HP ALM - Pratiques conseillées

À l'attention des professionnels d'ALM

Tests agiles - Pratiques conseillées

Date de publication du document : Mai 2016

Mentions légales

Garantie

Les seules garanties relatives aux produits et services HP sont décrites dans les déclarations de garantie expresses accompagnant lesdits produits et services. Rien dans le présent document ne doit être considéré comme constituant une garantie supplémentaire. HP ne saurait être tenu pour responsable des erreurs techniques ou éditoriales ni des omissions que pourrait comporter le présent document.

Les informations du présent document peuvent être modifiées sans aucun préavis.

Légende relative aux droits restreints

Logiciel informatique confidentiel. La possession, l'utilisation et la copie sont interdites sans licence valide délivrée par HP. Conformément à FAR 12.211 et 12.212, les logiciels commerciaux, la documentation des logiciels et les données techniques relatives aux articles commerciaux disposent d'une licence accordée au gouvernement des États-Unis conformément aux conditions de licence commerciale standard HP.

Avis de copyright

Marques

Microsoft® et Windows® sont des marques de Microsoft Corporation déposées aux États-Unis.

Oracle® est une marque déposée aux États-Unis par Oracle Corporation, Redwood City, Californie.

Mises à jour de la documentation

La page de titre de ce document contient les informations d'identification suivantes :

- le numéro de version du logiciel, qui indique la version du logiciel ;
  - le numéro avant le point correspond au numéro de la mise à jour majeure ;
  - le premier numéro après le point correspond au numéro de la mise à jour mineure ;
  - le second numéro après le point représente le numéro de la mise à jour mineure-mineure ;
- la date de version du document, qui change à chaque mise à jour du document ;
- la date de sortie du logiciel, qui indique la date de sortie de cette version du logiciel.

Pour vérifier l'existence de nouvelles mises à jour ou vérifier que vous utilisez l'édition la plus récente, rendez-vous à l'adresse URL suivante :

https://softwaresupport.hpe.com/group/softwaresupport/search-result?doctype=manuals?keyword=

Pour accéder à ce site, vous devez créer un compte HP Passport et vous connecter comme tel. Pour obtenir un identifiant HP Passport, accédez à l'adresse :

https://softwaresupport.hpe.com/

Vous pouvez également cliquer sur le lien New users - please register (Nouveaux utilisateurs - Inscrivez-vous) de la page de connexion à HP.

Vous recevrez également des versions récentes ou mises à jour si vous vous abonnez au service d'assistance du produit approprié. Pour plus d'informations, contactez votre représentant HP.

Assistance

Vous pouvez visiter le site Web en ligne de l'assistance HP Software à l'adresse :

https://softwaresupport.hpe.com/

Ce site Web fournit des informations de contact et des détails sur les produits, les services et l'assistance fournis par HP Software.

L'assistance logicielle en ligne de HP Software propose des fonctions de résolution autonome. Elle offre un moyen rapide et efficace d'accéder aux outils interactifs de support technique nécessaires à la gestion de votre entreprise. En tant que client bénéficiant de l'assistance HP, vous pouvez effectuer les opérations suivantes :

- effectuer des recherches dans les documents qui vous intéressent ;
- soumettre des incidents et suivre leur résolution, ainsi que des demandes d'amélioration ;
- télécharger des correctifs logiciels ;
- gérer vos contrats d'assistance ;
- rechercher des contacts d'assistance HP ;
- consulter les informations sur les services disponibles ;
- entamer des discussions avec d'autres utilisateurs du logiciel ;
- rechercher des formations et vous y inscrire.

La plupart des domaines de l'assistance nécessitent que vous soyez enregistré en tant qu'utilisateur HP Passport. Ils peuvent également nécessiter un contrat d'assistance actif. Pour de plus amples informations sur les niveaux d'accès à l'assistance, rendez-vous à l'adresse URL suivante :

https://softwaresupport.hpe.com/web/softwaresupport/access-levels

Pour obtenir un identifiant HP Passport, accédez à l'adresse URL suivante :

https://softwaresupport.hpe.com/

Table des matières

Bienvenue dans ce manuel

Bienvenue dans ce manuel des pratiques conseillées pour les tests agiles HP.

Ce manuel fournit des concepts, des consignes et des exemples pratiques pour la mise en œuvre des principes de test agile la mieux adaptée dans différentes organisations.

À propos des tests agiles

Avez-vous déjà eu un logiciel qui ne fonctionnait pas ? Et vous êtes-vous alors demandé : « Comment ce code a-t-il pu être publié ? ». Ou peut-être avez-vous en réalité travaillé sur un projet au sujet duquel vous avez dit à votre hiérarchie que le logiciel ne devait pas être publié, mais qui, à votre plus grande stupéfaction, l'a malgré tout été.

Il existe, bien sûr, de multiples raisons et situations qui peuvent conduire à la publication d'un mauvais code. Une cause majeure, toutefois, peut être la culture de l'entreprise.

Si les développeurs sont très estimés et les testeurs sont considérés comme étant moins importants, il est alors facile de comprendre pourquoi la voix du testeur peut être passée sous silence et son point de vue ignoré. Si les testeurs ne sont pas suffisamment techniques ou sont incapables d'écrire un rapport cohérent et convaincant, leur message peut être rejeté lors de la discussion générale ou considéré comme étant de peu de valeur en raison de la faiblesse de leur présentation. Si la décision de la publication a lieu à huis clos sans qu'il ne soit possible de présenter le rapport de test et d'en discuter, les informations concernant l'état réel du logiciel peuvent ne jamais être mises à la disposition de ceux prenant les décisions.

Il peut exister un certain nombre de raisons pour lesquelles, même lorsque le rapport de test est compris dans son intégralité, la publication est quand même validée et le code erroné, publié. Toutefois, ces raisons restent rares, le motif principal étant que le service de test est exclu de ce processus.

Les méthodes agiles constituent la dernière tentative des équipes de développement pour combler l'écart entre la croissance des attentes professionnelles et l'insuffisance de l'offre informatique. Alors qu'il est estimé que plus de 70 % des projets suivent encore le modèle en cascade (classique), la méthode agile s'impose rapidement comme la nouvelle norme aussi bien dans les petites organisations que dans les grandes. Avec la promesse de privilégier les résultats plutôt que les processus, les méthodes agiles sont au centre des efforts menés pour livrer de meilleurs logiciels.

HP a adopté des pratiques agiles pour ses processus de développement internes et estime que le partage de son expérience peut aider les clients à adapter leur méthodologie et leurs processus afin de proposer des logiciels de qualité.

Ces consignes devraient contribuer à réduire le temps de création initial et à tirer au maximum profit de l'utilisation d'HP ALM.

Public visé

Ce manuel est destiné aux personnes suivantes :

- Responsables de l'Assurance Qualité
- Ingénieurs spécialisés dans l'automatisation des tests
- Responsables du développement

Conditions préalables

Pour utiliser ce manuel, vous devez être familiarisé avec les principales phases du cycle de vie du développement des logiciels. Les processus métier d'organisations informatiques réelles doivent également vous être familiers.

