420-6GD-HY
Entretien de logiciels d'applications
(1-3-3)

Travailler en équipe avec la méthode Agile/Scrum

Arthur Ouellet
arthur@pwel.org
Local : B-2344C
Poste 2731

Cégep de St-Hyacinthe

Table des matières

1 Vue d'ensemble

Sans être encore très répandue, la méthode Agile est de plus en plus utilisée dans l'industrie du développement logiciel, et bien qu'elle puisse au premier abord sembler apporter de la lourdeur, elle se révèle être un très bon outil qui fait sauver du temps, en plus d'apporter de nombreux avantages. Être agile signifie de respecter certaines règles de base et de suivre une logique que l'on peut adapter selon nos besoins.

Historiquement, le travail d'équipe dans un projet de développement logiciel a souvent été centré sur un supérieur assignant des tâches à faire à son équipe de développeurs, qui eux acceptent ses décisions et les réalisent. Les développeurs n'ont pas nécessairement leur mot à dire, ils sont parfois considéré comme de simples exécutants (sans vouloir généraliser).

Avec la méthode de travail Agile, ce sont les développeurs qui sont au centre de l'action. Ils sont responsables de l'avancement du projet tout en restant supervisé. Avec agile, c'est l'équipe qui est au centre de tout. Les décisions se prennent en équipe, les tâches à faire sont déterminées en équipe, elles sont planifiée et développée en équipe et présentées au client (ou à son représentant) par l'équipe. Après tout, ce sont les programmeurs qui connaissent le mieu le produit.

J'ai vécu la transition vers Agile chez Ubisoft dans mon ancienne équipe et une fois bien rodée cette technique nous a permit d'être plus efficace, et surtout de se sentir plus impliqué dans *notre* projet.

Une des plus grosse difficulté dans un projet informatique est la planification du temps et des échéances. On entend souvent parler de gros projets informatiques qui sont retardés, qui coutent plus cher que prévu, etc. Agile n'apporte pas de solution miracle à ces problèmes, mais fournit des outils qui peuvent aider à la planification et au respect des échéances en granularisant la tâche et en aidant à prioriser les bons éléments.

Ce que je propose dans ce petit document sont un ensemble de règles inspirée directement de Agile, mais adaptée au cours et surtout au temps que nous avons (nous avons seulement 6 semaines, et on est loin du 40 heures/semaine). Ce guide est court et présente uniquement les éléments que nous pratiquerons en classe. Si le sujet vous intéresse, il existe une panoplie de livres sur le sujet. Certaines techniques pourtant centrale d'Agile comme le poker planning ont été mises de coté pour le cours.

2 Vocabulaire

Itération Une itération est une unité de temps pendant laquelle vous vous engagez à réaliser certaines tâches. C'est une façon de subdiviser le temps pour qu'il soit plus facilement planifiable et avec moins d'erreurs. Chaque itération doit apporter quelque chose augmentant la valeur du produit et cette valeur ajoutée doit être démontrable en fin d'itération. Voir plus bas pour plus de détails.

Tâche Tout ce qui sera fait sur le projet devra faire partie d'une tâche, incluant la préparation des tests, etc.

Backlog Le backlog est l'ensemble des tâches qui n'ont pas encore été planifiées dans une itération. Dans Bug Genie, cela est représenté comme une tâche qui n'est pas assignée à un jalon.

3 Rôles

3.1 Le client

Normalement en entreprise un projet n'a pas de raisons d'exister ni d'être modifié s'il n'y a pas de clients qui en font la demande. Dans notre cas, le client sera principalement Alain Auclair, coordonnateur du département informatique. Il est la personne la mieux placée pour savoir ce qu'il désire concernant le produit car il doit l'utiliser pour planifier les charges de cours en réunion départementale.

3.2 L'équipe

Comme mentionné précédemment, l'équipe est au centre de l'action. Dans le cours, vous serez tous placé en équipe de quatre.

3.3 Le scrum master

Le scrum master est le titre accordé à un des membres de l'équipe. Cela ne lui confère pas de rang hiérarchique quelquonque, mais le rend responsable de maintenir l'ordre dans les réunion d'équipe et de diriger adroitement son équipe sur le bon chemin. C'est un programmeur au même titre que les autres.

3.4 Le product manager

Le product manager est le représentant du client. C'est donc lui qui s'assure que les bonnes choses sont priorisées avec l'équipe. Je jouerai ce rôle.

4 Façon de travailler

Les façons de travailler présentées ici, bien que basées sur la méthodologie Agile, ont été adaptée au contexte du cours et à ses contraintes. En voici les principales étapes.

4.1 Division du travail en itérations

Le but est diviser le temps en itération. Une itération doit être assez courte sans être trop longue pour permettre d'être facilement planifiable. Un projet est considéré comme une suite d'itération qui se succèdes. Normalement une itération dure quelques semaines (environ entre 3 et 6 semaines), mais dans notre cas chacune d'elles durera une semaine. Nous planifierons donc chaque semaine de façon individuelle. Le fait de diviser le travail à faire en de relativement courtes itérations rend plus facile la prévision des choses à faire et le temps qu'elles prendront que si elles était prévues sur 6 mois par exemple.

