais pas de test de bout en bout.
- Intéressant d’avoir quelqu’un d’extérieur au développement pour le vérifier.
- C’est donc le testeur, Quelqu’un d’impartial

Le développeur et le testeur n’ont pas la même posture vis-à-vis du logiciel :

- Le développeur conçoit un logiciel à partir de spécifications (cahier des charges, expressions de besoins, etc.).
- Le testeur va chercher par tous les moyens à vérifier que le logiciel ne comporte pas de bugs.

2.3.2 Identify the relationship with the PO or project manager

- Le Product Owner représente la **voix du client**. C’est lui qui va rédiger les **User Stories**.
- Dans une équipe non agile, ce ne sera pas le chef de projet mais le business analyste qui rédigera les spécifications.
- Le testeur intervient à diverses **étapes du cycle produit** et travaille en étroite collaboration avec le PO. Par exemple, le testeur participe activement à l’élaboration des **critères d’acceptation**.

2.3.3 Les rôles ne sont pas figés

- Un développeur pourra remonter un problème utilisateur ou une opportunité de résoudre un problème après une discussion avec des utilisateurs du produit.
- Le PO pourra remonter un bug car il aura effectué quelques tests pour s'assurer que l'équipe de réalisation va dans le bon sens.

2.3.4 The tester's missions vary

- Only be an executor of the already written test plan and note whether the test is valid (“OK”) or not (“KO”) and, in the latter case, report the anomaly
- do automation, that is to say create scripts that will test the product without human intervention
- help design new features, support developers for better code quality, etc.
- carry out performance, load and stress tests
- manage test management (=Test Manager)
- track anomalies
- instill the spirit of quality within your team. (= Quality Coach)

2.4 Test Levels

2.4.1 Component testing

2.4.2 Integration test

2.4.3 System test

2.4.4 Acceptance test

2.4 Niveaux de test

Definition:

Acceptance testing: A level of testing that aims to determine whether to accept the system.

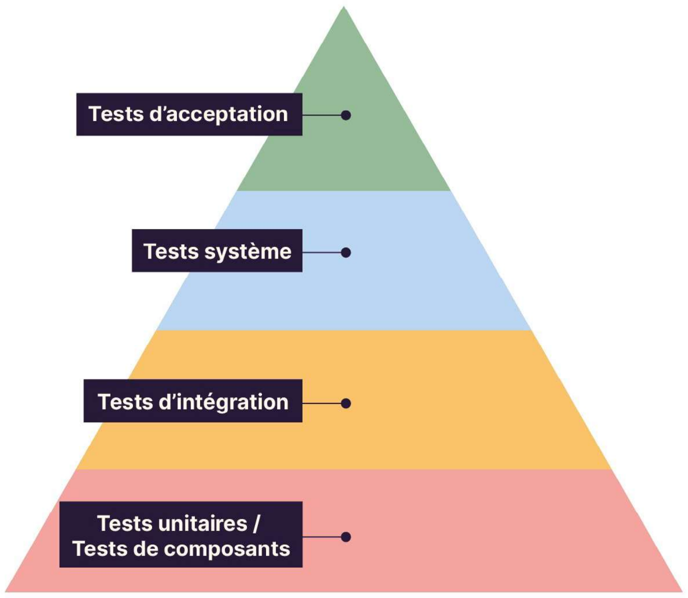
Integration testing: A level of testing that focuses on the interactions between components or systems.

Component Testing: A level of testing that focuses on individual hardware or software components.

Systems testing: A level of testing that focuses on verifying that a system as a whole meets specified requirements.

2.4 Niveaux de test

La pyramide des tests :



2.4 Example with an analogy

Let's take a customer who wants a 2m × 5m wall.

The workers (developers) will build the bricks. You will need to check that they are the right size. This corresponds to **component (or unit) testing.**

Then they will put the bricks together two by two. You will need to make sure that they fit together properly. This corresponds to **integration tests**.

Then they will build the wall. You will need to check that it is 2 m × 5 m (customer specifications). This corresponds to **system tests.**

And finally, the **customer will come and check the wall.** There, the customer is not satisfied with the result. He didn't want a brick wall but a stone wall!

This analogy allows you to clearly identify the different levels of testing and understand the objective of each. It also shows you that before starting development, you must clearly understand the final need and that the specifications must be written accordingly.

2.5 Types of tests

The tests are classified into several categories:

- functional tests
- non-functional tests
- Tests related to changes

2.5.1 Functional tests

Functional testing concerns testing the functionality of a product. This validates that the product works correctly and meets the need.