Structure

Ce manuel est organisé comme suit :

- Présentation des tests agiles
- Préparation en vue des tests agiles
- Assurer la réussite des tests agiles
- Conclusions

Commentaires

Si vous avez des questions, des commentaires ou des informations utiles sur les pratiques conseillées que vous souhaitez partager, envoyez un message à l'adresse e-mail suivante :

docteam@hpe.com

1 Présentation des tests agiles

Importance de la méthode agile

La demande en produits logiciels de haute qualité a poussé les professionnels des logiciels à identifier et à chiffrer les facteurs de qualité, tels que l'utilisation, la testabilité, la facilité de maintenance et la fiabilité, ainsi qu'à identifier les pratiques d'ingénierie prenant en charge la production de produits de qualité possédant ces attributs favorables. Le processus de développement logiciel, comme la plupart des artefacts d'ingénierie, doit être orchestré. En d'autres termes, il doit être conçu, mis en œuvre, évalué et mis à jour. À l'instar d'autres disciplines d'ingénierie, un processus de développement logiciel doit évoluer de manière cohérente et prévisible, et les meilleures pratiques techniques et de gestion doivent être systématiquement intégrées.

Les entreprises contraintes à produire des applications de qualité supérieure face aux exigences imposées par la concurrence se tournent vers les pratiques de développement agile. En réalité, les méthodes agiles et d'autres méthodes itératives deviennent la norme de facto du développement d'applications. Idéalement, la méthode agile consiste à fournir la valeur commerciale maximale plus rapidement en se concentrant sur les personnes et sur l'amélioration continue. La méthodologie agile est généralement considérée comme principalement applicable aux équipes de développement. Cependant, toute l'organisation doit s'adapter.

Le développement agile impose aux organisations deux défis importants : apporter, beaucoup plus tôt dans le processus de développement, de la qualité et de la stabilité aux applications afin de s'adapter à l'activité, et devenir suffisamment flexibles pour suivre la nature itérative de la méthodologie agile.

HP offre une approche et des solutions qui aident les équipes à adopter la méthodologie agile et à participer à sa réussite.

La méthodologie agile

La méthode agile, qu'est-ce que c'est ?

Le terme « agile » souffre certainement du syndrome du mot à la mode. « Agile » sonne si bien. Il fait penser à la fois à la vivacité, à la souplesse et à la puissance. Qui ne souhaiterait pas être agile ? Personne ne dit : « Merci, mais je préférerais être inflexible et peu réactif ». En outre, comme il s'agit d'un mot standard en français, tout le monde croit savoir ce qu'il signifie. Le problème avec les mots à la mode, c'est qu'on les utilise sans comprendre ce qu'ils signifient réellement.

Une recherche Google sur « développement agile » renvoie des millions de réponses et il n'est pas étonnant que de nombreuses personnes l'interprètent différemment. La partie principale du Manifeste Agile est bien connue :

> *Nous découvrons de meilleures façons de développer des logiciels en le faisant et en aidant les autres à le faire. Grâce à ce travail, nous estimons à présent :*
>
> > *Les individus et les interactions plutôt que les processus et outils*
> >
> > *L'utilisation de logiciels plutôt qu'une documentation complète*
> >
> > *La collaboration des clients plutôt que la négociation de contrats*
> >
> > *La réaction face aux changements plutôt que le suivi d'un plan*
>
> *Autrement dit, même si les éléments de droite ont de la valeur, nous accordons davantage d'importance aux éléments de gauche.*

Toutefois, la signification réelle du mot « agile » est parfois perdue dans la mer des différents produits dérivés de la méthode agile, notamment Extreme Programming, SCRUM, DSDM, Adaptive Software Development, Crystal, Feature-Driven Development et Pragmatic Programming. Même si elles sont subtilement différentes, ces méthodologies possèdent les mêmes valeurs fondamentales décrites ci-dessus. La méthode agile se matérialise par un effort implacable à fournir un flux continu de valeur commerciale, même en cas de changement constant.

Alors, qu'est-ce que la méthode agile et pourquoi les anciennes méthodes ne fonctionnent-elles pas toujours efficacement ?

Les heures des méthodes classiques sont comptées

Dans notre monde en constante évolution, les approches classiques ne fonctionnent pas toujours. Le développement en cascade se caractérise par des phases bien définies, chacune présentant un processus d'examen et d'autorisation complet qui doit être terminé avant de pouvoir poursuivre. En apparence, ce processus n'est pas intrinsèquement mauvais. En réalité, il est assez logique : Tout d'abord, déterminez tout ce dont vous avez besoin. Ensuite, passez à la conception. Ensuite, passez au codage. Ensuite, passez aux tests. Répétez ce processus.

Les problèmes surviennent lorsque le projet nécessite des réglages et des modifications qui n'ont pas été prévus dans les phases initiales. La réalité ruine le projet. Pour les projets qui peuvent être déterminés avec précision à l'avance, la cascade peut être le choix approprié. Malheureusement, l'expérience a prouvé qu'un ensemble d'exigences équilibré et immuable est rare.

La faute à qui ? La réponse varie selon la personne à qui vous posez la question. La hiérarchie condamne souvent l'équipe de produits pour son manque de rigueur. L'équipe de produits reproche à l'équipe de développement le manque de fiabilité du codage. L'équipe de tests reproche à l'administration de ne pas appliquer des délais et des limites plus stricts. « Vous avez approuvé les spécifications il y a deux mois ; nous ne pouvons pas ajouter cette fonction maintenant ! ».

Il n'est pas surprenant que ces projets soient à l'origine d'une telle frustration : à la fin, les équipes publient souvent des logiciels qui ne disposent pas des fonctions les plus importantes. Dans le même temps, il existe une multitude de documents bien montés (documents d'exigences du marché, spécification des exigences logicielles, schémas d'architecture, API et plans de test) qui n'apportent aucune valeur réelle aux clients.

La méthode agile est née

En raison de ces échecs, les équipes de développement commencent à procéder à des mises à jour fréquentes. Dans un premier temps, ces méthodes itératives ont été utilisées dans de petits cycles, mais un planning fixé à l'avance restait de vigueur (autrement dit, un budget fixe, un plan strict et des ensembles de fonctionnalités prédéterminés). Par conséquent, au lieu d'une mise à jour majeure, plusieurs mises à jour mineures étaient planifiées.

Ce concept a évolué : on parle à présent de mises à jour itératives sans plan fixe. En réalité, chaque mise à jour apportait quelque chose, ce qui permettait de prendre de meilleures décisions quant aux fonctions qui doivent être incluses par la suite. Le développement fonctionnait en petits cycles, la planification et la conception étant suffisamment avancées pour maintenir le flux. Grâce à cette plus grande flexibilité, la charge d'une documentation formelle excessive est grandement allégée. À présent, la phase de test est incorporée dans chaque étape du processus, pas seulement à la fin de chaque mise à jour.

Coexistence de la méthode en cascade et de la méthode agile

Tandis que la méthode agile gagne en popularité, l'approche classique conserve encore une place. En fin de compte, le succès de la méthodologie de développement est mesuré par la qualité des logiciels produits. Toutefois, même si votre organisation utilise des méthodologies en cascade classiques, vous pouvez toujours bénéficier de concepts agiles pour améliorer la qualité.

Si votre entreprise développe des logiciels de qualité, si le développement est durable et que l'activité est servie par ces logiciels de manière appropriée, il est vraiment inutile d'insister sur ce point. La méthode agile n'est pas la panacée pour les problèmes liés au développement, au processus et à la gestion. Toutefois, si vous êtes prêt à effectuer certaines des modifications nécessaires, elle peut s'avérer extrêmement bénéfique sur le long terme.

