Tests et Test Unitaire

Introduction

Liens avec un référentiel métier

- Tests d'intégration et d'acceptation d'un service
- Gestion d'environnements de développement et de test
- Réalisation des tests nécessaires à la validation d'éléments adaptés ou développés
- Réalisation des tests nécessaires à la mise en production d'éléments mis à jour

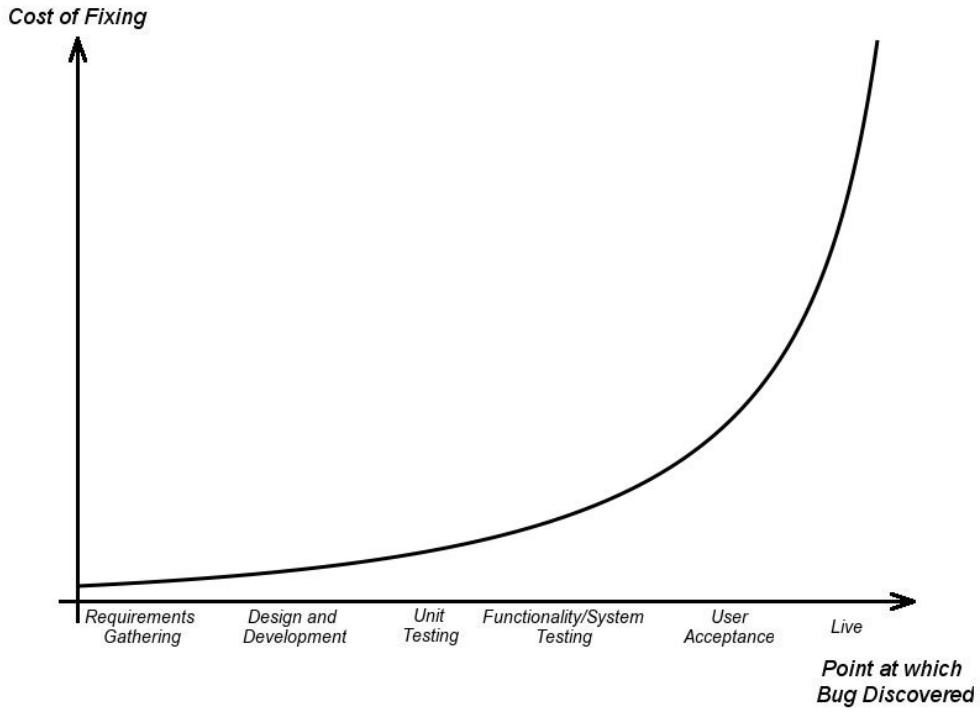
Savoirs associés:

- Test de performance, test de charge, test d'intrusion...

Le concept de tests unitaires n'est pas une nouveauté. Depuis le début de l'informatique, tester fait partie de l'activité quotidienne d'un développeur. Ce qui est nouveau aujourd'hui, c'est que l'on place cette activité, en particulier les tests unitaires, au coeur du processus de conception (ref. Méthodes Agiles, DevOps). Plusieurs raisons à cela :

- Concevoir le test unitaire d'un service avant même d'avoir codé ce dernier, **favorise la modularité** (petites unités à tester) et la **concision** (le développeur n'implémente que l'essentiel).
- Plus un bogue est détecté tôt plus facile sera sa correction, **moins il coûtera**.
- Disponibilité d'**outils** (***xUnit***) d'aide à la conception et exécution de tests unitaires (open source).
- Bonne **intégration** de ces outils dans les ateliers de génie logiciel.
- Augmente la **qualité** générale du code produit. Facile à maintenir (code modulaire) et à tester.
- Permet de **rejouer les tests** à volonté, afin de concourir aux tests de non-regression.

Mots clés : **test unitaire**, **xUnit**, **industrialisation du logiciel**, **intégration continue**



Une définition (simplifiée d'après wikipedia).

> **Un *test* est un ensemble de *cas à tester*, éventuellement accompagné d'une *procédure d'exécution* (séquence d'actions à exécuter). Il est lié à un objectif.**

Un *cas à tester* **ressemble à une expérience scientifique**. Il examine une hypothèse exprimée en fonction de **trois éléments clés** :

