Cours Génie Logiciel – M1 Informatique

Partie II – Test logiciel

Chapitre 1 – Fondamentaux des tests

Structure du chapitre 1

Niveau d’apprentissage

- Les niveaux de connaissance sont fournis pour chaque section et classés de la façon suivante :
  - **K1**: se souvenir
  - **K2**: comprendre
  - **K3**: utiliser
  - **K4**: analyser

1.1 Pourquoi les tests sont-ils nécessaires ?

- LO-1.1.1 : Décrire, avec des exemples, la manière par laquelle un défaut dans un logiciel peut causer des dommages à des personnes, à l'environnement ou à la société (K2)
- LO-1.1.2 : Faire la différence entre la cause initiale du défaut et ses effets (K2)
- LO-1.1.3 : Donner des raisons pour lesquelles les tests sont nécessaires en donnant des exemples (K2)
- LO-1.1.4 : Décrire pourquoi les tests font partie de l'assurance qualité et donner des exemples sur comment les tests contribuent à une qualité accrue (K2)
- LO-1.1.5 : Expliquer et comparer les termes faute, défaut, défaillance et les termes associés erreur et bug (K2)

LO → Learning objective = objective d'apprentissage

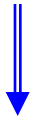
Termes importants (définition à retenir)

- Erreur (synonyme: méprise)
  - Action humaine produisant un résultat incorrect
- Défaut (synonymes: Bug, Faute)
  - Une imperfection dans un composant ou un système qui peut conduire à ce qu'un composant ou un système n'exécute pas les fonctions requises, par exemple une instruction ou une définition de données incorrecte. Un défaut, si rencontré lors de l'exécution, peut causer la défaillance d'un composant ou d'un système.
- Défaillance
  - Écart constaté du composant ou système par rapport au livrable, au service ou au résultat attendu
  - Incapacité d'un système ou d'un composant d'exécuter une fonction requise dans les limites spécifiées. Une défaillance peut être produite quand un défaut est rencontré

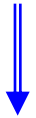
Motivation du test logiciel – Détecter des défauts avant la mise en production

Erreur

(Spécification, conception, programmation)



Défaut dans le logiciel



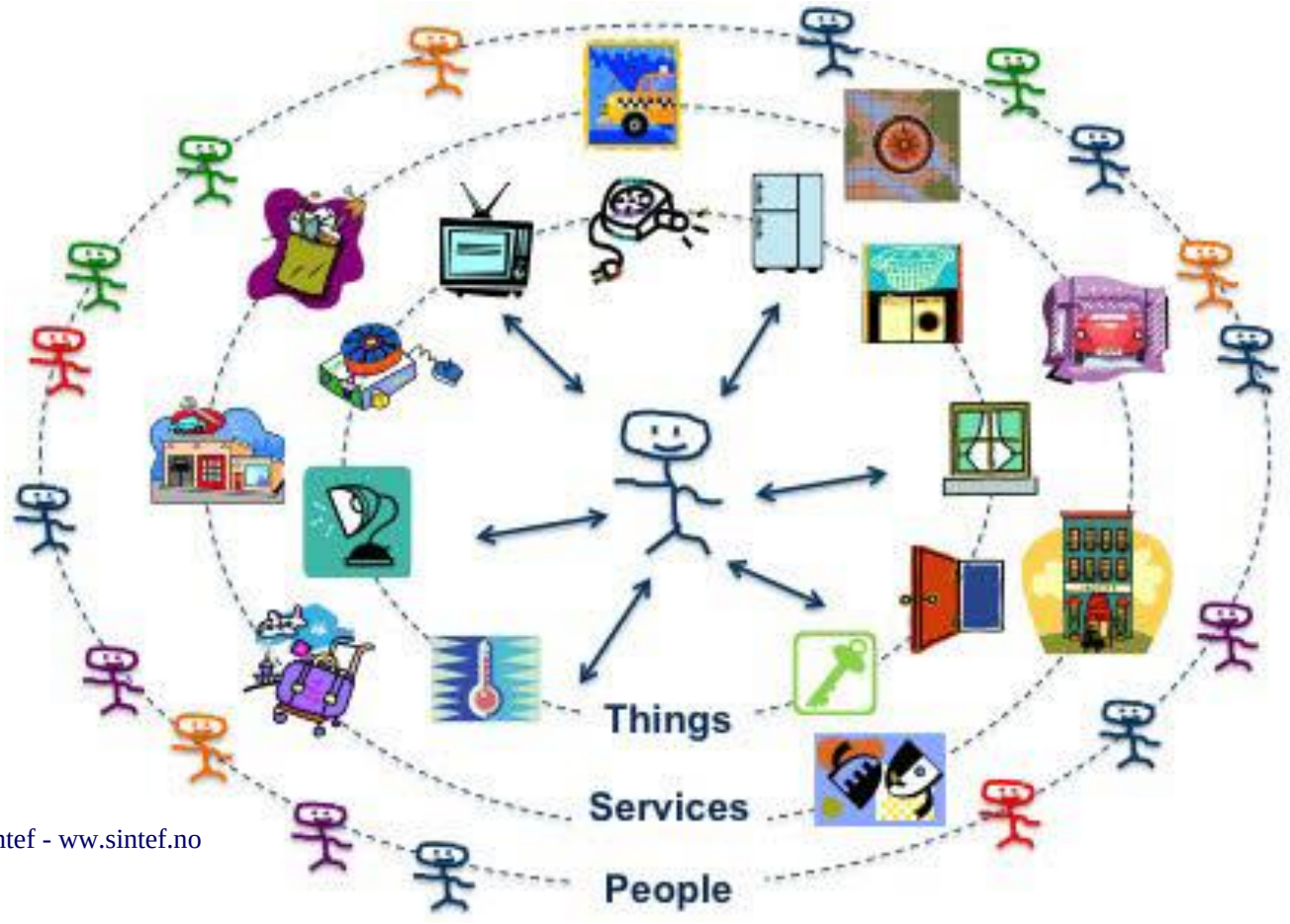
Défaillance du logiciel

(anomalie de fonctionnement)

➔ **Améliorer la qualité du logiciel : Fiabilité, Conformité aux spécifications, Robustesse, Performances, Sécurité ….**

1.1.1 Contexte des systèmes logiciels (K1)

- Le logiciel est de plus en plus omniprésent
- Les anomalies de fonctionnement ont des impacts très importants



Source : Sintef - ww.sintef.no

Impact des anomalies

Sur certains systèmes critiques (transport, énergie, …), l'impact des défaillances peut avoir des conséquences humaines.

Des bugs dans le logiciel d'une voiture Toyota ont bien entraîné la mort de son conducteur

31 octobre 2013 Sécurité 5 Comments

Avec de plus en plus de fonctionnalités critiques contrôlées par l'informatique embarquée, et des millions de lignes de code nécessaires pour réaliser les logiciels de nos voitures actuelles, aucun automobiliste n'est à l'abri d'un bug gênant.

Après une âpre bataille juridique qui aura duré pas moins de 7 ans, un tribunal américain a rendu son verdict : **Toyota a été reconnu responsable dans la mort d'une conductrice à cause de bugs dans le système embarqué de la voiture.**

Une décision juridique qui fait suite à l'accident en 2007 d'une Toyota Camry. Un accident provoqué par l'accélération incontrôlée du véhicule, qui a entraîné la mort de sa conductrice,

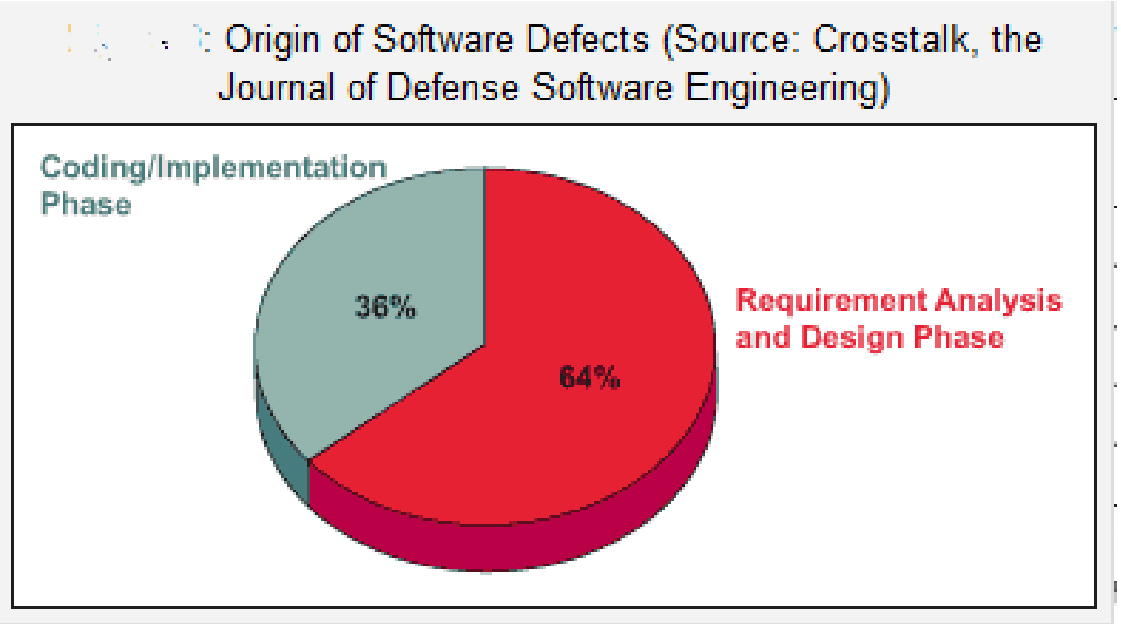
Source - http://www.nikopik.com/2013/10/des-bugs-dans-le-logiciel-dune-voiture-toyota-ont-bien-entraine-la-mort-de-son-conducteur.html

Impact des anomalies

- Pertes financières – ex. Erreur de calcul sur le versement des retraites
- Perte de temps – ex. Avions cloués au sol pour un problème logiciel
- Perte de réputation – ex. Panne de réseau chez un opérateur téléphonique
- Impact sur la vie humaine – systèmes logiciels critiques

1.1.2 Origine des défauts logiciels (K2)

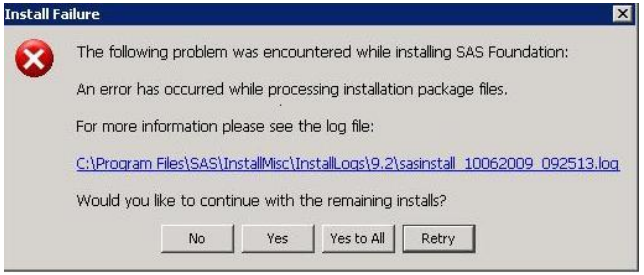
- Les défauts (bug) dans le logiciel proviennent d'erreurs commises tout au long du cycle de vie.



Exemple d'analyse de répartition de la provenance des défauts

Causalité : défaut → défaillance

causalité



Périmètre des tests logiciels

Des défauts dans les logiciels, systèmes ou documents peuvent générer des défaillances, mais tous les défauts ne le font pas.

Autres causes des défaillances

- Les défaillances peuvent aussi être causées par des conditions environnementales :
  - radiations,
  - magnétisme,
  - champs électronique,
  - pollution.
- Ces défaillances peuvent causer des défauts dans les microprogrammes ou influencer l'exécution du logiciel en modifiant les conditions matérielles.
  - Par ex : changement d'état d'une valeur d'un registre du microprocesseur suite à un champ électromagnétique trop proche de la carte embarquée (pollution électromagnétique). D'où la mise en place de norme de compatibilité électromagnétique évitant les perturbations des appareils entre eux.

