Scrum Shu Ha Ri

Par Christophe Addinquy

Aujourd'hui, agilité est souvent synonyme de Scrum. Cette popularité a engendré une mode : celle de mépriser cette approche à laquelle on attribue tous les maux. J'aurais beaucoup à dire de ces « analyses ». Mais je vais plutôt faire différemment. Je voudrais vous faire découvrir Scrum autrement, pour vous montrer ce qu'il recèle, vous aider à le comprendre et à en faire bon usage

Etre agile

Nouveaux venus à l'agilité, on entend souvent des organisations clamant leur décision courageuse de « faire de l'agile ». Décisions souvent sincères (mais pas toujours), mais surtout pas forcément abordées de a bonne manière. L'agilité n'est pas une recette miracle. Ce n'est d'ailleurs pas du tout une recette : on ne fait pas de l'agile comme on fait de la pâte à crêpes !

L'agilité est avant tout une culture et un état d'esprit : on EST agile.

Une rupture… pour une rupture de contexte

En quoi l'agilité est-elle une rupture, et surtout par rapport à quoi ? Eh bien, il s'agit avant tout d'une rupture sur la façon d'aborder le processus de développement logiciel. Les méthodes classiques, qu'elles soient en cascades ou itératives s'efforcent toujours de mettre en adéquation :

- Les rôles : qui participe au projet et avec quelle expertise.
- Les artefacts : ce qui est produit durant le projet.
- Les tâches : comment sont produit les artéfacts via les rôles.
- Optionnellement, on peut aussi définir des workflows, c'est à dire l'agencement de tout cela au cours du temps.

Unified Process[1], qui est pourtant probablement l'un des moins prescriptifs de ces processus l'illustre parfaitement[2].

[1] [JACOBSON+99], p. 18
[2] Voir également [KRUCHTEN04] p. 36

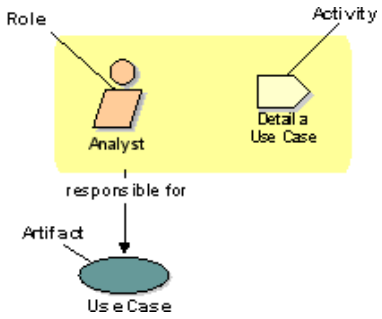


Figure 1: rôles, artefacts et tâches avec RUP

Les variations de ces modèles tournent toujours autour d'une même idée, le modèle prescriptif, héritier plus ou moins direct du management scientifique du travail de Taylor[3] ! Ce qui est bien avec le Taylorisme, c'est que l'on est à peu près certain de faire peur au gens. Il convient pourtant d'examiner son champs d'exploitation. On peut pour cela s'appuyer sur le modèle Cynefin[4]. Cela ne suffirait pas, mais nous laisserons de côté les autres aspects pour notre propos.

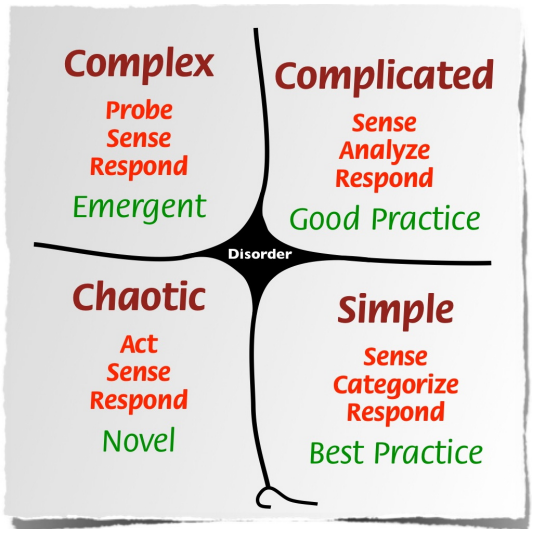


Figure 2: Le framework Cynefin

Bien que la nature primaire du modèle Cynefin soit dynamique, nous pouvons nous en servir pour partitionner les typologies de projet de manière statique.

- Le domaine « simple » : il permet la décomposition non ambiguë du travail à réaliser sous forme de tâches élémentaires. C'est le domaine incontesté du management scientifique du travail qui fournit les « best practices » nécessaires.
- Le domaine « compliqué » : il peut être analysé à priori, mais avec grande difficulté. Les « best practices » ne conviennent pas à tous les contextes, on parle alors simplement de « bonnes pratiques ». On est déjà à la limite de ce que peut proposer le Taylorisme, pour se tourner déjà vers de l'expertise.

[3] [TAYLOR11]
[4] [RUBIN13], p. 7

- Le domaine « complexe » : Il est caractérisé par des systèmes ne pouvant être décrits à priori, mais analysés à postériori. Ce sont les mécanismes émergents[5] qui sont utilisés pour guider l'évolution de ces systèmes.
- Le domaine « chaotique » sort un peu du cadre qui est le notre aujourd'hui. C'est une situation de bordel ambiant où l'on est cantonné à répondre dans l'immédiat à une situation. L'équivalent d'une guérilla urbaine, en quelque sorte…

En fait, Taylor lui-même ciblait le domaine d'application du management scientifique du travail au travail simple et répétitif, excluant de facto les travailleurs du savoir, par exemple. Les méthodes agiles ont elles une appétence naturelle pour les domaines compliqués et complexes, avec une préférence pour les seconds. Les créateurs de Scrum ont très tôt clarifié ce positionnement[6] et le représente comme suit.

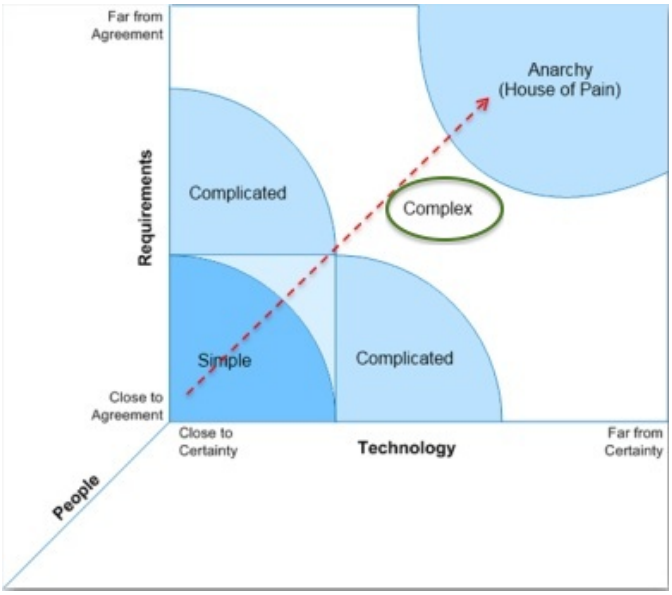


Figure 3: Le positionnement de Scrum

L'agilité n'est donc pas une méthode différente, ce n'est pas même une méthode ! C'est un autre paradigme, adapté à un contexte complexe. Comme nous l'avons dit, c'est un savoir être plutôt qu'un savoir faire !

Pour aborder un savoir être, il nous faut une approche différente d'un descriptif en profondeur. Il nous faut un chemin initiatique.

Le Shu Ha Ri

Alistair Cockburn nous propose un modèle d'évolution de ce savoir être inspiré de l'Aïkido : le Shu-Ha-Ri[7]. Ces 3 niveaux peuvent se résumer comme suit :

[5] [ADDINQUY12]
[6] [SCHWABER01], p. 94
[7] [COCKBURN06], p. 17

- Le Shu : C'est la phase d'apprentissage. Elle pourrait se traduire littéralement par « obéissance ». Il s'agit de suivre de manière exacte les gestes (le processus) proposé sans en dévier.
- Le Ha : Une fois le niveau Shu acquis, on passe au Ha qui est le perfectionnement. Ha signifie « rupture ». Il s'agit d'adapter les techniques acquise sans trahir l'esprit sous-jacent.
- Le Ri : On atteint le Ri quand on arrive au niveau de la maîtrise. Il s'agit de « l'abandon ». On innove et l'on crée sa propre façon d'être agile.

L'un des aspects les plus intéressant de Scrum à mon avis, est qu'il permet de nous accompagner au long de ces 3 phases. Pour cela, il faut encore en comprendre la véritable nature, ce qui est hélas rarement mis en avant.

Mon meuble Scrum

La plupart des nouveaux venus à Scrum, mais aussi hélas des vétérans voient Scrum comme un processus parfaitement balisé, surtout proche du prescriptif que j'ai évoqué précédemment.

Ce n'est pas ainsi que je vois Scrum. Scrum ne fait rien par lui-même, Tobias Mayer le compare à une class abstraite sans implémentation[8]. Pour ma part, j'aime bien la métaphore du meuble de rangement.

Figure 4: Scrum est un meuble... vide !

Mais pas n'importe quel meuble ! Un meuble très pratique, avec des étagères et des espaces de rangement fonctionnels. Mais ce meuble, s'il reste vide n'est d'aucune utilité. Il sera d'autant plus utile et pratique que l'on s'en sert utilement. On établira des usages de ses différentes étagères et tiroirs. Au fil du temps, on affinera l'exploitation du meuble afin qu'il nous aide à ranger et retrouver encore mieux nos affaires.

Pour débuter avec Scrum, et ensuite mieux savoir comment l'améliorer, il nous faut débuter au « niveau Shu » et voir comment le mettre en œuvre « by the book ».

[8] [MAYER13], p. 86

Scrum Shu

D'abord, un peu d'histoire…

Scrum s'enorgueillis de plus ancienne méthode de développement agile. Un titre un peu vide de sens à mon avis (et même faux, si l'on considère par exemple Evo[9]), mais que nous allons rapidement explorer. Scrum a été présenté officiellement pour la première fois à la conférence OOPSLA 1995[10] à Austin (Texas). Ce papier d'une vingtaine de pages cite comme source principale d'inspiration un article paru en 1986 dans le Harvard Business Review[11].

L'impact de Scrum reste toutefois confidentiel durant les années 90. En fait, sa notoriété sera même estompée par l'avènement de l'Extreme Programming[12] au début des années 2000, bien que le mouvement initié par Kent Beck lui soit postérieur !

Relayé au sein de la communauté des patterns[13], Scrum commencera à gagner doucement de l'audience au début des années 2000. Le premier ouvrage qui lui est entièrement consacré ne sera publié qu'en 2001[14], avec moins bruit que le texte de Kent Beck dédié à Extreme Programming.

