SCRUM
(extrait de Bi-Méthodes)

3.5 SCRUM

3.5.1 Les origines de Scrum

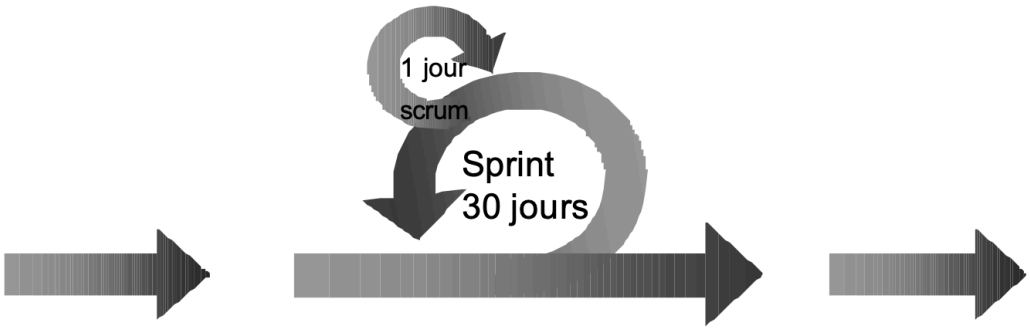
La méthode Scrum est née de la coopération de deux sociétés : Advanced Development Methods (ADM) et WMARK Software (éditeur d'environnement de développement orienté objet). Ces deux compagnies déploraient le man- que d'adéquation des anciennes méthodologies avec la programmation orientée objet et les architectures à base de composants. Scrum est le résul- tat de la volonté de mettre en place des processus empiriques, contrôlés par des mesures quantitatives.

Note : le terme Scrum est emprunté au rugby et signifie "mêlée". La méthode porte ce nom car dans un processus Scrum se tient chaque jour une réunion informelle, appelée Scrum, pour faire le point sur ce qui a été fait, ce qui va être fait le lendemain et ce qu'il reste à faire pour la suite.

3.5.2 Le processus Scrum : vue générale

Le cycle de vie d'un projet Scrum peut être divisé en trois parties :

- La phase d'initiation est une phase linéaire aux processus explicitement connus, et dont les inputs et outputs sont bien déterminés.
- Le "sprint" de développement est un processus empirique : beaucoup des processus de la phase de sprint sont inconnus et non identifiés. Ce n'est pas pour autant le chaos car des contrôles, incluant notamment la prise en compte du risque, encadrent chaque itération de la phase de sprint pour ne pas tomber dans le chaos, tout en préservant la flexibilité. Le projet est ouvert sur son environnement jusqu'à la phase de clôture. Les livrables peuvent être modifiés à tout moment pendant les deux premières phases.
- La phase de clôture est aussi une phase linéaire dont les processus sont clairement définis.



Planning & Architecture
- phases linéaires
- processus bien définis
- backlog
- conceptualisation
- analyse - conception
- architecture système

Sprints de développement
- phase non linéaire
- processus empirique
- flexibilité
- accueil du changement
- contrôles permanents pour éviter le chaos, notamment management du risque

Clôture
- préparation de la release
- documentation finale
- derniers tests
- livraison

La phase d'initiation comporte tout d'abord une activité de planning. Dans le cas du développement d'un nouveau système, il s'agit de réaliser la concep- tualisation et l'analyse du système. La phase de planning doit aboutir à la mise en place d'un "backlog", c'est à dire une liste de tâches restant à effec- tuer, à la définition de la date de livraison et des fonctionnalités à livrer ainsi qu'à la formation de l'équipe de développement. C'est à l'occasion de cette phase que les aspects de management du risque et des coûts sont traités.

Cette phase est en général relativement courte et ne met pas plus de 10 personnes à contribution. Elle requiert pourtant des compétences variées, allant de la connaissance du marché, des utilisateurs potentiels, à des res- sources techniques issues d'autres développement similaires ou utilisant les mêmes technologies.