1.1.3 - Rôle des tests dans le développement, la maintenance et l'utilisation des logiciels (K2)

Les tests contribuent à la qualité du logiciel en permettant **la détection des défauts** avant la mise en production.

Le test suit aussi d'autres motivations :

Exigences légales ou contractuelles

Respect de standards ou de normes industrielles

Impact des tests

- Des tests rigoureux des systèmes et de la documentation peuvent aider à réduire les risques d'occurrence de problèmes dans l'environnement opérationnel.

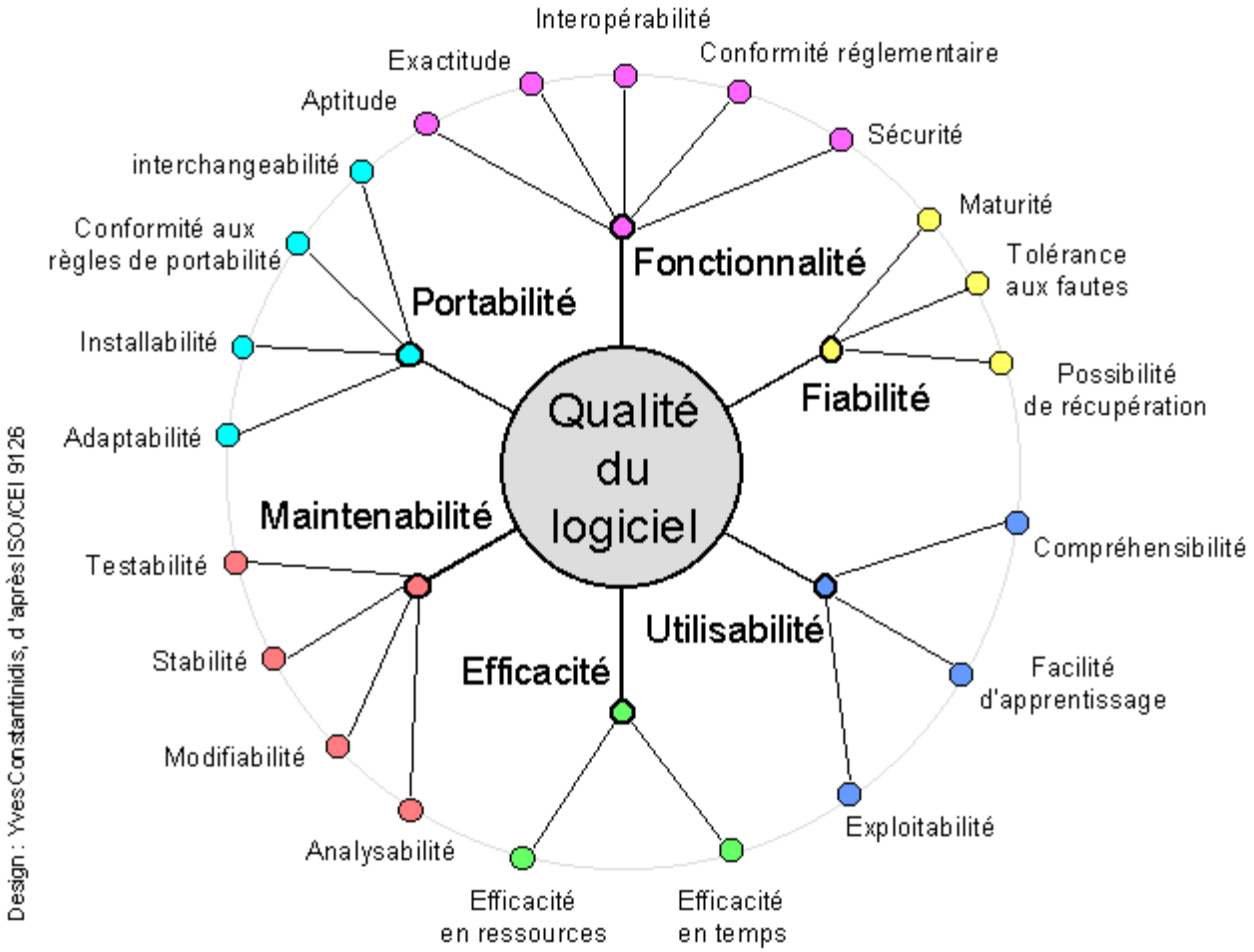
- Ils contribuent à la qualité des systèmes logiciels, si les défauts découverts sont corrigés avant la livraison du système pour un usage opérationnel.

1.1.4 – Test et qualité (K2)

- Qualité
  - Degré par lequel un composant, système ou processus atteint des exigences spécifiées et/ou des besoins ou attentes des clients ou utilisateurs
- Assurance Qualité
  - Partie de la gestion de la qualité visant à fournir l'assurance que les exigences qualité seront atteintes
- Risques
  - un facteur qui pourrait résulter dans des conséquences négatives futures, généralement exprimé comme un impact et une probabilité.

Critères de qualité du logiciel – ISO 9126

Le test vise les critères de qualité du logiciel



Impact des tests sur la qualité

- Les tests peuvent augmenter le niveau de confiance en la qualité d'un logiciel s'ils trouvent peu ou pas de défauts.
- Un test conçu correctement et qui est exécuté sans erreur réduit le niveau de risque général du système.
- Quand les tests trouvent des défauts, la qualité du système logiciel s'accroît quand ces défauts sont corrigés.

Test et amélioration continue

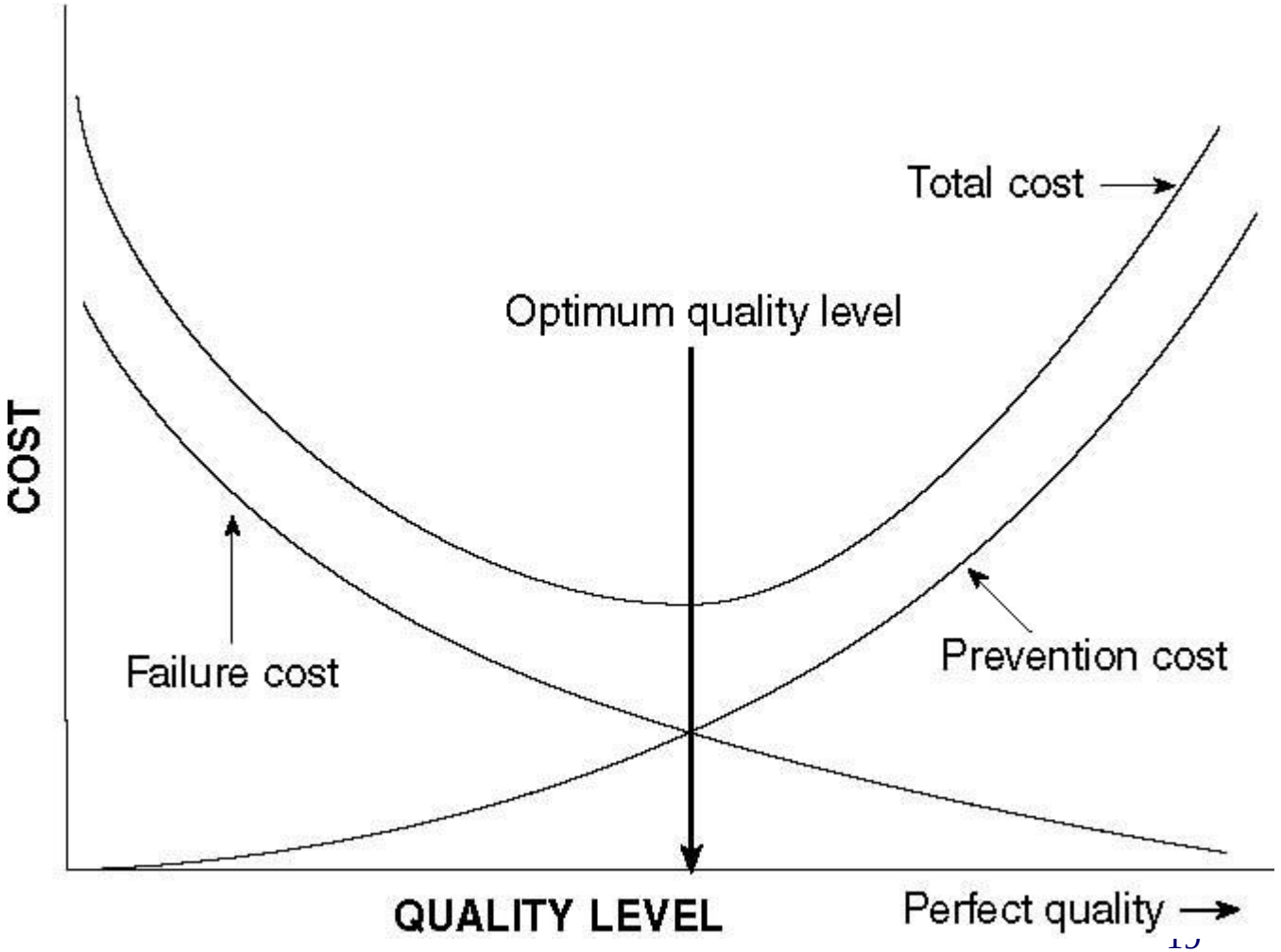
- Des leçons doivent être apprises à partir des projets précédents.
- En comprenant les causes premières des défauts trouvés dans d'autres projets, les processus peuvent être améliorés, ce qui ensuite peut prévenir l'apparition de ces défauts et, en conséquence, améliorer la qualité des systèmes futurs.

- Les tests sont une activité de l'assurance qualité (au côté des standards de développement, de la formation et de l'analyse des défauts, …).

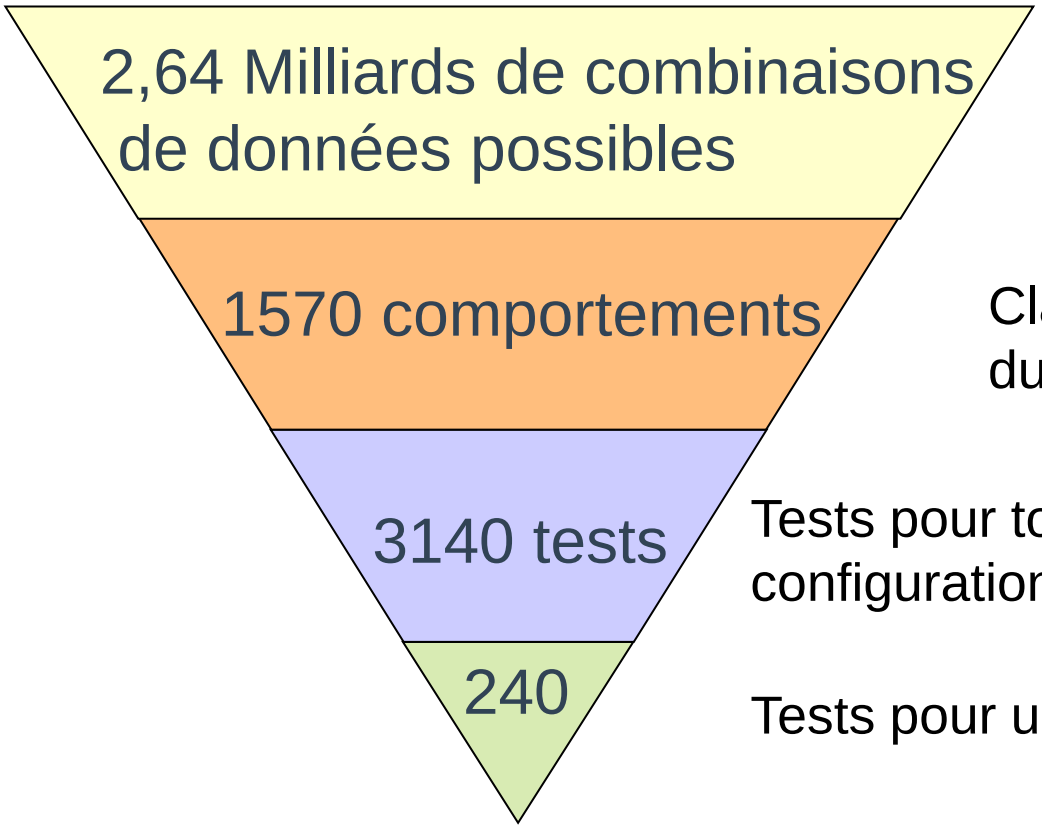
1.1.5 – Combien de test est suffisant ? (K2)

L'équilibre entre le coût de la qualité et le coût de la non-qualité.

