

# Tests fonctionnels

# Tests

Activité du cycle de vie logiciel

# Définition

## Tests

« activité dans laquelle un système ou un composant est exécuté dans des conditions spécifiées, les résultats sont observés ou enregistrés, et une évaluation est faite pour certains aspects du système ou du composant »

SOURCE: IEEE Std 610.12-1990

- ISO/CEI 25051:2014(fr) Ingénierie du logiciel — Exigences de qualité pour le logiciel et son évaluation (SQuaRE\*\*) — Exigences de qualité pour les progiciels et instructions d'essai

\*\* SQuaRE : Systems and software Quality Requirements and Evaluation

# Définition (suite)

## Tests

« activité dans laquelle un système ou un composant est exécuté

dans des conditions spécifiées,

les résultats sont observés ou enregistrés,

et une évaluation est faite pour certains aspects du système ou du composant »

➔ pouvoir comparer les résultats obtenus à ceux prévus.

# Objectifs des tests

- Vérifier la **conformité** d'un logiciel avec ses spécifications.
- Trouver des anomalies qui n'ont pas été détectées.

# Conformité / non conformité

Exemple :

- Introduction d'un numéro de client belge :
  - Une lettre suivie de 4 chiffres.
  - Obligatoire.
  - Lettre correspond à la province belge (**de A à J**) dans laquelle le client vit.

Tests du numéro de client :

- Entrée de A1476. **Ok**
- Essai d'entrer L1449. → **Message erreur (L). Ok**
- Essai de ne pas entrer de numéro de client. → **Message erreur (vide). Ok**
- Essai d'introduire uniquement le numéro de 4 chiffres 1449 → ***A été accepté. Anomalie, non-conformité.***

# Objectifs des tests

## Rappel

Les tests servent à vérifier que le logiciel réagit de manière **conforme** :

- Les tests de [vérification](#) visent ainsi à vérifier que ce système réagit de la façon **prévue par ses développeurs**.
- Les tests de [validation](#) visent ainsi à vérifier que ce système est conforme **aux besoins du client**.

# Vérification / validation

## Tests de vérification

- Tests unitaires.
- Tests d'intégration.

- Tests fonctionnels.

- Tests systèmes

(performance, volume, stress, fiabilité, sécurité, vitesse de croisière).

## Tests de validation

(d'acceptation, de recette)

- Formalisés par le client.

- Tests fonctionnels.

- Tests systèmes.



# Tests fonctionnels

# Tests fonctionnels

Objectif : vérifier la **conformité** d'un logiciel avec ses spécifications (**cahier des charges & analyse formelle**)

Ils mettent en œuvre des **scénarios** qui permettent de tester les tâches-métiers (fonctions) pour lesquelles le logiciel a été développé.

➔ valider l'exécution de la tâche-métier (fonction).

Ils mettent l'utilisateur **en situation de travail**.

Ils sont souvent développés en **technique de « boîte noire »**.

# Tests fonctionnels

Chaque test fonctionnel valide une partie du logiciel.

- Soit une action isolée.
  - La validation du nom de l'entreprise.
- Soit le scénario d'un cas d'utilisation.
  - Créer une entreprise.
- Soit un enchaînement de cas (scénario d'usage).
  - Indiquer le nom des personnes présentes lors de la confirmation de la participation de l'entreprise à 1 JE.

# Couverture fonctionnelle

- Couverture fonctionnelle : chaque fonctionnalité métier de l'application doit être vérifiée par au moins un **cas\*\* de test**.
  - Algorithmes permettent de calculer cela (partitionnement, classes d'équivalence...).

Un **cas de test** est un « **ensemble d'entrées, de conditions d'exécution et de résultats attendus**, développé pour un objectif particulier, tel que le cheminement particulier d'un programme, ou la vérification de la conformité à une exigence spécifique » [SOURCE: IEEE Std 610.12-1990]

Computer Society - Institute of Electrical and Electronics Engineers (IEEE)

# Couverture fonctionnelle

- **Cas de test** qui vérifie **chaque fonctionnalité** métier de l'application.
  - **Ensemble de toutes les combinaisons** possibles d'emploi des fonctionnalités du logiciel :
    - Ordre de saisie des données.
    - Ordre d'apparition des erreurs.
    - Chemins...
- ➔ **Tester d'abord les chemins pris par un utilisateur** (façon de travailler) et ensuite l'ensemble des cas limites.