Depuis, Scrum a pris l'ascendant sur le monde du développement agile, mais les approches Scrum et XP[15] sont souvent utilisées ensemble de manière complémentaire.

L'essentiel de Scrum

Scrum est une approche simple. Son guide officiel est d'ailleurs des plus succinct[16]. Il n'y d'ailleurs en fait pas de pratique intrinsèque à Scrum. Comme le dit Tobias Mayer : « Scrum ne vous dit pas comment faire du Scrum »[17]. On peut le résumer ainsi :

- 1 framework simple.
- 3 rôles : le Product Owner, le Scrum Master, les Equipiers
- 2 artéfacts : Le Product backlog et le Sprint backlog
- 4 réunions : le planning meeting, le daily Scrum, le sprint review et la rétrospective.
- 2 cycles : celui du sprint et celui de la journée.

[9] [LARMAN03], p. 212 ; [GILB05], p. 355
[10] [SCHWABER95]
[11] [TAKEUCHI+86]
[12] [BECK99]
[13] [SCHWABER+99]
[14] [SCHWABER+01]
[15] [KNIBERG07]
[16] [SUTHERLAND+13]
[17] [MAYER13] p. 22

L'essentiel de Scrum peut en fait être décrit en 5 minutes[18]. Peut-être 10 quand même…

Le cadre de Scrum

Le framework Scrum est souvent représenté graphiquement par le schéma suivant.

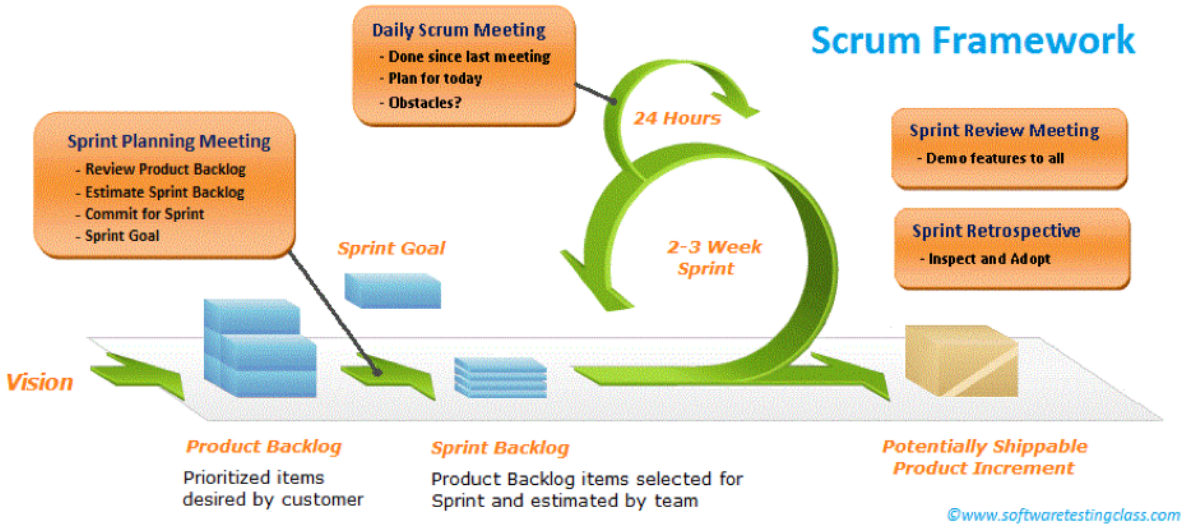


Figure 5: Le framework Scrum

La matière première alimentant Scrum est le Product Backlog, régulièrement alimenté et priorisé par le Product Owner.

Au début de chaque cycle de développement (qui dure typiquement entre 2 et 3 semaines, bien que les auteurs persistent à proposer jusqu'à 30 jours[19]), l'équipe et le Product Owner sélectionnent le sous-ensemble de plus haute priorité qui constitue le Sprint Backlog.

A la fin du cycle de développement, le produit augmenté de ces nouvelles fonctionnalités est démontré et est potentiellement livré.

Les 3 rôles

Scrum ne reconnaît que 3 rôles[20].

[18] [SOFTHOUSE06]
[19] [SCHWABER+12], p. 18 ; [SCHWABER04], p. 8
[20] [JOHNSON+12], p. 6

Figure 6: Scrum c'est 3 rôles et pas plus !

Le Scrum Master : A la fois facilitateur et protecteur[21], coach même, il guide l'équipe vers plus d'efficacité en suivant les principes fondamentaux de Scrum. Il veille à ce que l'équipe ne manque de rien et écarte les perturbations qui n'ont rien à voir avec le projet (sans être non plus un écran à la collaboration et aux informations).

Le Product Owner : Le Product Owner porte la vision fonctionnelle du produit[22]. A ce titre, il décide des fonctionnalités qui figureront dans celui-ci. Il a donc nécessairement un pouvoir décisionnaire. Mais il ne décide pas du comment et fait confiance à l'équipe pour la réalisation et les charges de travail associées.

L'équipe : Il n'y a pas de rôles spécialisés à l'intérieur de l'équipe[23]. Par exemple, il n'y a pas de testeurs. L'équipe est constituée autant que faire se peut de membres pluridisciplinaires, même si Scrum reconnaît que de l'expertise peut s'avérer nécessaire. Toutefois, il est de la responsabilité de l'équipe et d'elle seule de s'organiser pour mener le travail à bien en fonction des savoir-faire de chacun. Il faut simplement que l'équipe, en tant qu'équipe, possède en son sein toutes les compétences nécessaires.

Il était une fois un sprint

En pratique, comment se déroule un sprint ? Ceci peut être décrit en racontant l'histoire d'un Sprint, comme le fait Jeff Sutherland[24]. Nous allons tenter d'en donner une version rendue à l'essentiel.

[21] [COHN10], p. 118
[22] [PICHLER10], p. 3
[23] [RUBIN13], p. 195
[24] [SUTHERLAND+11], p. 55

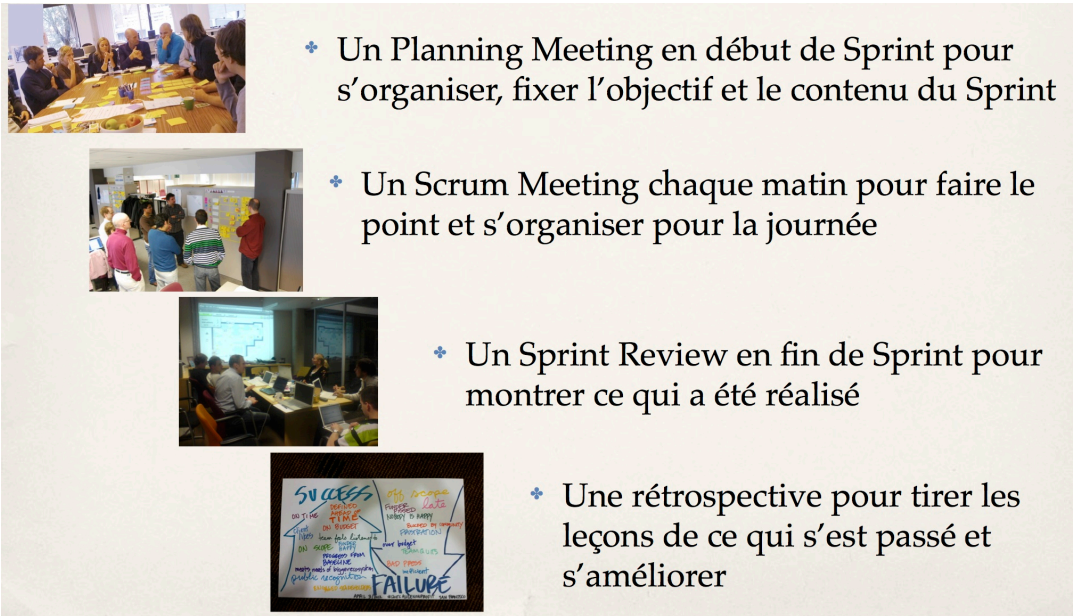


Figure 7: Le déroulement d'un sprint

En début de Sprint se déroule le planning meeting. Son but est d'établir le plan de bataille du sprint. L'équipe et le Product Owner discutent de l'objectif du sprint, de la compréhension des fonctionnalités souhaitées et de la manière dont elles peuvent être réalisées. Le produit du planning meeting est un plan constitué d'un sprint backlog qui devient généralement un Scrum Board où l'avancement est suivi et affiché, et dont les fonctionnalités sont découpées en tâches. La plupart du temps des estimations sont associées aux fonctionnalités.

Chaque jour l'équipe se réunit devant son Scrum Board pour faire état de son avancement, se synchroniser et préparer son plan d'action pour la journée qui commence. C'est le Daily Scrum (ou stand-up).

A la fin du Sprint, l'équipe réunit toutes les parties prenantes afin de faire état de l'avancement du projet, démontrer ce qui a été réaliser et recueillir le feedback[25].

Le sprint se clôt par une rétrospective[26] : un moment privilégié où l'on réfléchit sur la façon dont on a travaillé, ce qui a bien ou moins bien marché et comment cela pourrait être amélioré. On en repart avec un plan d'action d'améliorations.

C'est tout ?

Difficile de démarrer Scrum sans l'habiller d'un certain nombre de pratiques. Nombre d'entre elles sont aujourd'hui admises comme faisant partie intégrante de la démarche, bien que ce ne soit pas le cas. Ces « packaging » n'ont rien de mauvais en soi[27], mais il conforme parfois l'idée d'une approche prescriptive.

[25] [LACEY12], p. 179
[26] [DERBY+06] ; [GONÇALVES+13]
[27] Voir [AUBRY13] ou [RUBIN13] ou [LACEY12] ou encore [RAWSTHORNE+11]

C'est en commençant à considérer Scrum sous cet angle que l'on peut tomber dans certains travers.

Comment se fourvoyer efficacement

Loin d'être une liste exhaustive, ce qui suit est une sorte de « best of » de cas de figure que j'ai pu, hélas, rencontrer.

Le « mini V »

Une fois le périmètre du sprint établi, il est tentant, lorsque l'on vient de l'ancien monde, de découper le sprint en phases, en commençant par l'analyse et en terminant par les tests[28]. C'est le concept du « mini V ».

Figure 8: "min V", certes, mais tout autant maléfique !

