Université du Québec
École de technologie supérieure

RAPPORT TECHNIQUE
PRÉSENTÉ À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
DANS LE CADRE DU COURS LOG792 – PROJET DE FIN D'ÉTUDES EN GÉNIE

Implantation de la méthodologie SCRUM dans les grandes entreprises

MARTIN MAYER
MAYM24087501

DÉPARTEMENT DE GÉNIE LOGICIEL ET DES TI

Professeur superviseur

Alain April

MONTRÉAL, 17 AVRIL 2010
HIVER 2010

REMERCIEMENTS

Ce rapport est une résultante provenant, entre autre, d'un effort continu des gestionnaires de grandes entreprises dans leurs créativités pour gérer la demande toujours plus élevée de performance technologique sur un marché plus compétitif que jamais. En désirant repousser les limites de la performance, ils ont dû revoir les processus de gestion inhérents à la réalisation de leurs projets informatisés. La méthodologie SCRUM fait le pont entre le désir de l'entreprise et le développement logiciel.

Ce rapport n'aurait pas été possible sans l'aide de mes précieux collaborateurs qui ont pris le temps de partager leur expertise et leur expérience de la méthodologie SCRUM. M. Erik Lebel et M. Luc Dorval, tous deux évoluant au sein de Pyxis Technologie inc. en tant que professionnels Agile et SCRUM. Ils ont accompagné des équipes en entreprises dans l'implantation de SCRUM et de la méthodologie Agile. Aussi, M. Fernando Cuanca, un entraîneur Agile aguerri, évoluant auprès de Tek Systems qui utilise la méthodologie au quotidien. Leur savoir-faire de la méthodologie SCRUM m'ont permis de présenter un portrait des subtilités de l'application de celle-ci.

Enfin, il ne faut pas passer sous silence l'aide de M. Alain April, membre du Laboratoire de Recherche en Génie Logiciel de l'École de Technologie Supérieure (ETS) - Université du Québec. M. April a facilité le premier contact auprès des intervenants chez Pyxis Technologie Inc. chef de file de la grande région de Montréal dans l'application et l'intégration Agile et SCRUM.

Implantation de la méthodologie SCRUM dans les grandes entreprises

MARTIN MAYER
MAYM24087501

RÉSUMÉ

Les grandes entreprises en développement logiciel sont en croissance constantes et poursuivent le but de livrer du logiciel de qualité qui répond au besoin de l'utilisateur, dans les temps prescrits par le client. La problématique de celles-ci réside en l'utilisation des méthodes de développement traditionnelles qui ne satisfont plus entièrement aux besoins grandissants de qualité.

Les grandes entreprises essaient, en vain, d'instaurer de nouvelles méthodes de développement. L'objectif de ce rapport est de fournir aux grandes entreprises qui désirent optimiser la qualité du logiciel et l'efficacité de leurs équipes de développement, un guide d'implantation de la méthodologie SCRUM adaptée à leur réalité.

Pour arriver à ces conclusions, il a fallu étudier la méthodologie, l'appliquer dans une grande entreprise et valider les impacts de ces changements. Ainsi, ce rapport démontrera les caractéristiques de la méthodologie SCRUM en utilisant une combinaison de recherches bibliographiques, d'entrevues avec des professionnels SCRUM et des recherches internet.

Après un bref historique de la méthodologie SCRUM, le rapport traite du processus de développement, de l'influence importante des pratiques de la programmation extrême, du rôle des intervenants, des problématiques reliées à son implantation dans les grandes entreprises ainsi que des moyens d'implanter la méthodologie SCRUM au sein d'une équipe.

La méthodologie SCRUM est plus qu'une technique, elle est un changement dans la façon d'aborder le développement logiciel en passant par une plus grande participation des intervenants internes et externes du processus. C'est un engagement quotidien de toutes les parties prenantes vers un avenir ou la qualité logicielle et l'authenticité des rapports humains n'auront pas de compromis sinon que la synchronicité et le succès de leurs entreprises.

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

OOPSLA	« Object-Oriented Programming, Systems, Languages, and Applications »

INTRODUCTION

Les entreprises en développement logiciel sont en croissance constantes et poursuivent le but de livrer du logiciel de qualité qui répond au besoin de l'utilisateur, dans les temps prescrits par le client. Afin de répondre à ces critères, les entreprises doivent utiliser des processus de développement logiciel stricts. Il y a plusieurs processus disponibles qui apportent leurs avantages et leurs inconvénients. Depuis quelques années, l'utilisation de la méthodologie SCRUM semble gagner en popularité, mais peu d'entreprises s'y aventurent.

La méthodologie SCRUM est un processus de développement qui demande une coordination entre le client, les équipes de développement et la direction de l'entreprise. Ce processus exige d'implanter des méthodes de travail et de communication qui sont souvent nouvelles pour bien des individus. Elle requiert du client de s'impliquer activement dans le projet et à l'équipe de changer ses méthodes de travail. Ainsi, plus l'entreprise est grande, plus il peut être difficile d'implanter SCRUM.

SCRUM présente une solution intéressante pour les grandes entreprises qui aimeraient gagner en flexibilité. En utilisant une méthode évolutive de développement qui implique une plus grande participation du client dans le processus de développement, les deux parties voient leurs chances de succès augmentées proportionnellement à la qualité de leurs communications et de leurs relations. En utilisant la méthodologie SCRUM dans les grandes entreprises, la qualité du logiciel est accrue et les nouveaux besoins commandés par la réalité changeante du client sont considérés tout au long du processus. Une synergie qui gagnerait à être reconnue.

Ce rapport présentera l'implication, les risques et un moyen d'implantation de la méthodologie SCRUM dans la réalité des grandes entreprises. Afin de bien définir SCRUM nous aborderons au chapitre 1 : l'historique et feront un tour d'horizon de SCRUM. Le processus de développement sera défini au chapitre 2, pour ensuite traiter des pratiques de la

programmation extrême au chapitre 3 et des rôles des intervenants au chapitre 4. Au chapitre 5, nous analyserons les problématiques de l'implantation de la méthodologie SCRUM dans les grandes entreprises pour finir au chapitre 6 : moyen d'implantation de cette méthodologie. Ce rapport technique dressera, pour terminer, des recommandations face à l'utilisation de cette méthodologie dans la grande entreprise.

Avant de poursuivre, il est essentiel de mentionner que la méthodologie est parfaite pour la programmation orientée objet puisqu'elle permet de développer les spécifications les unes après les autres dans un environnement contrôlable. De ce fait, les projets de type procéduraux sont inappropriés à cette méthode puisqu'il demande un système entièrement programmé incluant l'ensemble des spécifications développées pour être livrer et fonctionner.

CHAPITRE 1

HISTORIQUE ET TOUR D'HORIZON DE SCRUM

Ce chapitre présente l'historique de SCRUM et fait un bref tour d'horizon. Il permettra d'apporter les informations nécessaires pour une meilleure compréhension des chapitres suivants.

1.1 Historique de SCRUM

En 1993, Jeff Sutherland [Sutherland 1993] et Ken Schwaber ont fait l'analyse des processus de développement logiciel de l'époque. Leurs conclusions étaient les suivantes :les processus analysés ne convenaient plus à la méthode empirique et devenaient imprévisibles ce qui les empêchaient d'être répétés.

Le tableau 1 suivant présente les résultats de leur étude comparative des méthodologies de développement logiciel existant soit : cascade, spiral, itératif et SCRUM.

Tableau 1 Comparaison des méthodologies de développement [Sutherland 1993]

	Cascade	Spiral	Itératif	SCRUM
Processus défini	Requis	Requis	Requis	Planification et fin de projet seulement
Produit final	Déterminé durant la planification	Déterminé durant la planification	Défini durant le projet	Défini durant le projet
Coût du projet	Déterminé durant la planification	Partiellement variable	Défini durant le projet	Défini durant le projet
Date de fin de projet	Déterminé durant la planification	Partiellement variable	Défini durant le projet	Défini durant le projet
Adaptable à l'environnement	Planification seulement	Planification seulement	À la fin de chaque itération	En tout temps
Flexibilité et créativité de l'équipe	Limité – approche livre de recettes	Limité – approche livre de recettes	Limité – approche livre de recettes	Illimité durant les itérations
Transfert de connaissance	Formation avant le début du projet	Formation avant le début du projet	Formation avant le début du projet	Formation entre les membres de l'équipe durant le projet
Probabilité de succès	Faible	Moyen à faible	Moyen	Élevé

Jeff Sutherland et Ken Schwaber, en s'appuyant sur l'analogie du jeu du rugby[1], ont alors proposé un nouveau processus de développement logiciel : le SCRUM.

L'analogie du SCRUM au rugby a été utilisée pour la première fois par Takeuchi et Nonaka [Takeuchi, Nonaka 1986] qui ont publié une étude dans le « Harvard Business Review ». Dans cette étude les auteurs comparaient les équipes performantes et multifonctionnelles aux mêlées utilisées par les équipes de rugby.

Le SCRUM est un processus itératif et incrémental qui s'adapte à la réalité des projets de développement logiciel. Le processus SCRUM a été ensuite combiné aux principes de la programmation extrême proposée par Mike Beedle. Cette version de SCRUM est la première qui a été présentée à la conférence internationale OOPSLA (« Object-Oriented Programming, Systems, Languages, and Applications ») de 1995.

En 2001, Ken Schwaber et Mike Beedle ont publié le premier ouvrage de référence sur le sujet intitulé : « Agile Software Development with Scrum ».

Finalement, c'est en février 2008 que Ken Schwaber, Mike Cohn, et Esther Derby ont fondé l'alliance SCRUM, une organisation sans but lucratif. Cette organisation a pour but et de soutenir les utilisateurs intéressés à SCRUM pour réussir l'implantation et l'utilisation de cette méthode par la promotion de publication d'articles, de ressources, de formations et d'événements s'y rattachant. Elle a pour mission de sensibiliser et de soutenir l'amélioration itérative de la profession du développement logiciel.

---

[1] Le SCRUM au rugby est une manière de relancer le jeu, soit après une violation accidentelle ou lorsque le ballon est sorti de la zone de jeu.

1.2 Tour d'horizon de SCRUM

SCRUM est une méthodologie agile qui permet de livrer un logiciel plus rapidement avec plus de qualité. SCRUM permet au client d'un logiciel développé de contrôler la qualité du travail à effectuer tandis que l'équipe contrôle la quantité de travail effectué. Cette méthodologie permet de s'adapter rapidement aux changements d'un client puisqu'à fréquence régulière (chaque fin d'itération) l'équipe et le client réévaluent les spécifications du logiciel. Ainsi, le client reçoit les spécifications qu'il a demandées plus souvent (jusqu'à une possibilité d'un livrable par semaine) ce qui ne fait qu'accroître sa satisfaction. Les spécifications développées sont toujours les plus pertinentes pour le client, car elles sont réévaluées avant d'être entamées. Ce qui permet d'optimiser les ressources de développement selon les besoins réels du client.

SCRUM pourrait se résumer par deux actions soient inspecter et adapter. Ces actions, à elles seules, décrivent presque toutes les situations rencontrées durant l'utilisation de cette méthodologie.

La durée d'une itération varie d'une à quatre semaines et sa valeur d'affaires est plus élevée dans l'itération la plus courte. Plus loin, nous reviendrons sur la notion de valeur d'affaires.

La méthodologie SCRUM propose les intervenants suivants :

- Le gestionnaire de produit (« Product Owner »);
- Le maître SCRUM (« ScrumMaster »);
- L'équipe de développement

La méthodologie SCRUM propose les activités suivantes :

- Planification d'itération (« Sprint Planning »);
- Revue d'itération (« Sprint Review »);
- Rencontre/mêlée quotidienne (« Daily Scrum Meeting »).

La méthodologie SCRUM propose la création d'artefacts :

- Le carnet de produit (« Product Backlog »);
- Le carnet d'itérations (« Sprint Backlog »);
- Le graphique d'avancement (« BurnDown Chart »).

La figure 1 donne un aperçu de la méthodologie SCRUM dans son ensemble qui sera traitée dans les prochains chapitres.

