Les objets factices

Test et validation logicielle

Par wahid bannour

Octobre 2022

problème

- Tester une classe qui est associée (ou dépend) à d'autres classes
- Pour que le test unitaire respecte les consignes évoquées
  - Rapide
  - Simple
  - Accès uniquement au CPU et la mémoire

=> Isoler la classe

Technique d’isolation

- Faire recours au 5 principe du SOLID
- Dependency Inversion principal
- Utiliser les interfaces pour créer un couplage faible entre la classe et ses dépendances.
- Généralement, la classe à tester fait partie des classes métiers (diagramme des classes participatives) qui découle directement du diagramme de use cases.
- La classe métier dépend de l’abstraction d’une classe (infrastructure ou présentation) mais pas de son concrétisation
- Les classes dépendantes à isoler avec les interfaces sont généralement des classes qui appartiennent l’infrastructure.

Fake Object (objet factice)

- Pour tester une classe dépendante:
  - Modifier la dépendance avec une interface
  - implémenter l'interface dans le projet de production avec la vrai classe (algorithme et dépendance avec les infrastructures)
  - dans le projet de test, implémenter l'interface avec une classe non réelle qui va jouer un rôle de démarrage pour la classe métier à tester
  - cette classe est appelé classe factice (fake).
  - l'implémentation des méthodes de l'interface dans cette classe se réduit à un simple retour de valeur (true, false,….) sans faire de traitement.

Stub

- Au moment de l'appel de la classe factice dans la méthode de test, on va voir quel rôle cet objet va jouer.
- Si la classe factice est ses méthodes ne vont être utilisé que pour le niveau ARRANGE ou ACTION de la méthode de test, alors on appel cette objet un STUB. C'est-à-dire le rôle de l'objet factice se réduit à démarrer la classe à tester.
- Les classes factices peuvent être créé d'une façon automatisée en utilisant un framework de mocking.

mock

- Dans le troisième scénario, le résultat de test n'est pas accessible directement (la fonction à tester ne retourne pas un résultat mais plutôt envoie le résultat vers une autre méthode appartenant à une classe dépendante-souvent une classe d'infrastructure)
- Pour cela, la personnalisation (redéfinition du comportement) de la fonction moquée nécessite une récupération du résultat injecté par la méthode à tester.
- Puisque l'objet factice joue un rôle d'objet possédant le résultat de test, alors on appel cet objet factice un MOCK.