Un travail sur l'architecture permet en outre de déterminer de quelle manière les éléments du backlog seront implémentés. Il s'agit en fait d'une part de déterminer l'architecture du système et d'autre part de débuter la conception détaillée.

Phase de "sprints"

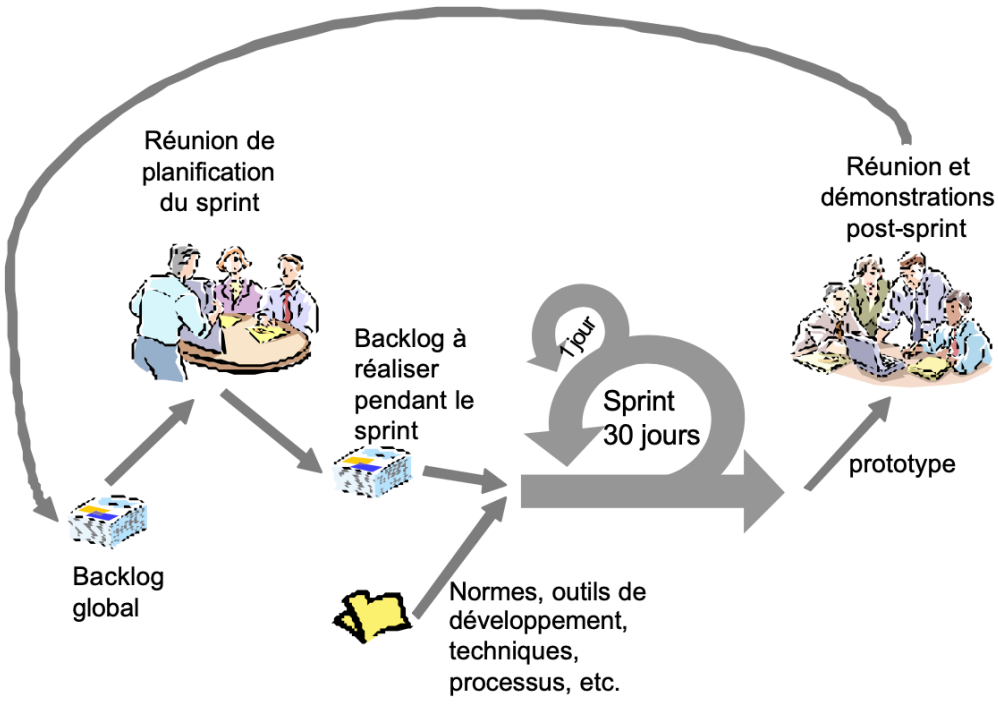
C'est la phase de développement itératif. Elle sera décrite dans la partie suivante.

Phase de clôture

Quand l'équipe de management estime que les variables de temps, exigen- ces, coûts et qualités concordent pour permettre le passage à une nouvelle release, ils déclarent la clôture de la release en cours. Cette phase de clôture prépare le produit pour une livraison : intégration, tests systèmes, documen- tation utilisateur, préparation de supports de formation, préparation de sup- ports marketing, etc.

3.5.3 Les "sprints"

Le développement à proprement parler se fait au cours de ce que la méthode Scrum appelle des "sprints". Les sprints sont guidés par des listes de tâches à réaliser, classées par ordre de priorité : les "backlogs". Un sprint est une période de 30 jours environ durant laquelle l'équipe est isolée de toute in- fluence extérieure, c'est à dire qu'aucun travail supplémentaire ne peut être ajouté à un sprint en cours, ni aucun modification du backlog ne peut être faite pour le sprint en cours. L'équipe est libre de déterminer le meilleur moyen de développer les fonctionnalités qui ont été retenues pour former la liste des tâches à effectuer pendant le sprint. Chaque jour, l'équipe entière participe à une réunion Scrum pour partager les connaissances acquises, faire un point sur l'avancement et donner au management une certaine visibi- lité sur la progression du projet.