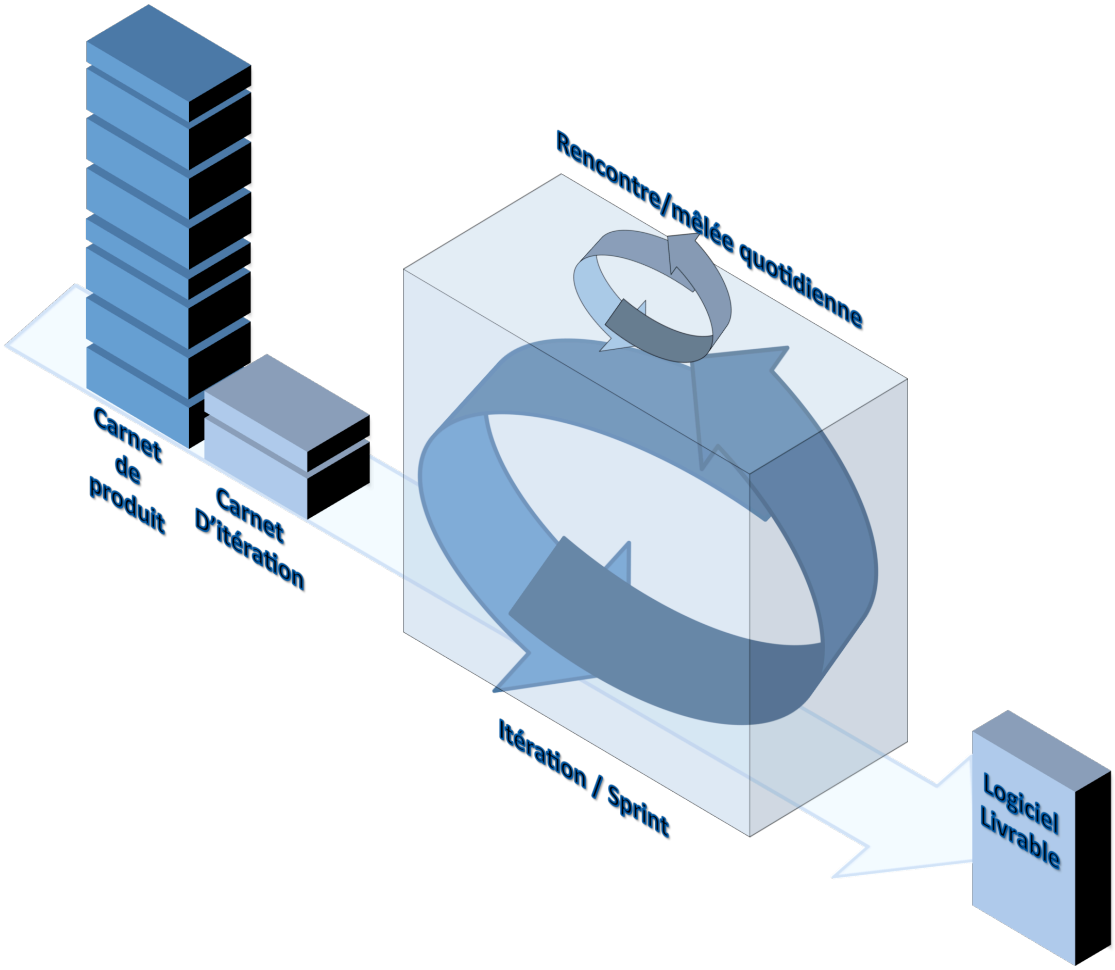


Figure 1 Cycle de vie de SCRUM [Sutherland 1993]

CHAPITRE 2

DÉFINITION DU PROCESSUS DE DÉVELOPPEMENT

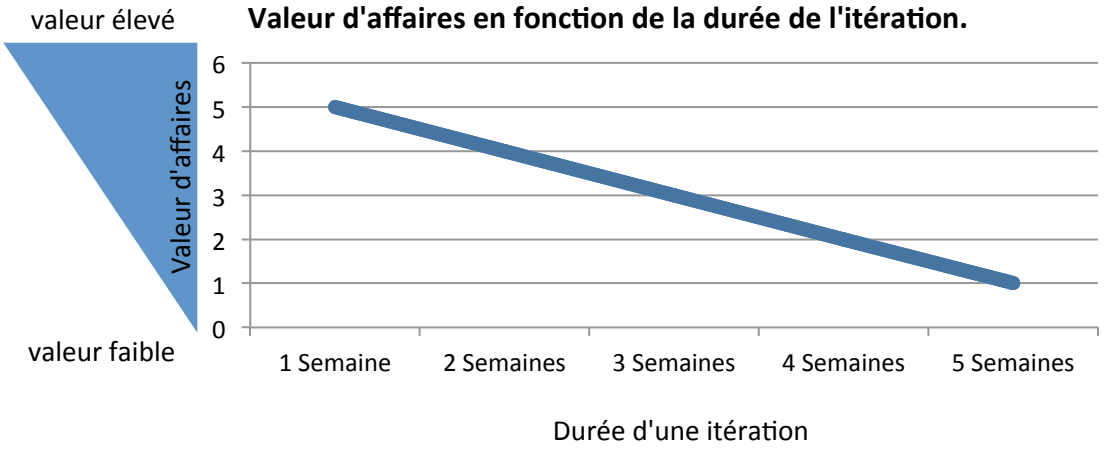
Jusqu'à présent nous avons vu un bref historique et un aperçu de SCRUM dans l'introduction. Ce chapitre définira davantage le processus de développement SCRUM.

2.1 Avant de débuter votre première itération

Avant de démarrer une première itération, la méthodologie SCRUM demande de créer le carnet de produit. Le carnet de produit représente la liste des spécifications logicielles classées selon leur valeur d'affaires respectives.

Avant de poursuivre, définissons ce qu'est la « valeur d'affaires » selon SCRUM. La valeur d'affaires est la valeur attribuée à la spécification qui apportera le plus de retour sur investissement après son développement pour le logiciel. C'est une mesure qui quantifie un ensemble de facteurs déterminés par le client et l'entreprise. Par exemple : un logiciel de traitement de texte qui n'aurait pas la possibilité de créer de nouveaux documents est impensable, c'est pour cette raison que cette spécification a une grande valeur d'affaires et serait la première à être développée comparativement à celle qui permet de changer la couleur du fond de l'écran.

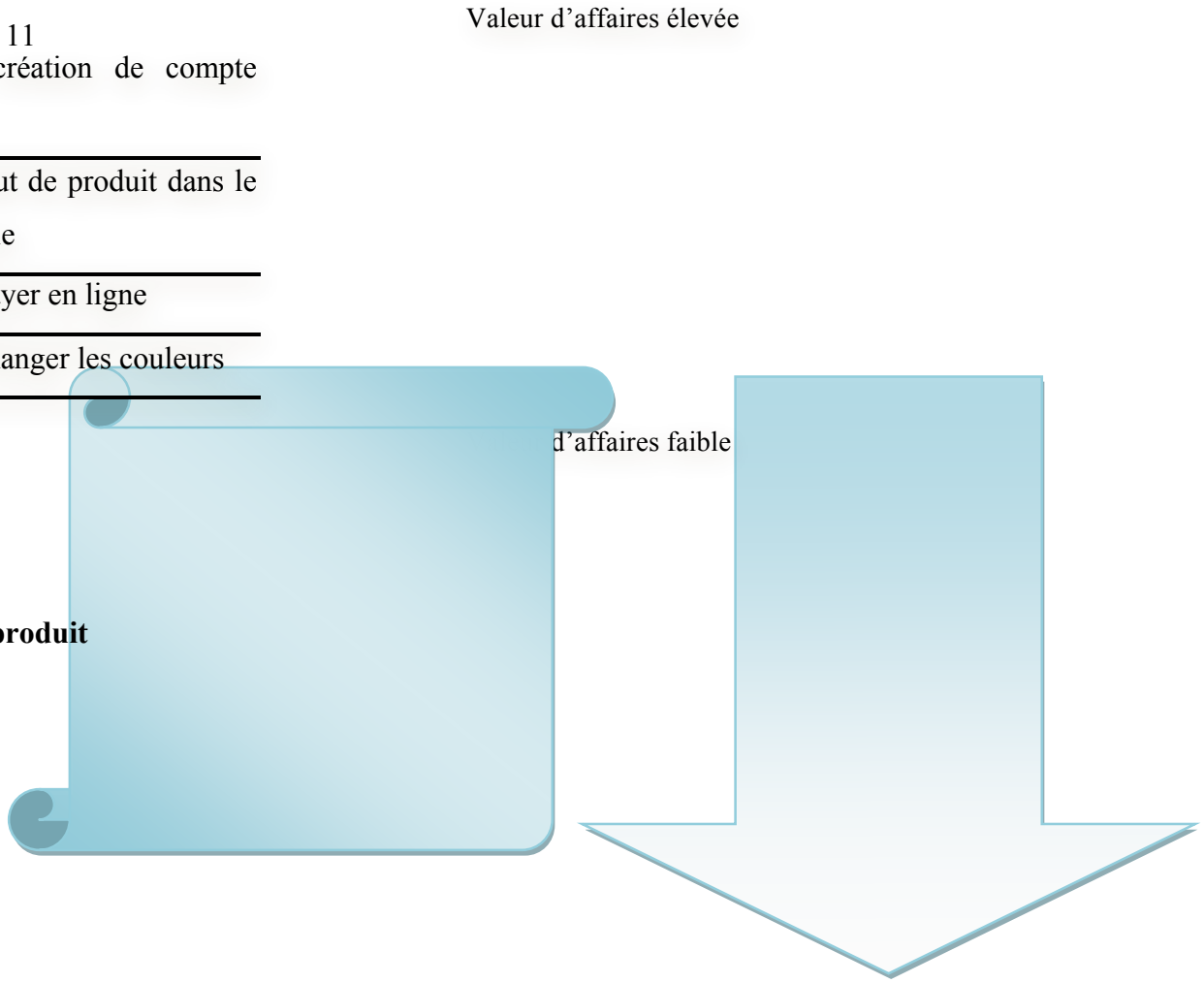
Le processus itératif de la méthodologie permet de rendre un rapport de causalité inversement proportionnel de la valeur d'affaires en vertu de la durée des itérations. La figure 2, démontre cette relation sans toutefois contenir de données statistiques réelles.



La liste des spécifications présentées dans le carnet de produit peut être répertoriée différemment soit :

- Selon les spécifications du client ou domaine d'affaires
- Autrement, sera désignée comme Liste d'histoires (« User Stories[2] »).

[2] Pour plus d'information sur ce sujet qui n'est pas couvert dans ce document consulté le livre « User Stories Applied: For Agile Software Development » de Mike Cohn



gure 3 Carnet de produit

Le gestionnaire de produits devra faire l'exercice de prioriser ses demandes selon des critères respectant la mission et les objectifs de son logiciel. En précisant la valeur d'affaires, il estime l'impact et le retour sur investissement qu'aura chacun des items dans le carnet de produit. Comme les domaines d'affaires varient et qu'ici nous nous adressons spécifiquement à de grandes entreprises, il est recommandé d'utiliser des indicateurs simples qui pourront être utilisés par l'équipe de développement afin d'évaluer la fonctionnalité. Par exemple : bas, moyen, élevé et très élevé.

Les spécifications prioritaires listées dans le haut du carnet de produits doivent être :

- Assez précises pour être estimées par l'équipe;
- Assez petites pour être testées et développées durant une itération.

La priorisation des spécifications du carnet de produit doit être fait attentivement en se rappelant les objectifs de l'implémentation de cette fonctionnalité. Cette codification permettra d'optimiser les ressources de développement afin que le client ait un retour sur investissement le plus tôt possible sur sa décision. Ce processus permet d'établir, entre autres, le plan de livraison du produit.

Les spécifications devront ensuite être estimées, l'ensemble des spécifications n'a pas besoin d'être entièrement estimé pour débuter une itération. L'exercice d'estimation peut être interrompu après l'équivalent de 10 jours de travail de développeur estimé à l'avance. Le fait d'estimer davantage de spécifications pourrait entrer en conflit avec l'effort d'optimisation déployé puisqu'il ne tient pas compte de la pertinence que le client lui accordera en temps voulu, le moment venu. L'idée reste toujours ici de faire profiter le client d'un retour sur investissement le plus tôt possible dans le développement de sa solution.

Le carnet de produit n'est pas un document final.

Le gestionnaire de produit, conserve le contrôle et le pouvoir d'agir en tout temps. Il peut ajouter, modifier ou effacer une spécification dans le carnet de produit à n'importe quel moment du développement. Ce qui lui permet de communiquer ses craintes et les changements que vit l'entreprise qu'il représente. Il conserve le droit de veto sur l'ordre d'exécution et peut ainsi répondre en temps réel aux changements inattendus qui surviendraient dans l'entreprise en offrant des outils tangibles et accessibles rapidement.

2.2 Planification d'itération

Une fois que le carnet de produit est avancé à un taux satisfaisant, on passe à la planification d'une itération.
Il est important de rappeler que les premières spécifications du carnet de produit doivent être :

- Assez, précise pour être estimé par l'équipe;
- Assez petite pour être testé et développé durant une itération.