Faire de la mini-cascade n'est pas agile :

- On génère un délai important entre le moment où l'on commence à travailler sur la fonctionnalité et le moment où l'on peut obtenir du feedback. Mini V signifie donc « mini tunnel » !
- Les tests arrivent à la fin : en cas de non-conformité, c'est une grande partie de l'itération, sinon sa totalité qui est mise en cause. Les sprints « time-boxés » rendent alors la situation difficile à vivre.
- Les « en cours » sont très important, en fait il s'agit de la totalité des fonctionnalités de l'itération : la priorisation des fonctionnalités n'est plus d'aucune aide et le « focus » sur une ou quelques fonctionnalités en cours est inexistant.

L'un des symptômes indiquant que l'on se dirige vers ce travers est l'utilisation du mot « phase » par des membres de l'équipe. Cela fait partie de mes alertes personnelles. Il n'y a pas de phases dans un projet agile.

[28] [WEST11]

L'équipe de tests séparée

Aux fonctionnalités figurant dans le backlog, Scrum nous impose de subordonner une « définition de terminé » à ces items. Cette définition est une check-list[29] de chose devant être faites ou vérifiées sur lesquelles l'équipe Scrum (comprenant le PO et le Scrum Master) et parfois les parties prenantes se sont mis d'accord. Si le contenu de cette liste peut être sujet à discussion, Scrum nous impose toutefois que les items développés soient potentiellement corrects, sinon comment les qualifier de « potentiellement déployables »[30] ?

Figure 9: Ca marche, il n'y a plus qu'à passer les tests...

L'idée de séparer l'équipe de tests de l'équipe de développement repose sur une vieille idée : on ne peut être juge et partie. C'est une logique de défiance dans laquelle on estime que si on laisse filtrer les éléments de validation vers l'équipe de développement, celle-ci va tricher afin de les faire passer en négligeant la qualité !

Non seulement cette approche par la méfiance conduit à produire des sprints non-déliverables (donc en fait non agiles), augmentant dans le même temps le délai de livraison, donc de feedback, mais cette approche rend difficile la convergence[31] entre équipes séparées qui interprèteront différemment une même spécification ! Remplacer la défiance par la collaboration est pourtant l'un des fondements du manifeste agile[32] !

Le test fonctionnel, en environnement agile, est une activité complètement intégrée à l'équipe de développement[33]. Il n'y a pas de phase de recette séparée, mais une validation continue.

[29] [RUBIN13], p. 74 ; [AUBRY13], p. 151
[30] [ADDINQUY13], [ADZIC11] p. 14
[31] [ADZIC11], p. 246
[32] http://agilemanifesto.org/iso/fr/ ;
[33] [CRISPIN+09], p. 22

Le « Scrum but… »

Il y a certaines expressions qui provoquent chez moi certaines associations d'idées. Par exemple « nous avons adapté Scrum à nos besoins » ou encore « nous avons pris le meilleur de différentes méthode ». C'est pour moi le signe d'une appréhension prescriptive de l'agilité, souvent d'ailleurs produite par des « départements méthodes » qui ne sont même pas impliqués dans les projets ! En fait, certaines méthodes prétendument agiles s'appuient justement sur cette approche, comme PUMA[34] ! Nous avons vu plus haut dans le modèle Cynefin que l'approche « best practices » est approprié uniquement dans le cas de contextes simples.

Lorsque l'on démarre avec Scrum dans le « Shu », il n'est nul besoin de trafiquer le framework ! Scrum est simplement, comme le stipule le Scrum Guide[35] dans son titre, un ensemble de règles du jeu. L'adaptation ou l'adoption des pratiques se fait de manière émergente en utilisant le feedback du projet lui-même, durant les rétrospectives.

Les chose ne seraient toutefois pas si catastrophiques si il ne s'agissait que de sélectionner des pratiques agiles. Hélas le plus souvent de pervertir le modèle agile afin de le faire ressembler au modèle de développement existant. Scrum n'est alors plus qu'un déguisement : on a emprunté son vocabulaire pour justifier les anciennes pratiques. C'est ce que j'appelle le Scrum « Canada Dry » : ça a le goût et l'odeur de l'agile, mais ce n'est pas de l'agile !

Figure 10: Scrum Canada Dry, le goût et l'odeur de l'agilité sans l'agilité

Beaucoup de détracteurs de Scrum s'appuient sur ces semblants de projets agile pour critiquer le framework. Il faut aussi dire que cette approche tend à se répandre[36]. Ron Jeffries appelle cette approche le « Scrum but »[37]. Il fournit un certain nombre de clés. J'ajoute ici ce que j'ai pu observer :

[34] http://www.entreprise-agile.com/ ;
[35] [SUTHERLAND+13], p. 1
[36] [WEST11]
[37] [JEFFRIES13]

- « Nous allons démarrer le projet en Scrum, mais nous avons besoin d'une phase préalable pour produire un backlog détaillé »[38]. Le backlog est ici vu comme une nouvelle incarnation du cahier des charges … avec son inévitable phase d'écriture !
- « Notre chef de projet sera le Scrum Master de l'équipe. C'est également lui qui fera les estimations, car il a le plus d'expérience ». Rien n'a changé ici, le chef de projet a simplement changé de nom pour devenir le Scrum Master, mais il fonctionnera en « contrôle / commande », plutôt que d'être là pour aider l'équipe[39]…
- « La planification du projet Scrum sera matérialisée sur un diagramme de Gantt, afin de pouvoir suivre les affectations et les retards ». L'équipe « Scrum » a ici été dépossédé de son auto-organisation, le projet est enclenché en mode prédictif d'où le feedback et l'adaptation ont été exclus.
- « Le client final n'a pas de temps à consacrer au projet. Le travail en relation avec l'équipe de développement sera confié à Josiane[40] qui était précédemment AMOA et sera PO proxy sur ce projet ». Rien n'a vraiment changé ici : l'assistance à maîtrise d'ouvrage qui isolait l'équipe de développement des véritables utilisateurs est toujours présente avec les mêmes travers. Le nom a juste changé !

Nous pourrions poursuivre très longtemps cette litanie. Le motif est toujours le même : Scrum est vidé de sa substance, seul le vocable est gardé pour déguiser les pratiques existantes, souvent afin de satisfaire un top management demandant de l'agile !

Le mode rush

Les versions précédentes du Scrum Guide évoquaient la notion d'engagement. Les équipes devant, au début d'un sprint, s'engager à livrer les fonctionnalités retenues en début de sprint. Si l'engagement moral est une valeur saine, cette notion d'engagement est souvent vue comme un moyen d'exiger plus des équipes via une sorte d'auto-asservissement.

La plupart du temps toutefois, cette pression résulte de l'équipe elle-même ! Le rythme de livraison toute les 2 ou 3 semaines s'avère très soutenu. Il est indispensable de ne prendre en charge que la quantité de travail pouvant être réalisée avec la qualité requise en respectant un rythme de travail décent[41]. Si le rythme n'est pas soutenable, on observera un épuisement de l'équipe, une chute de la productivité, une démotivation et souvent des départs (ce qui n'arrange pas la productivité).

[38] [ADDINQUY12B]
[39] [MESSAGER10], p. 184 ; [HIGHSMITH04] p. 57
[40] Le nom est bien entendu fictif !
[41] http://agilemanifesto.org/principles.html ; voir le principe n° 8

Figure 11: Le mode "rush" conduit à l'éffondrement.

Les équipes essayant d'en faire « trop », que ce soit par elle-même ou par une action du management obtiennent les résultats suivants :

- Augmentation de la dette technique[42] : car on ne prend pas le temps pour implémenter correctement les fonctionnalité, refactorer le code ou nettoyer le code mort.
- Baisse de la qualité des fonctionnalité, car on néglige les tests pour augment la vélocité apparente.
- Augmentation du coût d'ajout des nouvelles fonctionnalités, car la base de code n'est pas propre et le comportement applicatif pas maîtrisé.
- Augmentation des heures de travail pour compenser[43], d'où : stress, fatigue, démoralisation, perte d'attention.
- Augmentation des sacrifices de qualité pour tenter de sauvegarder la vélocité…

Il s'agit d'une spirale négative qui ne peut être endiguée qu'en revenant aux véritables principes agiles.

Des stories trop longues !

Spécialement lorsque le product owner débute, il est fréquent de voir des items de backlog de taille importante (par exemple représentant 20 jours de travail). On trouve plusieurs raisons à cela :

- Manque d'habitude à « trancher fin » les fonctionnalités.
- Manque de confiance dans l'équipe : le PO pense que s'il n'embarque pas tout ce qu'il veut maintenant, ce ne sera jamais fait.
- Intime conviction de savoir par avance tout ce qui est nécessaire plutôt que de s'appuyer sur les mécanismes de feedback[44] et d'émergence.

L'un des problèmes inhérents aux « grosses fonctionnalités » est la difficulté de fluidifier le développement et conséquemment la difficulté d'aligner la fin des

[42] [TATE05], p. 15
[43] [TATE05] , p. 32
[44] [ADDINQUY13B]

développement sur la fin d'itération[45] (sans compter bien sûr le délai de feedback et le risque de construire des choses inutiles).

Certains projets ont ainsi des items de backlog s'étalant systématiquement sur plus d'une itération, les itérations commençant d'ailleurs par la fin des items précédemment commencés. Certains items peuvent s'étaler sur 3 itérations, voir plus ! Dans ce cadre, les itérations perdent tout sens et n'existent plus que pour justifier une prétendue conformité à Scrum.

Scrum Ha

Une fois rôdé au fonctionnement de Scrum « by the book », il peut être temps de réfléchir à la manière dont on peut en faire « notre Scrum ». Je différencie là :

- Les évolutions « de l'intérieur », qui permettent de solidifier Scrum avec des pratiques adaptées.
- Les extensions externes, qui permettent d'englober dans le framework agile des activités qui n'y sont pas initialement prévues.

Mais avant d'aborder ces points, nous commencerons par évoquer le l'extension la plus naturelle : l'intégration de pratiques d'ingénierie dans Scrum.

Scrum et XP, une question de bon sens !

L'un des reproches les plus souvent fait à Scrum est l'absence de pratiques d'ingénierie. Ce n'est pas le but de ce cadre de développement qui n'est là que pour, rappelons-le, donner des règles du jeu. Une pratique couramment admise est d'emprunter les pratiques de développement d'Extreme Programming pour renforcer Scrum. Cette pratique avait été baptisée XBreed[46] il y a une dizaine d'année, mais cette appellation n'a pas perduré. Initialement constituée de 12 pratiques[47], XP est aujourd'hui formé de 13 pratiques[48], résumées ci-dessous.

