Jama software®

LIVRE BLANC

Tester les Exigences

Adapté de Karl E. Wiegers, Testing the Requirements, Microsoft Press 2006.

Quelqu'un m'a un jour demandé à partir de quand on pouvait commencer à tester son logiciel. J'ai répondu : " Vous pouvez commencer les tests dès la rédaction de votre première exigence. "

Il est difficile de visualiser le fonctionnement d'un système simplement en lisant les spécifications des exigences. Les tests basés sur les exigences permettent aux participants à un projet de mieux percevoir le comportement attendu d'un système. Et le simple fait de concevoir des tests révèlera de nombreux problèmes dans les exigences, bien avant que vous n'ayez fait des tests sur un système opérationnel. En commençant à développer des tests une fois les premières exigences établies, vous pouvez rapidement repérer les éventuels problèmes et les corriger à moindre coût.

Exigences et Tests

Il existe une synergie entre les tests et les exigences. Ce sont des représentations différentes d'un même système. En créant de multiples représentations d'un système (exigences écrites, schémas, tests, prototypes etc.), vous favorisez une compréhension du système bien plus riche qu'avec une seule représentation.

Les méthodologies agiles de développement logiciel privilégient la rédaction de tests de validation plutôt que d'exigences fonctionnelles détaillées. Envisager le système au travers des tests est faisable mais ne vous offre qu'une seule représentation des connaissances sur les exigences.

La rédaction de tests en boîte noire (tests fonctionnels) permet de cristalliser votre vision du comportement attendu du système sous certaines conditions. Les exigences vagues et ambiguës vous sauteront aux yeux car vous serez incapable de décrire le comportement attendu du système. Quand les analystes opérationnels, les développeurs et les clients passent en revue les tests ensemble,ils partagent une vision du fonctionnement du produit.

J'ai vécu une situation qui m'a vraiment fait comprendre l'importance de combiner les tests à la spécification des exigences. Un jour, j'ai demandé à Charlie, le gourou des scripts Unix de mon groupe, de créer une interface e-mail simple sous forme d'extension pour un système payant de suivi des anomalies que nous utilisions. J'ai rédigé une douzaine d'exigences fonctionnelles décrivant le comportement de l'interface e-mail. Charlie était enthousiaste. Il avait créé beaucoup de scripts, mais il n'avait encore jamais vu d'exigences écrites. Malheureusement, j'ai attendu deux semaines avant de rédiger les tests pour cette fonction d'e-mail. Évidemment, j'avais fait une erreur dans l'une des exigences. J'ai trouvé l'erreur parce que l'image mentale que j'avais du comportement de la fonction, représentée dans une vingtaine de tests, était en décalage avec une des exigences. Déçu, j'ai corrigé l'exigence erronée avant que Charlie n'ait terminé son

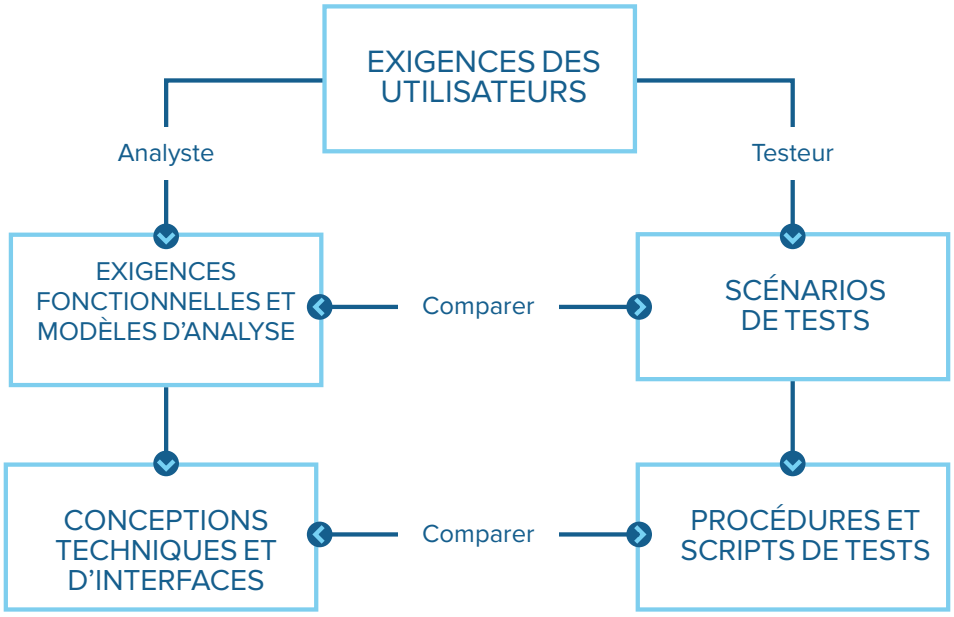

Figure 1. Les produits de développement et de test proviennent d'une même source.

implémentation et lorsqu'il a livré le script, celui-ci ne présentait aucune anomalie. Une petite victoire, mais les petites victoires font les grandes réussites.