Il y a récemment eu une prolifération d'approches *hybrides* dans le cadre desquelles certaines équipes travaillant sur un nouveau projet ou passant à une nouvelle technologie adoptent entièrement la méthodologie agile, tandis que d'autres prennent certains éléments adaptés à leurs objectifs et à la culture de leur entreprise, et que d'autres équipes, en particulier dans les grandes entreprises, suivent toujours la méthodologie classique simplement parce qu'elle fonctionne.

Dans les grandes organisations, bien souvent, différentes équipes travaillant sur différents composants d'une initiative plus importante utilisent différentes méthodes pour produire les logiciels. Cette particularité rend les rapports difficiles sur le statut des initiatives, les objectifs et les KPI de chaque équipe variant dans l'ensemble des différents processus. Gérer différentes nuances des méthodes agiles et en cascade au sein d'un même projet ajoute de la complexité à une opération déjà complexe. HP ALM permet de surmonter cette complexité à l'aide de projets sur modèle normalisés. Les méthodes agiles et en cascade sont toutes les deux proposées.

À l'instar de nombreux éléments du secteur de la technologie, il ne se dégagera probablement aucun vainqueur de cette lutte de méthodologies et les approches hybrides prévaudront finalement une fois que le buzz s'atténuera.

2 Préparation en vue des tests agiles

De la méthode en cascade à la méthode agile

L'approche en cascade du développement logiciel est un processus séquentiel au cours duquel le développement est considéré comme avançant de manière régulière et descendante (comme une cascade) dans les phases d'analyse des exigences, de conception, de mise en œuvre, de test, d'intégration et de maintenance.

Elle considère les tests comme une autre phase de la vie des produits. Voici certaines de ses qualités :

- D'abord, vous codez, ensuite vous testez.
- Les équipes de développement et de test sont distinctes, chacune travaillant dans son propre domaine.
- Chaque partie du processus est lourd et nécessite une multitude de documents :
  - Exigences
  - Documents de conception
  - Plans de test
  - Matrices de traçabilité
  - Tableaux de suivi des modifications
- Les tests deviennent un goulet d'étranglement en toute fin, ce qui rend le respect du planning pratiquement impossible.
- Les outils d'automatisation entrent en action après le codage, ce qui exige des efforts coûteux des équipes de test pour l'enregistrement/la lecture.

L'approche en cascade souffre d'un manque de participation, de l'absence de communication entre les développeurs et les testeurs, et d'un impact minimal sur les personnes chargées de prendre des décisions sur les produits.

L'introduction de la méthodologie agile promet d'alléger certains des problèmes importants de la méthodologie en cascade :

- Le codage et les tests deviennent inséparables.
- Les commentaires et la collaboration sont les éléments clés de la solution.

- Une phase de tests exploratoires révèle des conséquences inattendues et des attentes implicites.
- Dans la plupart des cas, le testeur doit faire preuve d'un sens de la réflexion critique, pas seulement de respect des documents relatifs aux exigences.
- La définition des tâches étant floue, diverses idées commencent à apparaître.
- Tous les types de personnes commencent à travailler en équipe.

Recettes d'une approche agile réussie

HP croit en une vue holistique de la gestion du cycle de vie des applications. Lorsque la méthode agile est ajoutée au mélange, elle permet d'intégrer l'AQ et les commentaires de l'utilisateur dans les processus de développement. L'équipe de développement et l'équipe d'AQ font alors partie de la même équipe, contrairement à la fragile trêve qui existe entre ces deux rôles aujourd'hui. Cette méthode permet également de s'adapter davantage aux besoins du client et aux changements du marché, de sorte que les produits répondent aux normes d'excellence auxquelles s'attendent les clients.

Voyons quelles sont les recettes qu'HP utilise pour convertir un processus de test en une méthode agile. Ces recommandations devraient contribuer à la constitution d'équipes agiles gagnantes qui fournissent de meilleurs logiciels.

Évaluer vos processus

De nombreuses personnes pensent que l'objectif des tests doit tout simplement être de repérer *tous* les problèmes. En réalité, cette tâche n'est pas entièrement réaliste ; elle est en fait impossible. Malgré leur fonction de responsables de l'« assurance qualité », les testeurs ne peuvent jamais vraiment *assurer* la qualité et on ne peut pas attendre d'eux qu'ils trouvent *chacun* des problèmes. L'objectif réel des tests doit être d'*améliorer* les logiciels. Il s'agit d'une vision des tests largement acceptée dans le monde de la méthode agile.

Malgré cela, la hiérarchie a encore tendance à penser que les tests peuvent mettre au jour l'ensemble des problèmes ou déterminer si les logiciels fonctionneront comme prévu. Elle confond, à tort, les tests avec la validation. Pour comprendre le rôle véritable des tests, il faut comprendre qu'il s'agit d'une méthode scientifique, d'une recherche continue d'informations. Au lieu de penser aux tests en termes de réussite ou d'échec, essayez de déterminer s'ils apportent ou non des informations utiles. Dans un environnement agile, les testeurs doivent être sollicités pour fournir ces informations utiles tout au long du cycle de développement, plutôt que d'indiquer simplement s'il s'agit d'une réussite ou d'un échec.

Pour que la méthode agile soit un succès, les tests doivent devenir le pivot du processus de développement. Elle peut avoir des débuts chaotiques, mais lorsqu'elle est exécutée correctement, la méthode agile n'est pas du tout chaotique. Il s'agit d'une méthodologie par laquelle les programmeurs créent des tests unitaires avant l'écriture du code, instillant ainsi la qualité comme principal facteur de réussite du produit logiciel. Elle suit un ordre différent de la méthodologie classique, un ordre qui porte sur le code plutôt que sur le papier et sur la valeur plutôt que sur la responsabilité.

Raconter une histoire

Oubliez les mots *cas d'utilisation* pour opter pour le terme *narration*. En théorie, c'est la même chose, mais, dans la pratique, la différence est bien là. Pour illustrer cette différence, essayez de faire une recherche sur chacun de ces termes dans Google Images : « Cas d'utilisation » donne des images de diagrammes et de schémas bruts et intimidants, alors que « récit » renvoie des images agréables et colorées.

Les testeurs auront une réaction subconsciente similaire : Demandez à un testeur de créer un cas d'utilisation et il pensera en termes bruts dénués d'originalité. Mais demandez-lui un récit et ses idées fuseront. De cette façon, vous l'encouragerez à être plus créatif et à ne pas se contenter d'un simple examen à l'écran. Vous donnez plus de pouvoir au testeur sans réellement utiliser le terme cliché de valorisation. Ce changement de terminologie lui permet de se mettre à la place de l'utilisateur du logiciel.

Les testeurs doivent être impliqués tout au long de la phase de conception. Ils doivent participer au développement des spécifications et à la collecte des exigences.

Dans un environnement agile, dans lequel les tests sont exécutés en continu, l'actualisation est particulièrement importante.

Définir la stratégie de test

Veillez à définir tôt votre stratégie de test. Commencez par comprendre que la spécification des exigences logicielles ressemble assez bien à un plan de test. Par conséquent, il n'y a aucune raison de conserver les deux types de documents dans le monde à évolution rapide de la méthode agile. Les testeurs savent comment trouver les erreurs. Ils savent également que les erreurs n'apparaissent pas seulement à la fin, mais tout au long du processus. Trouver des erreurs dans la logique ou dans l'utilisation est un must pour les chefs de produit au cours des phases de développement des spécifications ; or, leurs compétences sont bien souvent orientées vers d'autres domaines. Observer la spécification des exigences logicielles/le plan de test unifié avec les yeux d'un testeur apporte, par conséquent, beaucoup de valeur lors des premières phases.