# Approches

- **Approche manuelle centrée sur l'IHM.**
  - Approche intuitive, elle permet de reproduire le comportement de l'utilisateur final.
  - Elle permet de tester les cas d'utilisation d'une application (vue de l'utilisateur).
- **Approche automatisée par programmation.**
  - Elle demande des capacités que n'ont souvent pas les clients finaux.
  - Elle sera utilisée par des développeurs.
  - Elle sera souvent utilisée pour relancer les tests fonctionnels à chaque build ou à chaque itération.

# Faire évoluer les tests

Les tests fonctionnels doivent **évoluer** en fonction des modifications apportées au logiciel.

- Il faut donc garder les **données** qui ont permis de reproduire un bug et le scénario de test (ou CR).
- Il faut les intégrer dans le scénario global de tests de l'application.
- Ceci permettra de tester la non-réapparition du bug (**non-régression** des développements futurs).

# Rédiger les tests

- Les tests peuvent être exécutés de nombreuses fois pendant la durée de vie du logiciel.
  - ➔ Il faut prévoir que le document puisse être rempli, commenté, daté et signé à chaque exécution.
  - ➔ Le document est une présentation des tests à exécuter, son remplissage se passe lors de l'exécution des tests.
- Il y aura plusieurs documents de tests.



# Document de tests

- Description de l'environnement.
- Référence au(x) cas d'utilisation.
- Liste des prérequis.
- Description des données en input (déjà présentes avant l'exécution du cas).
- Description du scénario du test.
- Enfin, description des données d'output.

# Scénario de test

Un scénario comprend plusieurs étapes :

- Le cas de test le plus général dans lequel tout se passe bien.
- Toutes les autres combinaisons possibles dans lequel tout se passe bien.
- Les cas d'erreurs.

# Scénario de tests : cas général exemple

## Scénario :

L'utilisateur encode son nom (Legrand), son prénom (Jacques), son pseudo (LJJ), son mot de passe (Fbdfg1;2.3), son email ([j.legrand@student.vinci.be](mailto:j.legrand@student.vinci.be)).

Il choisit le bachelier en BIM.

Il pousse sur le bouton « S'inscrire ».

## Résultat :

L'utilisateur est créé dans la DB.

PAE 2015 : DIMOV Theodor  
DRAGOMIR Philippe  
OSTE Nicolas  
WAGEMANS Jeremy

The screenshot shows a web form titled "Inscription". It contains several input fields: "Nom" (with a blue border), "Prénom", "Pseudo", "Mot de passe", "Email", and a dropdown menu for "Bachelier en imagerie médicale" with a downward arrow. At the bottom is a large green button labeled "S'inscrire".

**ATTENTION, pas d'écran dans les tests fonctionnels!**

Les écrans sont affichés pour donner du contexte dans le cours d'organisation !

# Scénario de tests : cas alternatif exemple

## Scénario :

L'utilisateur encode son email ([j.lou@student.vinci.be](mailto:j.lou@student.vinci.be)) et son mot de passe (Jilou56.7).

Il introduit son nom (Gilles), son prénom (Lou) et son pseudo (Jilou).