Du point de vue du client, une itération est une promesse que certaines fonctionnalités seront développées avant la fin de l'itération. Comme vous commencerez le développement dans la deuxième semaine de cours, il y aura en tout 4 itérations avant la fin du projet. Ce sera à vous de décider ce qui devra être fait dans chacune des itérations pour que le travail soit complètement réalisé à l'échéance du projet.

Après chaque itération, le produit doit pouvoir être considéré *stable* et potentiellement livrable au client s'il en faisait la demande. Toute modification faite depuis le début de l'itération doit avoir été soumise (dans subversion) et les tests doivent avoir été fait. Le suivi des tâches doit lui aussi être à jour dans Bug Genie.

Dans votre cas, une itération commence en début de cours le lundi, et se termine en début de cours le lundi de la semaine suivante.

4.2 Meeting de début d'itération

Communément appelé *sprint planning*, ce meeting est fait en équipe et sert à déterminer toutes les tâches qui seront faites par l'équipe en cours d'itération. Les tâches doivent être entrées dans le logiciel de gestion de projet présenté en classe et représente un engagement avec le client à partir de ce moment. Les tâches se voient assigner une priorité et seront réalisées de la plus prioritaire à la moins prioritaire pendant l'itération pour s'assurer que s'il venait à manquer de temps, le plus important sera fait.

Cette étape doit être faite en gardant le produit final en tête. Suite à cela, le développement commence, et chacun des membre de l'équipe prend une des tâche en commençant par la plus prioritaire. Aussitôt qu'elle est terminée, il en prend une autre parmis les plus prioritaires restantes, et ainsi de suite jusqu'à ce qu'il n'en reste plus et/ou que l'itération finisse.

4.3 Meeting de fin d'itération

Aussi appelé *sprint review*, ce court meeting est le moment de faire le point sur l'itération qui vient de se finir et discuter avec l'équipe de ce qui a bien fonctionné et des problèmes rencontrés. Ce meeting sera fait en début de cours le lundi de façon à clore l'itération passée avant d'en commencer une nouvelle.

Tel que mentionné sur le calendrier, je demande que chacun des membre de l'équipe me remettre un document expliquant brièvement ce qui a bien été et mal été pour vous dans l'itération, les tâches que vous avez accomplie (chacun met uniquement les tâches qu'il a lui-même accomplies) et une justification si certaines tâches n'ont plus être réalisées.

4.4 Scrum meeting

Ce meeting survient normalement vers le début de journée dans un contexte réel de travail, dans notre cas nous le feront environ 15 minutes après le début du cours. Ce meeting se veut court, au maximum environ 10 minutes. C'est le moment idéal pour faire le point sur l'avancement du travail et permet à tout les membres de l'équipe de savoir ce que les autres font. Nous aurons donc 3 meeting par semaine, un au début de chaque cours.

Comme il doit être court et interactif, il se fait debout, tout les participant sont placés en cercles. Chacun parle à tour de rôle et informe ses coéquipiers des trois choses suivantes :

1. Sur quoi j'ai travaillé lors de la dernière journée
2. Quels problèmes j'ai rencontrés et/ou quelle information utile je pourrais transmettre à mon équipe
3. Sur quoi je prévoit travailler aujourd'hui

En tant que représentant du client (product manager), j'y assisterai dans le but de voir l'avancement du projet. Chaque équipe devra donc attendre son tour pour que je puisse me joindre a leur meeting en travaillant sur le projet en attendant.

4.5 Maintenance du backlog

Norlement, lorsque le besoin se fait sentir, l'équipe peut se réunir pour planifier certaines fonctionnalités qui devront être implémentées plus tard (dans une prochaine itération). Comme notre projet est relativement court, ce serait une perte de temps de le faire. Par contre, vous êtes encouragés à entrer des nouvelles tâche à mesure que vous avez des idées d'améliorations ou de modifications. N'ajoutez jamais ces tâches dans l'itération courante.

5 Gestion des imprévus

Il peut arriver que certains évènements se produisent en cours d'itération qui n'ont pas été planifié et qui ne sont donc pas comptés dans le temps prévu de travail. En voici quelques examples avec des piste de solutions pour leur faire face.

5.1 Les bugs

Lorsqu'un bug est trouvé ne touchant pas directement une tâche en cours, il est simplement ajouté dans le backlog et sera planifié dans une itération ultérieure.

5.2 Changement de spécifications du produit

Si une modification est demandée par le client en plein millieu d'une itération et qu'elle ne touche pas les tâches en cours, lui indiquer que ce sera planifié dans une prochaine itération. Si elle affecte l'itération en cours, s'ajuster en gardant en tête qu'il est normal que les échéances pour l'itération ne soit pas remplie (l'itération se termine toujours à la même date, mais certaines tâches seront peut-être restantes.

5.3 Imprévu technique

Au début, il est normal de mal prévoir le temps que peut prendre certaines tâches, ou bien les défis qu'elles représentent. Aviser le product manager des causes dans le document remis par chacun des coéquipier en fin d'itération (tel que mentionné plus haut)

5.4 Autre

Soyez agile ! Il faut savoir improviser, n'oublions pas que la méthode Agile n'est qu'un ensemble de règles directrices que l'on peux ajuster à une situation particulière. En cas de doutes, en parler avec le product manager.