Les tests ne constituent pas une phase dans les équipes agiles, ils représentent un mode de vie. Les équipes agiles testent en continu. C'est la seule façon de s'assurer que les fonctions mises en œuvre au cours d'une itération ou d'un sprint donné sont réellement appliquées. Un avantage à intégrer l'AQ tôt : les tests garantissent que les exigences de l'application pourront être testées. Il arrive très souvent dans le cadre de la méthode agile que les testeurs puissent commencer à créer des cas d'utilisation en test au cours de la phase initiale de conception.

Que tester ?

La matrice multidimensionnelle de ce qui doit être testé doit être définie tôt. L'importance de cette matrice a augmenté de façon exponentielle au cours des dernières années, la matrice se complexifiant. La définition des exigences de couverture est une phase essentielle de la procédure de spécifications qui peut donc être traitée précocement. La taille de la matrice de la couverture a un impact significatif sur les besoins en personnel et les coûts de l'assurance qualité. Pour cette raison, la définir clairement et tôt permet aux responsables d'allouer des ressources et de déterminer ce qui peut être fait en interne et ce qui doit être fait à l'étranger ou en externe pour une meilleure couverture.

Les mesures de test doivent être adaptées à l'activité

Les responsables des tests se concentrent souvent sur des informations de faible utilité. Ne l'oubliez pas lors de la collecte des données. Ils sont habitués à la compilation de rapports détaillés indiquant le nombre de cas de test rédigés (soit le nombre de cas exécutés, le taux d'échec, les niveaux de gravité, etc.), des informations certainement utiles pour les responsables des tests, mais pour quels autres membres de votre organisation ? Seraient-elles, par exemple, utiles dans la pratique pour l'équipe de développement de l'activité ?

Un type de données aurait son importance. Il s'agit des statistiques d'impact direct sur l'activité : les ventes n'ayant pas été conclues en raison de bugs, le taux de renouvellement de la clientèle et le coût relatif des défauts par type de détection. Ce type d'informations vaut de l'or aux yeux des décideurs. Si vous pouvez collecter ces données, faites-le immédiatement. Il vous suffit d'utiliser les données correctement. La collecte des mesures devient parfois une tâche en soi, tandis que le message ne parvient pas aux responsables.

Collaborer

Une réelle collaboration doit prendre place entre les équipes de développement et d'AQ. Sans la collaboration, la méthode agile devient un ensemble de procédures forcées et ne permet pas de créer des logiciels de qualité. Tous les membres de l'équipe doivent être exposés aux mêmes données en temps réel afin qu'ils puissent prendre le pouls du sprint. Cela signifie que tous les membres doivent avoir accès à une version commune de la vérité. Cela inclut les commanditaires en dehors du processus du cycle de vie du développement des logiciels sous la forme d'un rapport qui traite des besoins de l'activité et qui concerne également d'autres équipes de développement qui utilisent peut-être la méthode agile ou au moins une variante de cette méthode.

Alerter uniquement lorsque cela est nécessaire

Si toutes les anomalies détectées par le testeur reçoivent la priorité la plus élevée, les développeurs ignoreront forcément certaines de celles qui sont vraiment importantes.

D'un autre côté, si vous dites sans cesse « Ignorez ce bug ; il s'agit d'un problème connu » ou « Ignorez ce test, il n'est pas encore pertinent », votre équipe de test prendra l'habitude d'ignorer les bugs et ne tiendra pas compte de ceux dont vous souhaitez la prise en charge.

Soyez prudent lors de l'affectation de la gravité et de la priorité du bug afin que les développeurs comprennent l'importance du problème.

La clé : le temps

Les courts cycles de publication du développement agile peuvent souvent imposer des contraintes temporelles difficiles à gérer pour les équipes d'assurance qualité. L'une des manières de contourner ce problème est d'être en contact permanent avec les développeurs pendant l'itération. Plus tôt un bug est découvert ou les défauts des fonctions sont mis au jour, plus simple en sera la correction. L'implication de l'équipe d'assurance qualité permet d'effectuer les tests tandis que les ingénieurs se chargent du développement. Il est inutile d'attendre la fin d'une itération pour générer une liste d'anomalies. Au fur et à mesure que les anomalies sont détectées, ces informations sont transmises à l'équipe de développement, qui y répond dans le cadre du processus de développement. Le coût de résolution précoce des anomalies au cours du développement est bien plus faible que celui d'une résolution ultérieure. Le risque qu'une anomalie entraîne des problèmes avec d'autres parties de code associées est également réduit. Globalement, l'équipe aura davantage confiance dans la qualité du résultat du développement, sachant que les anomalies sont continuellement identifiées et traitées.

Des tests fonctionnels, mais pas seulement

Les tests faisant peu à peu partie du développement, et les activités de test étant de plus en plus condensées, la tendance est à oublier d'autres aspects de l'assurance qualité, à savoir la validation des performances des applications et l'évaluation de la sécurité.

À quel moment les performances de votre application commenceront-elles à se dégrader ? Combien d'utilisateurs simultanés peut-elle supporter ? Comment les différents composants de votre application se comportent-ils avec 50, 100 ou 1 000 utilisateurs actifs ? Où sont les goulets d'étranglement entre votre base de code, votre base de données et les processus de répartition de la charge ? Sans réponses à ces questions, vos efforts de test et de développement pourraient s'avérer inutiles. Si votre logiciel ne peut pas tenir sous la contrainte, la souplesse de vos méthodes n'a aucune importance. Pour être prêt, commencez par l'analyse des risques des performances tôt au cours de l'itération afin que vous puissiez cartographier l'ensemble des zones potentiellement faibles à l'avance, même avant que le code soit écrit.

De même, le seul logiciel qui compte est un logiciel sûr. Les attaques de logiciel entreprises par des pirates, des criminels et des personnes agissant de l'intérieur peuvent entraîner l'interruption de l'activité, entacher la marque, conduire à des pertes financières colossales et nuire à des personnes innocentes. Les cibles de ces attaques sont les vulnérabilités masquées des applications logicielles. Ces vulnérabilités s'accumulent dans le logiciel, dans l'attente d'être exploitées.

Pire encore, de nouvelles vulnérabilités sont constamment introduites dans les organisations par leurs propres groupes internes de développement logiciel, ainsi que par le biais d'approvisionnements de fournisseurs, de sociétés de sous-traitance et de projets libres.

Répondre aux éventuelles menaces de sécurité de manière rapide au cours des cycles d'interaction courts du développement agile augmente les chances de détection et de correction précoces. Si les développeurs sont informés de la menace qu'une fonction représente pour la sécurité du logiciel, qu'ils viennent juste d'archiver pour cette version, ils seront bien plus à même de résoudre ce problème de façon opportune et pourront peut-être même valider la correction pour la prochaine version. Par exemple, vous pouvez ajouter le récit d'un utilisateur malveillant au cours de la phase de planification afin qu'il soit traité au cours de la phase de validation de l'itération.

Se rencontrer souvent

Bien que Scrum ne soit pas la seule méthode pour mener des tests agiles, des réunions Scrum quotidiennes se sont révélées extrêmement utiles pour les équipes sur le long terme. Elles aident en effet les services à rester concentrés, stimulés et coordonnés.

Scrum soulage les équipes de la pression de la gestion. Les équipes sont autorisées à choisir leur propre tâche, puis à s'organiser par le biais d'une communication rapprochée et d'une entente commune au sein l'équipe sur la meilleure façon de réaliser le travail. Dans une méthode Scrum réussie, cette autonomie peut améliorer de façon significative la qualité de vie des développeurs et permettre aux responsables de conserver leurs employés.