Lors de cette planification, le gestionnaire de produit présente les spécifications logicielles qu'il désire inclure dans l'itération courante. C'est à ce moment que le gestionnaire de produit peut délibérément inclure une spécification qui serait devenue prioritaire suite à une nouvelle situation urgente; pertes de ressources et de connaissances inattendues qui presseraient l'entreprise à implanter une spécification répondant à ce besoin dans les meilleurs délais.

> ***Nous ne travaillons pas les uns pour les autres, mais les uns avec les autres. -STANLEY C.GAULT***

L'équipe évalue le nombre d'heures auquel ils se sentent en mesure de s'engager. Pour trouver ce nombre, il s'agit de demander à chacun des membres de l'équipe, d'évaluer le temps qu'il croit pouvoir s'engager à travailler sur cette itération par jour.

Questions à poser afin d'estimer l'engagement possible de l'équipe dans une itération.
Combien de temps crois-tu contribuer activement à l'itération par jour? (i.e 6 heures)

Multipliez par

Combien de jours seras-tu présent durant l'itération? (i.e 5 jours)

Exemple :

Pour une itération de 5 jours

Tableau 2 Planification d'itération - Calcul de l'engagement de l'équipe

Membre de l'équipe	Nombre heure par jour	Nombre de jours présent	Total en heures
Martin	5	5	25
Chantale	6	5	30
David	7	5	35
Roger	6	5	30
Steven	6	4	24
		Total	**144**

L'équipe peut donc s'engager à effectuer 144 heures de travail pour l'itération.

Dans le cas où l'équipe utilise la pratique de programmation par les paires alors on multiplie le total des heures par un facteur de maturité. Ce facteur dépend de la maturité de l'équipe et de ses performances précédentes, il varie d'une équipe à l'autre. Une valeur de départ pour une équipe débutante qui n'a jamais utilisé la programmation par les paires serait de 50%. Dans le tableau 2, on pourrait dire que l'équipe peut s'engager à travailler sur des spécifications n'excédant pas 72 heures de travail pour la durée de l'itération.

L'équipe estime ensuite les spécifications du haut du carnet de produits en leur attribuant un nombre d'heures globales en vue de leurs réalisations et s'engage à réaliser un certain nombre de spécifications dans l'itération tout en respectant la capacité de l'équipe soit 72 heures. Cette planification se nomme *évaluation de haut niveau*. Dans notre cas, l'équipe évalue les trois premières spécifications de sorte que la première est estimée globalement à 10 heures, la deuxième à 20 heures et la dernière à 40 heures pour un total de 70 heures de travail pour cette itération.

Le maître SCRUM dirige alors une activité qui a pour objectif de diviser les spécifications en tâches techniques. Les tâches doivent être assez petites pour être réalisées dans les meilleurs délais, idéalement moins de 8 heures. L'évaluation de haut niveau en heures de la spécification est ensuite comparée à la somme des tâches, et ce, pour l'ensemble des spécifications du carnet d'itération.

Tableau 3 Exemple d'un carnet d'itération

Spécification du carnet d'itération	Tâches	Responsable	Estimation en heures
Permettre la création de compte utilisateurs	Configurer la base de données	Chantal	2 heures
	Créer les objets du domaine d'affaire	Martin	6 heures
	Créer l'interface utilisateur	David	2 heures
Permettre l'ajout de produit dans le panier d'épicerie	Persister le panier d'épicerie	Steven	9 heures
	Créer les objets du domaine d'affaire	Chantal	10 heures
	Créer l'interface utilisateur	Martin	1 heure
Permettre de payer en ligne	Créer une transaction sécurisée	David	19 heures
	Créer la transaction avec la compagnie d'authentification de la carte de crédit	Roger	20 heures
	Créer l'interface utilisateur	David	1 heure

Pour poursuivre l'exemple mentionné ci-haut, disons que la somme des tâches à accomplir est de 130 heures. Dans le cas où il y a une différence significative entre la capacité de l'équipe (72 heures) et l'évaluation (70 heures) et le temps estimé pour les tâches (130 heures), une négociation doit être entamée avec le gestionnaire de produit pour rajuster l'engagement de l'équipe face à l'itération. Ici le gestionnaire de produit devra choisir et

reporté une ou plusieurs spécifications afin d'accéder à des résultats réalistes et satisfaisants pour l'itération.
Il est primordial que l'équipe s'engage sur le travail qu'il est possible de réaliser seulement durant une itération. Si vers la fin de l'itération l'équipe n'a plus de travail, il devra transférer une nouvelle spécification de taille égale ou moindre au temps restant dans l'itération courante. De plus, le gestionnaire de produit ne pourra pas changer aucune des spécifications incluses dans l'itération courante sans que l'équipe entière et le maître SCRUM acceptent le changement. La maturité de l'équipe fera fluctuer le facteur initial de 50% selon ses performances autant en développement qu'en planification. Ainsi, plus les évaluations et les engagements se rapprocheront de la réalité plus la vélocité de l'équipe augmentera.

Le fait même de transférer une spécification dans l'itération est un contrat entre l'équipe et le gestionnaire de produit. L'équipe s'engage à développer la spécification telle qu'elle est décrite par le gestionnaire de produit à ce moment. Tandis que le gestionnaire de produit s'engage à ne pas changer le contenu de la spécification qui fait partie de l'itération courante telle qu'il l'avait décrite au début de celle-ci.

Le but de cet exercice est de s'engager sur un nombre de spécifications raisonnables pour remplir le carnet d'itération. Trop de spécifications dans une itération pourraient nous faire manquer à notre engagement et ultimement empêcher la livraison d'un logiciel livrable et fonctionnel à la fin de l'itération.

2.3 Mêlée quotidienne

La mêlée quotidienne est un moment privilégiée durant la journée qui permet à tous les membres de l'équipe de faire le point sur les réalisations de chacun depuis la dernière mêlée quotidienne. Néanmoins, dans les équipes moins expérimentées, le maître SCRUM soutient cette réunion afin de faciliter les discussions.
Avant le début de la réunion, les membres de l'équipe se préparent à répondre brièvement à 3 questions.

1. Quel est le travail effectué depuis la dernière mêlée quotidienne dans le cadre du projet (et hors projet)?
2. Quelles sont les tâches qu'il s'engage à faire d'ici la prochaine mêlée quotidienne dans le cadre du projet (et hors projet)?
3. Quels sont les éléments susceptibles d'empêcher un des membres d'atteindre ses objectifs avant la prochaine mêlée?

La mêlée quotidienne dure, tout au plus, 15 minutes. Cette courte rencontre se fait debout, en cercle, et l'ensemble de l'équipe y participe incluant le gestionnaire de produit. La position debout permet de conserver le dynamisme et l'accent sur l'objectif de la réunion. L'ensemble des participants doit s'abstenir de poser des questions durant cette réunion. Ils doivent plutôt les garder pour après la mêlée quotidienne. À la fin de cette réunion, l'ensemble des membres de l'équipe doit être capable de brosser un statut d'achèvement et les étapes en cours d'exécution par ses collègues jusqu'à la prochaine mêlée.

Le maître SCRUM, en plus de préparer ses réponses pour les trois questions se doit de préparer la réunion. Il s'assure de la liste du statut d'achèvement des tâches débutées, celles ré estimées ou même celles ajoutées. L'ensemble de ces informations lui permet de mettre à jour le graphique d'avancement qu'il peut utiliser dans la mêlée quotidienne. (voir la figure 4 et la figure 5)

Durant la mêlée quotidienne, il doit soutenir son équipe en priorisant et en éliminant les éléments bloqueurs. Le maître SCRUM est aussi en charge de gérer les problèmes interpersonnels dans l'équipe afin de maintenir la productivité et l'efficacité de l'équipe à son plus maximum.

2.4 Le graphique d'avancement

Le graphique d'avancement permet à l'équipe de connaître l'état d'avancement de l'itération. Ce graphique démontre le nombre d'heures restant à effectuer d'ici la fin de l'itération. Quand une tâche est accomplie, son temps est soustrait du temps restant à effectuer. Ce graphique est mis à jour quotidiennement et permet d'évaluer si la performance de la vélocité de l'équipe. Une itération est considérée réussie quand le temps restant est égal à zéro.

Le carnet d'itérations présenté par le graphique d'avancement est établi dans la planification d'itération, mais peut être modifié durant l'itération.

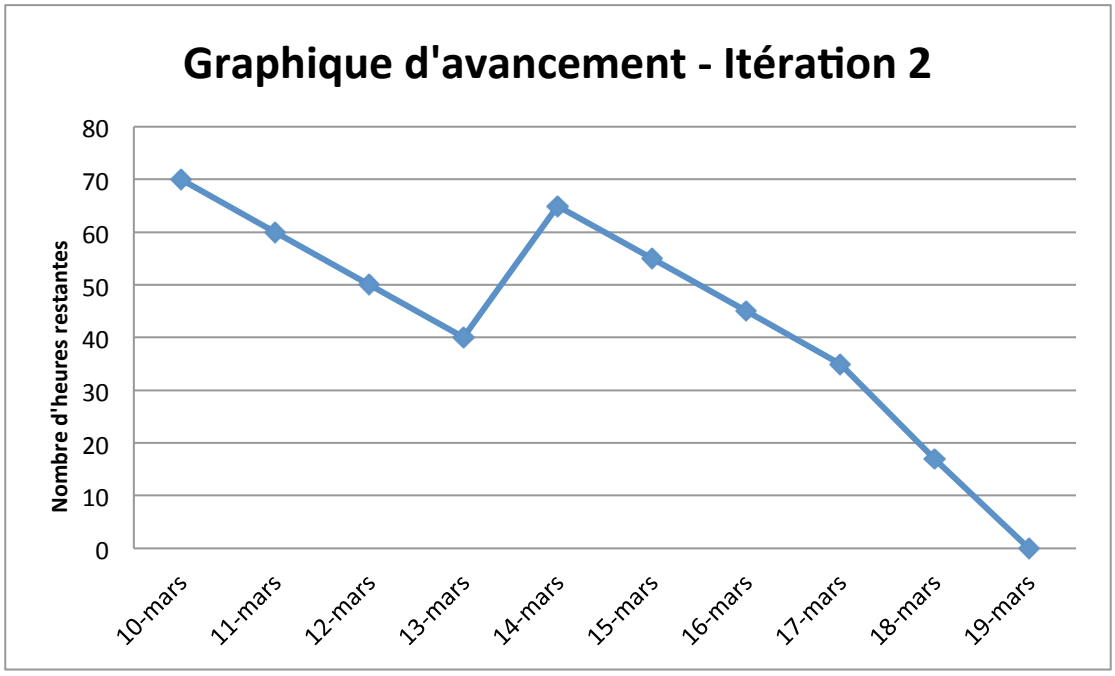


Figure 4 Exemple de graphique d'avancement simple

Plusieurs facteurs peuvent affecter le carnet d'itérations en voici quelques un :

- Sous-évaluation/surévaluation du temps requis par une spécification développée;
- Des tâches supplémentaires pour résoudre des problématiques identifiées par le gestionnaire de produit;
- Une séance de travail avec le gestionnaire de produit pour redéfinir la compréhension générale et l'objectif de l'itération.

Ces changements seront considérés par l'équipe et le gestionnaire de produit, s'ils sont nécessaires, mineurs et peuvent être inclus dans le temps alloués sans compromettre l'itération courante.

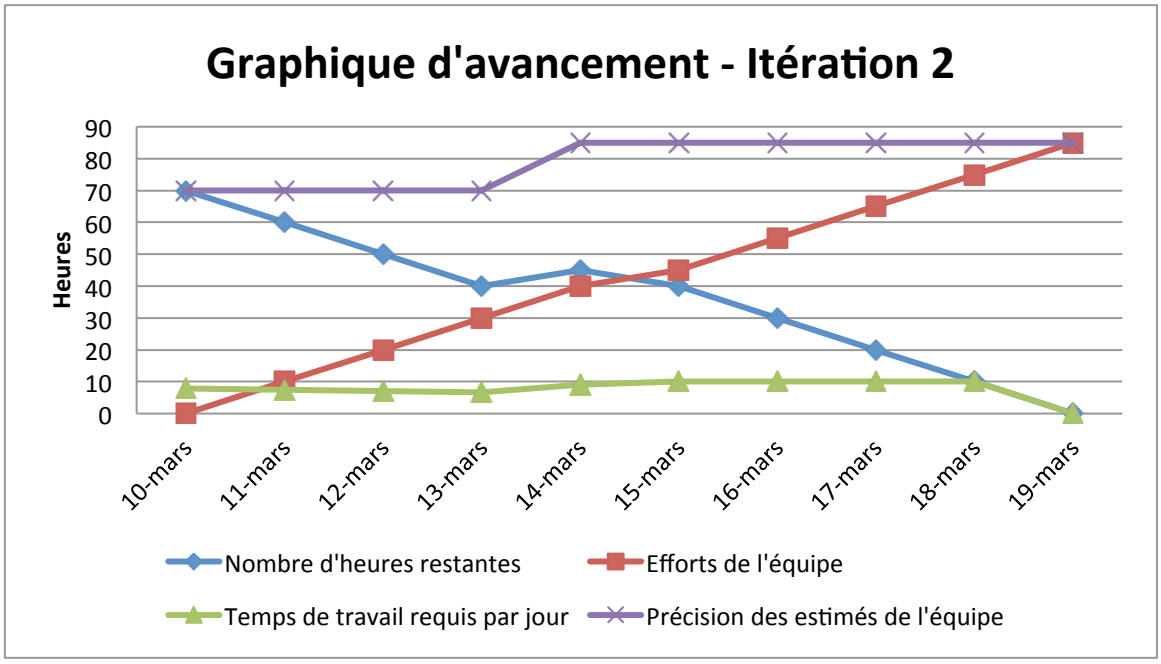


Figure 5 Exemple d'un graphique d'avancement plus complexe

2.5 Rencontre de revue d'itération

La rencontre de revue d'itération est effectuée à la fin de chaque itération. Elle est divisée en deux parties et dure environ 4 heures.