[45] [GOLDSTEIN13], p. 47
[46] [LARMAN03], p. 132
[47] [BECK99]
[48] [BECK+04]

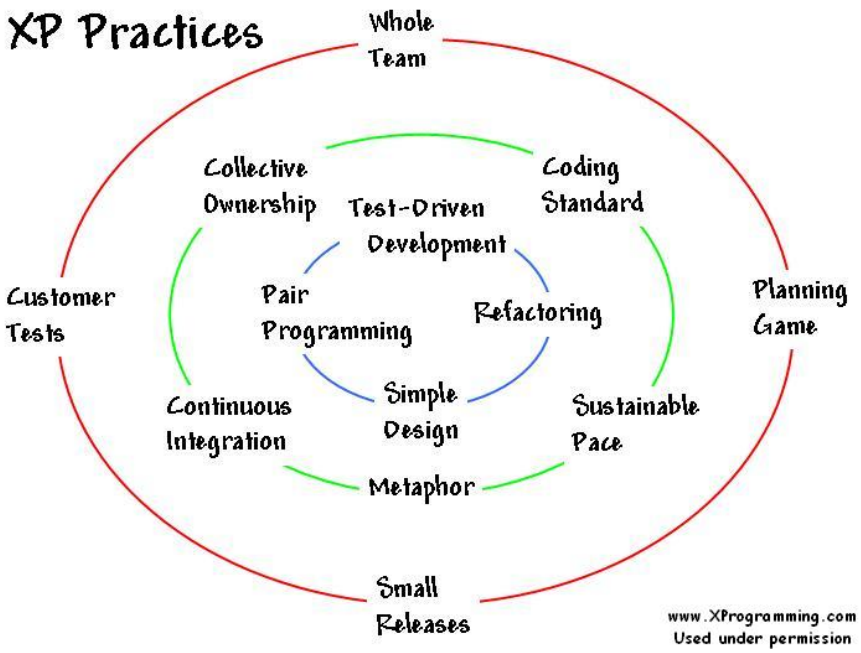


Figure 12: Les pratiques XP

Certaines pratiques sont tellement convenues dans Scrum que leur héritage de départ a été oublié. D'autres sont utilisées moins systématiquement, alors qu'elles apportent de réels avantages. Nous allons simplement évoquer brièvement certaines d'entre-elles.

Le Planning Game : c'est une façon d'aborder le planning meeting via des discussion[49] et une estimation collective des fonctionnalités identifiées sous forme de User Stories. La technique du Planning Poker[50] est souvent utilisée pour les estimations.

L'intégration continue : Il semble aujourd'hui inconcevable de débuter un projet agile sans plateforme d'intégration continue[51]. Celle-ci doit à chaque « commit » dans le système de versionning, non seulement construire l'application, mais jouer les tests et vérifier les indicateurs de qualité.

Test Driven Development : Cette pratique consiste à écrire les tests unitaires avant le code qu'il doit tester[52] ! Considéré comme une bonne pratique, il sa mise en œuvre progresse mais reste minoritaire.

Pair Programming : Le développement en binôme permettant de faire une revue de code permanente et en temps réel[53] est probablement la pratique issue d'XP qui peine le plus à s'imposer.

[49] [COHN04], p. 110
[50] [COHN06], p. 56
[51] [HUNT+00] p. 50 ; [BECK+04], p. 49 ; [DUVALL+07]
[52] [KOLKELA08], [BECK03] et [ASTELS03]
[53] [WILLIAMS+02]

Renforcer Scrum à l'intérieur

Au-delà des pratiques empruntées à l'Extreme Programming, un certain nombre de pratiques font leur apparition ou se généralisent. La liste qui suit n'est pas exhaustive.

Le développement guidé par les tests d'acceptance

Le développement guidé par les tests est avant tout une démarche de définition des besoins s'appuyant sur des exemples[54] avant d'être une pratique outillée[55]. L'écriture des tests fonctionnels y est réalisée de manière collaborative entre l'équipe de développement et le PO, complétant les spécification sur le comportement attendu du système dans différentes situations.

Les agile games

En progression constante depuis 7 ou 8 ans, l'utilisation des jeux pour innover[56], construire[57] et comprendre est en constante progression. Les jeux agiles sont des outils puissants pour favoriser l'engagement et permettre la co-construction.

Figure 13: Utiliser les jeux pour comprendre, construire et innover

Le management visuel

Issu de l'univers du Lean, inspiré par les Obeya room[58], les équipes Scrum déploient de plus en plus leur suivi de projet en éléments visuels affichés sur les

[54] [ADZIC11]
[55] [GÄRTNER13] et [CARDINAL13]
[56] [HOHMANN07]
[57] [GRAY+10]

murs, donc visibles de tous en permanence, devant lesquels les gens peuvent discuter et prendre des décisions. Les managers sont invités à se déplacer (plutôt que de recevoir des rapports dans leur boite mail), pour voir l'avancement tout en se rendant compte en pratique sur le terrain de ce qui se passe.

Figure 14: Un Scrum Board

Le craftmanship

Directement issu de l'extreme programming que nous avons évoqué précédemment, ce mouvement promeut la qualité du code au travers de quelques principes de base :

- Le « clean code »[59] : qui met en avant les principes d'écriture d'un code lisibles, bien conçu et testé dont on peut être fier.
- Une recherche continuelle du perfectionnement via des pratiques telles que les coding dojo[60] ou les communautés de pratique.
- Une très bonne maitrise des outils, des environnements et des langages utilisés.

Etendre Scrum à l'extérieur

De Scrum à Kanban

Les équipes ayant acquis un bon niveau de maturité sur leur capacité à délivrer très souvent réduisent parfois leur cycle de développement à une semaine, afin de minimiser le « stock » que constitue l'ensemble de fonctionnalités à estimer et démontrer. L'étape suivante consiste alors à éliminer complètement la notion

[58] [IGNACE+12], p. 105
[59] [MARTIN08], [HUNT+00], [BECK07], [SPINELLIS03] et [MCCONNELL04]
[60] [MARTIN11], p. 83

d'itération pour travailler en flux. Il n'y a plus de planification, une nouvelle fonctionnalité à réaliser (la plus prioritaire à ce moment là) prend place quand une autre sort. Ce fonctionnement en flux utilise la technique du Kanban[61], qui nous vient du Lean.

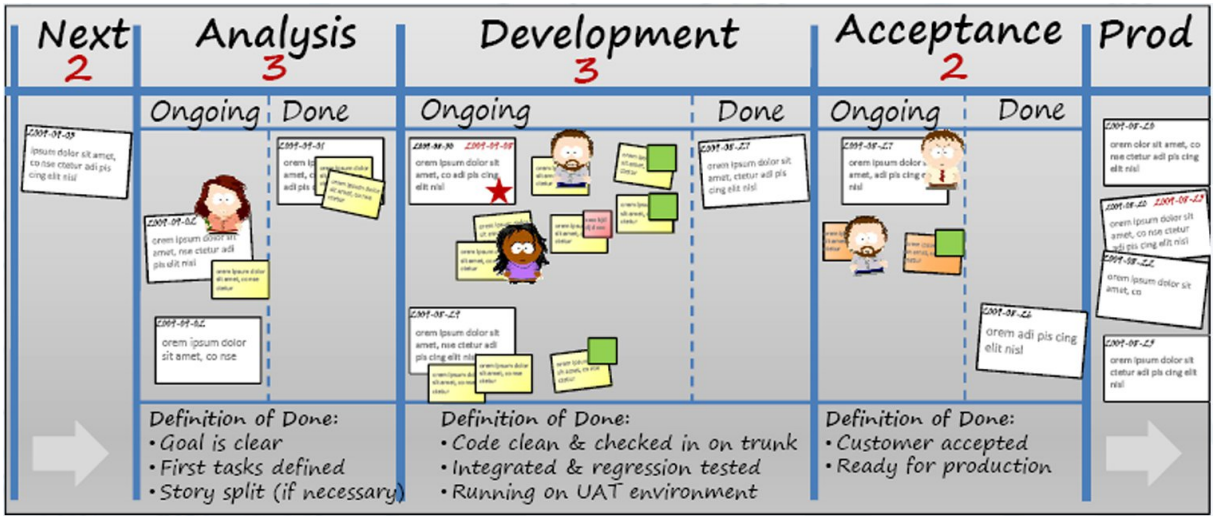


Figure 15: Utiliser un Kanban pour gérer son cycle de développement

La combinaison Scrum / Kanban est utilisé par un nombre croissant d'équipes[62]. Cet ensemble peut même être utilisé pour coordonner plusieurs équipes Scrum[63].

Agilifier la définition du produit

Scrum se concentre sur l'activité de réalisation en mode agile. Toute la gestion produit est déléguée au PO qui a pour responsabilité d'aligner son travail de définition de produit sur le cycle de développement. Mais Scrum ne propose rien à ce sujet[64]. Pourtant les activités liées au Marketing et à la gestion de produit couvrent déjà à elle-seules (par exemple) un spectre d'activité très large[65] !

[61] [MORISSEAU12]
[62] [KNIBERG+10] et [LADAS08]
[63] [KNIBERG11]
[64] [ADDINQUY12C]
[65] [RUBIN13], p. 179

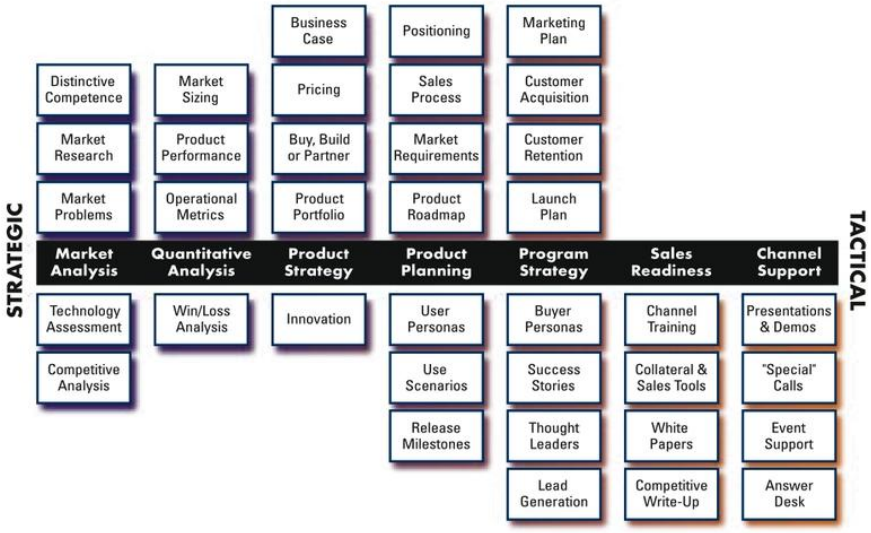


Figure 16: Pragmatic Marketing Framework

La réalité de nombreux projets Scrum est une réalisation agile à l'intérieur d'un carcan produit non-agile. Le backlog qui est la charnière entre les deux est alors vu comme :

- Au mieux, une facilité qui permet de construire le besoin de manière incrémentale, mais pas de le remettre en cause ou de bénéficier du mécanisme de feedback.
- Au pire une concession à Scrum, là où un bon vieux cahier des charges serait préféré…

Quelques pratiques et courants émergent pour penser différemment la construction incrémentale de produits.