Le modèle de Joseph Juran.



Maîtriser la combinatoire : exemple de la validation d'un lecteur de Carte Bancaire



MagIC 500

Classes de comportements du système

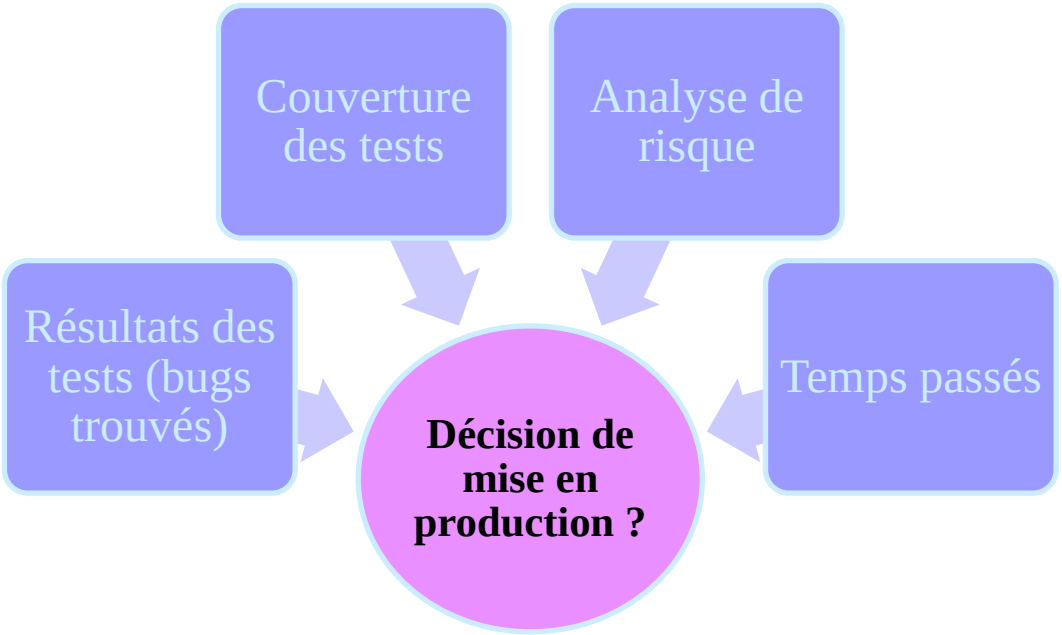
Tests pour toutes les familles de configuration et les valeurs aux bornes

Tests pour une configuration particulière

➔ Trouver le bon compromis est une question de gestion du risque

Arrêt des tests et mise en production

- Les tests doivent fournir suffisamment d'informations pour que les responsables puissent prendre des décisions informées concernant la mise en production du logiciel ou du système en cours de test, pour l'étape de développement suivante ou la livraison aux clients.



1.1 Pourquoi les tests sont-ils nécessaires ? - Quiz

- LO-1.1.1 : Décrire, avec des exemples, la manière par laquelle un défaut dans un logiciel peut causer des dommages à des personnes, à l'environnement ou à la société (K2)
- LO-1.1.2 : Faire la différence entre la cause initiale du défaut et ses effets (K2)
- LO-1.1.3 : Donner des raisons pour lesquelles les tests sont nécessaires en donnant des exemples (K2)
- LO-1.1.4 : Décrire pourquoi les tests font partie de l'assurance qualité et donner des exemples sur comment les tests contribuent à une qualité accrue (K2)
- LO-1.1.5 : Expliquer et comparer les termes faute, défaut, défaillance et les termes associés erreur et bug (K2)

Quiz – section 1.1

Q1 - Quelles affirmations parmi les suivantes sont vraies?

I) Des tests rigoureux et la correction des défauts constatés peuvent aider à réduire le risque de problèmes survenant lors de la mise en production du logiciel

II) Les tests logiciels sont principalement nécessaire afin d'améliorer la qualité de travail des développeurs.

III) Les tests logiciels peuvent être nécessaire pour satisfaire aux exigences légales ou contractuelles.

IV) Des tests rigoureux sont parfois utilisés pour prouver que tous les défauts ont été trouvés.

a) I, II et IV

b) I, II, III et IV

c) I et III

d) I et IV

Quiz – section 1.1

Q2 - De quel mot, le mot « bug » est-il synonyme?

a) Défaillance
b) Défaut
c) Méprise
d) Erreur

Quiz – section 1.1

Q3 - Quand doit on arrêter les tests sur une release de logiciel ?

a) Quand les tests prévus ont été exécutés

b) Quand il n'y plus de temps a y consacré

c) Quand les risques résiduels ont été estimé acceptable

d) Quand tous les défauts ont été corrigés

1.2 Que sont les tests ? (K2)

- LO-1.2.1 : Rappeler les objectifs habituels des tests. (K1)
- LO-1.2.2 : Décrire les objectifs des tests dans le développement logiciel, la maintenance et les opérations comme moyen de trouver des défauts, fournir la confiance et l'information, et prévenir les défauts. (K2)
- LO-1.2.3 : Faire la différence entre tester et déboguer (K2)

Termes (à retenir)

- Test
  - un ensemble d'un ou plusieurs cas de tests
- Objectif de test
  - une raison ou but pour la conception et l'exécution d'un tests.
- Cas de test
  - un ensemble de valeurs d'entrée, de préconditions d'exécution, de résultats attendus et de postconditions d'exécution, développées pour un objectif ou une condition de tests particulier, tel qu'exécuter un chemin particulier d'un programme ou vérifier le respect d'une exigence spécifique

Termes (à retenir)

- Exigence
  - une condition ou capacité requise par un utilisateur pour résoudre un problème ou atteindre un objectif qui doit être tenu ou possédé par un système ou composant pour satisfaire à un contrat, standard, spécification ou autre document imposé formellement
- Déboguer
  - le processus de trouver, analyser et éliminer les causes de défaillance dans les logiciels
- Revue
  - une évaluation d‘un état d‘un produit ou projet pour s‘assurer des déviations par rapport aux résultats planifiés et recommander des améliorations. Exemples : revue de gestion, revue informelle, revue technique, inspection et relecture technique

Processus de test

- Le test n'est pas limité à l'exécution du logiciel dans le but d'identifier des défaillances : il est aussi nécessaire de planifier, de définir des objectifs, de concevoir des conditions de tests, de prévoir des données de tests, des critères de début et d'arrêt, des environnements de tests et bien sûr de contrôler tout cela.

Les principales activités du test logiciel

Tests dynamiques / Tests statiques

- Les tests dynamiques concernent l'exécution de l'application pour la tester, dans un contexte donné (l'environnement de test) et des données de test spécifiques.

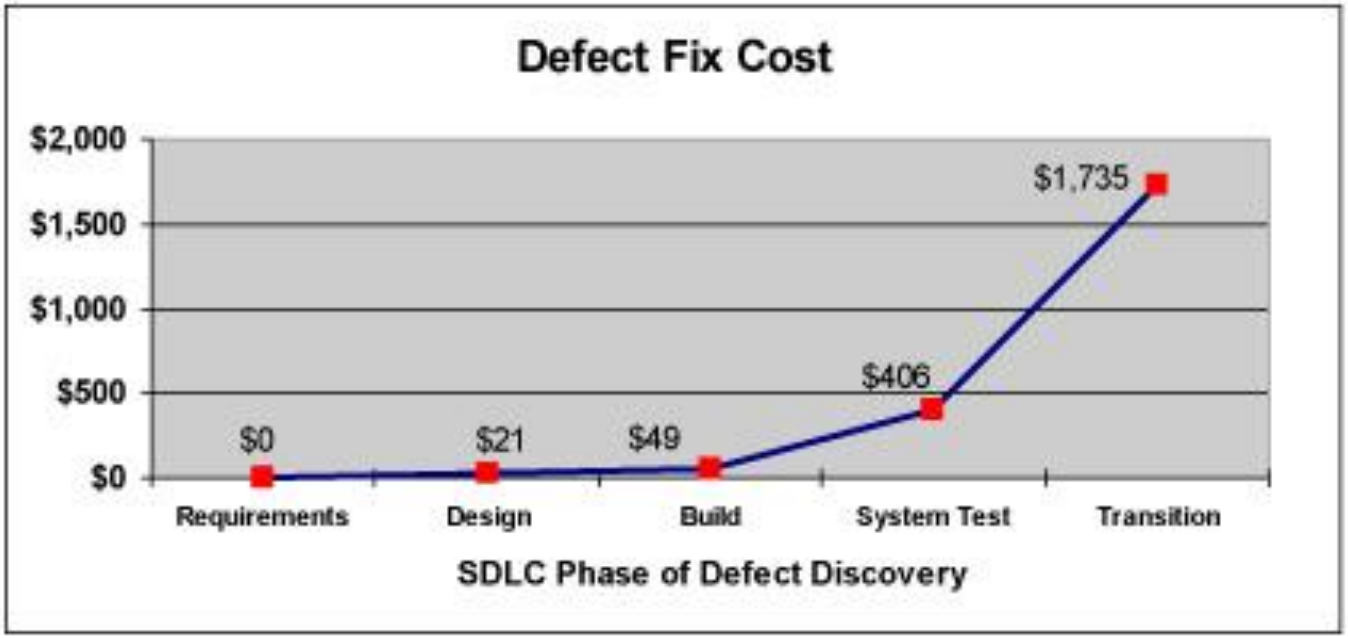
- Les tests statiques sont aussi appelés revues ou inspections. Ils sont utiles pour détecter des défaux sur l'ensemble des artefacts du cycle de développement du logiciel : exigences, spécifications, conception, code, tests, …

Les objectifs de test

Les objectifs de test sont variables d'un projet à un autre. Voici quelques exemples :

- Trouver des défauts (pour les corriger)
- Acquérir de la confiance sur le niveau de qualité (par mesure du nombre de défauts trouvés)
- Fournir de l'information utile aux prises de décision (par exemple sur la qualité d'un code)
- Prévenir des défauts (par l'impact des mesures prisent à partir des résultats du test).