Cependant, trop souvent, les réunions Scrum quotidiennes deviennent des réunions hebdomadaires, qui, à leur tour, deviennent des réunions mensuelles. Pour éviter cela, convenez d'un moment qui satisfasse tout le monde et faites de votre mieux pour respecter le planning.

3 Assurer la réussite des tests agiles

La réalité du théâtre actuel de développement logiciel pousse de nombreux responsables d'équipe à se demander comment il est possible de devenir une entité agile plus efficace. S'il existait un assistant de la méthode agile, un responsable d'équipe et l'assistant pourraient avoir la conversation ci-dessous :

Responsable d'équipe : « Comment puis-je faire pour que mon équipe devienne agile ? »

Assistant de la méthode agile : « Vous ne pouvez pas changer les gens, vous pouvez simplement les aider à changer d'eux-mêmes. »

Responsable d'équipe : « Comment puis-je les aider ? »

Assistant de la méthode agile : « Vous devez d'abord devenir vous-même agile. »

Responsable d'équipe : « Comment puis-je devenir agile ? »

Assistant de la méthode agile : « C'est simple… mais pas facile. »

Pour retirer le mystère entourant le lancement de la méthodologie agile, prenons en compte les aspects essentiels du chemin vers l'agilité.

Changement de rôles

L'introduction des méthodes agiles modifie les rôles traditionnels d'une organisation. Le triumvirat de l'analyste de l'activité, du développeur et du responsable AQ revêt les nouveaux rôles agiles d'expert du domaine, de programmeur et de testeur, chaque rôle ayant des responsabilités communes avec les autres. Voici certains de ces rôles :

- Maître Scrum

  Cette personne est responsable des délais et des pense-bêtes.

- Détenteur du produit

  Parfois appelé expert du domaine, cette personne définit en général ce qui doit être fait, mais sans trop de détails.

- Programmeurs

  Cette personne met en application des récits à forte valeur commerciale à l'aide de bonnes techniques et pratiques.

— Testeurs

  Cette personne fournit des informations sur l'état du projet.

— Responsables du développement/d'équipe

  Ces personnes participent à la planification des discussions et assument la responsabilité globale du groupe de programmeurs en termes de livraison.

En fonction du nombre de personnes et de la complexité du projet, les rôles supplémentaires suivants apparaissent :

— Chef de projet

  Cette personne fait tout ce qui est nécessaire pour garantir que les projets importants sont livrés, que ce soit maintenant ou à l'avenir.

— Client (agile)

  Cette personne s'assure que l'équipe travaille sur les récits qui offrent la plus grande valeur commerciale possible.

— Concepteur de l'expérience utilisateur

  Cette personne aide le client à déterminer les besoins d'utilisation, aide l'équipe à les satisfaire et vérifie qu'ils ont été satisfaits.

— Architecte

  Cette personne garantit que l'équipe applique les bonnes techniques et pratiques.

Notez que les définitions ci-dessus correspondent réellement à des rôles et à des responsabilités, et non à des personnes. Une personne pourrait remplir plusieurs rôles dans la structure d'une organisation agile.

Étant donné que l'un des principaux attraits de la méthode agile est la promesse de travailler en équipe unie et non selon les silos de l'approche classique, cette structure présente de nets avantages :

— Une équipe, et non des individus

  L'équipe doit fonctionner comme une unité, et non comme un ensemble d'individus. Le responsable d'équipe doit trouver le moyen de renouveler l'engagement des membres de l'équipe et leur objectif réciproques. N'oubliez pas : vous ne pouvez pas motiver tout le monde selon le même degré. Une personne très motivée peut faire beaucoup de travail, mais le contraire est également vrai. L'examen annuel de chaque membre de l'équipe doit se concentrer sur l'accomplissement personnel de chaque membre, et non sur l'accomplissement de son équipe.

— L'ensemble de l'équipe œuvre pour la qualité.

  La qualité est en permanence maintenue avec l'approche de développement agile. Les testeurs et les programmeurs ont davantage confiance dans le code : ils savent qu'il existe une chance limitée que des anomalies inconnues créent des problèmes en aval.

— L'ensemble de l'équipe est responsable des tests.

  Les testeurs effectuent également une évaluation des risques en classant les anomalies par ordre de priorité afin de s'assurer que celles ayant éventuellement le plus d'impact sur le logiciel reçoivent la plus grande attention et les meilleures ressources en développement.

— Les testeurs reçoivent de l'aide et une formation permanente.

  Les testeurs sont davantage intégrés dans le développement, leurs compétences sont également différentes de celles qu'ils avaient avant que la société ne passe à la méthode agile. Il existe un besoin en testeurs ayant un profil technique, leur rôle consistant à plonger davantage dans le code. Ils ne travaillent pas seulement sur une interface utilisateur graphique ou au niveau de l'interface utilisateur.

— Les développeurs assurent les tests.

  Les tests unitaires constituent la norme de facto des équipes agiles. Le meilleur scénario, adopté par peu d'équipes, consiste à rédiger les tests unitaires pendant le codage, voire même avant.

  En outre, HP Application Lifecycle Intelligence (ALI) confère à HP ALM la possibilité de surveiller les modifications apportées au code source, de repérer les modifications non testées, de signaler si une anomalie est associée à un changement de code et d'avertir si un changement de code n'est pas lié à une exigence professionnelle. Il améliore HP ALM par l'agrégation de données provenant de plusieurs sources, ce qui permet aux décideurs d'accéder aux informations pertinentes et de visualiser les risques éventuels de modification dans les projets d'application. Tout cela se produit dans l'environnement de développement familier choisi sans modification de la façon dont les développeurs travaillent tout en leur donnant un aperçu des exigences et de l'assurance qualité qui sont généralement oubliées dans l'approche de développement en cascade.

— Les testeurs et les développeurs s'associent pour résoudre certains problèmes.

  Chaque membre de l'équipe apporte son propre point de vue sur le problème, ce qui permet de se rapprocher de la résolution. Étant donné que tous les membres travaillent pour la même équipe, il n'existe aucun problème de rupture de communication et des intérêts personnels.

La collaboration pour une équipe solide

Les rôles du testeur et du développeur sont en conflit de nombreuses manières. Un développeur s'engage à la réussite dans toutes les créations qu'il entreprend. Un testeur essaie de réduire le risque d'échec et tente d'améliorer le logiciel en détectant les anomalies. Les développeurs s'attachent à la technologie, ce qui prend beaucoup de temps et d'énergie lors de la production de logiciels. Un bon testeur, d'un autre côté, a pour objectif de fournir à l'utilisateur le meilleur logiciel pour résoudre un problème.

La méthode agile s'efforce de mettre fin aux perceptions erronées et de créer une équipe unifiée par l'objectif de livrer un logiciel de qualité en temps et en heure. La méthode agile consiste à fournir la valeur commerciale maximale plus rapidement en se concentrant sur les personnes et sur l'amélioration continue. Tandis que la collaboration reste un thème clé, quelque chose de plus élémentaire doit être traité au tout début : l'importance de la compréhension mutuelle. Il s'agit de la base de la construction d'une équipe solide, dans laquelle les personnes communiquent dans une langue dont ils ont convenu ensemble, pour fournir un logiciel de qualité. Sans cette coopération, le responsable des tests aurait du mal à définir les critères de test et à s'assurer que les niveaux de qualité nécessaires sont atteints. La seule solution possible est que les tests et le développement fonctionnent de pair. Les testeurs peuvent être des alliés puissants du développement et, tout au long du processus, ils peuvent passer d'adversaires à partenaires. Ceci est possible parce que la plupart des testeurs souhaitent être utiles. Ils n'ont besoin que d'un peu de considération et de soutien. Pour ce faire, cependant, un environnement doit être créé afin d'exploiter le meilleur des aptitudes d'un testeur. Le testeur et le responsable du développement doivent établir la phase de coopération au début du cycle de développement et communiquer tout au long du cycle. Dans de nombreux cas, cela se traduit par le rapprochement physique des équipes de développement et de test, ce qui encourage les interactions.