- La communauté Kanban découpe le produit en Minimum Marketable Feature (MMF) et Minimum Marketable Release (MMR)[66].
- Le Story Mapping[67] nous aide à construire un plan de release en structurant les User Stories selon des axes processus et priorité.
- L'Impact Mapping[68] nous propose de repenser complètement notre notion de de roadmap produit en substituant le « pourquoi » (inspiré du « Start with the Why » de Simon Sinek[69]) à la notion de périmètre.

[66] [ANDERSON10], p. 151 ; [MORISSEAU12]
[67] [ADDINQUY13C], il s'agit d'un ensemble de pointeurs vers des ressources traitant du Story Mapping, en attendant l'ouvrage de Jeff Patton sur le sujet
[68] [ADZIC12]
[69] [SINEK09]

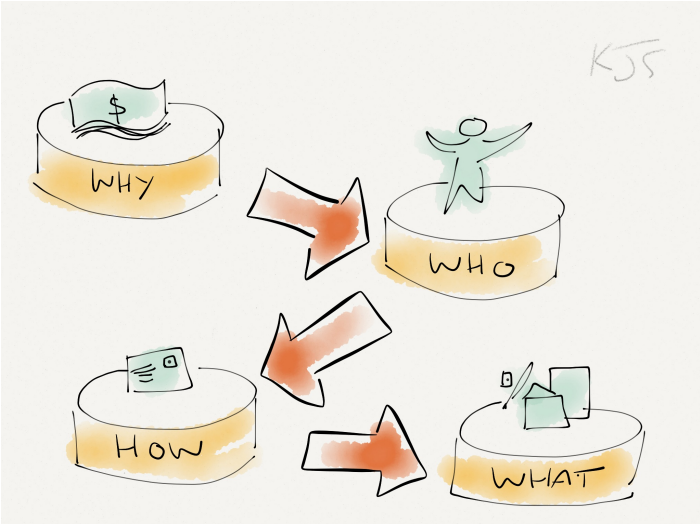


Figure 17: L'impact Mapping

- L'analyse de la valeur agile[70], qui substitue la notion de budget et de différenciateur à celle d'estimation.

Le Lean Startup

L'agilité peut embrasser plus que la réalisation et la définition de produit : elle peut englober le business model d'une entreprise elle-même ! Le Lean Startup[71] est un cycle « d'apprentissage validé » permettant d'orienter le produit ou le segment de marché adressé en fonction du feedback de l'utilisateur final. On n'y parle plus d'expression de besoin, mais d'hypothèse nécessitant une validation (ou invalidation) du terrain.

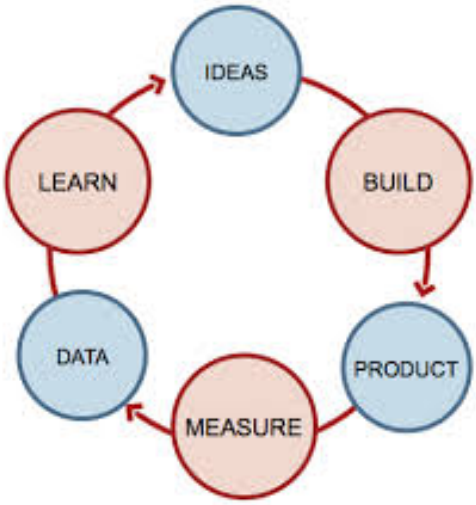


Figure 18: Le cycle Lean Startup

[70] [PIXTON+09]
[71] [RIES11]

Particulièrement adapté aux projets innovants (et pas seulement aux startups), cette approche abandonne les idyliques et erronée business plans aux profits de business models réalisés sur des canevas A3[72].

Pour être complètement opérationnel, le cycle Lean Startup doit s'appuyer sur une capacité de mise en production très réactive : c'est le domaine du devops.

Du développement à la production avec devops

Développement et production ont traditionnellement des optiques et des besoins très différents[73].

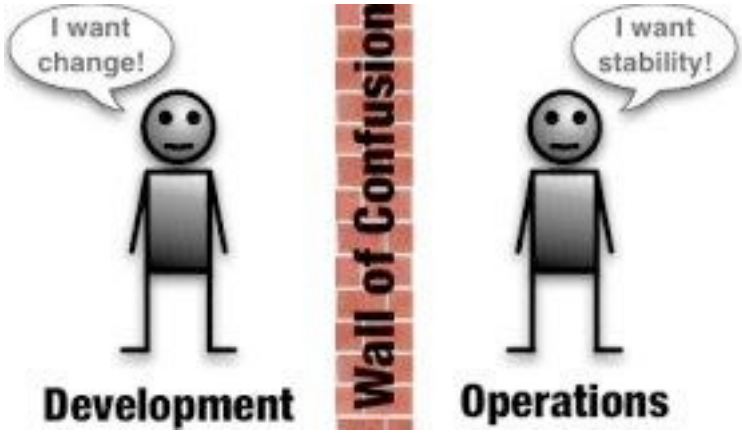


Figure 19: Développement versus opérations

- Le développement veut pousser constamment de nouvelles évolutions, là où les opérations veulent assurer la stabilité des plateformes.
- Le développement a le regard tourné vers le futur, ce qu'il y a encore à développer ; les opérations se focalisent sur le présent et doit fonctionner.
- Le développement est focalisé sur les aspects fonctionnels des applications ; les opérations les veulent opérables et monitorables… ce que le développement a tendance à prendre pour acquis !

Ces visions divergentes sont néanmoins complémentaires. Dans les grandes structures, elles sont souvent confiées à des directions différentes, ce qui est générateur de tension. Mais plus encore : si le développement est à même de livrer toutes les 2 ou 3 semaines (voir plus), les équipes d'exploitation bloquent souvent ces versions pour ne mettre en production que tous les 3 mois, voir plus. De nouveau, nous retrouvons le syndrome du cycle agile dans un carcan non-agile.

[72] [OSTERWALDER+10] et [MAURYA12], p. 27
[73] [HUTTERMANN12], p. 40

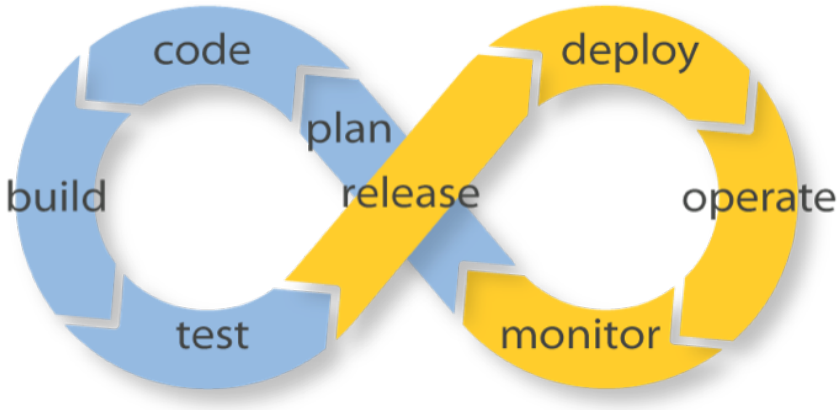


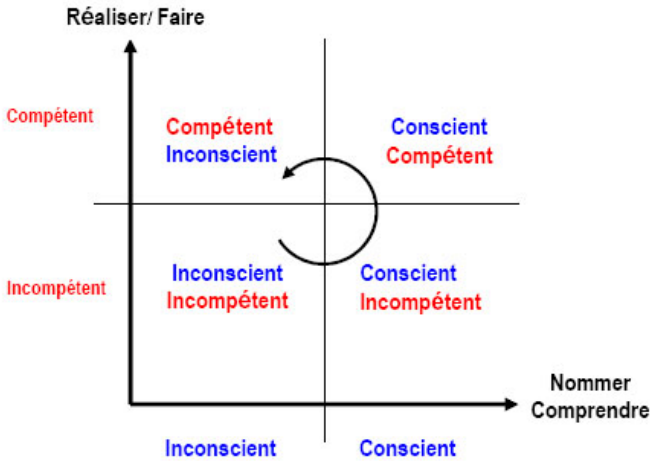
Figure 20: Le cycle de collaboration devops

Le mouvement devops s'appuie sur la collaboration directe entre équipes de développement afin de permettre des mises en productions plus fréquentes. Elles peuvent amener les équipes à opérer un déploiement continu[74], jusqu'à plusieurs dizaines de fois par jour dans certaines entreprises. Mais les mise en production quotidiennes ne sont pas rares.

La prise en considération des opérations sur le développement ne s'arrête pas aux mises en production, elle s'étend aux aspect architecturaux des applications[75], en terme de stabilité, performance et scalabilité.

Scrum Ri

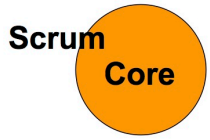
Vivre le « Scrum Ri » est à la fois plus simple et plus compliqué à décrire ! Etre dans le Ri ne nécessite pas de guide ou de règle car on sait « vivre agile ». On peut situer le niveau Ri sur le 4 niveaux de l'apprentissage au niveau ultime du « compétent inconscient »[76].



[74] [HUMBLE+11]
[75] [NYGARD07]
[76] [MESSAGER12], p. 64

Figure 21: Les 4 étapes de l'apprentissage

Le processus Scrum, au niveau Ri, peut être décrit de manière très succincte. Alistair Cockburn l'a fait dans sa présentation « current topics on agile »[77], en décrivant le « core scrum ».