Backlog global

Le backlog global regroupe toutes les caractéristiques, fonctionnalités, tech- nologies, améliorations et corrections qui constitueront la future release de l'application. Au cours d'un processus Scrum, le backlog n'est pas statique mais il évolue au fur et à mesure que l'environnement évolue. Les managers font constamment évoluer les spécifications pour s'assurer que le produit sera le plus approprié possible.

Tous les éléments qui constituent le backlog doivent être classés par ordre de priorité décroissante.

Chaque élément du backlog est estimé en termes de coût de développe- ment, temps de développement (en jours), risque, complexité, etc. Les élé- ments du backlog peuvent avoir diverses origines et être dictés par des be- soins du marché, des contraintes de marketing, des contraintes techniques, etc.

Sprint Backlog

Après chaque sprint, le suivant se prépare au cours d'une réunion de planifi- cation réunissant toute l'équipe. A cette occasion, développeurs et managers réunis définissent le backlog qui devra être réalisé au cours de l'itération suivante. Sur la base du backlog global qui aura été préalablement mis à jour et reclassé par ordre de priorité, l'équipe sélectionne les tâches qui seront accomplies au cours des 30 jours suivants. Si plusieurs équipes travaillent en parallèle, un backlog est établi pour chacune d'entre elles de manière cohé- rente. La sélection des tâches à accomplir et des fonctionnalités à dévelop- per se fait d'abord en fonction de l'ordre de priorité mais ce dernier peut être modifié pour des raisons de développement (par exemple, plusieurs tâches préalablement disjointes peuvent être rapprochées si leur développement est facilité par la mise en place d'une interface commune ou d'un module com- mun). Tout au long du sprint, chaque membre de l'équipe a la responsabilité de mettre à jour les éléments du backlog dont il est le propriétaire, ceci dans un souci de suivi permanent de l'avancement du projet.

Sprint

Un sprint, c'est à dire une itération de 30 jours de la phase de développe- ment, est lui aussi subdivisé en itérations d'une journée terminées par une réunion quotidienne qui porte le nom de scrum (mêlée).

Au cours d'un sprint, l'équipe ne peut recevoir aucun changement du backlog venu de l'extérieur. Toutes les modifications qui interviennent ne prendront effet qu'à l'itération suivante. Cette disposition a pour but de protéger le tra- vail de développement fourni par l'équipe du chaos qui règne à l'extérieur et de permettre aux développeurs de se focaliser sur un certain ensemble de fonctionnalités fixes à mettre en place. Par contre, les membres de l'équipe peuvent modifier le backlog pour atteindre l'objectif du sprint fixé au départ en termes de fonctionnalités.

Tout au long du sprint, le travail réalisé est mesuré et contrôlé de manière empirique. Chacun doit se focaliser sur les trois mêmes points : qu'est ce qui a été fait pendant la journée ? que reste-il à faire ? quels sont les obstacles qui gênent l'avancement du projet ? L'avancement du projet est évalué à travers quatre variables : coût, planning, fonctionnalités et qualité. Ces quatre variables ne sont pas indépendantes, c'est à dire qu'on ne peut pas fixer arbitrairement une valeur pour chacune d'entre elles. Grâce à cette surveil- lance de l'avancement, les déviations sont repérées assez tôt pour pouvoir être corrigées. Si le sprint devait demander plus de temps que prévu, on peut choisir soit d'augmenter le temps imparti, soit diminuer les fonctionnalités, soit de diminuer la qualité. En général, le coût ne peut pas subir d'augmenta- tion pendant un sprint.

La productivité est améliorée par la petite taille des équipes (4 à 7 personnes incluant concepteurs, architectes, développeurs, responsables qualité) et par l'isolation des équipes pendant le sprint (espace de travail isolé, mais aussi perturbations extérieures éliminées). De plus, le travail collaboratif permet le partage des connaissances et du savoir-faire.