Une bien meilleure communication entre développeur et responsable AQ simplifie grandement le processus car elle se traduit par moins de bureaucratie et par plus de codage, pour une livraison plus rapide. Les testeurs sont représentés à chaque étape du cycle de vie du développement des logiciels et influencent rapidement les décisions ayant trait à la qualité. La conception et l'exécution des tests prenant place plus tôt dans le cycle, il est toujours possible d'agir et de modifier une partie de code compromise.

Pour résumer les informations ci-dessus, il existe de nets avantages à appliquer la méthode agile :

- Une équipe.

  L'AQ n'a jamais été aussi proche du développement et devient partie intégrante du cycle.

- La qualité est désormais un problème pour l'équipe dans son intégralité et pas seulement pour le testeur.
- Le niveau des compétences d'une personne de l'assurance qualité augmente par l'interaction permanente avec les développeurs.
- Tout se passe très rapidement.

  Plus les tests sont réalisés rapidement, meilleure en est l'influence.
- Le niveau de qualité s'améliore grâce aux tests et à la détection des anomalies précoces.

Les approches comme les tests unitaires (pour vérifier que l'élément approprié est développé correctement) et les tests d'acceptation (pour vérifier que l'élément approprié est développé) ne sont pas nouvelles. Par le passé, toutefois, ces approches ont souvent été gérées de façon isolée. Avec la méthode agile, une solution pragmatique complète est privilégiée. Elle porte sur l'exigence elle-même, tout en tenant compte des parties prenantes. Vous pouvez trouver cette solution dans HP ALM. Elle permet de surmonter divers obstacles :

- Les obstacles se dressant entre les phases de projet et les activités de projet.

  Parce que le codage et les tests avancent plus étroitement ensemble.
- Les obstacles se dressant entre les types d'artefact.

  Parce que le code et les spécifications exécutables sont rédigés sur la même infrastructure unifiée.
- Les obstacles se dressant entre les rôles de projet.

  Parce que les tests sont rédigés en collaboration, avec des mécanismes pour utiliser des termes proches du domaine du problème.

Cycles rapides

La méthodologie classique en cascade de développement des systèmes est essentiellement une approche séquentielle de développement des solutions. Problème courant avec le modèle en cascade : le temps avant livraison du produit peut être excessif.

En revanche, le développement agile accélère la livraison du produit. Un petit système initial en état de fonctionnement est créé et rapidement livré, puis amélioré lors d'une série d'itérations. L'avantage est que les clients reçoivent au moins rapidement certaines fonctionnalités. Autre avantage : le produit peut être mis en forme par des commentaires itératifs. Par exemple, les utilisateurs n'ont pas besoin de définir chaque fonction correctement et en détail au début du

cycle de développement, mais peuvent réagir aux résultats de chaque itération. Avec l'approche agile, le produit évolue sans cesse au fil du temps. Il n'est pas statique et peut ne jamais être « achevé » au sens classique du terme.

L'expérience des équipes agiles internes de HP, ainsi que celle de ses nombreux clients, révèlent qu'il existe des périodes communes dans la nature cyclique :

- Les itérations, généralement appelées « sprints » dans la langue SCRUM, durent entre 2 et 6 semaines, les cycles de 4 semaines étant les plus populaires. Le résultat du sprint est la version contrôlable du code.
- Les jalons, lorsque la version en état de marche du produit est créée et envoyée au client de l'entreprise, se produisent tous les 3 à 4 sprints. À ce stade, la planification du jalon suivant est effectuée. Les clients ont la possibilité de voir une version incomplète mais en état de marche du produit final, et de fournir des commentaires qui sont incorporés dans le plan.
- Des réunions quotidiennes afin de discuter de l'état actuel de l'équipe. Ces réunions, généralement effectuées à proximité de la planche à dessin et remplies de pense-bêtes, réunissent les membres de l'équipe, et permettent le partage des problèmes et des idées de manière libre. Les membres de l'équipe choisissent les prochaines tâches à effectuer comme ils l'entendent, tout en gardant à l'esprit les compétences de chacun.
- Des statuts simples sont prévus pour suivre la progression. En général, ils comprennent les suivants :
  - À faire : statut initial après retrait de la tâche de la file d'attente.
  - En cours : lorsque la fonction est en cours de développement.
  - Terminé : lorsque la fonction réussit un test d'acceptation.
  - Obstruction : ce qui vous empêche d'accomplir votre tâche. Le blocage peut être d'origine technique ou non.

  Ces statuts, bien entendu, ne sont que des exemples. Les différentes entreprises peuvent améliorer la liste avec des valeurs supplémentaires qui correspondent mieux à la nature de leur activité.

Ces courtes périodes encouragent une réponse rapide du testeur et une bonne collaboration avec le développeur. Sans ces conditions, l'intervention peut s'avérer trop tardive pour pouvoir affecter les décisions concernant le produit, qui évoluent au fil du temps.

Les équipes agiles apportent de plus petites améliorations et utilisent une planification collaborative juste à temps afin de s'assurer que les modifications essentielles peuvent être publiées plus tôt. Si un élément est important, il peut être intégré dans une mise à jour ou une itération antérieure. Il peut déplacer une autre fonction qui n'est pas aussi importante, mais il ne perturbera pas toute une mise à jour. Résultat : une livraison plus rapide.

La planification est la clé

Le développement logiciel, c'est l'application d'exigences. L'exigence est le pivot et le moteur d'une publication logicielle.

Les tests classiques basés sur les exigences prévoient que la définition du produit soit finalisée et même figée avant la planification détaillée des tests. Avec le développement agile, la définition du produit et les spécifications évoluent indéfiniment. Autrement dit, rien ne vaut une spécification figée. Une définition des exigences et une conception du système complètes ne seront probablement jamais documentées.

En d'autres termes, la phase de planification est probablement la partie la plus importante du processus. Pendant des discussions importantes, l'ensemble du contenu du projet, parfois appelé *file d'attente*, est considéré comme la base de l'effort important suivant. La file d'attente est ensuite divisée en parties plus petites appelées *récits utilisateur* qui sont essentiellement les exigences de ce dernier, mais sans trop de détails et décrites dans un langage humain.

Les responsables d'équipe tirent des *tâches* des récits utilisateur, des activités dont la réalisation ne devrait idéalement pas prendre plus de 3 ou 4 jours. S'il est prévu qu'une tâche prenne plus de temps, il est nécessaire de diviser le récit utilisateur en plusieurs récits utilisateur plus petits afin qu'ils puissent être associés à la tâche à la fin.

La tâche devient une unité de travail finale, et les développeurs la prennent du pool général à leur gré, en fonction de leurs compétences.