1. (Demo or) **Deliver every sprint.**
2. **Let the team decide.**
3. **Inspect & Adapt every day and every sprint.**

4. **Someone has spare capacity to remove blocks (ScrumMaster)**
5. **Business speaks through 1 vocal cords (Product Owner)**

Figure 22: Le core scrum, par Alistair Cockburn

Peu de livres tentent de faire comprendre le Scrum au niveau Ri. Le seul réellement centré sur ce sujet est de point de vue celui de Tobias Mayer[78].

Inventez !

La règle du « Scrum Ri » est qu'il n'y a pas de règles, seulement des repères. Le Scrum Master devient « Scrum Explorateur »[79].

Figure 23: Prêt à devenir "explorateur agile" ?

[77] [COCKBURN10]
[78] [MAYER13]
[79] [MAYER13], p. 49

Le repère le plus connu est sans aucun doute le manifeste agile[80]. Mais le praticien « Ri » devra aussi construire les siens. Pour illustrer cela, je vais vous parler de ce dont je sais le mieux parler, à savoir de moi-même !

Mes repères agiles

Au fil du temps, j'ai développé mes propres repères, mes indices me permettant de porter mon attention sur des points d'améliorations. Tous ces repères ont fini par se regrouper autour de trois grands pôles qui constituent aujourd'hui ce qui m'importe réellement en agile. Mon étoile du nord en quelque sorte. Il s'agit d'une étoile triple.

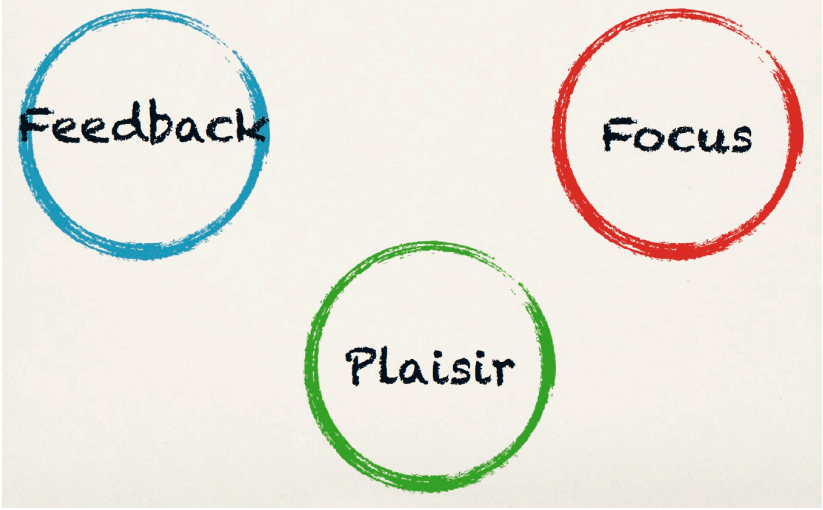
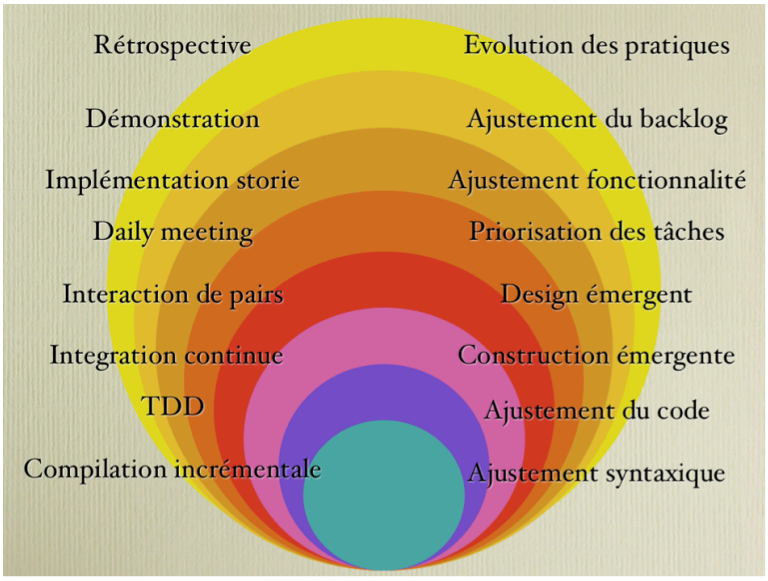


Figure 24: Mes repères agiles

Le feedback

Je regarde souvent la dynamique de travail agile comme une imbrication de boucles de feedback, allant de la seconde à quelques semaines.



[80] [COCKBURN06], p. 261

Figure 25: Les boucles de feedback de l'agilité

Chaque action que nous entreprenons a son pendant en terme de feedback, nous permettant de nous ajuster de mesurer ce que nous avons fait, si nous devons garder, modifier (corriger), amplifier ou jeter. Cela fait partie de notre boucle d'apprentissage. En fait, c'est notre boucle d'apprentissage !

En tant que coach, quand je vois une équipe ou le management décider quelque chose, je cherche la boucle de feedback correspondant. La présence ou l'absence de cette boucle de feedback me donne une idée de la maturité de l'équipe. En tant qu'agiliste expérimenté, avant d'entamer une action vous devez être capable de dire comment vous pourrez l'évaluer. C'est en quelque sorte un « TDD everywhere ».

Le focus

On ne peut avancer efficacement qu'en se concentrant sur un objectif. Là aussi, c'est vrai à différent niveaux sur un projet agile.

- Au niveau des objectifs du projet. Le rôle du management est de donner du sens à ce à quoi ils contribuent[81]. Il doit être capable de formuler celui-ci de manière claire[82].
- Au niveau des objectifs de sprint[83] : celui-ci doit se focaliser sur un objectif, qui se décline ensuite en user stories.
- Au niveau de la définition des user stories. Celles-ci doivent permettre de se concentrer sur un aspect précis[84], de concentrer l'effort et l'attention de l'équipe sur un aspect précis du système ou sur un segment de marché (d'utilisateurs) particulier[85].
- Dans la stratégie d'avancement quotidien : plutôt que d'entamer de développement tout azimuts, concentrer l'effort de l'équipe sur quelques stories, les plus prioritaires à ce moment, et les « passer dans le rétroviseur ».
- Dans l'implémentation : favoriser les « petits pas »[86] afin de favoriser l'avancement sur des éléments très précis… et en obtenir le feedback.

Ne pas se focaliser, c'est se disperser et perdre le sens de l'effort que l'on fournit. Dans mon travail de mentoring au quotidien, je cherche à discerner si l'action ou l'effort est bien focalisé ou s'il embarque de la « matière grasse » dont on devrait se débarrasser !

Le plaisir

L'une des assertions les plus déplaisantes de Scrum concerne « l'hyper productivité » à laquelle les équipes devraient arriver[87] ! Cette assertion hasardeuse

[81] [PIXTON+09], p. 60
[82] [APPELO10], p. 170
[83] [SUTHERLAND+13], p. 9
[84] [ADZIC13], p. 29
[85] [ADZIC13], p. 48
[86] [BECK07], p. 20

ou embarrassante (selon la manière dont on l'aborde) a induit certains travers comportementaux pour obtenir « plus de productivité ».

On n'obtient pas ceci en mettant la pression, en exigeant de terminer plus tôt sous le joug de menaces.

On l'obtient en mettant les membres de l'équipe en situation de l'accomplir. En satisfaisant :

- Les besoins extrinsèques : Environnement, outils, etc.
- Les besoins intrinsèques : ce qui compte réellement pour les individus.

Les besoins intrinsèques peuvent varier. Toutefois, je trouve très souvent 2 dénominateurs communs :

- Donner du sens à notre travail. Ce que nous avons abordé avec le « focus ».
- Avoir du plaisir dans ce que nous faisons.

La manière dont chacun prend plaisir dans son travail peut varier. Je la constate le plus souvent de deux manières :

- Comment fonctionne la dynamique d'équipe, les interactions, la confiance entre les membres, etc.
- Le « one on one »[88], la discussion ouverte en face à face pour prendre le pouls de manière individuelle et comprendre ce qui fait avancer chacun.

Le plaisir ne se décrète pas : il peut se favoriser en créant le bon cadre permettant l'erreur, l'échange et le fun. Il se fait aussi en écoutant les individus et en créant les opportunités adéquates.

Au final, le thermomètre est très simple : avez-vous plaisir à vous rendre au travail le matin ?

Scrum en dehors du développement logiciel

On peut aussi étendre Scrum au-delà du monde du logiciel. Pour certains (mais pas pour moi), le « Ri » peut se manifester dans la vie personnel. Parmi les cas concrets que j'ai rencontré : la rénovation d'une maison, la sélection d'un conjoint (!), l'organisation d'un mariage, de vacances, etc.

On peut aussi l'appliquer dans le monde professionnel en-dehors de l'univers logiciel. Jeff Sutherland, co-créateur de Scrum, est marié à une pasteur qui gère son église avec Scrum[89] ! J'ai bien un peu de mal à me faire une idée de la chose…

L'exemple le plus frappant reste tout de même celui de Wikispeed.

---

[87] [SCHWABER+01], p. 15
[88] [ROTHMAN+05], p. 150
[89] http://scrum.jeffsutherland.com/2009/06/scrum-in-church.html ;

Figure 26: Wikispeed

Wikispeed est un constructeur automobile qui a débuté en 2011 la construction de voitures modulaires aux USA. Son fondateur, Joe Justice applique Scrum en réalisant ses voitures sur des sprints d'une semaine[90] !

Transformation d'organisation avec Scrum

Nous avons évoqué les barrières qui peuvent empêcher un projet d'être réellement agile en étant enfermé dans un carcan non-agile. Cette réflexion peut s'étendre à la totalité d'une organisation, en incluant les fonctions support !

Fonctions support ou fonctions à supporter ?

Je ne vais pas vous apprendre que les entreprises se transforment de plus en plus en entreprises numériques, et nos projets construisent alors littéralement les produits de l'entreprise. Ce n'est pas toujours le cas. Dans la majorité des cas, les projets informatiques construisent des application pour soutenir, renforcer ou démultiplier le véritable business de l'entreprise. Ces projets sont donc des fonctions support de l'entreprise.