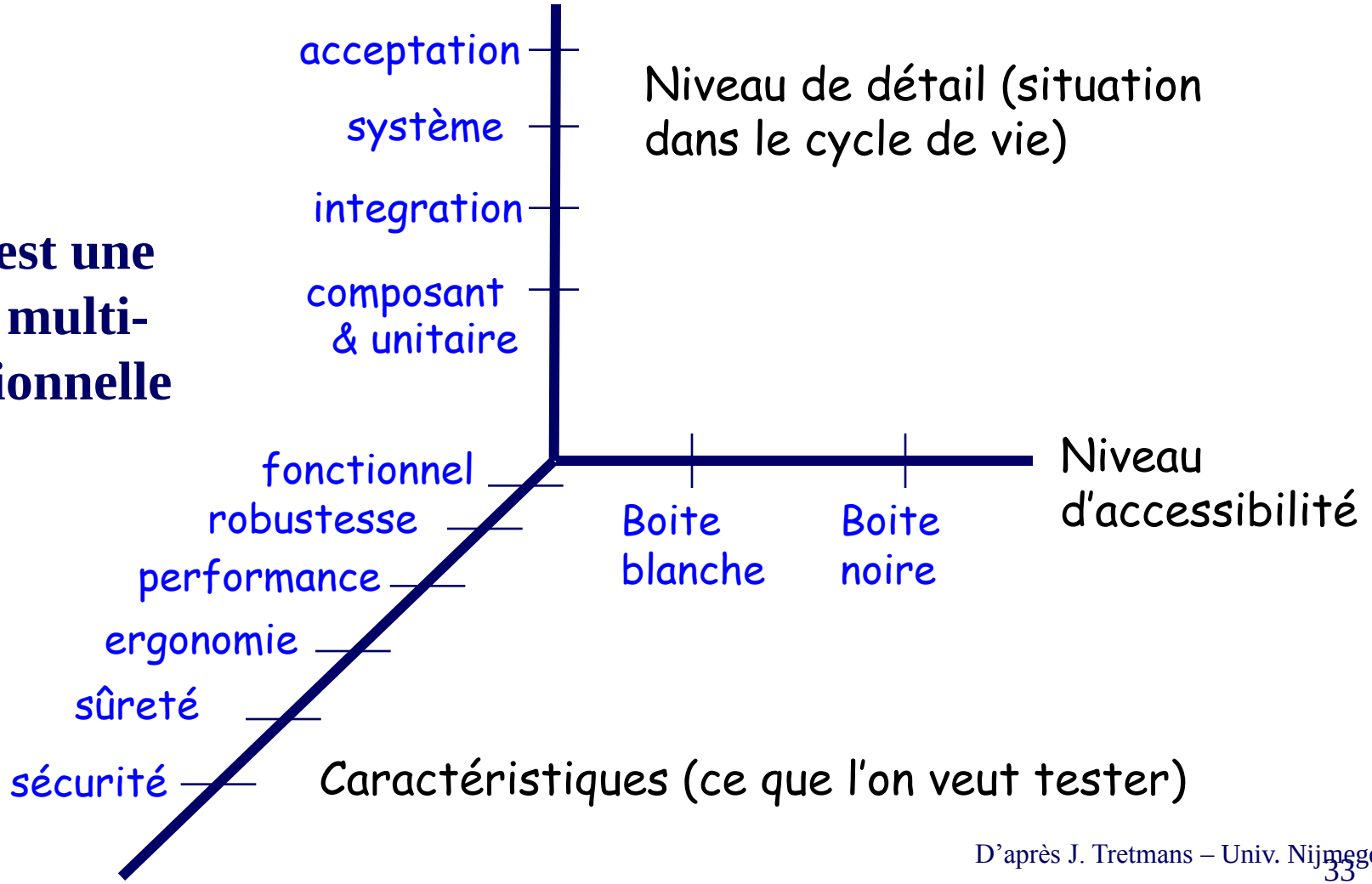
Coût de la correction d'un défaut



- Les activités de test doivent commencer le plus tôt possible dans le cycle de développement du logiciel.
- Les revues de documents (p.ex. exigences), l'identification et la résolution de problèmes peuvent aussi aider à prévenir l'apparition de défauts dans le code.

Aspect multi-dimensionnel du test

Le test est une activité multi-dimensionnelle



D'après J. Tretmans – Univ. Nijmegen

Différents points de vue pour le test en fonction du projet

- Par exemple, dans les tests de développement (p.ex. tests des composants, d'intégration ou système), l'objectif principal peut être de détecter le plus de défaillances possible de façon à identifier et à corriger les défauts dans le logiciel.

- Dans les tests d'acceptation, l'objectif principal peut être de confirmer que le système fonctionne comme attendu, pour s'assurer qu'il répond aux exigences.

- Dans certains cas, l'objectif principal des tests peut être d'évaluer la qualité d'un logiciel (sans chercher à trouver des anomalies), de façon à fournir aux responsables de l'information sur les risques de distribuer un système à un moment précis.

Différents points de vue pour le test en fonction du projet

- Les tests en phase de maintenance du logiciel incluent souvent des tests pour s'assurer que de nouvelles anomalies n'ont pas été introduites pendant le développement des évolutions. On parle alors de test de regression.

- Pendant les tests de mise en opération (test alpha / bêta), les objectifs principaux peuvent être d'évaluer les caractéristiques du système telles la disponibilité ou la fiabilité en situation réelle.

Tester n’est pas déboguer !

- Les activités de test mettent en évidence des défaillances et/ou des défauts.

ID	P	Status	Severity	Version	Summary
107*	P2	RESOLVED	enhancement	0.4	[BZ] Support integration with deadline
117					
118					ling sort order
119	P3	RESOLVED	normal	0.4	[BZ] Make default status !CLOSED
120	P3	RESOLVED	normal	0.4	[BZ] Support spaces in comma separated list
123*	P3	RESOLVED	normal	0.4	Create alias for name to product so that column name is called product and same as field search
104	P4	NEW	normal	unspecified	[BZ] Client site sortable columns
116	P4	NEW	normal	unspecified	[BZ] Dodgy characters in comments can break page
110	P5	NEW	normal	unspecified	[BZ] Hide headers if id explicitly set and less than 3 ids set, also make it clearer that it's a task when inlined
122	P5	NEW	enhancement	unspecified	Support clickable heading to sort

107 Added generic column handling to allow +deadline to add the deadline column

- Les activités de debogage permettent de localiser le défaut et de le corriger.
- Un test de confirmation doit être réalisé après correction. .

L’activité de débogage

- consiste à détecter, localiser et corriger des fautes dans un programme
- identifie l’emplacement des défauts dans le code
- identifie la cause du défaut
- nécessite une connaissance approfondie de la conception et du code
- est effectué par un développeur

1.2 Que sont les tests ? (K2)

- LO-1.2.1 : Rappeler les objectifs habituels des tests. (K1)
- LO-1.2.2 : Décrire les objectifs des tests dans le développement logiciel, la maintenance et les opérations comme moyen de trouver des défauts, fournir la confiance et l'information, et prévenir les défauts. (K2)
- LO-1.2.3 : Faire la différence entre tester et déboguer (K2)

Quiz – section 1.2

Q1 – Un entreprise achète un composant logiciel sur étagère pour l'intégrer dans son système de gestion. Des tests sont prévus sur ce composant avant sa mise en production. Quelle est leur raison principale pour réaliser ces tests?

a) Détecter des défauts dans ce composant

b) Avoir confiance en ce composant

c) Obtenir des preuves en vue d'une action en justice

d) Former les utilisateurs

Quiz – section 1.2

Q2 – Le role du testeur est-il plutôt

a) de concevoir et réaliser les tests pour rendre compte des résultats

b) de détecter et corriger les défauts dans le logiciel

c) de déboguer le logiciel

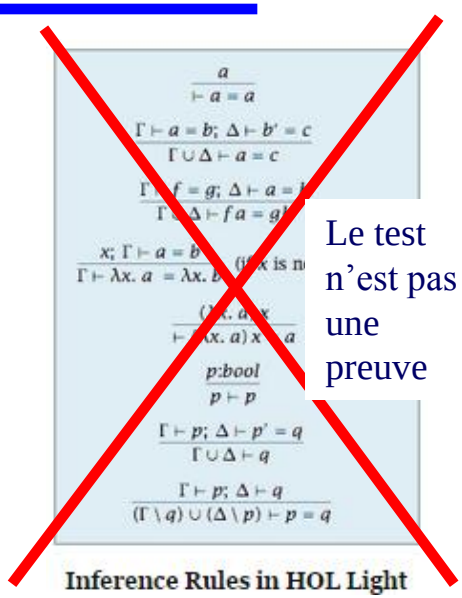
d) de trouver les causes initiales premières de chaque anomalies de fonctionnement

1.3 Principes généraux des tests (K2)

- LO-1.3.1 : Expliquer les 7 principes généraux des tests (K2)
  - Principe 1 : Les tests montrent la présence de défauts
  - Principe 2 : Les tests exhaustifs sont impossibles
  - Principe 3 : Tester tôt
  - Principe 4 : Regroupement des défauts
  - Principe 5 : Paradoxe du pesticide
  - Principe 6 : Les tests dépendent du contexte
  - Principe 7 : L’illusion de l’absence d’erreurs

Principe 1 : Les tests révèlent la présence de défauts (pas leur absence)

- Le test permet de détecter des défauts et des anomalies.
- Mais, ce n'est pas un preuve de correction du logiciel

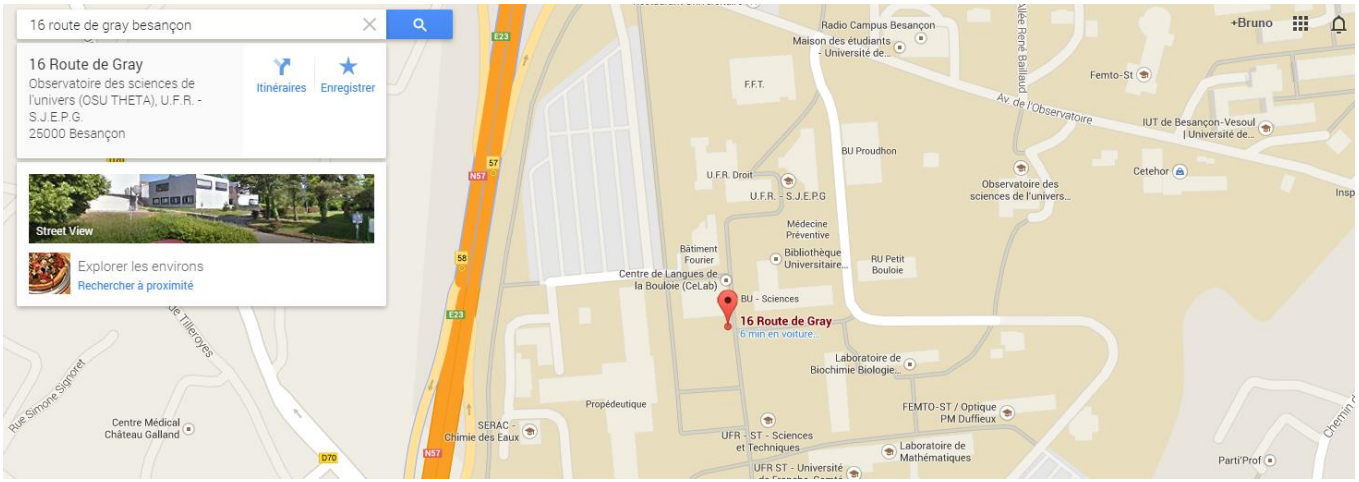


- Les tests réduisent la probabilité que des défauts restent cachés dans le logiciel.

=> le test est une méthode de vérification partielle de logiciels

Principe 2 : Les tests exhaustifs sont impossibles

- Combien de cas de tests pour tester Google Maps de façon exhaustive ?



Tests exhaustifs

- une approche des tests selon laquelle la suite de tests comprend toutes les combinaisons de valeurs d'entrée et de préconditions.