Tests Conceptuels

Évidemment, il n'est pas possible de tester le système lors de la phase de création des exigences car aucun logiciel exécutable n'a encore été créé. Néanmoins, vous pouvez commencer à dégager des tests conceptuels grâce aux cas d'utilisation ou aux scénarios utilisateurs.

Vous pouvez ensuite utiliser les tests pour évaluer les exigences fonctionnelles, les modèles d'analyse et les prototypes. Les tests doivent couvrir le déroulement normal du cas d'utilisation, des déroulements alternatifs ainsi que les exceptions identifiées lors de la création et de l'analyse des exigences.

Par exemple, prenons une application appelée Système de Suivi des Produits Chimiques. Un cas d'utilisation appelé " Voir la commande " permet à un utilisateur de récupérer une commande spécifique de produits chimiques dans la base de données et de voir les détails associés. Voici quelques exemples de tests conceptuels :

- L'utilisateur entre le numéro de la commande à afficher, la commande existe, l'utilisateur a passé la commande. Résultat attendu : Afficher les détails de la commande.
- L'utilisateur entre le numéro de la commande à afficher, la commande n'existe pas. Résultat attendu : Afficher le message " Désolé, cette commande est introuvable. "
- L'utilisateur entre le numéro de la commande à afficher, la commande existe, l'utilisateur n'a pas passé la commande. Résultat attendu : Afficher le message " Désolé, cette commande n'est pas la vôtre. Vous ne pouvez pas la consulter. "

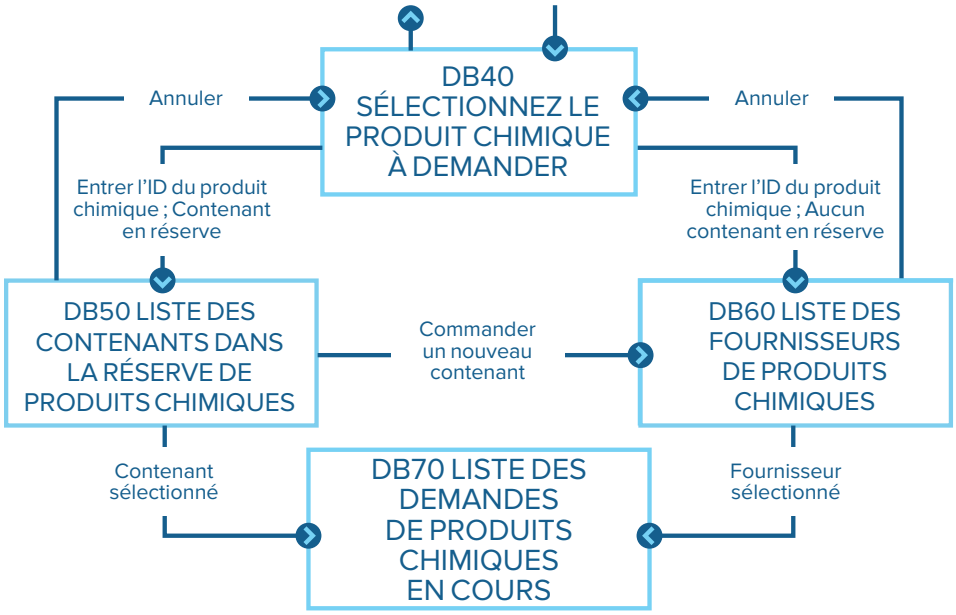

Figure 2. Extrait de la dialog map pour le cas d'utilisation " Demander un produit chimique ".

Dans l'idéal, un analyste commercial rédigera les exigences fonctionnelles et un testeur rédigera les tests en partant du même point : les exigences des utilisateurs, comme montré dans la figure 1. Les ambigüités dans les exigences des utilisateurs et leurs différentes interprétations mèneront à des incohérences entre les représentations des exigences fonctionnelles, des modèles et des tests. Trouver ces incohérences permet de révéler les erreurs. À mesure que les développeurs traduisent les exigences en interfaces utilisateur et en conceptions techniques, les testeurs peuvent transformer les premiers tests conceptuels en procédures de test détaillées pour les tests de système formels.

Exemple

L'idée de tester les exigences peut vous sembler abstraite de prime abord. Voyons comment l'équipe du Système de Suivi des Produits Chimiques a mis en lien la spécification des exigences, la modélisation d'analyse et la génération rapide de scénarios de tests. Voici un cas d'utilisation, quelques exigences fonctionnelles, un extrait d'une dialog map et un test, en relation avec la tâche de demande d'un produit chimique.