Durant la première moitié de la rencontre, le gestionnaire de produit présente à toutes les personnes concernées les spécifications complétées du carnet d'itération. Par la suite, le gestionnaire de produit entame une revue des prochaines spécifications pour l'itération suivante.

Pour la deuxième partie, le maître SCRUM anime une rétrospective de la dernière itération. Il passe en revue les bons et les moins bons coups de l'équipe, afin d'améliorer les pratiques de celles-ci pour les itérations futures. Lors de cette revue, l'équipe apporte des solutions aux difficultés rencontrées. Cette rencontre permet à chacun des membres de l'équipe de tirer parti du passé sans compromettre la vélocité de l'équipe. Inspecté et adapté, ici l'application des principes SCRUM se fait non seulement au développement, mais aussi aux principales ressources de production; les membres de l'équipe.

L'ensemble de la méthodologie SCRUM est défini dans la figure 6

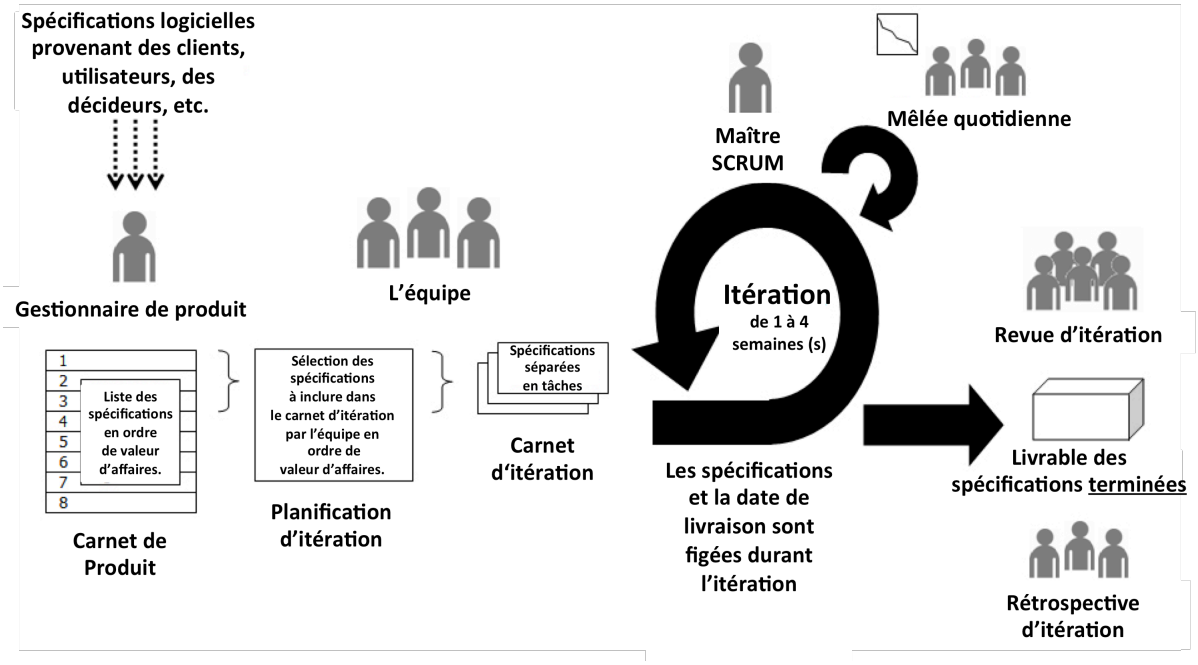


Figure 6 Processus SCRUM [Sutherland 1993]

2.6 SCRUM de SCRUM

Lorsqu'un projet est trop gros pour être réalisé par une seule équipe, il est alors divisé et distribué à plusieurs équipes. Une équipe SCRUM est fonctionnelle lorsqu'elle est limitée à moins de 10 membres. Ainsi, la responsabilité de réalisation est distribuée par autant de membres que composent les équipes incluant un maître SCRUM par équipe.

Le SCRUM de SCRUM consiste à une mêlée quotidienne entre les maîtres SCRUM d'un même projet. Cette activité n'est pas aussi fréquente que la mêlée quotidienne elle peut arriver qu'une fois par semaine ou plus souvent selon les besoins du projet. Lors de cette activité, les maîtres SCRUM doivent répondre à ces questions :

1. Qu'est-ce que votre équipe a fait depuis notre dernière mêlée?
2. Qu'est-ce que votre équipe fera d'ici la prochaine mêlée ?
3. Est-ce que votre équipe a rencontré des problèmes depuis la dernière mêlée ?
4. Allez-vous ajouter des éléments qui pourraient impacter une autre équipe d'ici la prochaine mêlée?

Le SCRUM de SCRUM n'est pas identifié à la figure 6 puisqu'il s'agit d'une répétition de la méthode originale.

CHAPITRE 3

LES PRATIQUES DE LA PROGRAMMATION EXTRÊME

Nous avons vu jusqu'à maintenant, la définition du processus de développement SCRUM dans le chapitre suivant, nous aborderons la programmation extrême. La programmation extrême va de pair avec SCRUM. La combinaison de ces pratiques a été proposée par M. Mike Beedle. Leurs similitudes dans les pratiques ont permis une intégration naturelle de la programmation extrême à celles de SCRUM.

A priori, l'utilisation de ces pratiques peut s'avérer difficile, voir même absurde. Afin d'en tirer profit, il est recommandé d'appliquer intensivement, une pratique à la fois, durant plusieurs semaines avant d'en mesurer les effets et de choisir lesquelles seront maintenues.

Il existe plusieurs pratiques de programmation extrême, sans en faire une liste exhaustive, ce chapitre présente les pratiques les plus courantes.

3.1 Programmation en paires

La programmation en paires est une pratique qui demande à deux programmeurs d'utiliser la même station de travail pour développer une partie du code. Ceci permet à deux programmeurs d'être informés simultanément des changements effectués sur la base de code. De plus, elle confère des avantages, dont une qualité du travail accrue, et permet de surmonter plus facilement les obstacles. Chacun des programmeurs tiendra, en alternance, l'un des rôles suivants:

1. Le conducteur utilisera la station et concentrera son attention sur la précision de ce qu'il inscrit dans la base de code.

2. Tandis que le navigateur effectuera la révision du code en temps réel. Il révise, analyse et apporte de nouvelles idées au conducteur afin d'améliorer la qualité et la pertinence du code. Il s'assure de planifier les prochaines étapes à effectuer pour respecter l'engagement annoncé lors de la dernière mêlée. Ce rôle est essentiel, car il optimise simultanément le temps et la qualité du code.

3.2 Travail énergisé (« Energized work ») et développement à un rythme soutenu

Les programmeurs sont reconnus pour être des gens passionnés par leur travail et à l'affût des nouvelles technologies. Ils sont particulièrement attirés par l'analyse et la résolution de problèmes complexes. Aussi, il n'est pas rare de voir des programmeurs développer et programmer durant leurs temps libres, à toutes heures du jour ou de la nuit.

Ils ne peuvent être plus efficaces que la précision des objectifs à atteindre. Ainsi, pour permettre une motivation soutenue, il est important de leurs offrir des objectifs précis avec une responsabilité collective suffisante.

La pratique du travail énergisé consiste à respecter les règles de vie suivantes :

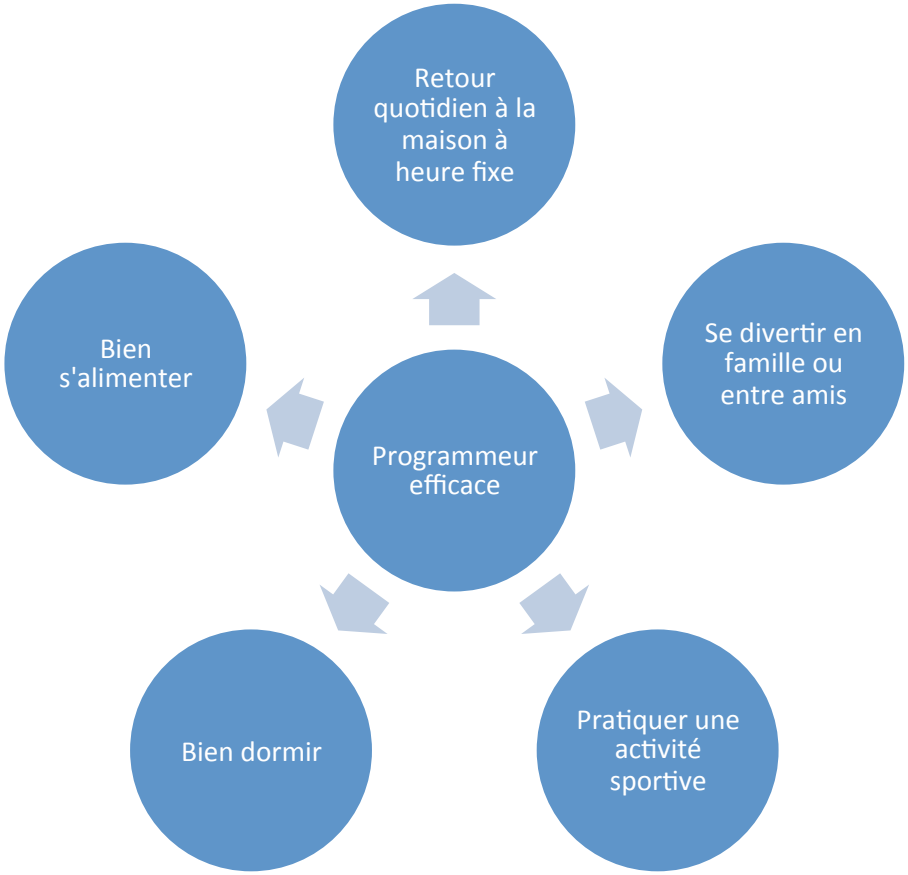


Figure 7 Les règles de la pratique énergisée

3.3 Développement piloté par les tests

Le développement piloté par les tests est une pratique qui demande à l'équipe de développer les tests unitaires de la fonctionnalité voulue. Ces tests sont développés avant même d'inscrire la première ligne de code de la fonctionnalité. Ce qui établit, en quelque sorte, un filet de sécurité. Ensuite, on développe les lignes de code exécutables strictement essentielles pour la fonctionnalité. En réussissant les tests précédemment développés, on s'assure que la fonctionnalité est efficace avant de poursuivre le développement. On termine en développant le code qui respectera les règles d'équipe et les standards commandés par le gestionnaire de produit.

Les tests unitaires se veulent précis et rapides à l'exécution malgré qu'ils doivent tester une fonctionnalité à la fois. Plus les tests sont rapides, plus ils permettent de corriger en temps réels les possibilités de bris, dont ceux liés à l'ajout de nouvelles fonctionnalités sur la base de code existant.

3.4 Intégration continue

Le principe d'intégration continue donne aux développeurs une boucle de rétroaction rapide sur le statut du nouveau code qui vient d'être intégré à la base de code commune. Ce qui permet au développeur d'être avisé rapidement que cette dernière intégration n'a pas impactée une ou plusieurs autres fonctionnalités. Le serveur d'intégration continue, exécute les tests unitaires du départ, autant de fois qu'il y a intégration d'une nouvelle section de code à la base de code commune.