Principe 2 : Les tests exhaustifs sont impossibles

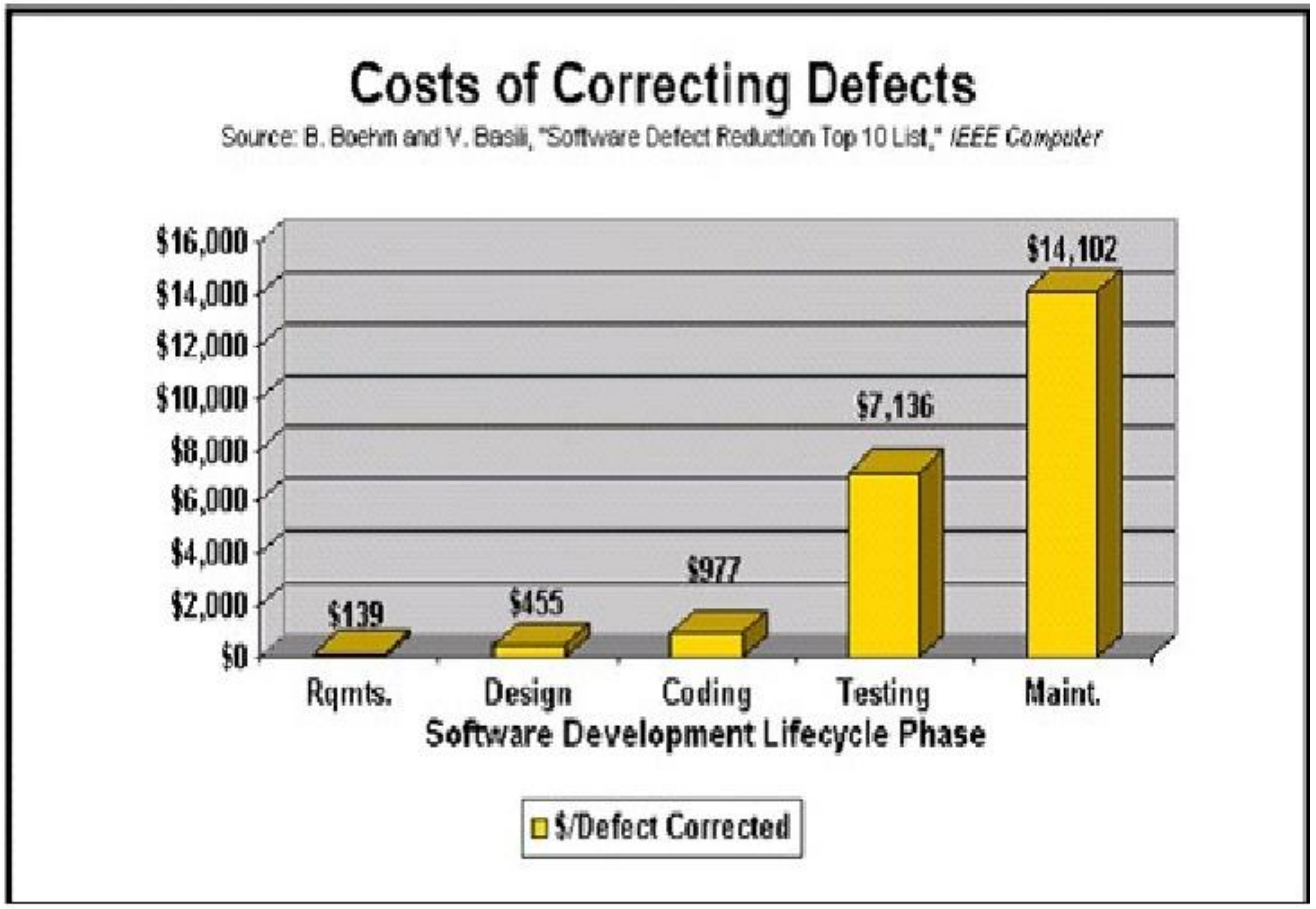
- Tout tester (toutes les combinaisons d'entrées et de pré-conditions) n'est pas faisable en pratique

➔ Exemple : un logiciel avec 5 entrées codées sur 8 bits admet 2^{40} valeurs différentes en entrée

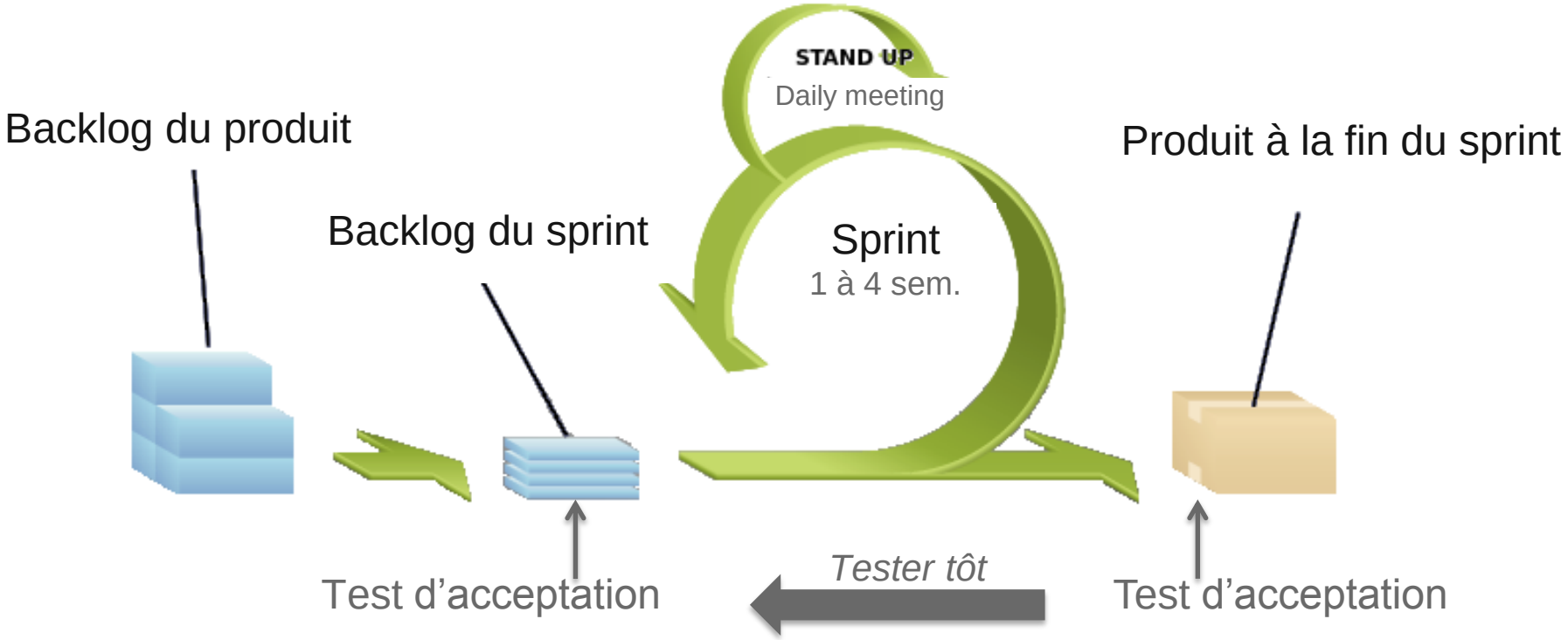
- Plutôt que des tests exhaustifs, nous utilisons l'analyse des risques et des priorités pour focaliser les efforts de tests.

Principe 3 – Tester tôt

- Plus une anomalie est détectée tard, plus son coût de correction est élevé



Principe 3 – Tester tôt



Equipe Scrum

Principe 4 – Regroupement des défauts

- Un petit nombre de modules contiennent généralement la majorité des défauts détectés lors des tests
- ou affichent le plus de défaillances en opération.

- Il faut adapter le test à la densité des défauts en fonction des modules

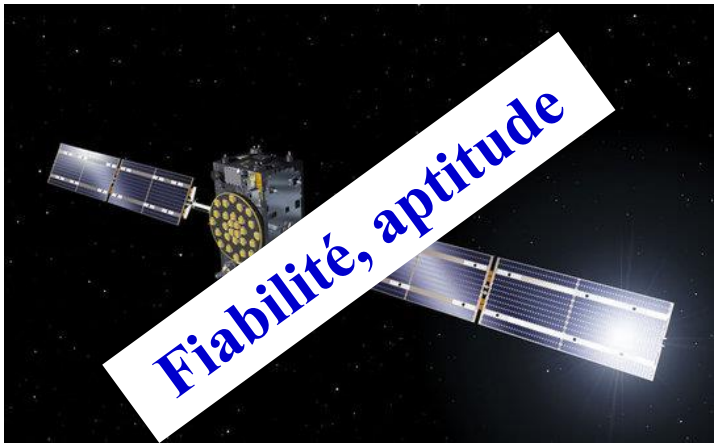
Principe 5 – Paradoxe du pesticide

- Si les mêmes tests sont répétés de nombreuses fois, il arrivera que le même ensemble de cas de tests ne trouvera plus de nouveaux défauts.

  - Pour éviter ce « paradoxe du pesticide », les tests doivent donc être renouvellé pour couvrir de nouvelles données et/ou des scénarios differents

Principe 6 – Les tests dépendent du contexte

- Les tests sont effectués différemment dans des contextes différents.
  - Les objectifs et techniques de test vont être adaptées à la nature du système testé

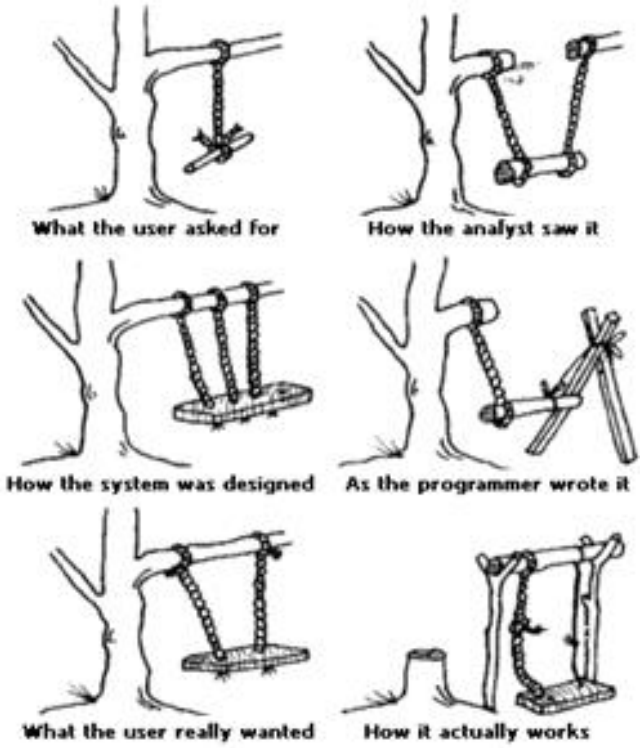


Principe 7 – L'illusion de l'absence d'erreurs

- Trouver et corriger des défauts n'aide pas si le système conçu est inutilisable et ne répond pas aux besoins et aux attentes des utilisateurs.

Du logiciel :

1. qui répond aux besoins
2. qui soit de qualité



Section 1.3 - Quiz

Q1 - Une équipe de test a trouvé 147 défauts, et 7 défauts ont été trouvé après la mise en production. Le client demande que pour la prochaine release, il soit garanti que tous les défauts seront trouvés. Lesquels des principes suivants peuvent être utilisés comme argument pour lui dire que cette garantie n'est pas possible ?