Cas d'utilisation

Un des cas d'utilisation fondamentaux du système consiste à " Demander un produit chimique ". Ce cas d'utilisation comprend un chemin qui permet à l'utilisateur de demander un contenant de produit chimique déjà disponible dans la réserve. Cette option permettrait à l'entreprise de réduire les coûts en réutilisant les contenants qu'elle possède déjà au lieu d'en acheter des nouveaux. Voici la description du cas d'utilisation : Le demandeur spécifie le produit chimique qu'il souhaite demander en entrant son nom ou son numéro d'identifiant. Le système peut soit offrir au demandeur un contenant du

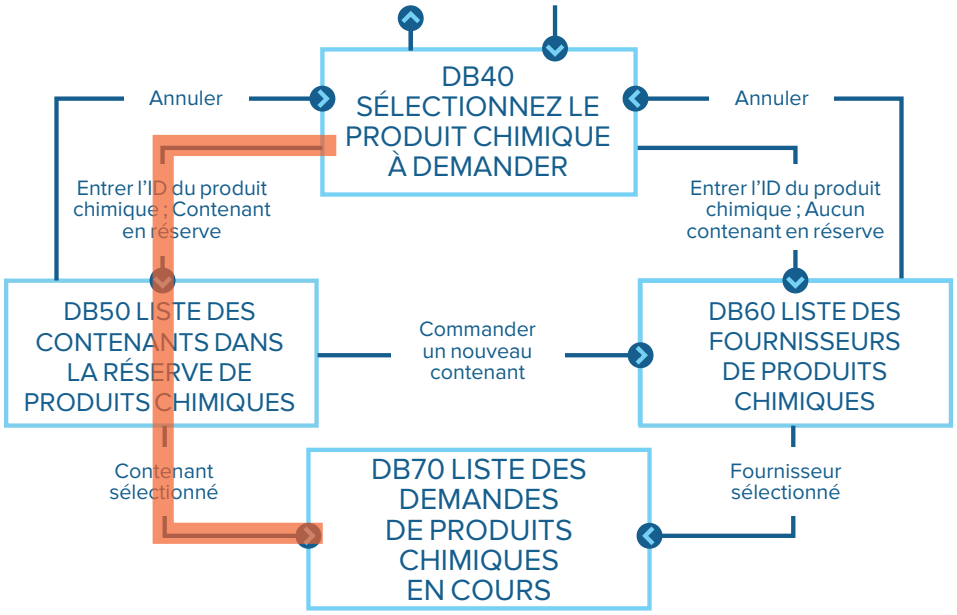

Figure 3. Traçage d'un test sur la dialog map du cas d'utilisation " Demander un produit chimique "

produit chimique présent dans la réserve, qu'il soit nouveau ou usagé, soit permettre au demandeur de créer une demande pour commander un nouveau contenant auprès d'un fournisseur.

Exigence fonctionnelle

Voici quelques fonctionnalités associées à ce cas d'utilisation :

1. Si la réserve contient des contenants du produit chimique demandé, le système affiche une liste des contenants disponibles.
2. L'utilisateur peut soit sélectionner un des contenants affichés, soit demander un nouveau contenant auprès d'un fournisseur.

Dialog map

La figure 2 présente un extrait de la dialog map du cas d'utilisation " Demander un produit chimique " liée à cette fonctionnalité.

Une dialog map est une représentation de haut niveau de l'architecture d'une interface utilisateur, sous forme de diagramme d'états transitions. Les boîtes de cette dialog map représentent les affichages de l'interface utilisateur (des boites de dialogue, en l'occurrence), et les flèches représentent les chemins de navigation possibles pour passer d'un affichage à un autre.

Test

Comme ce cas d'utilisation a plusieurs chemins d'exécution possibles, on peut envisager de nombreux tests afin de traiter le déroulement normal, les déroulements alternatifs et les exceptions. Voici un test basé sur le chemin qui montre à l'utilisateur les contenants disponibles dans la réserve de produits chimiques :

À la boîte de dialogue DB40, entrez un identifiant de produit chimique valide ; il y a deux contenants de ce produit chimique en réserve. La boîte de dialogue DB50 apparaît et affiche les deux contenants. Sélectionnez le deuxième contenant. DB50 se ferme et le deuxième contenant est ajouté à la fin de la liste actuelle des demandes de produits chimiques, dans la boîte de dialogue DB70.

Ramesh, directeur des tests pour le Système de suivi des produits chimiques, a écrit plusieurs tests comme celui-ci en partant de sa propre compréhension de la façon dont l'utilisateur peut interagir avec le système pour demander un produit chimique. Les tests abstraits de ce genre ne sont pas affectés par les détails de l'implémentation. Ils n'indiquent pas s'il faut cliquer sur des boutons précis ou d'autres techniques d'interaction spécifiques. À mesure que les activités de conception ont avancé, Ramesh a transformé ces tests abstraits en procédures de test spécifiques.