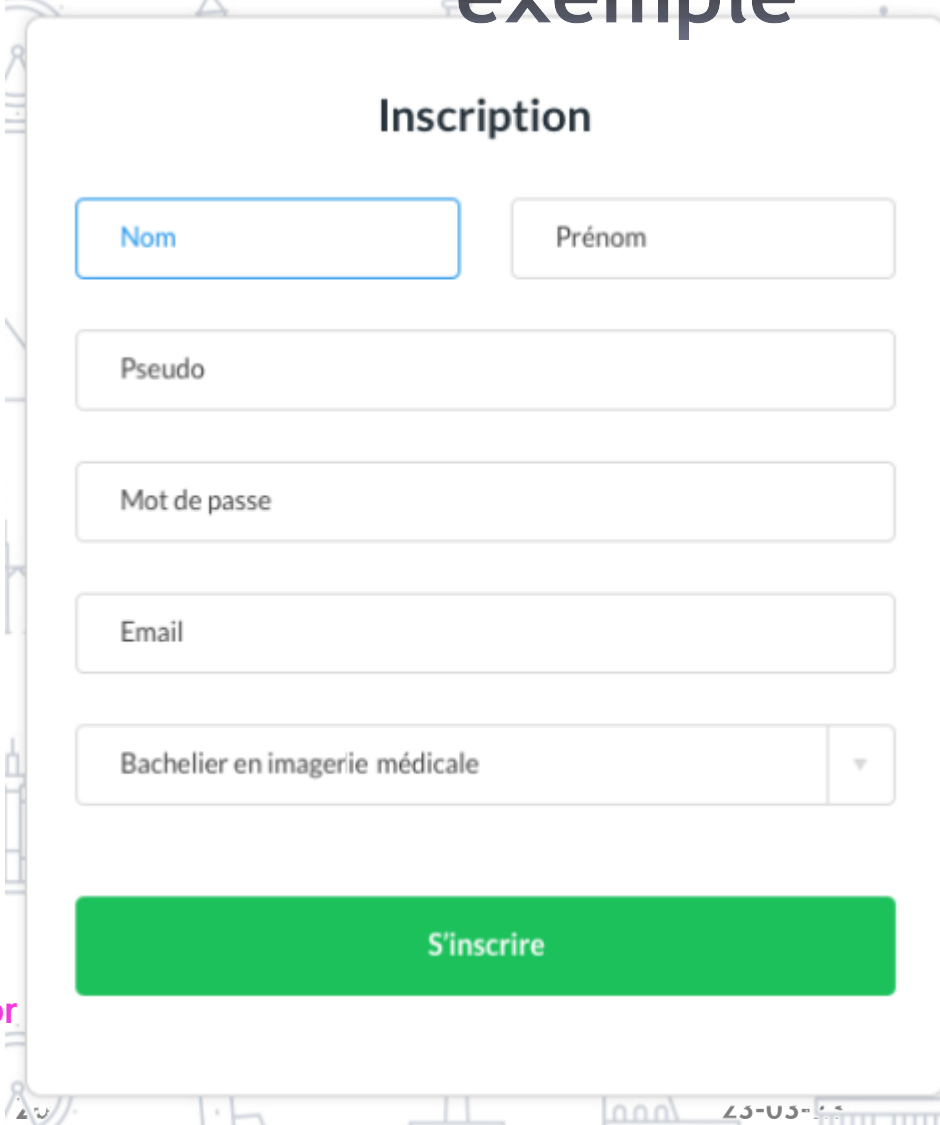
Il choisit le bachelier en BIN.

Il pousse sur le bouton « S'inscrire ».

## Résultat :

L'utilisateur est créé dans la DB.

PAE 2015 : DIMOV Theodor  
DRAGOMIR Philippe  
OSTE Nicolas  
WAGEMANS Jeremy



The screenshot shows a web form titled "Inscription". It contains several input fields: "Nom" (highlighted with a blue border), "Prénom", "Pseudo", "Mot de passe", "Email", and a dropdown menu for "Bachelier en imagerie médicale" with a downward arrow. At the bottom is a large green button labeled "S'inscrire".

# Scénario de tests : cas d'erreurs

- Tests des erreurs isolées
  - Saisie de données erronées.
  - Absence de valeurs.
- Tests des cas extrêmes
  - Fourchette avec un min plus grand que le max.
  - Valeurs autorisées : valeur maximale, valeur max + 1, combinaisons inexistantes.
  - Valeurs nulles.
  - Formats invalides.
  - Date trop ancienne ou dans le futur.
  - Nombre trop grand.
  - Nombre de caractères plus grand que celui prévu en DB ...

# Scénario de tests : cas d'erreurs

## (2)

- Tests des combinaisons d'erreur.
- Tests de données erronées dans un input.
- Tests de recherche :
  - Aucune donnée ne correspond aux critères.
  - Trop de données correspondent aux critères(ne pas afficher plus de x données dans un tableau).
- ...

# Scénario de tests : cas d'erreur

## exemple : erreur isolée

### Scénario :

L'utilisateur encode son email ([p.pou@student.vinci.be](mailto:p.pou@student.vinci.be)) et son mot de passe (PAPou!9;).