3.5 Réingénierie

La réingénierie logicielle est utilisée pour s'assurer que la base de code soit facile à maintenir et suffisamment extensible pour l'ajout de nouvelles fonctionnalités. Elle consiste à changer l'implémentation d'une fonctionnalité sans toutefois en modifier sa nature. Il est fondamental d'avoir une bonne couverture de tests afin de les exécuter sur la fonctionnalité après la réingénierie. C'est la seule méthode pour s'assurer que la fonctionnalité concorde toujours aux caractéristiques initiales.

3.6 Démonstration d'itération, petite livraison

La démonstration d'une itération, au gestionnaire de produit, est l'étape qui permet d'approuver qu'une spécification soit livrée. Durant cette rencontre, l'équipe révèle l'ensemble des spécifications produites dans l'itération en faisant la démonstration de la fonctionnalité. Avec l'accord du gestionnaire de produit, les spécifications approuvées peuvent faire partie d'une livraison partielle. Ce qui augmente la valeur d'affaires du produit avant la livraison finale et permet au gestionnaire de produit d'accroître sa confiance en l'équipe.

3.7 Standard de code

Un standard de code est un ensemble des caractéristiques utilisé par l'équipe dont le code, les procédures stockées d'une base de données, les commentaires ou tout autre artefact définissant le code d'une application.

Un consensus dans l'équipe doit être atteint afin de définir les caractéristiques du standard de code utilisé. La consistance et le consensus de l'équipe prime sur le nombre de standards de code au détriment de la perfection du ceux-ci; puisqu'ils peuvent être amandés en tout temps.

3.8 Principe du bien collectif

Le principe du bien collectif est plus simple à dire qu'à faire. Il s'agit de considérer l'ensemble des artefacts et décisions d'un projet dans sa globalité comme étant une responsabilité collective. Ainsi, chacun des membres de l'équipe doit s'approprier la responsabilité des artefacts, du code inscrit et de l'application de toutes décisions prises en équipe.

3.9 Conception simple

Ce principe permet de produire plus de code par itération. Pour ce faire, les programmeurs privilégient l'application de la solution la plus simple au détriment de concepts complexes qui possiblement apporteraient une meilleure valeur d'affaires.

En réduisant la complexité de la solution initialement, on réduit, par le fait même la complexité d'une éventuelle correction suivant une réingénierie. Toutefois, durant le développement il sera possible d'appliquer des concepts plus avancés en fonction de l'évolution des besoins.

Il faut se rappeler que l'optimisation du temps aura un impact significatif sur la valeur d'affaires et sera plus apprécié du gestionnaire de produit.

CHAPITRE 4

LES RÔLES

Jusqu'à présent nous avons définis le processus de développement SCRUM et l'importance des principes de la programmation extrême dans la méthodologie SCRUM. Dans ce prochain chapitre nous aborderons les rôles des joueurs clés de la méthodologie SCRUM .

4.1 Le gestionnaire de produit

Le gestionnaire de produit est responsable de la vision du produit demandé par le client. Il est l'intermédiaire traduisant les besoins du client vers l'équipe SCRUM et vice versa. Il détient un droit de veto sur les spécifications développées et tout le succès de l'opération repose sur ses talents de communicateurs ainsi que ses compétences. Par exemple, il est le seul à déterminer si la spécification est bel et bien complétée et si l'équipe passe à la spécification suivante du carnet suivant ou une autre de son choix indépendamment des choix de l'équipe SCRUM.

Important : Le rôle de gestionnaire de produit doit être tenu par une seule personne par équipe SCRUM afin d'éviter des situations ambiguës. Dans le cas où le projet utilise plusieurs équipes, les gestionnaires de produit de toutes les équipes doivent se synchroniser afin d'avoir la même vision pour le produit.

4.2 Le maître SCRUM

Le maître SCRUM soutient l'équipe afin de garantir les meilleures chances de succès des engagements pris par celle-ci. Sa tâche se résume à résoudre les bloqueurs techniques, humains et matériels qui surviendront durant le développement. Le maître SCRUM, malgré son expérience, ne devrait pas être perçu comme un superviseur au sens classique du terme. Tout au plus un excellent technicien qui voit au bon fonctionnement et au respect des normes à suivre. Il est souvent un facteur déterminant dans l'appréciation de la méthode SCRUM auprès de l'équipe qu'il soutient.

Ce rôle peut être tenu par un membre de l'équipe, un consultant ou un spécialiste à la seule exception du gestionnaire de produit. Dans ce dernier cas, la crédibilité et l'engagement du maître SCRUM pourraient en souffrir lors de décisions importantes à prendre.

4.3 L'équipe

L'équipe est responsable de développer le produit logiciel commandé par le gestionnaire de produit. Elle est constituée de 5 à 10 personnes ayant l'expertise nécessaire pour assurer la réalisation du produit. Au-delà de ce nombre, l'équipe perd de son efficacité et de sa flexibilité particulièrement lors des mêlées quotidiennes. Dans un tel cas, il est plutôt conseillé de se subdiviser en plus petites équipes et de fonctionner avec des SCRUM de SCRUM (expliqué en 2.6).

CHAPITRE 5

PROBLÉMATIQUES SCRUM DANS LA GRANDE ENTREPRISE

Maintenant que nous avons présenté tous les connaissances requises pour le projet, dans ce chapitre nous allons traiter de quelques problématiques susceptibles d'être rencontrées lors d'implantation et l'utilisation de cette méthode dans un contexte de grande entreprise.

5.1 Les implications de SCRUM

SCRUM demande aux équipes d'être entièrement autonomes, malheureusement, l'autonomie dans les grandes entreprises est quelquefois alourdie par la bureaucratie et la hiérarchie des patrons de gestions utilisés actuellement.

L'entreprise actuelle divisera ses équipes selon la nature de leurs tâches et imposera une hiérarchie interdépartementale difficile à manœuvrer. SCRUM forment plutôt de petites équipes multidisciplinaires entièrement autonomes responsables collectivement du succès du projet. Ainsi, une équipe composée de chargé d'équipe, d'architectes et de programmeurs, doit fonctionner dans une hiérarchie horizontale (figure 9) plutôt que verticale (figure 8).

Cette nouvelle façon d'organiser le travail exigera des membres de l'équipe une ouverture d'esprit et une saine communication.

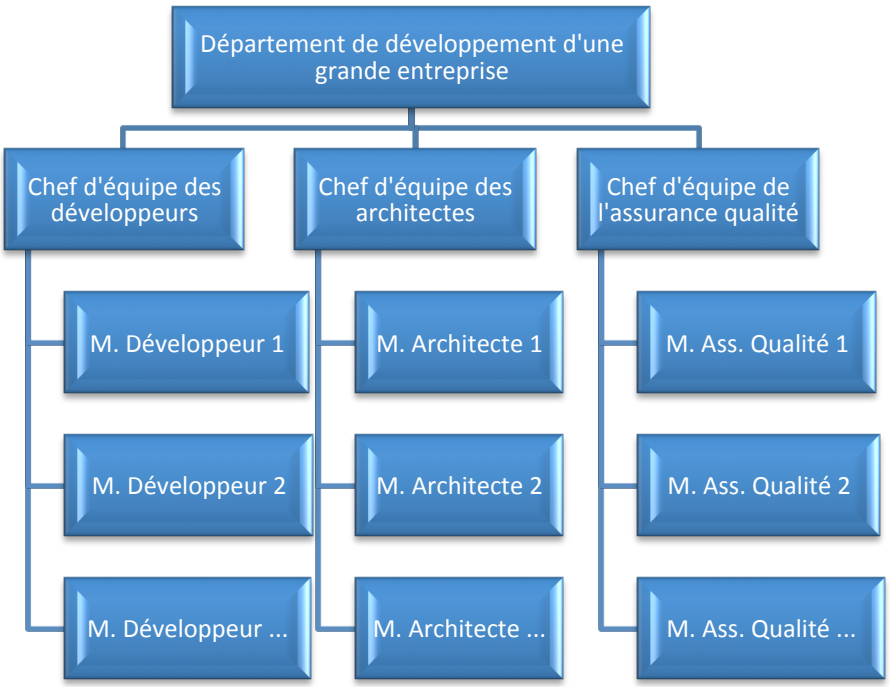


Figure 8 Hiérarchie verticale d'une grande entreprise dans le développement logiciel

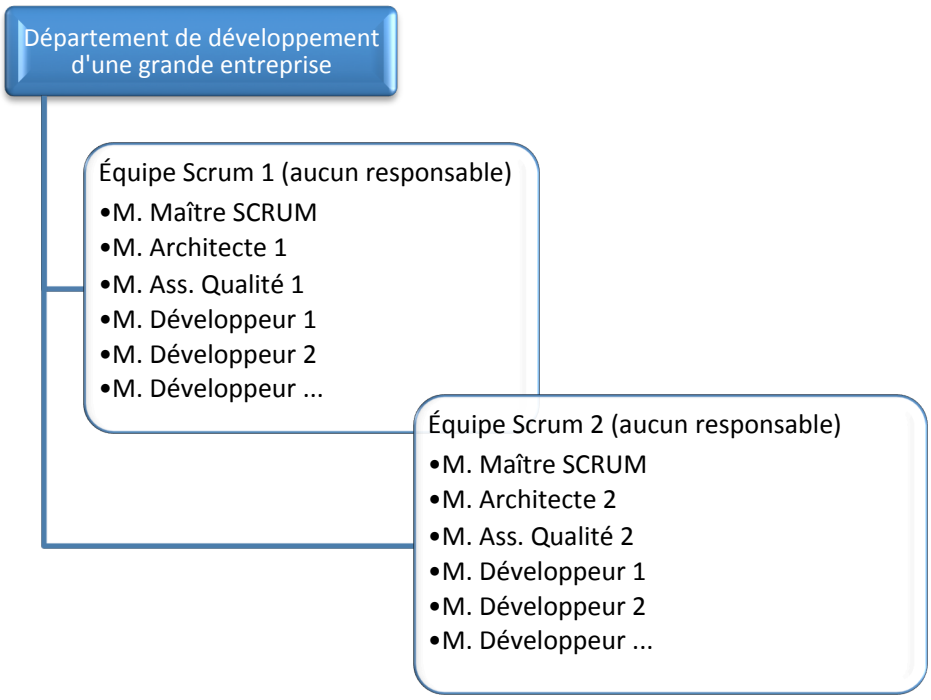


Figure 9 Hiérarchie horizontale d'une grande entreprise dans le développement logiciel

5.2 La gestion des risques liés à l'implémentation

5.2.1 La dette technique

La dette technique représente du travail à faire sur la base de code commune qui n'est pas relié à une spécification. Elle représente souvent une conception initiale plus faible lors du développement initial. Elle peut aussi être causée par l'ajout de spécifications dans la base de code sans avoir fait de réingénierie. Personne n'est à l'abri de la dette technique. En utilisant SCRUM on priorise la norme d'efficacité en solutionnant un problème par sa solution la simple. La révision d'une solution déterminée antérieurement ne se base pas sur sa simplicité initiale. Il est entendu que certaines solutions seront revisitées selon les besoins et les décisions du gestionnaire de produit. On doit se rappeler que le gestionnaire peut modifier ponctuellement l'ordre des spécifications à réaliser ainsi que le détail de celles-ci à chaque début d'itération.

La simplicité de la solution initiale réduira la difficulté et la complexité des modifications à apporter lors d'une révision. Ainsi, la créativité de l'équipe se voit encadrée afin d'augmenter l'efficacité nécessaire à chaque fois.

Pour contrer la dette technique, l'équipe peut se doter de différentes règles et normes qu'ils définiront acceptables. Ces règles deviendront les critères de base utilisée par chacun des membres de l'équipe afin de sélectionner les solutions souhaitables à privilégier lors de dilemmes rencontrés durant le développement de l'itération. En plus de s'assurer du niveau de connaissance suffisant et de la bonne compréhension des spécifications à développer, définir les plateaux techniques de l'interface utilisateur vers l'accès aux données ou intégrer des couches d'architectures ne sont que quelques exemples pour réduire la dette technique.

5.2.2 La motivation passe par la rigueur.

Ce n'est pas un secret à ce point-ci, la méthode SCRUM va à l'encontre des standards existants dans le domaine de développement logiciel et représente un changement de paradigme afin d'atteindre les objectifs. Il est donc crucial de supporter tous les intervenants, les membres de l'équipe, anciens comme nouveaux, durant la période d'essai ou l'adoption de la méthode.

Ainsi, le maître SCRUM et l'équipe doivent s'engager à devenir rigoureux dans l'application de ces nouvelles méthodes et ne pas créer ou encourager des préjugés favorables ou défavorables durant la période d'essai ou le premier mandat. Autrement, les fruits du changement seront moins importants, voir même anéantis. Ce qui mènerait l'équipe vers un échec suivi d'un abandon prématuré de la méthode sans fondement réel.