I) Le paradoxe du pesticide

II) Le regroupement des défauts

III) Les tests exhaustifs sont impossibles

IV) L'illusion de l'absence d'erreurs

V) Les tests montrent la présence de défauts (par leur absence)

VI) Tester tôt

a) I, V et VI

b) III

c) IV et V

d) III et V

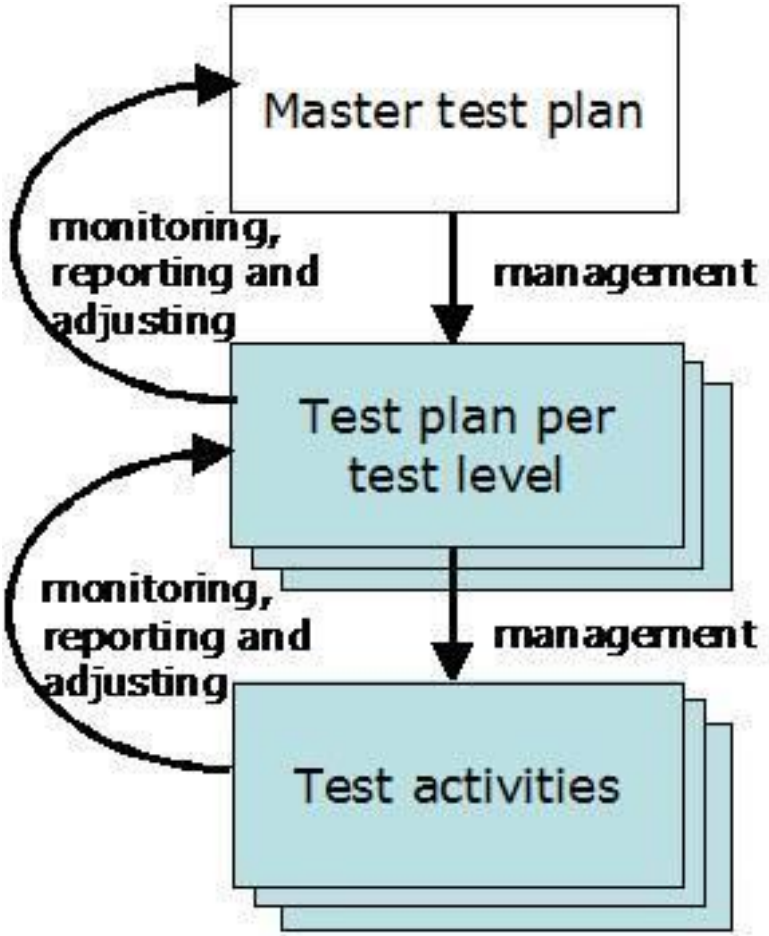
1.4 Processus de tests fondamental (K1)

- LO-1.4.1 : Rappeler les 5 activités de test fondamentales et les tâches associées, de la planification aux activités de clôture (K1)

1.4.1 Planification et contrôle des activités de test (K1)

- La planification des tests consiste à définir les objectifs du test et à spécifier les activités de test à mettre en œuvre pour atteindre les objectifs.

- Le contrôle des tests est une activité continue de suivi d'avancement des tests.



Source - TMap

Plan de test – Exemple de table des matières – d'après le standard IEEE 829 – 2008 (1/2)

1) **Introduction**

1.1 Objet : courte description des objectifs du plan de test

1.2 Définition : définition des termes et des acronymes utilisés dans le document

2) **Références** – Énumérer les références à d'autres documents appropriés.

Normes et standards

Plan de projet et plan d'assurance qualité

Cahier des charges,

Documents d'analyse

Documents de conception…

3) **Périmètre couvert par le plan de test** – Énumérer les éléments à tester (composants, produits, classe, module...). Ainsi que la version du composant et autres informations techniques.

4) Caractéristiques de qualité testées – Énumérer les propriétés testées par ce plan, d'un point de vue utilisateur.

Mentionner le risque sur le produit en cas d'échec. (élevé, médium, faible)

Mentionner le risque potentiel de découvrir des défauts (élevé, aucun)

5) **Exigences non couvertes par le plan de test** – Énumérer les propriétés qui ne sont pas testées par ce plan d'un point de vue utilisateur et expliquer les raisons.

6) **Processus et stratégie de test** – Décrire les activités, techniques et outils qui seront utilisés pour les tests avec suffisamment de détail pour permettre l'estimation des ressources et du temps nécessaires. Cette section définit aussi la stratégie globale en termes de priorité de test.

.

Plan de test – Exemple de table des matières – d'après le standard IEEE 829 – 2008 (2/2)

7) **Critères d'acceptation des tests** – Définir les critères de passage et d'échec des éléments testés.

8) **Critères d'arrêt des tests** – Décrire les circonstances où l'exécution des essais est interrompue et les conditions pour continuer.

9) **Identification des documents livrables** – Énumérer et définir les livrables qui doivent être élaborés durant les tests.

Référentiel de tests

Rapport d'exécution : verdict de chacun des cas de test lors de l'exécution

Rapports des anomalies : description et suivi des anomalies découvertes lors de l'exécution des tests

Rapport sommaire : Sommaire des activités et résultats des tests

10) **Gestion des anomalies** – Décrire comment rapporter, suivre et résoudre les anomalies détectées.

11) **Définition des activités de test** – Description des différentes activités réalisées lors des phases de test.

12) **Outillage et infrastructure de test** – Définition des outils et de l'infrastructure de test (environnement de test, base de recette, moyens matériels et réseau).

13) **Responsabilité** – Niveaux de responsabilité lors des phases de test (répartition entre client, développement et équipe de test).

14) **Équipe de test** – Définition de l'équipe de test, des compétences et du plan de formation nécessaires.

15) **Planning prévisionnel** – Planning prévisionnel global des phases de test incluant les principaux jalons.

16) **Gestion des risques** – Risques identifiés et gestion prévisionnelle.

1.4.2 - Analyse et conception des tests (K1)

Taches principales:

- Réviser les bases du test (telles que les exigences, le niveau de risque, les rapports d'analyse de risque, l'‟architecture, la conception et les interfaces)
- Evaluer la testabilité des exigences et du système
- Identifier et prioriser les conditions de test sur la base de l'analyse des articles de test, la spécification, le comportement et la structure du logiciel
- Concevoir et prioriser les tests de haut niveau
- Identifier les données de test nécessaires pour les conditions de test et les cas de test
- Concevoir l'‟initialisation de l'environnement de test et identifier les infrastructures et outils requis
- Créer une traçabilité bi-directionnelle entre les bases de test et les cas de test

1.4.3 Implémentation et exécution des tests

- Les tests peuvent être exécutés manuellement ou automatiquement
  - Manuel : Implémentation → documentation précise étapes de chaque test
  - Automatique : Implémentation → utilisation d'outillage de test automatique sur l'IHM ou développement de scripts de test sur les APIs

Le choix va dépendre de considérations spécifiques au projet :

- Nombre de fois que le test sera exécuté (non-régression)
- Complexité de mise en oeuvre de l'automatisation
- Savoir faire au sein du projet
- …

1.4.3 Implémentation et exécution des tests

Taches principales:

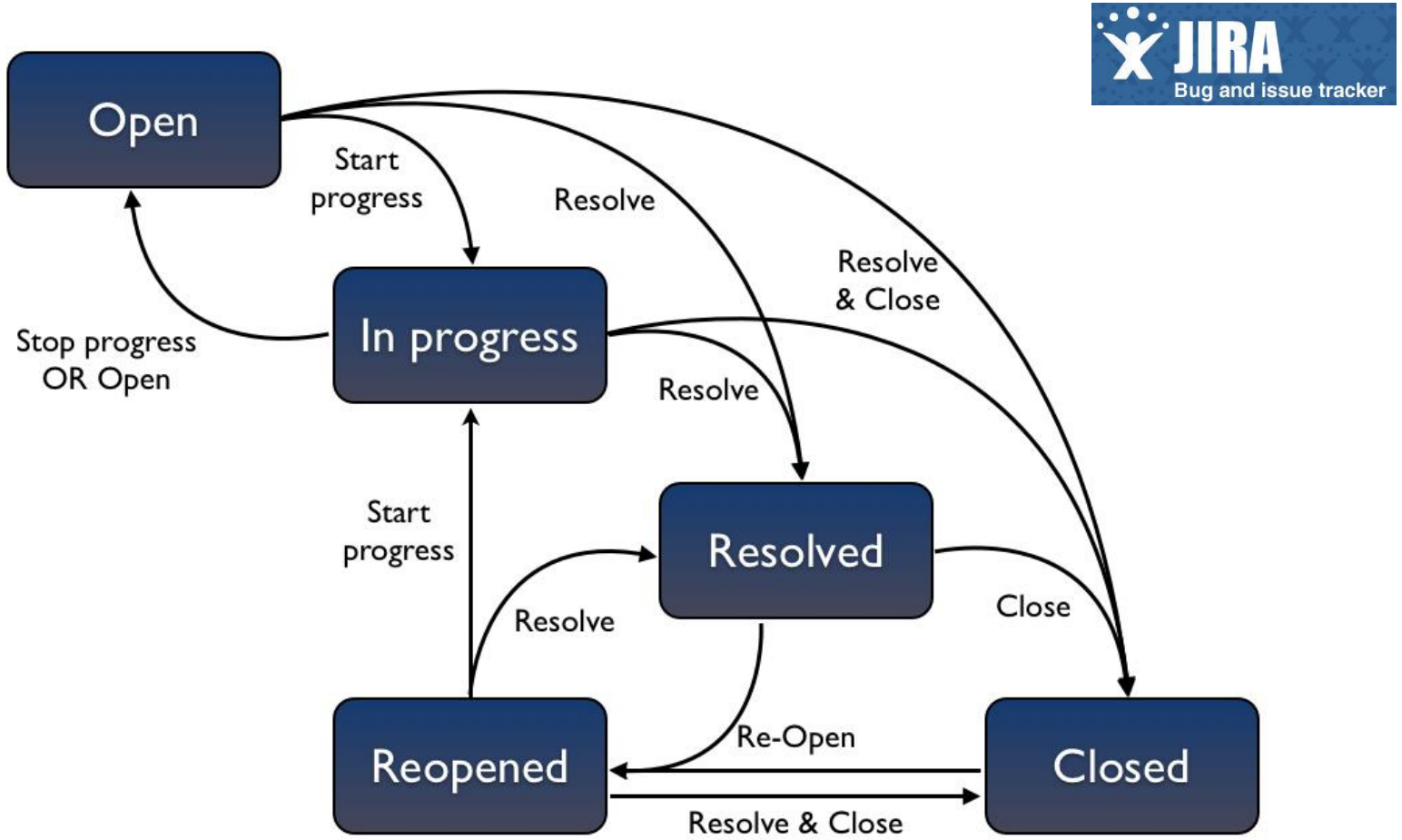
- Finaliser, développer et prioriser les cas de test (y compris l'identification des données de test)
- Développer et prioriser les procédures de test, créer les données de test et, éventuellement, préparer les harnais de test et écrire les scripts de tests automatiques
- Créer des suites de tests à partir des procédures de test pour une exécution rentable des tests
- Vérifier que les environnements de tests ont été mis en place correctement
- Vérifier et mettre à jour la traçabilité bi-directionnelle entre les bases de test et les cas de test
- Exécuter les procédures de test soit manuellement soit en utilisant des outils d'exécution de tests, en suivant la séquence planifiée

1.4.3 Implémentation et exécution des tests

Taches principales (suite):

- Consigner les résultats de l'exécution des tests et enregistrer les identités et versions des logiciels en test, outils de test et testware
- Comparer les résultats obtenus et les résultats attendus.
- Signaler les divergences comme des incidents et les analyser de façon à établir leur cause
- Répéter les activités de test en réponse aux actions prises pour chaque divergence :
  - » Test de confirmation → validation que la correction des correctement réalisée
  - » Test de regréssion (aussi appelé test de non-régression) →s'assurer que des défauts n‟ont pas été introduits dans des secteurs non modifiés du logiciel ou que le défaut corrigé n‟a pas découvert d‟autres défauts

Suivi des anomalies – Cycle de vie



1.4.4 Evaluer les critères de sortie et informer (K1)

Taches principales :

- Vérifier les registres de tests en fonction des critères de sortie spécifiés dans la planification des tests
- Evaluer si des tests supplémentaires sont requis ou si les critères de sortie doivent être changés
- Ecrire un rapport de synthèse des tests pour les parties prenantes