L'agilité nous apprend à être en ligne avec les fonctions supportées. C'est une saine évolution par rapport au paradigme précédent où l'on exigeait de nos utilisateurs qu'ils se plient aux fourches caudines de processus inadaptés : cahier des charges exhaustifs à rédiger en amont, spécifications détaillés à signer, processus de gestion de changement, acceptation d'un produit conforme aux spécifications mais pas aux besoins réel.

[90] http://www.forbes.com/sites/stevedenning/2012/05/10/wikispeed-how-a-100-mpg-car-was-developed-in-3-months/ ;

Aujourd'hui, cette situation se retrouve au sein des fonctions support sensées épauler les projets (mais aussi les autres départements de l'entreprise) : achats, RH, financiers, services généraux pour ne citer qu'eux !

- Des services généraux qui font obstacles aux besoins spécifiques des équipes au nom de la standardisation d'entreprise.
- Des achats objectivés aux économies, qui brident les projets sur les choix des prestations ou du matériel dont ils auraient besoin. De fausses économies qui s'avèrent finalement des dépenses plus élevées pour les projets et les clients.
- Des services financiers qui figent les budget annuels 6 mois avant le début de celle-ci, alors même que la situation a déjà changé quand l'année commence vraiment !

Des frémissements apparaissent pour briser cette logique, comme le Lean Accounting[91] ou le Beyond Budgeting[92] côté finance. Plus ambitieux, le Stoos Network[93] se veut un « réseau de réseau », initié par des personnes comme Stephen Denning ou Jurgen Appelo. Il s'agit ici de réformer le management des organisations pour créer l'organisation agile du 21ème siècle.

Quand l'organisation est un produit !

Quel est le rapport entre Scrum et l'agilification des organisations ?

La transformation agile d'une organisation peut se concevoir comme un projet. Un projet agile, bien entendu. Il peut donc se mener avec Scrum :

- Le produit est l'organisation agilifié.
- Le backlog, issu de la vision agile de l'organisation, est la liste des actions à mener.
- Le Product Owner est le dirigeant de l'organisation.
- Le coach organisationnel en sera le Scrum Master.
- Chaque Sprint délivrera un incrément de l'organisation agile !

[91] [MASKELL+05]
[92] http://www.bbrt.org/beyond-budgeting/beybud.html ;
[93] http://www.stoosnetwork.org/ ;

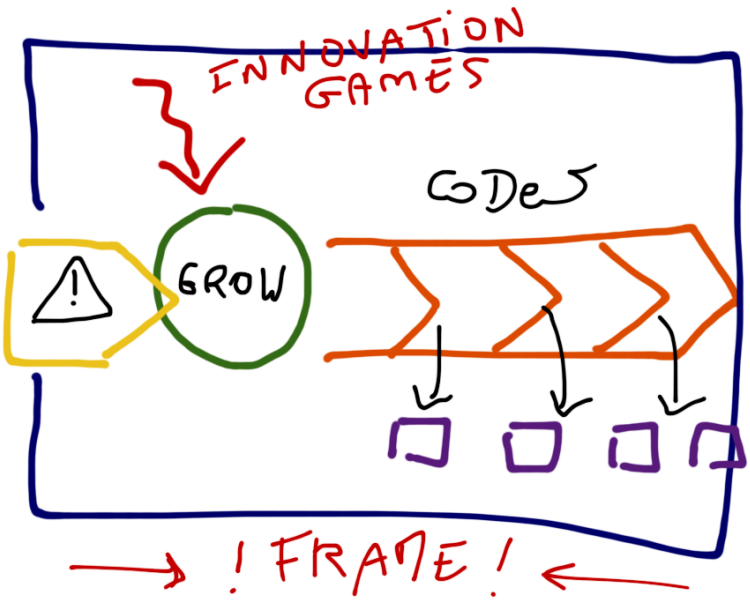


Figure 27 : Délivrer une organisation agile en incréments (c) Rupture 21

Suivez le guide !

Nous voici au bout de notre exploration. Revenons brièvement sur les 3 étapes que nous avons évoqué.

Commencer

Débutez par le Scrum simple et sans fioriture, sans substituer l'illusion de la forme au fond ! Résistez à le « perfectionner » d'emblée. Un processus n'est pas meilleur parce qu'il est complexe. Ce serait plutôt le contraire…

Echappez au tentatives de déguisement des vieilles pratiques sous couvert du vocable Scrum.

Adapter

Vous savez désormais faire « tourner » Scrum ? Ne vous arrêtez pas au « Scrum Shu ». Utilisez le pouvoir des rétrospectives pour trouver où vous améliorer. Devenez un explorateur de l'agilité : allez à la découverte des pratiques, rencontrez d'autres praticiens.

Vous souhaitez ajouter quelque chose ? Regardez aussi ce que vous pouvez enlever. Agilité et alourdissement ne riment pas.

Se détacher

N'écoutez pas ceux qui vous disent qu'ils font du meilleur agile parce qu'ils font du Lean au lieu du Scrum, ou de l'XP… Certes on y trouve de bonnes choses. Mais ils s'intéressent au processus et non au savoir être. Ce savoir être passe par la recherche, le questionnement, l'adhésion aux valeurs. On ne passe pas au « Ri ». On y arrive insensiblement, en se focalisant sur ce qui est important.

Ressources

[ADDINQUY12] : Les vertus de l'émergence – Christophe Addinquy - http://issuu.com/addinquy/docs/vertus-emergence ;

[ADDINQUY12B] : En finir avec le backlog ? – Christophe Addinquy - http://freethinker.addinq.uy/post/29279011758/en-finir-avec-le-backlog ;

[ADDINQUY12C] : En finir avec le Product Owner ? – Christophe Addinquy - http://freethinker.addinq.uy/post/30440618994/en-finir-avec-le-product-owner ;

[ADDINQUY13] : Done or done-done ? http://freethinker.addinq.uy/post/53875261316/done-or-done-done ;

[ADDINQUY13B] : L'agilité, une question de feedback – Christophe Addinquy - http://freethinker.addinq.uy/post/68206060492/lagilite-une-question-de-feedback ;

[ADDINQUY13C] : A la conquête du Story Mapping – Christophe Addinquy - http://freethinker.addinq.uy/post/66131430961/a-la-conquete-du-story-mapping ;

[ADZIC11] : Specification by Example, How successful teams deliver the right software – Gojko Adzic – Manning 2011 – ISBN : 978 1617290084

[ADZIC12] : Impact Mapping : Making a big impact with software products and projects – Gojko Adzic – Provoking Thoughts Ltd. 2012 – ISBN : 978-0-9556836-4-0

[ADZIC13] : 50 Quick Ideas to Improve your User Stories – Gojko Adzic – Leanpub 2013

[ANDERSON10] : Kanban, Successful evolutionary change for your technology business – David J. Anderson – Blue Hole Press 2010 – ISBN : 978 0 9845214 0 1

[APPELO10] : Management 3.0, Leading agile developers, developing agile leaders – Jurgen Appelo – Addison Wesley / Signature series 2010 – ISBN : 978-0-321-71247-9

[ASTELS03] : Test-Driven Development, A practical guide – David Astels – Prentice Hall / Coad series 2003 – ISBN : 0-13-101649-0 ; EAN : 978-0-131-01649-1

[AUBRY13] : Scrum, le guide pratique de la méthode agile la plus populaire, 3ème édition – Claude Aubry – Dunod 2013 – ISBN : 978 2 100 59446 7

[BECK99] : Extreme Programming Explained: Embrace change - Kent Beck - Addison Wesley 1999 - ISBN: 0-201-61641-6

[BECK03] : Test-Driven Development by example - Kent Beck - Addison Wesley / Signature series 2003 - ISBN: 0-321-14653-0

[BECK+04] : Extreme Programming Explained: Embrace change, 2nd edition – Kent Beck & Cynthia Andres – Addison Wesley / XP series 2004 – ISBN: 0-321-27865-8

[BECK07] : Implementation Patterns – Kent Beck – Addison Wesley / Signature series 2007 – ISBN : 0-321-41309-1 ; EAN : 978-0-321-41309-3

[CARDINAL13] : Executable Specifications with Scrum, a practical guide to agile requirements discovery – Mario Cardinal – Addison Wesley 2013 : ISBN : 978 0 321 78413 1

[COCKBURN06] : Agile Software Development, second edition : The cooperative game – Alistair Cockburn - Addison Wesley 2006 – ISBN : 0-321-48275-1 ; EAN : 978-0-321-48275-4

[COCKBURN10] : Current Topics on Agile – Alistair Cockburn 2010 - http://issuu.com/addinquy/docs/keynote-alistair ;

[COHN04] : User Stories Applied, for agile software development – Mike Cohn – Addison Wesley / Signature series 2004 – ISBN: 0-321-20568-5; EAN: 978-0-321-20568-1

[COHN06] : Agile Estimating and Planning – Mike Cohn – Prentice Hall 2006 – ISBN: 0-13-147941-5; EAN: 978-0-131- 47941-8

[COHN10] : Succeeding with Agile, Software development using Scrum – Mike Cohn – Addison Wesley 2010 – ISBN : 0-321-57936-4 ; ISBN13 : 978-0-321-57936-2

[CRISPIN+09] : Agile Testing, A practical guide for testers and agile teams – Lisa Crispin & Janet Gregory – Addison Wesley / Signature series 2009 – ISBN: 0-321-53446-8 ; EAN: 978 0 321 53446 0

[DERBY+06] : Agile Retrospectives, Making good teams great – Esther Derby & Diana Larsen – Pragmatic Bookshelf 2006 – ISBN : 0-9776166-4-9

[DUVALL+07] : Continuous Integration, Improving software quality and reducing risk – Paul M. Duvall, with Steve Matyas & Andrew Glover – Addison Wesley / Signature series 2007 – ISBN : 0-321-336338-0 ; EAN : 978 0 321 33638 5

[GÄRTNER13] : ATDD By Example, A practical guide to acceptance test-driven development – Markus Gärtner – Addison Wesley / Signature series 2013 – ISBN : 978-0-321-78415-5

[GILB05] : Competitive Engineering – Tom Gilb – Elsevier Butterworth-Heinemann 2005 – ISBN : 9780750665070

[GOLDSTEIN13] : Scrum Shortcuts without cutting corners, agile tactics, tools & tips – Ilan Goldstein – Addison Wesley / signature series 2013 – ISBN : 978 0 321 82236 9

[GONÇALVES+13] : Getting Value out of Retrospectives, a toolbox of retrospective exercices – Luis Gonçalves & Ben Linders – InfoQ / Leanpub 2013