Le développement se fait de manière itérative : à l'intérieur de chaque sprint se retrouvent les activités de planification, architecture, conception, dévelop- pement et implémentation.

Chaque sprint produit un livrable visible et fonctionnel (voir démonstration post-sprint)

Scrum (réunion quotidienne)

Au cours d'un sprint, une réunion quotidienne informelle de toute l'équipe a lieu, si possible toujours à la même heure et au même endroit. Ne peuvent intervenir que les personnes qui travaillent effectivement sur le développe- ment. Les extérieurs y sont invités pour suivre l'avancement du projet mais n'interviennent pas, toujours dans ce souci de ne pas perturber le travail de l'équipe par le chaos venu de l'extérieur.

Cette réunion n'est pas sensée durer plus de 30 minutes et consiste en un tour de table où chacun doit répondre aux trois questions : qu'est ce qui a été fait depuis la veille ? que reste-il à faire ? comment le faire ?

Le manager ou Scrum Master a pour rôle de prendre les décisions que l'équipe est incapable de prendre par elle même et s'engage à apporter une solution à tout ce qui entrave le développement du projet, soit par ses pro- pres connaissances, soit en faisant appel à des ressources extérieures.

L'utilité de telles réunions est assez évidente : elle permet d'une part, une synchronisation quotidienne de l'équipe et d'autre part, le partage des connaissances.

Réunion et démonstration post-sprint

La réunion post-sprint est l'occasion d'une part de montrer au client ce qui a été développé pendant les 30 jours précédents et d'autre part de confronter les résultats du travail de l'équipe avec la complexité et le chaos de l'environ- nement dans lequel l'application sera utilisée.

Cette réunion se déroule de manière informelle. L'équipe y présente ce qui a été fait, la manière dont elle a procédé pour atteindre les objectifs fixés et fait une démonstration de l'application, en l'état où elle est à la fin du sprint.

En se basant sur le backlog, c'est à dire sur ce qu'il reste à faire, sur un nou- vel état des lieux du marché et des besoins des utilisateurs et sur le proto- type issu du sprint précédent, cette réunion aboutit à la décision de publier ou non une release du logiciel et à la formulation des objectifs à atteindre à la fin de l'itération suivante.

3.5.4 Forces et faiblesses

Scrum présente des avantages certains par rapport aux méthodes plus tradi- tionnelles : cette méthode permet effectivement de répondre au changement avec une certaine souplesse et permet de modifier le projet et les livrables à tout moment, de façon à livrer la release la plus appropriée possible.

D'autre part, bien que flexible, cette méthode se veut rigoureuse puisque le contrôle des processus est omniprésent. En effet, le projet est constamment

managé à la lumière des quatre variables précédemment citées que sont délais, coûts, fonctionnalités, et qualité.

Du point de vue de la progression, Scrum accorde une place importante au partage des connaissances au sein d'une équipe, notamment au cours des réunions quotidiennes. Ceci est bien entendu un facteur qui améliore la pro- ductivité et la capitalisation des savoirs.

Enfin, la pratique qui consiste à isoler l'équipe du chaos extérieur pendant le sprint autorise les développeurs à se concentrer pleinement sur les fonction- nalités qu'ils ont à développer, sans être perturbés par des changements incessants des spécifications dictés par le client ou les exigences du marché. Ainsi, les développeurs peuvent conserver toute leur énergie à concevoir les solutions les plus astucieuses pour atteindre leurs objectifs.

On peut toutefois se demander si cette pratique n'est pas en contrepartie une sorte de frein à l'agilité. En effet, si on imagine qu'une modification des spéci- fications est ordonnée par le client en cours de sprint, est-il nécessaire de continuer à développer en ignorant cela jusqu'au bout du sprint ? Prendre en compte les modifications au fur et à mesure de leur arrivée conduit, certes, à une baisse de productivité, mais d'un autre côté, limite la production de code voué à la destruction. Ce point particulier mérite donc d'être examiné attenti- vement avant de mettre en place cette méthode.