Les nouvelles équipes agiles ont tenté de simplifier à l'excès ou d'ignorer le processus de planification pour son caractère formel excessif. Ne laissez pas ceci se produire. La planification ne doit jamais être ignorée ou devenir inutile. Au contraire. Investir le temps nécessaire, amener toutes les parties concernées autour de la table et répartir les éléments de la file d'attente en récits utilisateur de taille appropriée garantira un flux d'événements bénéfique au cours de la période charnière. Lorsque vous travaillez dans des équipes agiles, utilisez vos compétences agiles pour étalonner le processus de planification : allongez-le ou raccourcissez-le selon les besoins de votre activité.

La communication, le vecteur principal de la méthode agile

L'ensemble de l'équipe, y compris les programmeurs, les testeurs et les clients, étant responsable du résultat, seul un engagement ferme de chacun des membres de l'équipe le rend possible. Pour faciliter ce résultat, chaque membre de l'équipe doit être accessible et communiquer activement tout au long du projet.

Les ingénieurs en assurance qualité suivant la méthode en cascade ont tendance à se reposer sur des documents. Les testeurs agiles ne disposent pas d'un document volumineux sur les exigences comme base des cas de test et ne passent pas trois mois à écrire des cas de test avant qu'une partie de code soit capable de fonctionner. Ils se laissent porter par le flux de communication, qui peut être verbal, écrit ou virtuel, et assimilent les informations dont ils ont besoin. Ils apprennent à poser les bonnes questions au moment opportun et fournissent la quantité de commentaires idéale au bon moment.

Malheureusement, les testeurs ont en général du mal à :

- Comprendre ce que les clients veulent savoir.
- Supprimer les informations inutiles ou inintéressantes (le faire à l'excès n'est *pas* mieux).
- Voir l'éventuel impact des problèmes techniques sur l'activité.
- Présenter les données de façon à ce qu'elles soient facilement comprises.

Afin d'améliorer l'attitude générale vis-à-vis des testeurs et de communiquer efficacement leur valeur, il est nécessaire de faire preuve d'efforts dans l'élaboration d'informations à l'attention des nombreuses parties prenantes de l'équipe et de la société. Utilisez des outils modernes tels qu'un wiki pour produire une documentation sur les tests et fournir des statuts et des résultats de test. Ces deux éléments peuvent se trouver sur la même page wiki et servir de suivi de la conception des tests et des résultats des tests. Assurez la visibilité des informations.

Si vous souhaitez atteindre un haut niveau de qualité dans vos systèmes, vous devez intégrer des professionnels de la qualité au sein des équipes de développement afin qu'ils examinent le travail en cours et fournissent des conseils adaptés de manière itérative (réduisant ainsi le temps de communication et limitant le risque de perte d'informations via la documentation). Notez que vous pouvez toujours disposer d'une équipe de test indépendante, qui valide en externe le travail de l'équipe de projet, mais la plupart des professionnels de la qualité seront intégrés dans l'équipe.

L'automatisation est le mot d'ordre

Les cycles de développement ayant été raccourcis et les testeurs n'ayant pas des mois pour concevoir et exécuter des cas de test, l'approche agile exige des délais d'exécution rapide des activités de test.

Des cycles plus courts augmentent la souplesse. Heureusement, avec les projets agiles, le logiciel est prêt aux tests pratiquement dès le début. En règle générale, les équipes agiles utilisent plusieurs niveaux de test pour trouver différents types d'information :

- Des tests unitaires automatisés

  Ces tests vérifient le comportement de chaque fonction/méthode et des interactions entre objets. Ils répondent à la question : « Le code donne-t-il ce que le programmeur prévoyait ? ». Les tests unitaires fonctionnent souvent et fournissent des commentaires en quelques minutes.

- Des tests d'acceptation automatisés

  Lorsque vous avez besoin de vérifier le comportement du système de bout en bout, utilisez les tests d'acceptation. Ces tests sont les plus importants puisqu'ils apportent des réponses aux questions telles que « Le système fait-il ce que le client souhaite ? ». Ils fonctionnent généralement sur le code archivé sur une base continue, fournissant des commentaires en une heure environ.

- Des tests de régression automatisés

  En raison de la nature cyclique des projets agiles, il est nécessaire de vérifier l'état du produit pour voir si ce qui fonctionnait fonctionne encore. L'exécution des tests de régression manuels est plus longue et, parce qu'un homme doit être disponible, ces tests peuvent ne pas commencer immédiatement. Le délai avant réception des commentaires peut être des jours ou des semaines.

Il n'est pas étonnant que les projets agiles privilégient les tests automatisés. En fait, sans l'automatisation, il est pratiquement impossible de suivre le changement constant qui émane de chaque cycle. Par conséquent, il existe des idées simples qui permettraient à un testeur agile d'avoir toujours un temps d'avance sur la courbe :

- Planifier l'itération en cours.

  Concevez des tests pour les fonctionnalités ou les récits à faire à court terme. Bien souvent, la planification proactive est synonyme de retravail.

- Utiliser des catalogues de tests réutilisables.

  Ne revoyez pas à l'infini la conception des mêmes tests. Conservez plutôt des listes de vérification réutilisables.

— Conserver une liste des risques classés par ordre de priorité, mise à jour.

Quel type d'informations les testeurs recherchent-ils ? La liste des risques doit répondre à cette question.

Associée au code source, à l'automatisation des versions, au déploiement automatique vers un environnement de test et à l'intégration continue, l'automatisation des tests est l'un des principaux moteurs de livraison rapide de qualité de produits logiciels dans un monde agile.

Toutefois, les tests manuels, en particulier les tests exploratoires manuels, restent importants, même sur des projets agiles. Le bon sens habituel concernant les tests veut qu'il soit impossible de lancer les tests d'une fonction tant qu'elle n'est pas accessible à partir d'une interface externe, comme une interface graphique. Mais il est inutile d'attendre. L'automatisation des tests peut faciliter l'exploration manuelle qui apporte des informations complémentaires et couvre les lacunes de l'automatisation, et est nécessaire pour développer les tests automatisés.

Il existe, bien sûr, davantage de types de test qui doivent être appliqués au logiciel (par exemple, les tests inter-environnementaux, les tests dans des environnements propres au client, des tests d'internationalisation et de localisation). À l'instar des tests fonctionnels, ils peuvent et doivent être automatisés autant que possible, et traités pendant la période agile.

Commencer doucement

L'une des étapes importantes dans la diffusion de la méthodologie agile au sein de la société est de mener un projet pilote. Son objectif est le suivant :

- Vérifier les concepts énoncés dans les manuels ou par les formateurs à la méthode agile.
- Montrer la valeur initiale de la création d'une équipe agile.
- Commercialiser le concept d'un développement agile au sein de l'organisation.

Généralement, le processus commence avec une équipe ou un projet, et, en fonction des résultats, il est déployé dans l'ensemble de l'organisation.

Pour mener à bien le projet pilote, une organisation doit déterminer la cible initiale de l'équipe agile nouvellement formée et définir le contenu exact d'un projet spécifique de développement. Lors de la préparation d'une phase pilote, il est nécessaire de traiter autant d'aspects de la méthode agile que possible, mais sur une moindre échelle :

- Planification du projet pilote
- Mener des sessions de présentation générale sur les principes agiles
- Intégrer éventuellement un formateur à la méthode agile
- Identifier le contenu du projet et ses représentants commerciaux
- Regrouper les membres des équipes en un seul endroit
- Définir les membres du projet pilote
- Établir une chronologie du projet pilote et repérer les points de livraison
- Documenter la progression de chaque étape

L'expérience des clients HP se révélant, l'équipe du projet pilote se transforme tout en travaillant sur un produit existant ou sur son amélioration. Les entreprises créent de petites équipes dont la durée de vie, courte à moyenne, est prédéfinie, et les regroupent en un même endroit. L'équipe reçoit la liberté de définir sa structure et ses méthodes de travail, mais prend intégralement la responsabilité du résultat. Même si un projet pilote est mené sur une petite échelle, il est essentiel d'obtenir l'appui de la hiérarchie. La transmission régulière de rapports sur la progression et les problèmes au commanditaire permettra de garantir le soutien de la hiérarchie.