5.2.3 La résilience au changement

Nous entretenons cette fausse conception qui prétend que si la technologie évolue à un rythme fou que ces artisans sont immunisés contre les effets du changement et ne requerront pas de soutien en ce sens. Ce que nous oublions souvent c'est que la principale composante du changement réside en la capacité de résilience de l'être humain qui est tout sauf naturelle et rassurante chez l'humain.

Les équipes qui intègrent ces principes pour une première fois seront confrontées à des ralentissements dans leurs capacités à livrer un produit. Ils feront face à différents degrés de résistance aux changements qui menaceront certains au point de devoir être rassurés sur leurs compétences et leurs raisons d'être au sein de l'équipe.

Lors de l'implantation, la direction et les intervenants impliqués, devront s'engager à se respecter dans l'adoption de ce changement en réduisant le facteur de maturité afin de se laisser l'espace et le temps nécessaire aux changements dans l'équipe. Ce sera une victoire d'équipe que de voir ce facteur augmenter avec le temps, l'expérience et la satisfaction du client.

5.3 Manifestations à la résistance aux changements.

Les manifestations aux changements sont aussi nombreuses que différentes. Avant d'aborder quelques comportements fréquemment remarqués lors d'intégration de la méthode SCRUM. Il est essentiel de comprendre ce qu'est une équipe afin de remettre en contexte rapidement la façon d'intervenir dans une équipe SCRUM. Ensuite, nous verrons les problématiques observées régulièrement lors de l'introduction de SCRUM auprès de ressources qui utilisaient d'autres paradigmes de développement de logiciel :

Imaginez que vous renversez votre café bouillant sur vous en conduisant votre véhicule en route pour un important rendez-vous d'affaires. À l'achat du café, la caissière vous avait suggéré une tasse de voyage solide que vous avez décliné puisque la vôtre était à la maison. Vous viendrait-il à l'esprit de vous déculpabiliser en accusant votre main, le verre, votre insouciance d'avoir causé cette commotion et de vous imposer un châtiment tel celui de ne plus consommer de café? Pourquoi donc?

Parce que ce n'est ni la faute à votre main, du verre cartonné, à votre compétence à prendre des décisions. C'est l'ensemble de ces facteurs combinés aux facteurs extérieurs qui ont occasionnés le malheureux incident. Vous êtes seulement responsable d'avoir été présent et d'apprendre à gérer les risques inhérents à cette situation pour éviter qu'un tel drame vous fasse perdre à nouveau un temps fou à vous et vos clients.

Au même titre, l'équipe vivra différents incidents dans leurs évolutions. Il arrivera que le café finisse sa course sur l'un des membres de l'équipe. Il faudra alors reculer l'objectif et observer les problématiques dans leurs globalités plutôt que de chercher un coupable absolu. Chaque membre est responsable d'être présent et de participer quotidiennement, pour tout le reste, la gestion des risques de récidive et l'apprentissage prendront une place importante au sein de l'équipe. Ceci fait aussi partie des moyens pour augmenter le facteur de maturité de l'équipe de façon permanente.

5.3.1 La contrepartie de la transparence

SCRUM revisite des comportements acquis autrefois acceptables qui maintenant occasionneront des problématiques qui seront soulevées quotidiennement par d'autres membres de l'équipe durant les mêlées quotidiennes. Il faut comprendre que peu d'entre nous aime se remettre en question devant les autres, surtout s'il y eut préjudice involontaire.

La mêlée quotidienne se veut un processus où règnent la transparence et la communication. SCRUM permet de soulever une problématique qui vise l'optimisation de l'équipe, donc il faut s'assurer de désamorcer la crainte du jugement et encourager l'entraide et l'ingéniosité au sein de l'équipe.

5.3.2 La fréquence

Parce qu'elle est fréquente, la mêlée quotidienne demande une discipline, une capacité à la rétrospective et la planification qui ne sont pas la force de chacun. Avec le temps, l'équipe pourra rapprocher les estimations de leurs capacités à développer. Les encouragements et l'ouverture des membres éviteront un abandon prématuré de la méthode.

5.3.3 Le droit à un bassin de connaissance élargi

SCRUM permet l'apprentissage et donc par défaut accepte l'erreur. Néanmoins, pour éviter que des initiatives soient prises sans les connaissances appropriées, la méthode invite à élargir le bassin des connaissances de l'équipe en intégrant la participation d'un conseiller expert, un ingénieur, un architecte ou tout autre professionnel offrant des pistes de solutions afin de prendre la décision qui convient le mieux lors d'une impasse. L'architecte ou l'ingénieur pourrait, par exemple, analyser l'architecture actuelle du logiciel et proposer des solutions. Malgré les suggestions, la responsabilité des moyens utilisés pour résoudre l'impasse reste une décision d'équipe qui ne peut être reportée sur le gestionnaire de produit.

5.3.4 Délaisser un département pour l'autonomie d'une équipe.

Les grandes entreprises qui désirent implanter SCRUM font souvent face à une organisation qui isole en silos les membres selon leurs tâches respectives. Ainsi, on y retrouve des départements complets d'architectes, de développeurs, de gestionnaires de projet évoluant chacun de leurs côtés. SCRUM, en favorisant de plus petites équipes multidisciplinaires et autonomes, exige des membres une meilleure communication afin de résoudre ensemble les difficultés rencontrées. Par conséquent, il faudra libérer les participants au projet SCRUM d'une partie des responsabilités assumées par ceux-ci dans leurs départements respectifs et planifier des remplacements ou une redistribution des tâches selon la durée du projet.

La méthodologie privilégie que les membres sélectionnés le soient sur la base de leurs ouvertures au changement, leurs désires de rencontrer de nouvelles personnes et leurs talents en communication.

5.4 Analyse des Problématiques particulières vécues en entreprises

Nous venons de voir les problématiques documentées et connues. En voici d'autres qui ont été expérimentées en entreprises lors de l'implantation du projet ou racontée par des spécialistes de la méthodologie SCRUM.

5.4.1 Optimisation anticipée

Les équipes qui sont forcées d'utiliser SCRUM anticipent sans en faire l'expérience que cette méthode ne tient pas compte de leurs besoins entièrement. Ainsi, elles ont tendances à faire l'effort lorsque c'est facile et refuser catégoriquement lorsque le niveau d'efforts demandé passe de facile à un peu moins facile.

Ce comportement démontre le manque d'ouverture de l'équipe à utiliser cette nouvelle méthode. Pour l'éviter autant que possible, le maître SCRUM forcera une utilisation rigoureuse des principes de la méthodologie durant quelques semaines. Après ces semaines, il entamera avec l'équipe une discussion pour vérifier la pertinence de maintenir la méthode ou de l'adapter.

5.4.2 La difficulté du coût de projet lié à SCRUM

Les hautes instances des organisations ont souvent une idée précise des objectifs à atteindre et des budgets accordés pour les atteindre. Malheureusement, SCRUM étant évolutif et itératif, il ne présente pas une idée immuable ni du produit ni ne permet de chiffrer les efforts dans un budget.

En utilisant SCRUM, on s'assure de la stabilité et de l'engagement de l'équipe à développer un produit et de tendre vers l'objectif évolutif du gestionnaire de produit. Il est alors recommandé que la haute direction fasse part de leurs inquiétudes au gestionnaire de produit afin qu'il puisse diriger ses décisions dans le sens désiré.

Il existe une façon d'évaluer en attribuant des pointages de complexité afin d'évaluer un projet en heures. Néanmoins, loin d'être une science exacte, il permet de brosser un portrait qui peut être exécuté annuellement, pour valider l'estimation du pointage initial des spécifications ou dans le but de planifier l'évolution possible du développement. Plus l'évaluation vieillira plus il s'éloignera de la réalité et deviendra désuet.

Dans cette problématique il n'y a pas vraiment de solution les gestionnaires des hautes instances des organisations doivent faire confiance au gestionnaire de produit et se rabattre sur sa capacité à faire ressortir la valeur d'affaires dès les premiers livrables. Le travail de l'équipe sera toujours présenté à chaque itération donc visible à toutes les personnes concernées.

5.4.3 Le maître ou son élève

Lors de l'implantation de SCRUM, il est essentiel de se doter de ressources expérimentées pour accompagner l'équipe dans cette nouvelle aventure. Lorsque le maître SCRUM est novice, il transmet une certaine nervosité et méconnaissance qui affectera sa crédibilité aux yeux des autres membres. Ceci laisse planer un doute que le maître SCRUM ne sait pas ce qu'il fait et donc la méthodologie ne vaut pas la peine d'investir les efforts nécessaires pour la réaliser. Il existe une exception à ce phénomène, si le maître SCRUM a été choisi par l'équipe et que l'enthousiasme du projet a déjà fait son œuvre chez les membres de l'équipe.

5.4.4 Diviser pour régner

Utiliser SCRUM, c'est une méthode avec des règles à suivre comme les autres méthodes. Néanmoins, les compétences et les qualifications de chacun diffèrent et apportent son lot de difficultés dans l'intégration du code à la base de code commune. Par exemple, quelqu'un qui n'a pas suffisamment de compétences pour ajouter son code à la base de code commune et qui aurait le réflexe de remettre cet échéance à plus tard débutera plusieurs autres tâches

qui lui paraîtront plus importantes que celle d'intégrer le code existant à la base de code commune.

Dans un tel cas, le danger se fera sentir soit lors d'une perte massive de données ou lors de l'intégration en soulignant le manque d'uniformité et le risque de difficultés à intégrer la somme du code à la base du code commune. Possiblement que les changements auront une incidence sur d'autres spécifications développées parallèlement par un autre membre de l'équipe. En soumettant les changements fréquemment on facilite la fusion, on augmente l'uniformité du code on permet aux autres membres de l'équipe d'apporter les changements nécessaires sur leurs parties du code récemment produites.

Outre le rappel et la rigueur, c'est l'acquisition de cet apprentissage qui éliminera le problème. Avec le temps et l'expérience, l'équipe pourra diviser pour régner, afin de livrer plus rapidement, elle subdivisera davantage les tâches à accomplir et travaillera ensemble sur la même spécification en accomplissant cette fonctionnalité en deux fois moins de temps que prévue normalement.

5.4.5 Le super héros contre l'équipe

Le super héros de l'équipe est la personne qui connaît toutes les réponses de toutes les questions. Il conserve les informations et les partage peu afin de conserver son statut au sein de l'équipe. Malheureusement, cette personne, menace l'évolution de l'équipe SCRUM parce que les autres membres de l'équipe:

- perçoivent ce dernier comme omniscient et cessent de rechercher leurs propres solutions ou informations
- attendent le héros pour débuter un travail
- cesse d'apprendre à réaliser des tâches complexes qui sont souvent laissées aux soins du héros

- se retrouvent dépourvus lorsque la personne quitte l'équipe avec ses précieuses connaissances
- resteront en adoration de longues heures devant la prise d'otage du clavier par le héros durant les séances de programmation par les paires

La solution à cette situation est de ramener le héros aux services de l'équipe par tous les moyens possibles. Le héros doit devenir plus qu'une source d'information. Le maître SCRUM doit lui faire comprendre sa grande valeur et la nécessité de soutenir les autres membres de l'équipe dans l'acquisition et la pratique des connaissances qu'il détient. Le héros y trouvera son avantage, car il pourra ensuite se consacrer à l'acquisition de nouvelles connaissances pour le bénéfice de l'équipe. D'autres stratégies sont souvent utilisées dont : laisser à l'ensemble de l'équipe la possibilité de créer des spécifications complexes, de forcer le partage de connaissances par des activités de transfert de connaissances ou établir un ordre pour d'élocution lors des réunions en prenant soins de laisser le héros en dernier et radicalement changer le héros d'équipe.

5.4.6 Un manque de qualité ou du zèle?

Dans une équipe qui débute avec SCRUM on risque de retrouver des membres de différent niveau technique. C'est spécifiquement cette différence qui uniformisera la base de données commune en laissant la place aux différentes façons d'inscrire le code. Ce serait une mauvaise utilisation du temps des membres de l'équipe et peu judicieux de réécrire tout le code d'une section qui est fonctionnelle et qui ne requiert pas de changements. Cette base de code est commune à tous les membres de l'équipe et tous les membres de l'équipe devraient être consultés avant une telle opération, aussi justifiée soit-elle. Avec le temps les membres de l'équipe apprendront à mieux communiquer entre eux et ce type de difficulté disparaîtra d'elle-même

5.4.7 Être compétent... et avoir besoin d'aide

La compétence technique dans une équipe SCRUM valeur ajoutée inestimable pour résoudre des problèmes dans une équipe. Néanmoins, le jour où un membre reconnu pour ses compétences par ses pairs demande du soutien pour lui-même résoudre une problématique. Sa demande risque de passer inaperçue par les autres membres de l'équipe.