1.4.5 Activités de clôture des tests (K1)

- Objectifs principaux des activités de clôture
  - Améliorer les processus, méthodes et outils existant grâce à des retours d'expérience et demandes de changements
  - Rendre disponibles des éléments réutilisables pour des versions ultérieures de l'application testée ou par des projets ayant des similarités
  - Transmettre aux acteurs suivants (exploitation et maintenance par exemple) des informations utiles (défauts résiduels) et des éléments réutilisables (tests automatisés)

Termes importants (définition à retenir)

- Politique de tests
  - un document de haut niveau décrivant les principes, approches et objectifs majeurs de l'organisation ayant trait aux tests
- Stratégie de test
  - un document de haut niveau définissant, pour un programme, les niveaux de tests à exécuter et les tests dans chacun de ces niveaux (pour un ou plusieurs projets)
- Niveau de test
  - un groupe d'activités de test qui sont organisées et gérées ensemble. Un niveau de test est lié aux responsabilités dans un projet. Les exemples de niveaux de test sont les tests de composants, les tests d'intégration, les tests système et les tests d'acceptation

Termes importants (définition à retenir)

- Base de tests
  - tous les documents à partir desquels les exigences d'un composant ou système peuvent être déduites. La documentation sur laquelle les cas de tests sont basés
- Conditions de tests
  - article ou événement d'un composant ou système qui pourrait être vérifié par un ou plusieurs cas de tests; par exemple une fonction, une transaction, un attribut qualité ou un élément de structure
- Couverture de tests
  - degré, exprimé en pourcentage, selon lequel un élément de couverture spécifié a été exécuté lors d'une suite de tests

Termes importants (définition à retenir)

- Critère (de sortie, de succès, de complétude)
  - règle de décision utilisée pour déterminer si un élément de tests (fonction) ou caractéristique a réussi ou échoué lors d'un tests
- Données de tests
  - donnée qui existe (p.ex. dans une base de données) avant qu'un test ne soit exécuté, et qui affecte ou est affectée par le composant ou système en test
- Exécution des tests
  - processus consistant à exécuter un test sur un composant ou système en test, en produisant des résultats actuels

Termes importants (définition à retenir)

- Procédure de tests
  - document spécifiant la séquence d'actions pour l'exécution d'un test
- Rapport de synthèse de tests
  - document synthétisant les activités et résultats de tests. Il contient aussi une évaluation des articles de tests correspondants par rapport aux critères de sortie
- Testware
  - artefact produit pendant le processus de test afin de planifier, concevoir et exécuter les tests, tel que la documentation, les scripts, les entrées, les résultats attendus, les procédures de mise en place et de nettoyage, les fichiers, bases de données, environnements et tout logiciel ou utilitaires supplémentaire utilisé dans les tests

Section 1.4 - Quiz

Q1 – Laquelle de ces activités ne fait pas partie du processus de test ?

a) Planification des tests et contrôle

b) Analyse et conception des tests

c) Implémentation et exécution des tests

d) Correction des anomalies détectées

e) Evaluer les critères de sortie et Reporting

f) Activités de clôture des tests

1.5 La psychologie des tests (K2)

- LO-1.5.1 : Rappeler les facteurs psychologiques ayant une influence sur le succès des tests (K1)

- LO-1.5.2 : Comparer la mentalité d'un testeur avec celle d'un développeur (K2)

Termes importants (définition à retenir)

- Estimation d'erreur / Modèle de faute
  - une technique de conception de tests où l'expérience du testeur est utilisée pour anticiper les défauts pouvant être présents dans le composant ou système en cours de tests, comme résultat des erreurs faites, et pour concevoir des tests spécifiques afin de les exposer
- Indépendance du test
  - séparation des responsabilités qui favorise la réalisation des objectifs de test

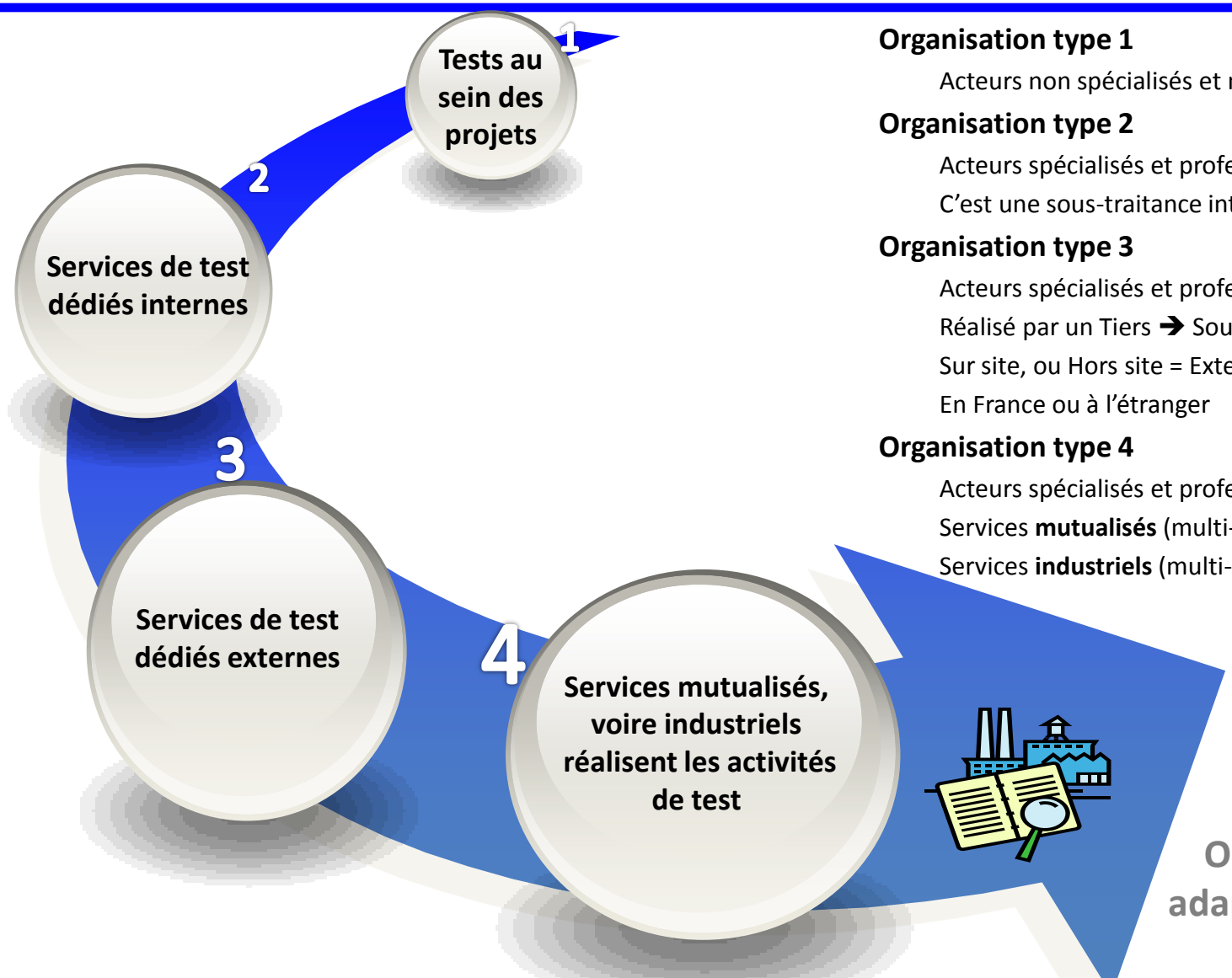
Psychologie du test – Tester ≠ Développer / Déboguer

- La tournure d'esprit à utiliser pendant les tests et les revues est différente de celle utilisée lors de l'analyse ou du développement
  - Avec la mentalité appropriée, des développeurs sont capables de tester leur propre code
  - Confier cette activité à un testeur contribue à concentrer l'effort et apporte des bénéfices d'une approche indépendante et d'un autre point de vue sur les exigences à tester

Niveaux d'indépendance du test

- Tests conçus par la personne qui a écrit le logiciel
  - Niveau d'indépendance bas
  - Peut manquer d'objectivité (parti pris de l'auteur)
- Tests conçus par d'autres personnes appartenant à l'équipe de développement que celle qui a produit le code
  - Niveau d'indépendance moyen
  - Généralement plus d'objectivité que l'auteur
- Tests conçus par d'autres personnes appartenant à un groupe différent de l'organisation
  - Niveau d'indépendance haut
  - Généralement une équipe de test indépendante
- Tests conçus par une autre entreprise
  - Niveau d'indépendance très haut
  - Infogérance ou organisme de certification

Exemples d'organisation des tests



Organisation type 1

Acteurs non spécialisés et non professionnalisés

Organisation type 2

Acteurs spécialisés et professionnalisés

C'est une sous-traitance interne ➔ refacturations interne

Organisation type 3

Acteurs spécialisés et professionnalisés

Réalisé par un Tiers ➔ Sous-traitance externe

Sur site, ou Hors site = Externalisé

En France ou à l'étranger

Organisation type 4

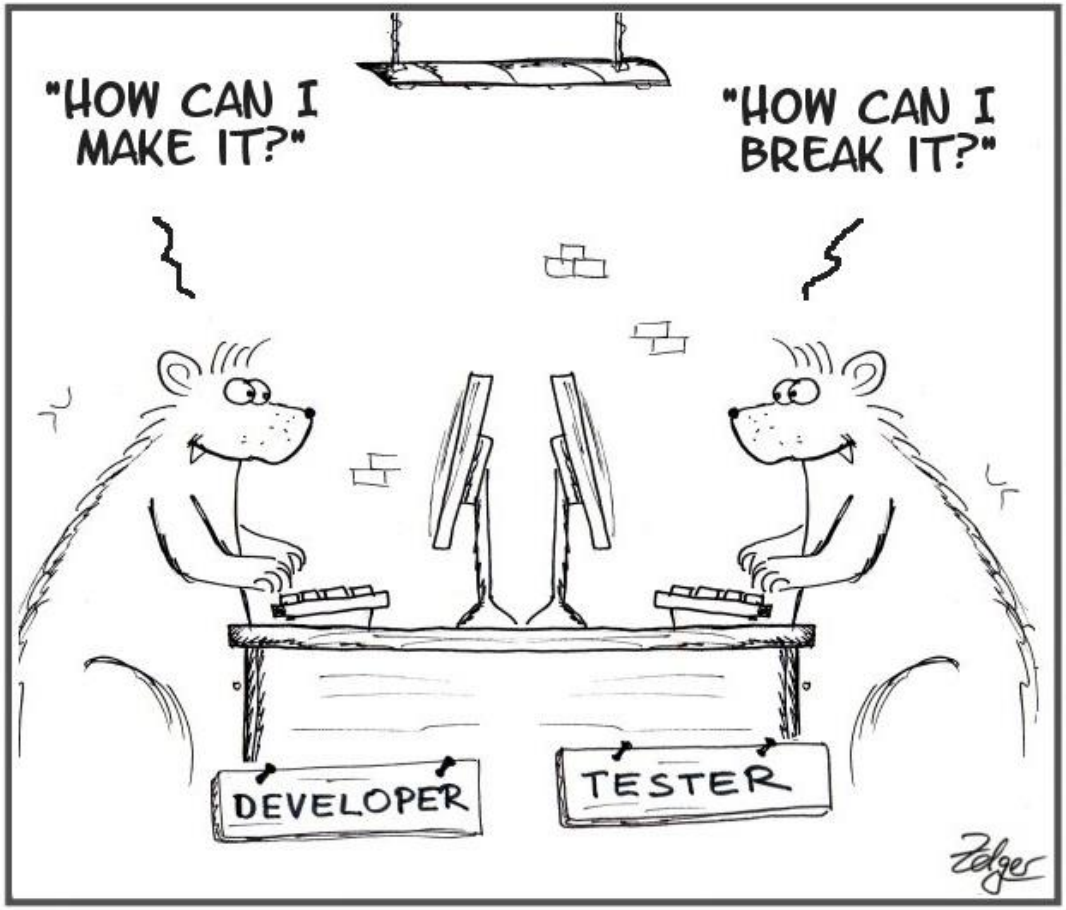
Acteurs spécialisés et professionnalisés

Services **mutualisés** (multi-projets / Domaines)

Services **industriels** (multi-clients / Co-entreprise)

Organisation adaptée au contexte

Différence de point de vue entre testeur vs developpeur – 1/2



They are not so much different, but they have different path for the same goal, to improve quality!!!