- Evaluate the functionalities to be implemented
- Performed at all test levels
- Measurement of functional coverage
- Generally require “professional” knowledge

2.5.2 Non-functional tests

- **robustness**: to ensure that the site meets robustness criteria, such as so many requests in so much time. For example, simulating a peak of users over a very short time;
- **performance:** response time for loading a page of a website for example;
- increase in load: gradually increase the number of requests to a website to check when it can no longer support the load;
- **compatibility:** does the software work on Windows and Mac OS? ;
- **ergonomics and usability**: is the site easily usable on a phone, tablet, computer or television? ;
- **security**: to ensure that hackers cannot recover or use user data.

2.5.3 Tests related to changes

Confirmation test (or retest)

- During the first execution of a test, a failure is detected and is the subject of an anomaly report sent for correction.
- Once the correction has been made and delivered back to the test environment, it is necessary to ensure the correctness of the correction.
- It is therefore necessary to rerun the test that detected the anomaly.

The retest is the minimum test that must be performed

Only the retest can guarantee that the fault has been corrected.

The purpose of a confirmation test is to confirm that the original fault has been successfully repaired.

2.5.3 Tests related to changes

Regression testing

- **A change made to the software can have unintended side effects which are called regressions**
- **It is therefore necessary, following a delivery (corrective or evolution) to check that the unmodified parts of the system which were functioning before the delivery continue to function.**
- **Regression test suites are run multiple times and typically evolve slowly, which is why regression tests are good candidates for automation**

Part II : Carry out functional and exploratory tests on a website

1. Test Writing Practice Workshop

Objective : To concretely apply the Knowledge acquired by writing test cases.

1. Test Writing Practice Workshop

- The test plan can contain passing cases and non-passing cases

Example of a passing case: Connect to the customer account of the ecommerce site with an email test@mail.com. Result: I am well connected

Example of a non-passing case: Connect to the customer account of the ecommerce site with a testmail.com email. Result: I get an “Invalid Email” error message

- **Limit cases** are tests that will push the software to its limits. This might involve entering special characters in input fields or checking what happens when you enter 1,000 characters.

1. Test Writing Practice Workshop

The key elements of a test case:

- Test Case Description: This is a brief summary of what the test case is supposed to verify.
- Preconditions: These are the conditions that must be met before executing the recipe case.
- Test Steps: These are the step-by-step instructions to execute the test case.
- Test Data: These are the specific inputs that will be used to execute the test case.
- Expected Results: This is what the system should do in response to each test step.
- Actual results: These are the actual behaviors of the system when executing the recipe case.
- Status: It indicates whether the test case passed or failed.

1. Test Writing Practice Workshop

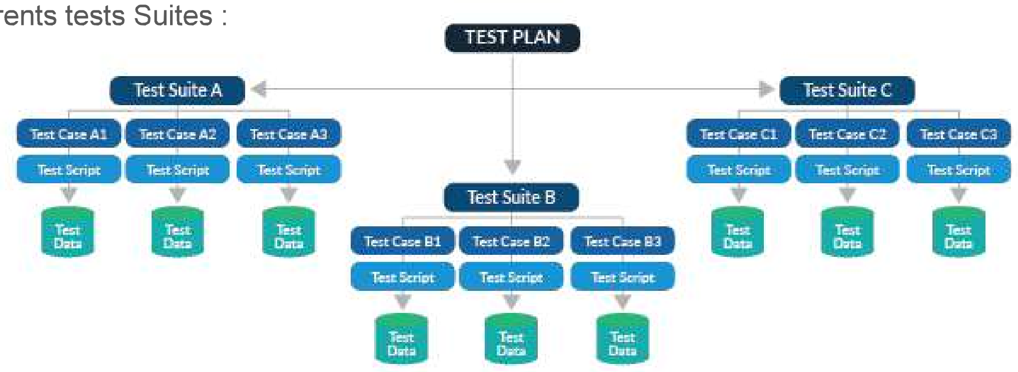
- Test case description: Verify login functionality.
- Preconditions: The user has a valid account.
- Testing Steps: 1. Go to the login page. 2. Enter a valid username and password. 3. Click on the “Connection” button.
- Test data: Username: “testuser”, Password: “testpassword”.
- Expected results: The user is redirected to the home page after successful login.
- Actual Results: To be completed when running the test.
- Status: To be filled when running the test.

1. Test Writing Practice Workshop

Test plan with sur **le login et création de compte du site** leroymerlin.fr dans un fichier excel avec les cas passants et les non passants.

-> Faire un tableau excel avec les différents tests Suites :

- Priority (Hight, medium, low)
- Test case description
- Test Steps
- Test Data (option)
- Expected Results
- Status:



JUNIA ISEN