Il introduit son nom (Pou) et son pseudo (Papou);

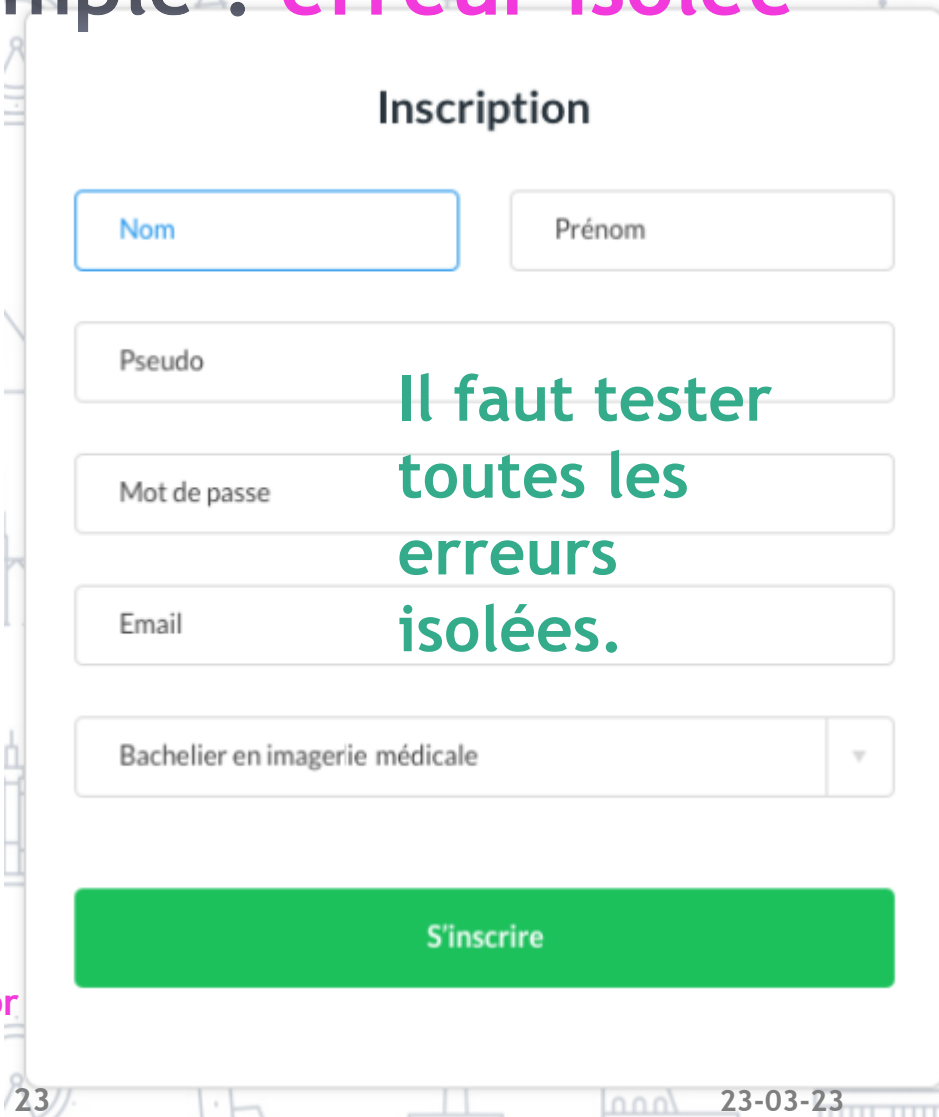
Il choisit le bachelier en BIN;

Il pousse sur le bouton « S'inscrire ».

### Résultat :

Le message d'erreur apparaît :  
« **vous devez introduire votre prénom** ».

PAE 2015 : DIMOV Theodor  
DRAGOMIR Philippe  
OSTE Nicolas  
WAGEMANS Jeremy



The screenshot shows a web form titled "Inscription". It contains several input fields: "Nom" (highlighted with a blue border), "Prénom", "Pseudo", "Mot de passe", "Email", and a dropdown menu for "Bachelier en imagerie médicale". At the bottom is a large green button labeled "S'inscrire". Overlaid on the right side of the form is the text "Il faut tester toutes les erreurs isolées." in green.

# Formalisme

- **Pas de copie d'écran**, car ce serait vite non maintenable.
  - Description de l'environnement.
  - Référence au(x) cas d'utilisation.
  - Liste des prérequis.
  - Description des données en input (déjà présentes avant l'exécution du cas).
  - Décrire le scénario du test.
  - Enfin, description des données d'output.

Exemple : JE

voir [TESTFONCT\\_0001-CreationEntreprise.pdf](#)



# Exercices