Vient ensuite la partie intéressante : le test des exigences. Ramesh a d'abord assigné les tests aux exigences fonctionnelles. Il a vérifié que chaque test pouvait bien être exécuté par l'ensemble des exigences actuelles. Il a également vérifié qu'au moins un test permettait de couvrir chaque exigence fonctionnelle. Cette assignation permet généralement de détecter des omissions. Ensuite, à l'aide d'un surligneur, Ramesh a tracé le chemin d'exécution de chaque test sur la dialog map. La ligne orange sur la figure 3 montre le tracé de l'extrait de test précédent sur la dialog map. En traçant le chemin d'exécution de chaque test sur le modèle, on peut trouver des exigences incorrectes ou manquantes, corriger des erreurs dans la dialog map et affiner les tests. Supposons qu'après avoir " exécuté " tous les tests de cette façon, le chemin de navigation sur la figure 2 intitulé " Commander un nouveau contenant ", qui relie DB50 à DB60, ne soit pas surligné. Deux interprétations sont possibles :

- Le passage de DB50 à DB60 n'est pas un comportement autorisé dans le système. L'analyste opérationnel doit supprimer ce chemin de la dialog map. Si la feuille de spécification des exigences contient une exigence décrivant cette transition, l'analyste opérationnel doit également supprimer cette exigence. .
- La navigation est un comportement légitime dans le système, mais le test prouvant ce comportement manque à l'appel

Quand je trouve ce genre d'incohérence, je ne peux pas définir quelle est la bonne interprétation. Je sais cependant que toutes les représentations des exigences (feuille de spécification, modèles et tests) doivent s'accorder.

Supposons qu'un autre test énonce que l'utilisateur peut faire une action pour passer directement de DB40 à DB70. Cependant, la dialog map ne contient pas ce chemin de navigation, le test ne peut donc pas être " exécuté ". De nouveau, il existe deux interprétations :

- La navigation de DB40 à DB70 n'est pas un comportement autorisé pour le système, le test est donc erroné.
- La navigation de DB40 à DB70 est une fonction légitime, mais la dialog map, et possiblement la feuille de spécification des exigences, ne contiennent pas d'exigence permettant d'exécuter le test.

Dans ces exemples, l'analyste opérationnel et le testeur ont combiné les exigences, les modèles d'analyse et les tests afin de détecter les exigences manquantes, erronées ou inutiles bien avant que le moindre code ne soit créé. À chaque fois que j'utilise cette technique, je trouve des erreurs dans toutes les exigences que je compare.

Comme le consultant Ross Collard l'a très bien remarqué : " Les cas d'utilisation et les tests ont deux synergies : si les cas d'utilisation pour un système sont complets, précis et clairs, le processus de création des tests est simple. Si les cas d'utilisation sont perfectibles, on pourra les améliorer plus facilement en essayant de créer des tests à partir d'eux. " Les tests conceptuels d'exigences de logiciel sont une technique efficace pour gérer les coûts et les délais d'un projet en détectant rapidement les ambiguïtés et les erreurs dans les exigences.

À PROPOS DE KARL WIEGERS

Karl a assuré des formations et des services de conseil partout dans le monde sur de nombreux aspect du développement de logiciel, de la gestion et de l'amélioration des processus. Il est l'auteur de 5 ouvrages techniques, dont Software Requirements, et a écrit plus de 175 articles. Avant de créer Process Impact en 1997, il a travaillé pour Eastman Kodak Company pendant 18 ans. Il y a occupé les postes de chercheur en photographie, développeur d'applications, responsable logiciels, et directeur processus de logiciels et amélioration de la qualité. Karl a dirigé des activités d'amélioration de processus dans des petits groupes de développement d'applications, dans le groupe de développement web de Kodak, et dans une division de 500 développeurs qui créé des logiciels embarqués et hôtes d'imagerie numérique.

À PROPOS DE JAMA SOFTWARE

Jama Software fournit la meilleure plateforme pour la gestion des exigences, des risques et des tests. Les équipes en charge de la production de produits, systèmes et logiciels complexes peuvent s'appuyer sur Jama Connect et sur des services pour raccourcir les temps de cycle (par secteur industriel), améliorer la qualité, réduire les modifications et minimiser les efforts tout en garantissant la conformité du produit. À la pointe des évolutions en matière de développement, Jama compte une clientèle toujours plus nombreuse, avec aujourd'hui plus de 600 organisations, dont SpaceX, Boston Scientific, Lyft, Deloitte, Alight, Samsung et Caterpillar.