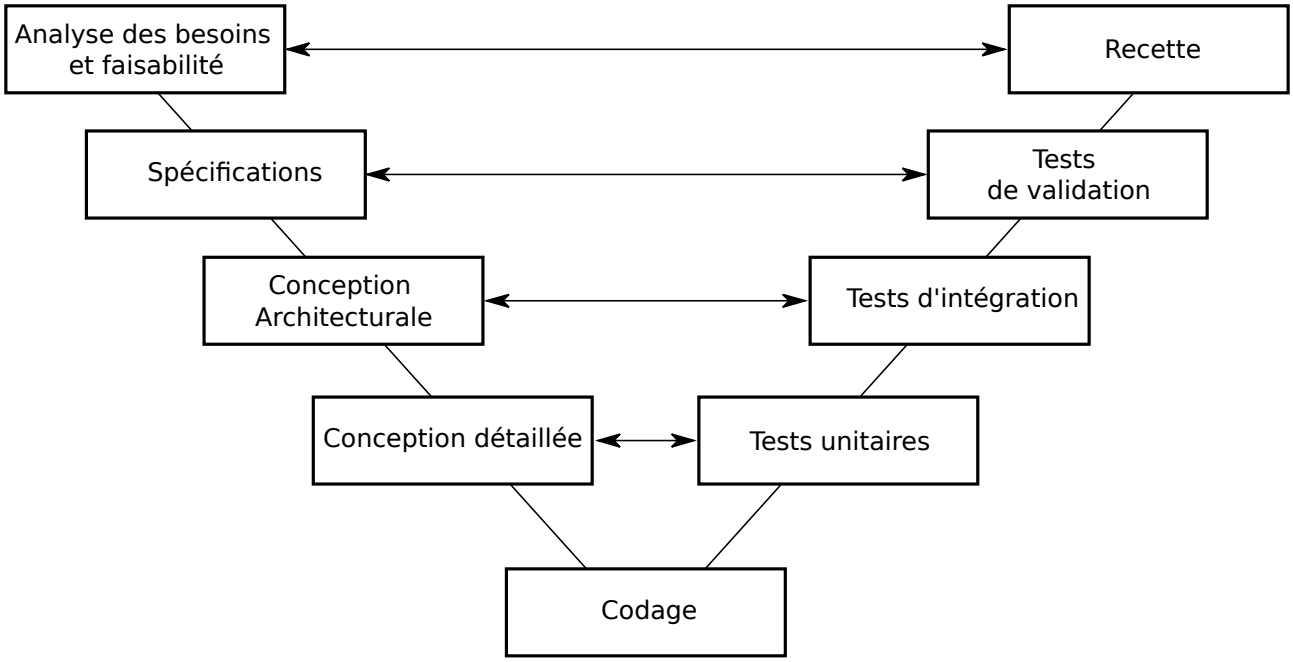
1. **les données en entrée,**
2. **l'objet à tester**
3. **les observations attendues.**

Cet examen est effectué sous conditions contrôlées pour pouvoir tirer des conclusions. Un bon test respecte également l'**exigence de répétabilité**.

L'*activité de test* d'un logiciel est un des **processus du développement de logiciels** . Elle utilise différents types et techniques de tests pour vérifier que le logiciel est conforme à son cahier des charges (vérification du produit) et aux attentes du client (validation du produit).

Les tests de vérification ou de validation visent à s'assurer que ce système réagit de la façon prévue par ses concepteurs (spécifications) ou est conforme aux attentes du client l'ayant commandé (besoins), respectivement.

Structure historique du cycle de vie en V, aujourd'hui intégré dans un cycle itératif et incrémental, voire repensé par une approche *test first*.



https://fr.wikipedia.org/wiki/Cycle_en_V

Plus le nombre d'erreurs trouvées est important, plus il y a de chances qu'il y ait davantage d'erreurs dans le composant logiciel visé. Inversement, *l'absence d'erreurs ne signifie pas que le composant en est exempt*.

Un objet ne peut être testé que si on peut déterminer *précisément* le comportement attendu en fonction des conditions auxquelles il est soumis. Si la spécification ne permet pas cette détermination, la propriété du logiciel qu'elle définit ne peut être testée.

Un test vise à mettre en évidence des défauts de l'objet testé. Cependant il n'a pas pour objectif :

- de diagnostiquer la cause des erreurs,
- de les corriger,
- de prouver la correction de l'objet testé.

Une classification par niveau couramment acceptée :

Niveau 4	**Test d'acceptance (la recette)**
Niveau 3	**Test Système (validation)**
Niveau 2	**Test Intégration**
Niveau 1	**Test Unitaire**

Quelques courtes définitions

Test de recette : L'application doit fonctionner dans son environnement de production (très souvent chargé d'autres services). On s'assure que l'application répond acceptablement à la demande initiale.

Test de validation : Les scénarios de cas d'utilisation des services sont vérifiés, à base de spécifications fonctionnelles (métier).

Test d'intégration : L'application est déployée dans un environnement le plus proche possible de l'environnement de production. On s'assure que le système n'en est pas perturbé et que les tests unitaires fonctionnent correctement dans cet environnement (test de non-regression).

Test unitaire : Chaque service (fonction, méthodes) est testé le plus indépendant des autres, au regard de ses spécifications techniques.