[GRAY+10] : Gamestorming, a playbook for innovators, rulebreakers and changemakers – Dave Gray, Sunni Brown & James Macanufo – O'Reilly 2010 – ISBN : 978 0 596 80417 6

[HIGHSMITH04] : Agile Project Management, Creating innovative products – Jim Highsmith – Addison Wesley / ASD series 2004 – ISBN: 0-321-21977-5

[HOHMANN07] : Innovation games, creating breakthrough products through collaborative play – Luke Hohmann – Addison Wesley 2007 – ISBN : 978 0 321 43729 7

[HUMBLE+11] : Continuous delivery, Reliable software releases through build, test and deployment automation – Jez Humble & David Farley – Addison Wesley / Signature series 2011

[HUNT+00] : The Pragmatic Programmer: From journeyman to master - Andrew Hunt & David Thomas - Addison Wesley 2000 - ISBN: 0-201-61622-X

[HUTTERMANN12] : DevOps for Developers, Integrate development and operations the agile way - Michael Huttermann - Apress 2012 - ISBN : 978-1430245698

[IGNACE+12] : La pratique du Lean Management dans l'IT, agilité et amélioration continue – Marie-Pia Ignace, Christian Ignace, Régis Medina & Antoine Contal – Pearson France 2012 – ISBN : 978-2-7440-6544-6

[JEFFRIES13] : Fractional Scrum or « Scrum-but » - https://www.scrum.org/ScrumBut ;

[JACOBSON+99] : The Unified Software Development Process - Ivar Jacobson, Grady Booch & James Rumbaugh - Addison Wesley / O.T. series 1999 - ISBN: 0-201-57169-2

[JOHNSON+12] : Scrum, a Breathtakingly Brief and Agile Introduction – Hillary Louise Johnson & Chris Sims – Dymaxicon 2012 – ASIN : B007P5N8D4 (Kindle edt.)

[KOLKELA08] : Test Driven, practical TDD and acceptance TDD for Java developers – Lasse Kolkela – Manning 2008 – ISBN : 1-932394-85-0 ; EAN13 : 978 1932394856

[LACEY12] : The Scrum Field Guide, practical advice for your first year - Mitch Lacey – Addison Wesley 2012 : ISBN : 978-0-321-55415-4

[LADAS08] : Scrumban, Essays on Kanban systems for lean software development – Corey Ladas – Modus Cooperandi Press 2008 – ISBN : 978 0578002149

[LARMAN03] : Agile and iterative development, a manager's guide – Craig Larman – Addison Wesley / ASD series 2003 – ISBN: 0-13-111155-8

[KNIBERG07] : Scrum and XP from the Trenches, how we do Scrum – Henrik Kniberg – InfoQ 2007 – ISBN : 978 1 4303 2264 1

[KNIBERG+10] : Kanban and Scrum, making the most of both – Henrik Kniberg & Mathias Skarin - InfoQ 2010 – EAN : 978 0 557 13832 6

[KNIBERG11] : Lean from the Trenches, managing large-scale projets with Kanban – Henrik Kniberg – Pragmatic Bookshelf 2011 – ISBN : 978-1-934356-85-2

[KRUCHTEN04] : The Rational Unified Process, an introduction, third edition – Philippe Kruchten – Addison Wesley / O.T. series 2004

[MARTIN08] : Clean Code, a handbook of agile software craftsmanship – Robert C. Martin – Prentice hall / Robert C. martin series 2008 – ISBN: 0-13-235088-2 ; EAN: 978 0 13 235088 4

[MARTIN11] : The Clean Coder, A code of conduct for Professional Programmers - Robert C. Martin - Prentice Hall / Robert C. Martin series 2011 - ISBN : 978 0 13 708107 3

[MAURYA12] : Running Lean 2nd edition, iterate from plan A to a plan that work – Ash Maurya – O'Reilly 2012 – ISBN : 978-1-449-30517-8

[MASKELL+05] : Lean Accounting : What's it all about ? – Brian H. Maskell et Bruce L. Baggaley – Lean Accounting Summit 2005 - http://issuu.com/addinquy/docs/leanaccountingdefined-target ;

[MAYER13] : The People's Scrum, Agile ideas for revolutionary transformation – Tobias Mayer – Dymaxicon 2013 – ISBN : 978 1 937 965150

[MCCONNELL04] : Code Complete, a handbook of software construction - Steve McConnell - Microsoft press 2004 - ISBN : 0-7356-1967-0

[MESSAGER10] : Gestion de projet agile, avec Scrum, Lean, Extreme Programming, 3ème édition – Véronique Messager-Rota – Eyrolles 2010 – EAN : 978 2 21212750 8

[MESSAGER12] : Coacher une équipe agile, guide à l'usage des Scrum Masters, managers … et de leurs equipes – Véronique Messager – Eyrolles 2012 – ISBN: 978-2-212-13414-8

[MORISSEAU12] : Kanban pour l'IT, une nouvelle méthode pour améliorer les processus de développement – Laurent Morisseau – Dunod 2012 – ISBN : 9782100578672

[NYGARD07] : Release It ! Design and deploy production-ready software – Michael T. Nygard – Pragmatic Bookshelf 2007 – ISBN : 0-9787392-1-3 ; EAN : 978 0 9787392 1 8

[OSTERWALDER+10] : Business Model Generation - lexander Osterwalder & Yves Pigneur - John Wiley & sons 2010 – ISBN : 978-2-7440-6487-6

[PICHLER10] : Agile Product Management with Scrum, Creating products that customers love – Roman Pichler – Addison Wesley / Signature series 2010 – ISBN : 978 0 321 60578 8

[PIXTON+09] : Stand Back and Deliver, Accelerating business agility – Pollyanna Pixton, Niel Nickolaisen, Todd Little & Kent McDonald – Addison Wesley 2009 – ISBN : 978 0 321 57288 2

[RAWSTHORNE+11] : Exploring Scrum : The fundamentals – Dan Rawsthorne with Doug Shimp – CreateSpace 2011 – ASIN : B0058WS4T4

[RIES11] : The Lean Startup – Eric Ries – Crown Business 2011 – ISBN : 978-0-307-88789-4

[ROTHMAN+05] : Behind Closed Doors, secrets of great management – Johanna Rothman & Esther Derby – The Pragmatic Bookshelf 2005 – ISBN: 0-9766940-2-6

[RUBIN13] : Essential Scrum, A practical guide to the most popular agile process – Kenneth S. Rubin – Addison Wesley / Signature series 2013 – ISBN : 978 0 13 704329 3

[SCHWABER95] : SCRUM development process – Ken Schwaber – Conference Oopsla 1995 ; http://issuu.com/addinquy/docs/scrum_oopsla_95 ;

[SCHWABER+99] : SCRUM, a Pattern Language for Hyperproductive Software Developpement, Ken Schwaber, Martine Devos, Yonat Sharon, Mike Beedle and Jeff Sutherland, *in* Pattern Languages of Program Design, vol. 4 - Neil Harrison, Brian Foote & Hans Rohnert edt. - Addison Wesley / Software Patterns series 2000 - ISBN: 0-201-43304-4 ; pp. 637 - 651

[SCHWABER+01] : Agile Software Development with Scrum - Ken Schwaber & Mike Beedle - Prentice Hall / series in agile software development 2001 - ISBN: 0-13-067634-9

[SCHWABER04] : Agile Project Management with Scrum – Ken Schwaber – Microsoft Press 2004- ISBN: 0-7356-1993-X

[SCHWABER+12] : Software in 30 days – Ken Schwaber & Jeff Sutherland – John Wiley & sons 2012 – ISBN : 978-1-118-20666-9

[SINEK09] : Start with Why, How great leaders inspire everyone to take action – Simon Sinek - Penguin Books 2009 – ISBN : 978 1 59184 2804 1

[SOFTHOUSE06] : Scrum in 5 Minutes – Softhouse

[SPINELLIS03] : Code Reading, The open source perspective – Diomidis Spinellis – Addison Wesley / Effective Software Development series 2003 – ISBN: 0-201-79940-5

[SUTHERLAND+11] : The Power of Scrum – Jeff Sutherland, Rini Van Solingen & Eelco Rustenberg – CreateSpace 2011 – ISBN : 1463578067 (Kindle edition)

[SUTHERLAND+13] : The Scrum Guide 2013, The definitive guide to Scrum : the rules of the game – Jeff Sutherland & Ken Schwaber – Scrum.org 2013 ; http://issuu.com/addinquy/docs/scrum-guide_2013 ;

[TAKEUCHI+86] : The New New Product Development Game - Hirotaka Takeuchi, Ikujiro Nonaka – *in* Harvard Business Review jan-feb 1986

[TATE05] : Sustainable Software Development, an agile perspective – Kevin Tate – Addison Wesley / ASD series 2005 – ISBN: 0-321-608-28608-1

[TAYLOR11] : The Principles of Scientific Management - Taylor, Frederick Winslow - Harper & Brothers 1911

[WEST11] : Water-Scrum-Fall Is the Reality of Agile for Most Organizations Today – Dave West – Forrester Research 2011 ; http://issuu.com/addinquy/docs/water-scrum-fall_0 ;

[WILLIAMS+02] : Pair Programming Illuminated - Laurie Williams & Robert Kessler - Addison Wesley 2002 - ISBN: 0-201-74576-3

L'auteur

Christophe Addinquy fut d'abord un adepte de la première heure des "Design Patterns". Au début des années 2000, il a suivi cette communauté après qu'elle ait donné naissance à la communauté agile.

Pour le contacter ou le suivre

mail addinquy@computer.org

twitter	@addinquy
LinkedIn	addinquy
Blog	http://freethinker.addinq.uy/
Google+	addinquy
Quora	addinquy

Remerciements et crédits

Cet article fait suite à la présentation du même nom faite au Printemps Agile à Caen en Mars 2014.

Remerciements

Tout d'abord, merci à Jean-Luc Lambert pour m'avoir invité au Printemps Agile 2014. Même si je l'ai exprimé à ma façon, avec mes convictions, l'idée majeure du Shu Ha Ri adapté à l'agilité est à mettre au crédit d'Alistair Cockburn à qui j'ai donc emprunté beaucoup. Merci à Laurent Sarrazin de m'avoir guidé sur les propos de la transformation agile d'organisation. Et un immense merci aux nombreux membres de la communauté agile avec qui j'échange régulièrement : ces échanges m'enrichissent et me poussent toujours au questionnement.