Au risque de répéter la solution, ici elle réside en la communication. De plus, le maître SCRUM qui entend toutes les demandes de solutions pour les bloqueurs ne peut exclure la demande parce qu'elle proviendrait d'une personne plus compétente. Ce serait un non-sens dans l'utilisation des principes SCRUM.

5.4.8 L'idée c'est de finir ensemble et plus vite que ça!

Une équipe SCRUM a un seul et unique but durant une itération, terminez l'ensemble des spécifications avant la fin de celle-ci. Chacun des membres de l'équipe a son lot de tâches à accomplir, mais tout le monde n'arrive pas en même temps à la fin de son lot. Il est important de se rappeler que si l'un des membres de l'équipe a terminé son lot il peut contribuer à augmenter l'efficacité de l'équipe en aidant ses collègues à compléter le leur.
Avec l'expérience les membres apprendront à estimer plus efficacement le nombre de temps qu'ils peuvent consacrer à une tâche durant une itération.

La mentalité SCRUM d'avoir un but commun et le fait d'être propriétaire à 100% collectivement sont des concepts difficiles pour les équipes à accepter. Pour contrer la perte de temps ou la prise de décision peu judicieuse de démarrer une autre tâche ne faisant pas partie de l'itération courante est de répéter le principe le plus souvent possible. Au départ ce rôle est souvent tenu par le maître SCRUM.

5.4.9 Une équipe SCRUM peut-elle être composée de spécialistes?

Il existe une fausse perception provenant de la méconnaissance de la méthodologie SCRUM, elle consiste à l'exclusion des spécialistes (gestionnaire de base de données, architecte, etc.) comme membres de l'équipe. Rectifions les faits, la méthodologie privilégie toute personne qui est en mesure d'effectuer le travail relié à une spécification. La vérité c'est que l'équipe peut contenir des spécialistes qui devront participer activement à livrer du logiciel et aider leurs coéquipiers en tant qu'égaux et non subalternes. Ils se verront distribués des tâches reliées ou non à leurs champs d'expertise et ils devront les réaliser indépendamment de leurs compétences.

5.4.10 Faire du temps ou « Tant qu'a faire… »?

Une nouvelle équipe SCRUM doit s'habituer à respecter de nouvelles règles afin d'optimiser leurs temps et respecter les échéanciers. Les membres doivent s'exercer à modifier leurs façons de voir les choses. Entre autres, ils doivent remplacer cette expression« Tant qu'à faire! » par « Que puis-je faire pour économiser du temps sur l'itération courante? »

Le changement de mentalité est plus difficile à opérer qu'il en paraît. Il arrive parfois que des membres d'une équipe, dans un élan de professionnalisme ou de générosité, ajoutent du code sur des spécifications qu'ils doivent développer :
« Tant qu'à faire je vais améliorer en ajoutant ceci et cela! ».

Voici ce qui risque de se produire réellement dans l'itération courante avec ce type de comportement des membres de l'équipe :

- L'équipe risque de ne pas atteindre l'échéancier d'une itération, car ils ont concentré leurs efforts à ajouter du code superflu qui n'ajoute aucune valeur ajoutée au produit.

- Ils ont privé l'équipe du soutien nécessaire pour aider d'autres équipes à finaliser leurs lots de codes.
- Si l'intégration du code superflu s'avère à impacter d'autres sections du code essentiel, l'équipe devra prendre le temps d'investiguer et corriger le code impacté inutilement.
- Malgré que les itérations et les échéanciers soient respectés, l'équipe aura manqué à son engagement SCRUM puisqu'elle n'aura pas privilégié l'avancement du projet en travaillant sur les tâches de la spécification requise dans le carnet de produit.

Le gestionnaire de produit pourrait, pour éviter cette situation, préciser plus exactement ce qu'il désire qui soit fait pour la livraison à chacune des spécifications. Il pourrait développer sur la spécification, donner des exemples (si nécessaire), préciser les conditions de succès, donner des aperçus des écrans ou des rapports, etc.

Le maître SCRUM pourra répéter souvent, aux membres tentés de faire du tant qu'à faire de le transformer par que puis-je faire pour aider mon équipe à gagner du temps?

CHAPITRE 6

MOYEN D'IMPLANTATION DE LA MÉTHODOLOGIE

Dans les chapitres précédents nous avons présenté toutes les connaissances requises reliées à SCRUM, nous avons aussi fait état des problématiques liées à l'utilisation de SCRUM. Ce chapitre présentera un moyen d'implanter cette méthodologie au sein d'une équipe d'une grande entreprise. Cette méthode est le résultat de l'implantation réussi de SCRUM au sein d'une équipe de développement en entreprise. Les détails du cas vécu en entreprise seront abordés tout au long de ce chapitre et présentés dans un encadré afin de le différencier de la méthode d'implantation. La méthode présentée a aussi été validée auprès d'autres entreprises qui utilisent la méthodologie SCRUM.

6.1 Implanter SCRUM

Avant de débuter une migration vers la méthodologie SCRUM, mentionnons que les chances de succès sont diminuées quand la direction impose la méthodologie à une équipe existante désintéressée au changement. Il est plutôt conseillé de former les volontaires sur la méthodologie SCRUM et de lancer une période d'essai d'environ 90 jours. Après quoi, l'équipe pourra juger de la pertinence d'adopter cette méthodologie.

Cas vécu en entreprise :
La méthode a été proposée par la direction auprès de l'équipe. L'équipe et toutes les parties concernées du projet ont participé à une formation sur la méthodologie SCRUM. Après quoi, ils ont décidé d'adopter cette méthode de travail.

Pour la grande entreprise, il s'agit souvent de remanier les effectifs existants de façon à libérer des représentants de chacun des départements sur la base du volontariat, offrir la formation et sélectionner le maître SCRUM.

Les étapes pour l'adoption et l'utilisation de la méthodologie SCRUM comprennent des éléments nécessaires tels que le carnet de produit, le carnet d'itération, etc. Mais laisse une grande latitude à l'équipe pour établir les bases de leur fonctionnement d'équipe. Cette section du document traitera d'un cheminement clairement défini pour implanter la méthodologie en mentionnant les étapes à franchir ainsi que leurs définitions possibles.

6.1.1 Sélectionner les intervenants

Une migration vers la méthodologie SCRUM nécessite plusieurs intervenants. Premièrement, un gestionnaire de produit connaissant les caractéristiques et l'utilité du produit final. Il détiendra tout le pouvoir décisionnel concernant le produit et guidera l'équipe dans la conception et la réalisation du produit final. Le gestionnaire de produit peut être tenu par un gestionnaire de projet, un membre de la direction ou un utilisateur du futur produit ou du produit actuel qui sera amélioré. Le volontariat est un moyen de s'assurer d'avoir une personne intéressée et qui sera motivée par l'expérience.

Le gestionnaire de produit passera beaucoup de temps à analyser et détailler le carnet de produits. De plus, il devra répondre aux questions des autres membres de l'équipe, et ce, quotidiennement. Il approuvera aussi régulièrement des spécifications selon les échéanciers. Il est donc fortement suggéré de libérer cette personne de ses tâches régulières.

Cas vécu en entreprise :
Le gestionnaire de produit, devait continuer à exercer ses fonctions habituelles en plus de fournir le travail à l'équipe, une solution loin d'être idéal. Il a quand même été présent et complètement dédié au projet durant toutes les étapes de l'itération 0.

Deuxièmement, le maître SCRUM doit être sélectionné et idéalement sa participation devrait être volontaire. Qu'il soit développeur, architecte ou ingénieur, ce membre clé devra partager

son temps pour effectuer les tâches du maître SCRUM et celle de développeur. Selon la portée du projet, la quantité de travail du maître SCRUM variera, mais pourrait accaparer tout son temps. Dans ce dernier cas, il est possible qu'il ne développe pas le produit, mais assumerait tout de même le rôle. La personne sélectionnée devra aussi être libérée temporairement de ses fonctions régulières pour concentrer son attention sur le travail exigeant de maître SCRUM.

Cas vécu en entreprise :
Le maître SCRUM sélectionné au début du projet ne faisait pas partie de l'équipe, il était un expert provenant d'une firme et était présent pour faciliter l'adoption. Après quelques itérations, un maître SCRUM de l'équipe interne à été formé pour remplacer le maître SCRUM provenant de la firme de consultants.

Troisièmement, l'équipe de développement doit être choisie en fonction de leurs intérêts pour développer le produit et leur ouverture d'esprit (après tout ils devront utiliser la méthodologie SCRUM et l'adopter). Dans la plupart des grandes entreprises, il sera difficile de procéder à la sélection des développeurs puisque les ressources seront assignées au projet selon les règles établies par le département de ressources humaines ou par les membres de la haute direction. Cette réalité des grandes entreprises peut fragiliser l'équipe puisque ces membres seront contraints à participer au projet plutôt que volontaires.

Cas vécu en entreprise :
L'équipe était déjà sélectionnée par la direction. La direction avait pris soin de libérer, pour la durée du projet, les effectifs de l'équipe de leurs tâches régulières (soit quatre jours et demi sur 5 par semaine) afin qu'ils se consacrent principalement au projet.

Pour une équipe débutante avec la méthodologie SCRUM, il est conseillé de faire appel à un expert en la matière. Cet expert, quoiqu'optionnel, assistera l'équipe à traverser toutes les étapes de démarrage d'un nouveau projet SCRUM. Qu'il s'agisse d'une personne contractuelle ou salariée, il sera généralement engagé pour quelque mois; le temps que l'équipe se familiarise avec la méthodologie SCRUM. La durée du mandat dépendra des aptitudes de l'équipe à adopter la méthodologie, il peut arriver que ce processus prenne plus d'une année tellement les membres de l'équipe sont habitués de travailler différemment.

6.1.2 Itération 0

L'itération 0 est le début de l'utilisation de SCRUM. Elle est la fondation des itérations à venir, elle doit donc être soigneusement exécutée. Dans cette section nous aborderons l'ensemble des étapes de l'itération 0 présenté à la figure 10.

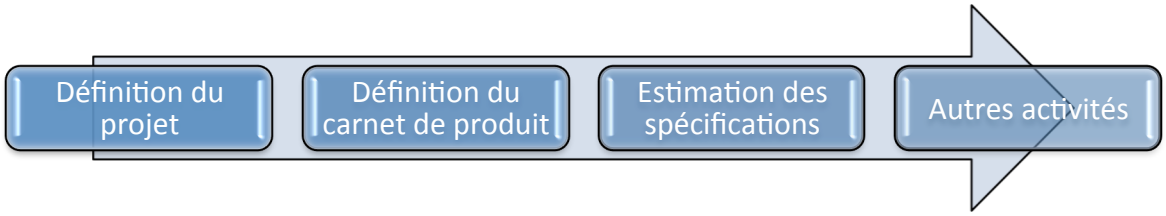


Figure 10 Progression des étapes de l'itération 0

6.1.2.1 Définition du projet

Maintenant que l'équipe est officiellement formée, la première itération demande une préparation particulière pour sa mise en place et celles qui la suivront. La première tâche d'une équipe SCRUM est de définir le projet. À lui seul, le gestionnaire de produit pourrait s'acquitter de cette tâche, néanmoins lorsqu'elle est effectuée en équipe, celle-ci aura une meilleure compréhension du produit et des demandes qui en découleront. La définition du projet consiste à définir les objectifs et les risques potentiels. Cette étape est optionnelle, mais peut aider l'équipe à amorcer leurs collaborations.

Cas vécu en entreprise :
La définition du projet s'est fait en équipe avec le gestionnaire de produit. Il a même clarifié les objectifs du projet. Cette activité a permis à l'équipe de s'approprier la définition afin d'y revenir tout au long des itérations lorsqu'une spécification semblait diverger des objectifs du projet.

6.1.2.2 Le carnet de produit, une étape obligatoire

Ayant une meilleure idée du produit ainsi que de l'objectif du produit à atteindre, l'équipe et le gestionnaire de produit peuvent passer à la création d'un carnet de produit de haut niveau. Encore une fois, cette tâche pourrait être fait exclusivement par le gestionnaire de produit, mais lorsque l'équipe est impliquée on voit un niveau de compréhension du projet supérieur ce qui aide l'équipe à se mobiliser. Une fois la première ébauche du carnet de produit réalisé, le gestionnaire de produit peut découper les spécifications de haut niveau vers des spécifications plus précises.

Cas vécu en entreprise :
L'ensemble de ces activités a été réalisé en groupe permettant à l'équipe de bien saisir le domaine d'affaires et au gestionnaire de produit de bien articuler les spécifications requises.

