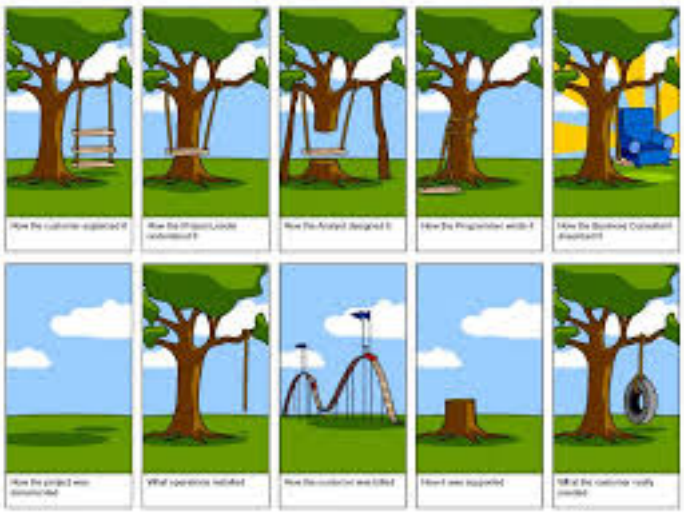
02/04/14 - Rédiger des spécifications agiles

Un constat

La documentation d'un projet a une longue durée de vie. C'est un constat indéniable dans l'industrie logicielle. Comment maintenir cette « doc » à jour ? Comment la rendre compréhensible par un utilisateur comme un développeur ? Autrement dit, comment rendre des « specs » **fiables** et **non ?**



La vision du besoin, de l'analyste au développeur...et au final, celle de l'utilisateur ! Vous connaissez tous les constats suivants :

- Les utilisateurs arrivent à formuler ce qu'ils veulent après avoir vu une première version;
- Spécifier intégralement un logiciel interactif est impossible;
- Les besoins évoluent dans le processus de développement d'une application;
- La communication en interaction directe (face à face) est la technique la plus efficace pour transmettre une information.

Des principes basiques

Face à ces constats, nous pouvons dresser les principes suivants:

Spécifier à plusieurs

Les « métiers » et les « techniques » élaborent les spécifications ensemble :

• Les personnes du métier déterminent les objectifs à atteindre, la raison, la valeur ajoutée;
• Les développeurs/architectes/concepteurs amènent la vision technique de l'infrastructure de l'application;
• Les testeurs identifient les cas d'utilisations (et notamment les cas d'utilisation aux limites).

Identifier et implémenter les tests en amont

• Les tests servent à gérer la bonne couverture du besoin;
• Les tests permettent de vérifier l'absence de bug;
• Les tests peuvent guider le développement et la conception de la solution

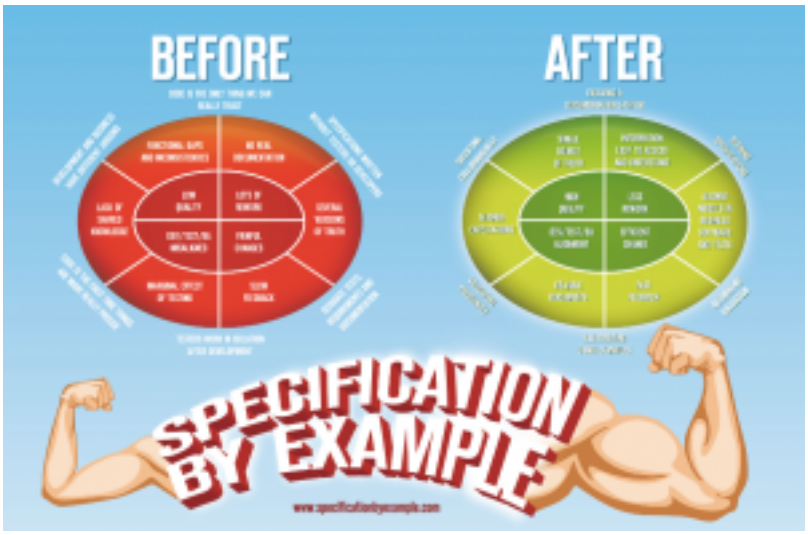
Exclure les données techniques (notamment IHM) dans la spécification écrite

• Les détails d'IHM sont décrits dans des maquettes au fil de fer;
• La doc technique est enregistrée dans un document à part.

Utilisez le vocabulaire de l'utilisateur

• Rédiger les stories ou les cas d'utilisation avec le vocabulaire métier permet de mettre en place un glossaire (donc un modèle de domaine, donc aider aux diagrammes de classes tout en parlant « l'utilisateur »....);
• Utiliser le présent et des verbes d'action (« Effectuer un paiement » pour un titre de UC par exemple ou « Payer par CB » pour une story).

Spécifier par l'exemple



L'acteur métier/ le PO identifie des exemples associés à une fonctionnalité. Puis, c'est en général le testeur et/ou le développeur qui se chargent de les rédiger. Après les développements, l'acteur métier pourra se reposer sur ces spécifications pour s'assurer qu'une nouvelle fonctionnalité est bien « terminée ».

Mise en pratique

Une story se **discute** : Le PO (product Owner) décrit et discute un comportement avec un testeur et un développeur et un expert métier. Ils l'affinent ensemble. Une spécification est élaborée.

Une story se **valide** : Le PO décide que l'ensemble des spécifications couvrent suffisamment le comportement demandé pour l'itération courante : le périmètre est vérouillé. Tout ce qui est en dehors de ce périmètre ou qui ne sera pas développé sera intégré dans les prochaines itérations

Une story se **développe** : les développeurs écrivent tout d'abord le test (vérifiant ainsi qu'il échoue puisque le code qu'il teste n'existe pas) puis implémentent ensuite la fonctionnalité (jusqu'à ce que le test passe)

Un story se **démontre** : une fois passés tous les tests, le PO constate que la fonctionnalité est correctement implémentée. La story passe à l'état « terminé ».

Lecture

Pour terminer ce billet, voici une lecture d'une article de Gojko Adzic que je vous

recommande chaudement : article

Bonne lecture.