Source : http://sachinm203.blogspot.fr/2012/02/relationship-between-developer-and.html

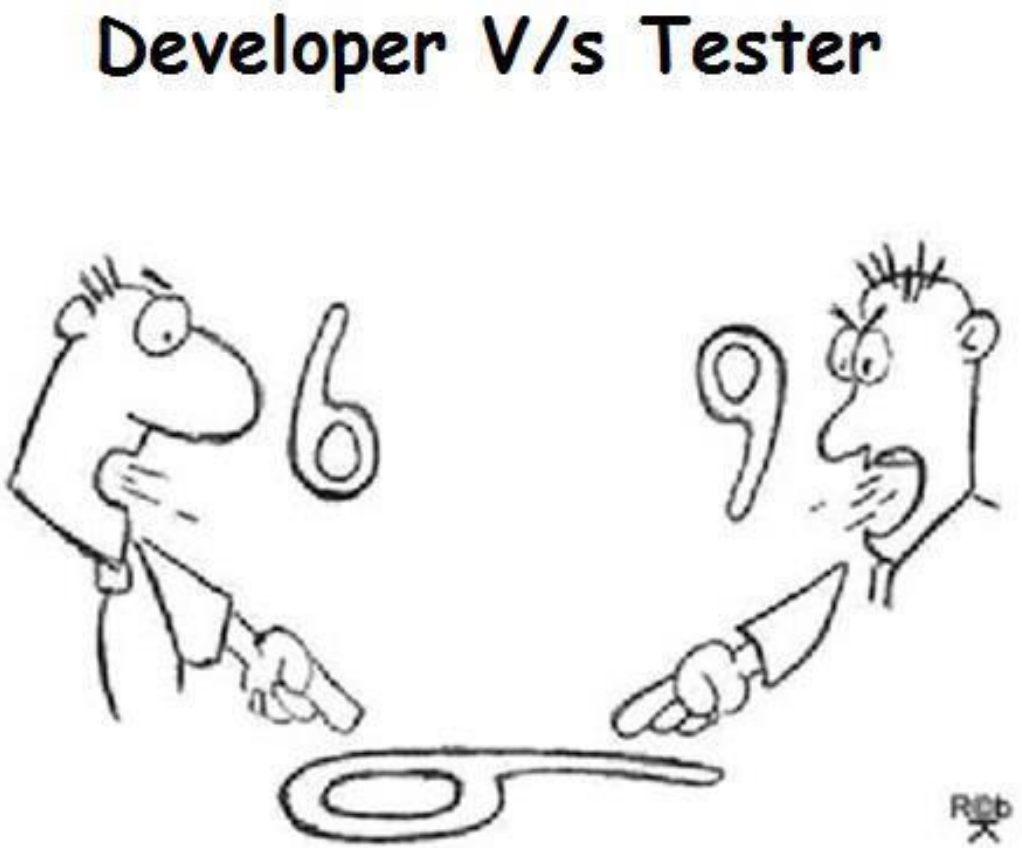
Aspects spécifiques de l'activité de test

- Rechercher des défaillances dans un système requiert :
  - de la curiosité,
  - du pessimisme professionnel,
  - un oeil critique,
  - une attention au détail,
  - une bonne communication avec les développeurs,
  - de l'expérience sur laquelle baser l'estimation d'erreurs.

Communication – Point important du test logiciel

- Commencer par une collaboration plutôt que par des conflits – rappeler à chacun l'objectif commun de systèmes de meilleure qualité
- Communiquer les découvertes sur le produit de façon neutre et factuelle sans critiquer la personne responsable, par exemple, écrire des rapports d'incidents (ou des résultats de revues) objectifs et factuels
- Essayer de comprendre ce que ressent une autre personne et pourquoi elle réagit comme elle le fait
- Confirmer que l'autre personne a compris ce que l'on a dit et vice versa

Différence de point de vue entre testeur vs developpeur – 2/2



Source : http://www.cutorcopy.com/developer-vs-tester/

Une bonne communication est nécessaire

- Les testeurs et le responsable des tests ont besoin de bonnes compétences relationnelles pour communiquer des informations factuelles sur les défauts, les progrès et les risques, de manière constructive
- La description d'un défaut doit aider l'auteur à améliorer ses compétences
- Les défauts trouvés et corrigés pendant les tests permettront de gagner du temps et de l'argent plus tard, et de réduire les risques

Testeur – Méthode de travail

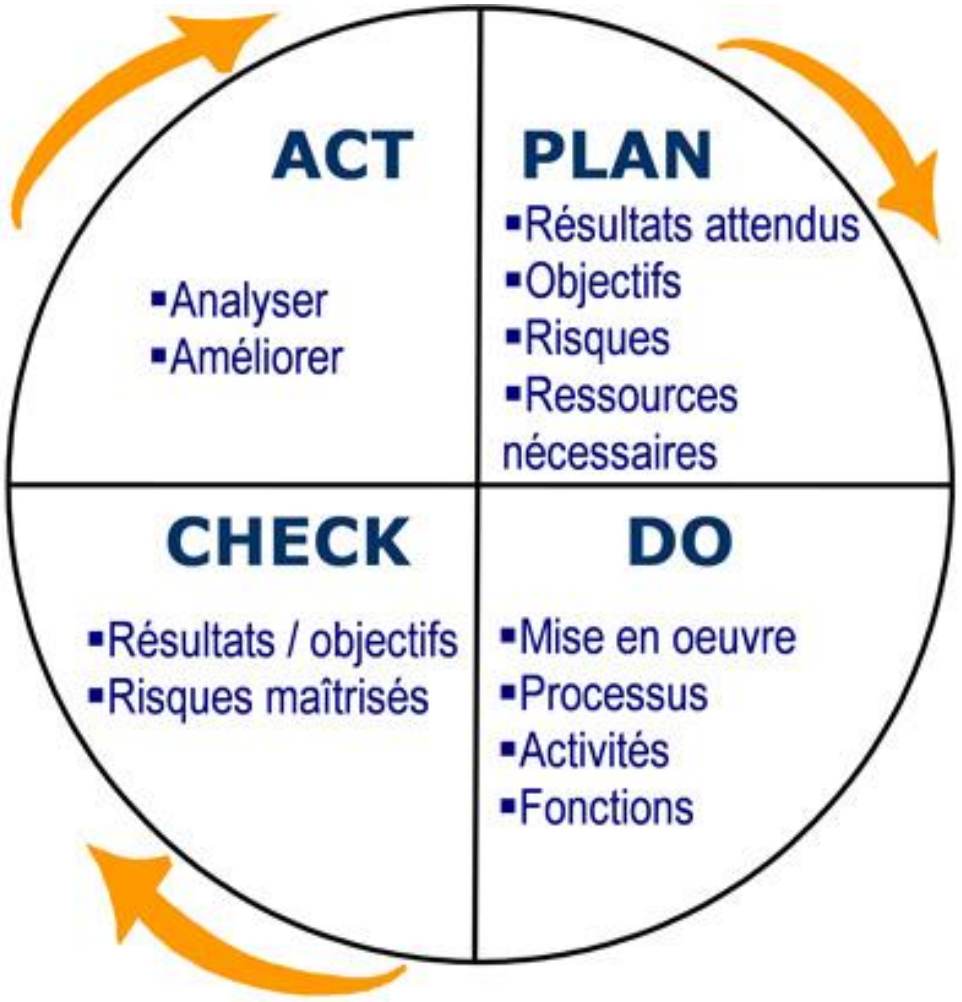
- Les personnes et les projets sont dirigés par des objectifs.
- Les personnes ont tendance à aligner leurs plans en fonction des objectifs mis en place par le management et les autres responsables, par exemple, pour trouver des défauts ou confirmer qu‟un logiciel fonctionne.

➔ il est important de spécifier clairement les objectifs des tests.

Testeur – Méthode de travail

Méthode – PDCA

→ Roue de Deming



Source : http://leblogdumanagementdeprojet.com/2011/07/26/cultivez-vos-talents-projet-et-qualite-de-nombreux-points-communs/

Section 1.5 - Quiz

- Q1 – Quelles phrases correspondent aux qualités les plus importantes d'un testeur ?

I) Avoir une vision critique et interrogative

II) Etre curieux et aller dans le détail

III) Etre un bon orateur

IV) Savoir communiquer sur les anomalies détectées

V) Avoir le sens de l'autorité

a) I, II et IV

b) II et IV

c) I, II et V

d) Tous

1.6 Code éthique

- Les testeurs manipulent des données et informations sensibles (par exemple par rapport à la vie privée)
- Comme tout professionnel, ils doivent respecter un code éthique
- Code éthique de l'ISTQB
  - PUBLIC – les testeurs de logiciels certifiés doivent agir en fonction de l'intérêt public
  - CLIENT ET EMPLOYEUR – les testeurs de logiciels doivent agir pour l'intérêt de leur client et de leur employeur tout en respectant l'intérêt public
  - PRODUIT – les testeurs de logiciels doivent assurer que les fournitures qu'ils produisent (concernant les produits et les systèmes qu‟ils testent) répondent le plus possible aux standards professionnels

1.6 Code éthique

- Code éthique de l'ISTQB (suite)
  - JUGEMENT – les testeurs de logiciels doivent conserver leur intégrité et leur indépendance dans leur jugement professionnel
  - GESTION – les chefs de projet de test de logiciels et les responsables doivent respecter et promouvoir une approche morale dans la gestion de projets de test de logiciels
  - PROFESSION – les testeurs de logiciels doivent mettre en avant l'intégrité et la réputation du métier en cohérence avec l'intérêt public
  - COLLEGUES – les testeurs de logiciels doivent être loyaux, aider leurs collègues, et promouvoir le partenariat avec les développeurs de logiciels
  - PERSONNELLEMENT – les testeurs de logiciels doivent participer en permanence à de la formation pour leur métier et doivent promouvoir une approche morale concernant sa pratique.

Ce qui a été vu dans le chapitre 1

- 1.1 Pourquoi les tests sont-ils nécessaires ? (K2)
  - 1.1.1 Contexte des systèmes logiciels (K1)
  - 1.1.2 Origine des défauts logiciels (K2)
  - 1.1.3 Rôle des tests (K2)
  - 1.1.4 Tests et qualité (K2)
  - 1.1.5 Combien de test est suffisant ? (K2)
- 1.2 Que sont les tests ? (K2)
- 1.3 Principes généraux des tests (K2)
- 1.4 Processus de test fondamental (K1)
  - 1.4.1 Planification et contrôle des tests (K1)
  - 1.4.2 Analyse et conception des tests (K1)
  - 1.4.3 Implémentation et exécution des tests (K1)
  - 1.4.4 Evaluer les critères de sortie et informer (K1)
  - 1.4.5 Clôture des tests (K1)
- 1.5 La psychologie des tests (K2)
- 1.6 Code d'éthique