Un des objectifs de cette activité est de définir l'orientation du carnet de produit. Sans en préciser les fins détails, chaque spécification à développer doit y être inscrite. Certaines spécifications seront d'ailleurs trop complexes pour les inclure dans une seule itération. Le temps venu, elles seront subdivisées en plus petites spécifications. Toutefois, il est nécessaire d'attribuer une priorité d'exécution en considérant la valeur d'affaires pour chacune des spécifications afin de déterminer de quoi sera composée la première itération. Le gestionnaire de produit devra, dans un autre temps, détailler chacune des spécifications

sélectionnées faisant partie de la section supérieure du carnet de produits pour fournir le travail relié à la première itération.

Il est permis de faire appel à plus d'un intervenants externes au projet pour élaborer la définition du carnet de produit dans la mesure qu'ils aident à préciser les spécifications du produit à développer.

Le gestionnaire de produit devra s'habituer à préciser les spécifications du haut du carnet de produit afin de toujours être prêt pour l'itération suivante. Les spécifications pour l'itération suivante devront être précises et sans équivoques.

6.1.2.3 Estimation des spécifications

Le gestionnaire de produit a fourni à l'équipe une liste de spécifications toutefois, il n'a aucune idée de l'effort nécessaire pour les développer. L'équipe n'est pas en mesure de fournir un nombre d'heure au gestionnaire de produit, en plus d'être imprécis cela demanderait trop d'efforts et ne seraient pas significatif. Néanmoins, l'équipe peut fournir une autre unité de mesure au gestionnaire de produit; un pointage qui mesure le niveau de difficulté qu'une spécification prend par rapport à une autre.

On peut voir cette évaluation comme une grandeur de chandail, on évalue le niveau de difficulté d'une spécification en se demandant si c'est un petit, moyen, large ou un extra large. Plutôt que d'utiliser une grandeur de chandail, on suggère d'utiliser la suite de Fibonacci (0,1,2,3,5,8,13,21,34,55...) puisque par définition son incrément est toujours plus complexe que le précédent.

Pour débuter cette activité, nous devons étalonner une spécification. Cet étalon deviendra notre point de référence pour l'évaluation des autres spécifications. Il est important de bien choisir l'étalon puisqu'un étalon trop petit ou trop grand sera difficilement utilisable.

Par exemple : Nous savons que la spécification A est complexe à créer, mais représente-t-elle un 5 ou un 8? Dire qu'elle représente un 8 veut dire qu'elle est au moins 2 fois plus complexe qu'une spécification de 5 points. Pour trouver son pointage, on doit donc la comparer à notre étalon et les spécifications déjà évaluées. Dans notre exemple la spécification A est complexe à créer, mais pas assez pour la qualifié de 8 points, on lui attribue donc la valeur précédente soit 5 points.

Par la suite, l'équipe évalue l'ensemble des spécifications et attribue des valeurs selon le pointage. Ces pointages peuvent faire l'objet de révision en tout temps. À la fin de l'activité, il est conseillé de faire une révision de l'ensemble des spécifications et de les comparer entre elles pour s'assurer que l'on a bien fait l'effort nécessaire pour développer cette spécification.

Cette étape est optionnelle, elle permettra d'établir la grandeur du carnet de produit. De plus, si la capacité de l'équipe est connue alors il sera possible de développer un échéancier réaliste du projet dans la mesure où le carnet de produit ne serait pas modifié au cours du processus de développement.

Cas vécu en entreprise :
Afin de réduire la pression sur l'équipe, les gestionnaires de la direction n'avaient ni de date de fin officiel et ni d'objectifs clairs pour ce projet. Ils désiraient simplement récolter des résultats. L'équipe a donc évalué le haut du carnet de produit et répété l'exercice entre chaque itération.

6.1.2.4 Autres activités

Dans certaines équipes, diverses activités sont réalisées afin d'éviter certaines problématiques d'équipe durant les premières itérations.

On y retrouve entre autres :

- Une définition de ce que veux dire terminer pour l'équipe et le gestionnaire de produit;
- La définition des normes de développement;
- La création des premières règles d'équipe;
- L'estimation de la vélocité de l'équipe (travail pouvant être réalisé durant une itération).

Cas vécu en entreprise :
Dans le cadre du projet, l'ensemble des autres activités mentionnées ont été exécuté afin de fixer les limites des membres de l'équipe. Après quelques itérations, le résultat de ces activités a été mis à jour pour refléter la réalité changeante des équipes SCRUM.

6.1.3 La première itération

La première itération d'une nouvelle équipe SCRUM peut être exigeante et il est donc conseillé de s'assurer de suivre les règles de SCRUM à la lettre.
Par exemple :

- Durant la mêlée quotidienne, les membres de l'équipe doivent répondre aux questions les uns après les autres dans le délai prescrit de 15 minutes;
- Les spécifications doivent être développées simplement, sans initiative du développeur, dans le doute le développeur doit vérifier avec le gestionnaire de produit;
- Développer une dynamique d'équipe qui se fait dans le respect mutuel des membres.

Cas vécu en entreprise :
Durant toute la première année du projet, le maître SCRUM de l'équipe devait rappeler sans cesse à l'équipe de suivre les règles établies par SCRUM et l'équipe elle-même. L'effort du maître SCRUM et sa détermination ont permis de faciliter l'adoption des règles. À ce jour l'entreprise utilise toujours SCRUM, grâce aux efforts soutenus du maître SCRUM et de l'adoption et l'application par l'équipe de la méthode.

6.1.4 Continuer avec SCRUM

Les entreprises qui persistent à utiliser la méthodologie SCRUM remarquent des changements positifs auprès des membres de leurs équipes et du développement de leur projet. Ils remarquent :

- Une augmentation de la vitesse de développement puisque seulement les spécifications nécessaires et ayant une valeur d'affaires sont mises en chantier;
- Un alignement des objectifs individuels et d'entreprises puisque les membres de l'équipe sont au cœur des décisions reliées au projet;
- Un changement de culture dans l'entreprise qui devient une culture gérée par la performance puisque les individus travaillent pour s'acquitter de leurs engagements et livrer le produit selon les échéanciers de chacune des itérations;
- Un support accru de la part de la direction plus consciente de la valeur d'affaires du produit créé par l'équipe découlant des multiples livraisons et de la transparence des mêlées quotidiennes;
- Une meilleure communication et collaboration entre les membres et les parties intéressées découlant des fréquences et de la transparence des mêlées quotidiennes;
- Une évolution professionnelle des membres de l'équipe mobilisée par le décloisonnement et le partage des connaissances des membres entre eux.

Cas vécu en entreprise :
Le niveau de satisfaction des clients et des usagers de l'application était très élevé. Ceci est principalement dû à l'ouverture d'un canal de communication entre les utilisateurs du système et le gestionnaire de produit.

CONCLUSION

SCRUM est un changement de mentalité drastique pour plusieurs développeurs. Il demande de la transparence dans les interventions avec le gestionnaire de produit, de la collaboration entre les membres d'une équipe et demande aux individus une adaptation constante aux nouvelles règles d'équipe.

D'un autre côté la méthodologie SCRUM est flexible, à chaque itération elle consent d'adapter le produit en fonction des nouvelles spécifications. De plus, elle permet aux entreprises de livrer le produit au moment voulu puisqu'à la fin de chaque itération le produit est livrable et fonctionnel. Les membres de l'équipe profitent de l'expérience acquise durant les itérations passées et se mobilise dans un seul but commun.

SCRUM est toutefois mieux adapté pour des petites équipes de moins de 10 personnes ce qui n'est pas toujours la réalité des grandes entreprises. De plus, les membres de l'équipe auront plusieurs rôles à jouer que ce soit architecte, analyste, développeur ou gestionnaire de base de données. Un membre d'une équipe SCRUM a un seul titre et cumule les responsabilités. Cette réalité peut convenir difficilement à certaines entreprises.

Donc même si la première itération est laborieuse, les bénéfices ressentis de la méthodologie SCRUM sont enviables à bien des points de vus. Comme le mentionne son créateur Jeff Sutherland: « SCRUM est ardu, mais c'est beaucoup mieux que ce que l'on faisait avant! »

RECOMMANDATIONS

Les grandes entreprises doivent parfaire leur communication avec leurs clients, c'est ce que la méthodologie SCRUM leur permettra d'améliorer afin qu'elles deviennent des chefs de file dans le développement logiciel. Toutefois, comme ce rapport en fait état, la méthodologie SCRUM demande une véritable ouverture d'esprit autant des clients qui commandent la création des logiciels que des entreprises qui développent les mêmes logiciels. Il s'agit d'un partenariat moderne entre le client et l'entreprise. De plus, les professionnels, artistes et artisans qui développent le logiciel sont appelés à modifier leurs visions et leurs approches des méthodes de travail traditionnelles.

Donc, il est favorable pour une entreprise en développement logiciel d'utiliser la méthodologie SCRUM si elle est préparée à investir les efforts nécessaires à son implantation. Le changement de paradigme se fait doucement, une équipe à la fois dans les projets compatibles (programmation orientée objet seulement). L'adoption et le succès de la méthodologie SCRUM vécu par une équipe mobiliseront de plus en plus d'équipes à participer au changement en appliquant, à leurs tours, la méthodologie à d'autres projets. La connaissance et la maîtrise de la méthodologie se fera progressivement plus présentes tout au long des projets. Les règles d'équipe, les normes et tous les artefacts pourront être transférés d'un projet à l'autre ce qui réduira la durée de l'itération 0 entre les projets.

> ***« Le temps, l'argent et les efforts requis pour faire progresser les membres d'une équipe ne portent pas leurs fruits immédiatement, mais au fil du temps. » -John C. Maxwell***

BIBLIOGRAPHIE

Bernd Bruegge. 2006. «Methodologies: extreme programming and scrum ». In Scientific Literature Digital Library And Search Engine. En ligne. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.135.951&rep=rep1&type=pdf>. Consulté le 21 février 2010.

Don Wells. 2010. «Extreme Programming: A gentle introduction ». En ligne. < http://www.extremeprogramming.org>. Consulté le 21 mars 2010.

Hirotaka Takeuchi et Ikujiro Nonaka. 1986. «New New Product Development Game ». In Harvard Business Review. En ligne. < http://hbr.org/product/new-new-product-development-game/an/86116-PDF-ENG>. Consulté le 26 février 2010.

James Shore, Shane Warden. « The Art of Agile Developement », États-Unis, O'Reilly Media,2008, 409 p.

Jeff Sutherland. 1993-2007. «The Scrum Paper : Nuts, Bolts, and Origins of an Agile Process » En ligne. < http://jeffsutherland.com/scrumpapers.pdf >. Consulté le 26 février 2010.

Jeff Sutherland. 2010. « Scrum Log Jeff Sutherland». En ligne. < http://jeffsutherland.com/>. Consulté le 21 février 2010.

John C. Maxwell. « Les 17 LOIS INFAILLIBLES DU TRAVAIL EN ÉQUIPE», Nashville, Maxwell Motivation Inc., 2001, 264 p.

Jonas Bandi, 2010. «The Risks Of Scrum ». In Closed-Loop. En ligne. < http://blog.jonasbandi.net/2010/01/risk-of-scrum.html >. Consulté le 2 avril 2010.

Kelly Waters. 2010. «How To Implement Scrum in 10 Easy Steps? ». In Agile Software Development. En ligne. < http://www.agile-software-development.com/2007/09/how-to-implement-scrum-in-10-easy-steps_20.html >. Consulté le 25 mars février 2010.

Ken Schwaber. 2007. « The Enterprise and Scrum ». In Books24x7. En ligne. <http://common.books24x7.com/book/id_18638/book.asp>. Consulté le 14 mars 2010

Kent Beck. « Extreme Programming Explained: Embrace Change, Second Edition», États-Unis, Addison-Wesley Professional, 2004, 224 p.

Mountain Goat Software. 2010. «Mike Cohn's Blog - Succeeding with Agile». En ligne. < http://blog.mountaingoatsoftware.com/>. Consulté le 14 mars 2010.

Ralph Jocham. 2009. «The Risks Of Scrum? ». In SlideShare. En ligne. < http://www.slideshare.net/choose/ralph-jocham-the-risks-of-scrum >. Consulté le 2 avril 2010.

Scrum Alliance. 2009. « ScrumAlliance transforming the world of work ». En ligne. < http://www.scrumalliance.org>. Consulté le 26 février 2010.

SCRUM User Group Montréal. 2010. «SCRUM User Group - Montréal ». En ligne. < http://www.scrumusergroup.ca/>. Consulté le 2 avril 2010.

SlideShare. 2006. «What is scrum? ». In SlideShare. En ligne. < http://www.slideshare.net/scrumology/what-is-scrum-1526293 >. Consulté le 21 février 2010.