À l'issue du processus, il est de bon ton de mener une session d'analyse, parfois appelée « rétrospective », à la fois pour les membres de l'équipe du projet pilote que pour leurs clients. Des commentaires complets vous permettent d'éviter, à l'avenir, les erreurs inutiles, et facilitent l'expansion rationalisée de la méthodologie agile. Si possible, le commanditaire exécutif doit participer à cette session d'analyse. Dans tous les cas, un document décrivant le processus et ses conclusions, pouvant être utilisé ultérieurement sous la forme d'un plan de déploiement général, doit être disponible.

La centralisation est également une preuve d'agilité

Tandis que la plupart des équipes agiles ressemblent à la structure décrite ci-dessus, il existe généralement certaines équipes inter-projets spécialisées dotées de compétences uniques en infrastructure, gestion des laboratoires de tests, tests de charge, évaluation de la sécurité, etc. Rien d'étonnant vu les compétences uniques requises des membres de ces équipes. La maîtrise de l'art de l'ingénierie des performances ou de l'analyse de la sécurité prend beaucoup de temps. Toutefois, la nature spécifique de ces équipes ne doit pas les empêcher de participer aux processus agiles. Dans les grandes organisations, il n'est pas rare pour les équipes centralisées de trouver des avantages au soutien des initiatives des entreprises. Elles peuvent en effet apporter une vue sur l'avenir différente et plus holistique sur tous les problèmes non fonctionnels.

Des principes essentiels doivent être suivis lors de l'intégration d'un spécialiste :

- Cette personne travaille exclusivement avec un point de contact dans chaque équipe.
- Dans certains cas, un spécialiste est dédié à une équipe spécifique. Dans les autres, un ingénieur intervient dans plusieurs équipes. Cela dépend vraiment de la taille et de la complexité du projet.
- Chaque mission doit commencer par la réunion de lancement formel, à l'occasion de laquelle chaque partie prend connaissance de ses objectifs et des délais.
- Chaque mission est conclue par un document signé à l'aide d'un modèle prédéfini.
- La participation nécessite l'engagement du responsable de l'équipe de développement afin de s'assurer que les aspects non fonctionnels sont pris en compte et traités.

La nature itérative des projets agiles correspond parfaitement aux objectifs des tests non fonctionnels : s'impliquer et avoir une influence dès les premières phases du projet permettent de traiter les problèmes inhérents et donnent un excellent taux de résolution.

Problèmes avec la méthode agile

Un autre aspect de l'adoption de la méthode agile concerne les différentes variantes du processus. Aucune entreprise n'est la même, ce qui se reflète dans la manière dont le développement et les tests des applications sont effectués. HP, grâce à sa large base clientèle, a la chance de pouvoir suivre différentes formes de l'application de la méthode agile et les techniques de test agiles.

La méthode agile relève bien plus de la culture d'entreprise, des relations humaines et des procédures commerciales que d'outils spécifiques, de slogans et d'astuces. Elle est bien plus rapide que la méthode en cascade, mais elle peut avoir bon nombre des problèmes de certains des cycles de projet classiques : exigences inadaptées, problèmes de communication, absence de documentation.

Bien que la méthode agile présente un fort potentiel et un excellent taux d'adoption, elle comporte certains problèmes et restrictions. Les responsables doivent prendre en compte les éléments suivants avant de déployer de manière globale de la méthodologie :

- L'approche agile semble mieux fonctionner avec de petits groupes.
- Elle est bien plus facile à gérer lorsque les personnes sont regroupées en un endroit. Toutefois, on trouve toujours des exemples d'équipes agiles triomphantes se trouvant sur plusieurs sites.
- Autre avantage du regroupement physique : le développement à l'étranger est problématique et impose une énorme pression sur l'activité au quotidien.
- Il est parfois vraiment difficile de tester quelque chose avant la fin de l'ensemble du projet même si chaque test d'itération a été marqué comme ayant été terminé. Bien souvent, ce statut est accordé par les responsables AQ sans satisfaction complète.
- Bien souvent, les tests de l'interface utilisateur graphique sont problématiques avec la méthode agile, tout pouvant éclater lors de l'itération suivante.
- La phase de la planification prend encore un temps considérable, du moins à chaque jalon.
- Une mauvaise planification au début du jalon entraîne des problèmes ; autrement dit, si les éléments de la file d'attente ont mal été répartis, le récit utilisateur est trop important ou trop petit. Ceci peut avoir une incidence sur l'ensemble du cycle de développement.

Il doit être clair dès le début que la méthode agile n'est pas la panacée pour les problèmes liés au développement, au processus et à la gestion. Toutefois, si vous suivez les conseils de HP et examinez sans cesse la situation, elle peut être extrêmement bénéfique pour la qualité du logiciel produit.

4 Conclusions

L'approche agile existe depuis quelque temps maintenant. Elle s'est avérée précieuse dans la plupart des projets complexes réalisés par les entreprises, petites et grandes. Les entreprises les plus innovantes d'aujourd'hui et de demain continueront à exiger toujours plus de la méthode agile. Pour elles, être capables de définir, de planifier et d'exécuter efficacement des projets dans un environnement dynamique à grande vitesse fera toute la différence entre simplement survivre et prospérer. L'agilité, c'est prendre les bonnes décisions aussi bien au cours de l'*exécution* d'un projet que de la *planification*.

D'un point de vue des tests, le plus grand défi à relever lorsque vous êtes membre d'une équipe de projet agile est que le logiciel est *en constante* évolution. Le nouveau code étant intégré dans l'environnement de test tous les jours, voire toutes les heures, le système que vous testez devient une cible mobile. Traiter ce type de changement peut requérir différentes approches aux tests, avec différentes compétences, tactiques et outils. Certains aspects des tests de logiciels ne changent pas simplement parce que l'équipe de projet utilise une approche agile pour mettre en œuvre le logiciel. C'est le cas notamment des mécanismes de l'exécution des tests, mais certains testeurs peuvent avoir besoin d'apporter des modifications significatives à leur approche de test s'ils souhaitent ajouter de la valeur à un projet logiciel agile. Les testeurs agiles doivent prendre des décisions quant aux tâches à effectuer par la suite, à la manière dont ils vont s'y prendre, aux moyens de les rendre pertinentes pour le client et à la façon d'exécuter l'application de manière à améliorer la compréhension du fonctionnement du processus et des risques.

Nous souhaitons tous améliorer la qualité des applications que nous produisons pour le compte de nos services, secteurs d'activité et entreprises respectifs. Ce qui est nécessaire pour rendre cet objectif réalisable est envisager la qualité sous l'angle du cycle de vie, une planification et une conception appropriées de l'environnement, l'utilisation cohérente des meilleures pratiques, des outils de pointe tels qu'HP ALM et une connaissance approfondie de l'activité de l'entreprise.

Nous espérons que les pratiques conseillées répertoriées dans ce document faciliteront l'adoption de la méthodologie agile et l'utilisation d'HP ALM dans votre organisation.