Extrême Programming

Les tests d'acceptances

vérifier que l'application fonctionne correctement dans des ces concrets : une liste d'inputs/ outputs permettant de vérifier si le programme fonctionne correctement. Le client est aussi responsable de vérifier que le programme correspond à son attente. On laisse le client tester chaque fonctionnalité.

Par contre, on aimera faire une automatisation des tests d'acceptances, pour pouvoir refaire les tests, mais sans devoir attendre le client, ou devoir le faire manuellement.

Cela ressemblera beaucoup aux tests unitaires, sauf que dans ce cas-ci, il faudra passer directement par l'interface de l'application pour faire les tests. Alors que dans les tests unitaires, on passait directement par le code pour tester.

La librairie Selenium permet de piloter un navigateur et de faire les tests.

Permet :

- La non-régression du code.
- Prendra plus de temps pour les tests unitaires. on parle d'ordre de grandeur de 1 seconde à plusieurs minutes. Rendra les tests plus compliqué à être lancer. Demande aussi de devoir déployer le code en ligne, d'avoir toute l'infrastructure en plus du code qu'on veut tester ... Assez encombrant
- Tests d'integration.
- Maintenance lourde : prend beaucoup de temps pour être mit en place.

Les tests exploratoires

Une autre personne que le développeur va commencer à essayer de faire tout et n'importe quoi sur l'application ; de se comporter comme le pire des utilisateurs. Il faut essayer de faire les choses de manière "bête", et tester si le programme tient le coup.

Design

- Simplicité : Extrême programming promeut le code le plus simple possible, avec un code qui fait seulement ce qu'il faut, sans autre fioriture.
- Facile de lecture : La documentation va être par référence le code. Il doit donc être simple, bien commenté, ...

Coding standards

Mettre en place dans les équipes d'un standart lors de l'écriture du code. C'est un choix cosmétique, mais c'est toujours très utile pour que les codes soit ressemblant. Cela aidera aussi chacun à aller retoucher les codes des autres. Dans Extrême programming, on encourage les gens à aussi aller retoucher le code des autres.

Outils pour vérifier le code : sonarqube, fxCop.

Refactor

Prendre du code pour le modifier et l'améliorer. Le rendre plus facile à comprendre, plus facile à lire, mais sans changer la manière dont l'application se comporte.

Vélocité

Concept spécifique d'Extrême programming, pour éviter un soucis : qu'on se dise que chaque semaine, on peut faire 40h00 de travail par semaine. C'est un résonnement logique de première abord, mais qui ne marche pas dans un cas réel. On ne travail pas à 100% pendant les 40heures de la semaine. On en prend pas en compte aussi les paperasses administratives, les coups de fils des gens, ...

La solution pour éviter c'est soucis, c'est de mesurer la réalité, mesurer concrêtement le temps qu'on consacre au travail à 100% par semaine. Par contre, l'idée de faire cela, c'est pas de se dire qu'on a pas bien travaillé cette semaine, pour se faire taper les doigts. C'est pour pouvoir dire au manageur ce qu'il se passe, voir qu'on a plus ou moins bosser que la semaine passée, et de permettre de stabiliser le temps de travail. Permet aussi de planifié les prochaines productions en fonction de ce qu'on avait déjà fait jusque-là