Disciplined Agile Delivery

IBM

Le développement agile n'est pas un effet de mode. Au moment de l'arrivée sur le marché des méthodes agiles mainstream - Scrum et Extreme Programming - les idées véhiculées n'étaient pas nouvelles, encore moins révolutionnaires. En fait, elles avaient été pour beaucoup décrites dans d'autres méthodes telles que Rapid Application Development (RAD), Evo et dans différentes versions d'Unified Process (UP), sans parler des classiques en la matière : tel que *'The Mythical Man Month'* de Frederick Brook. Dans ce contexte, il n'est pas surprenant qu'une collaboration étroite et unifiée entre équipes, travaillant sur le même site pour développer et livrer des logiciels, donne de meilleurs résultats que des projets menés dans des silos spécialisés plus préoccupés par des considérations individuelles que par la performance de l'équipe. Il n'est pas non plus surprenant que la diminution des volumes de documentation et des tâches administratives permette de réaliser des économies substantielles et d'accélérer les délais de livraison.

Le développement agile était autrefois considéré comme viable uniquement pour les petites équipes travaillant sur le même site. Plus récemment, il est apparu que certaines améliorations apportées à la qualité des produits, à l'efficacité des équipes et à la rapidité de livraison (pratiques agiles) ont permis à des équipes plus étendues d'adopter des principes agiles au sein de leurs environnements. Une récente étude menée par le magazine *'The Agile Journal'* a fait ressortir le fait suivant : 88 % des entreprises - dont un grand nombre de plus de 10 000 employés - utilisent ou envisagent d'utiliser des pratiques agiles dans le cadre de leurs conduites de projets. Le développement agile est en passe de devenir la méthode de développement logiciel dominante sur le marché. Cette tendance s'appuie également sur d'autres études du secteur dont une étude menée par *'Dr. Dobb's Journal'* qui note un taux d'adoption des techniques agiles de 76 %, et parmi les entreprises qui les ont adoptées, 44 % des équipes en moyenne appliquent des techniques agiles sous une forme ou une autre.

Toutefois, ces chiffres doivent être modérés : selon une étude ultérieure conduite par Ambysoft, il s'est avéré que seules 53 % des personnes qui se déclaraient être des "équipes agiles" étaient réellement agiles. Il est clair que les méthodes agiles ont été surmédiatisées depuis quelques années, ce qui a conduit à un abus et à une mauvaise compréhension du concept ; l'utilisation de processus était faible, voire absente. Pour un grand nombre d'équipes, cela s'est traduit par une situation anarchique et chaotique, entraînant des échecs cuisants de projets et un retour de bâton de la communauté informatique, qui préfère les approches plus traditionnelles.

Mais le développement agile, lorsqu'il est correctement exécuté, n'est pas une excuse pour être indiscipliné. Il est clair que l'exécution des méthodes de développement agile mainstream a toujours exigé une approche disciplinée - plus disciplinée peut-être que les approches traditionnelles, par exemple les méthodes de développement en cascade. Il ne faut pas confondre le cérémonial de nombreuses méthodes traditionnelles avec de la discipline : c'est plutôt là un signe de bureaucratie rampante - et souvent hors de contrôle. Toutefois, les méthodes de développement agile mainstream n'offrent généralement pas une quantité suffisante de conseils aux entreprises. Les implémentations matures de développement agile font ressortir un besoin fondamental dans les entreprises pour un degré de rigueur - gouvernance, planification de l'architecture et modélisation - que les méthodes de développement agile stratégique considèrent comme inutile. La plupart des méthodes de développement agile mainstream admettent que leurs stratégies nécessitent de nombreux ajouts et ajustements pour les équipes de plus de huit personnes travaillant à proximité. De plus, parmi les entreprises et les services publics du classement Fortune 1000, la plupart dispose d'équipes étendues et réparties géographiquement. Par conséquent, les efforts d'adaptation requis peuvent dans ce cas s'avérer coûteux et risqués. Il est donc temps de proposer une nouvelle génération d'infrastructures de processus agiles.

Voici les principales idées développées dans ce livre blanc :

- Les équipes sont les principales déterminantes du succès des projets informatiques.
- Aller vers une approche disciplinée de livraison agile constitue la première étape de l'élaboration des stratégies d'entreprise agiles.
- L'approche DAD (Disciplined Agile Delivery) représente une infrastructure de processus logiciels hybrides.
- Des stratégies agiles doivent être appliquées tout au long du cycle de vie de livraison des projets.
- Les équipes agiles sont plus faciles à diriger que les équipes traditionnelles.

L'importance du contexte : modèle d'évolution agile

Pour bien comprendre les besoins en matière d'infrastructure DAD, il convient tout d'abord d'appréhender la réalité de la situation telle qu'elle se présente. Le modèle ASM (Agile Scaling Model) représente une infrastructure contextuelle qui définit un plan d'adoption et d'adaptation de stratégies agiles, visant à relever les défis spécifiques que rencontrent les équipes agiles de développement logiciel. La première étape de ce processus consiste à adopter un cycle de vie Disciplined Agile Delivery qui tienne compte de l'évolutivité des stratégies de construction agiles, afin d'englober la totalité du processus de livraison - du commencement du projet au déploiement en environnement de production. La deuxième étape consiste à identifier les facteurs d'évolution applicables (le cas échéant) à une équipe projet, puis à adapter vos stratégies pour répondre aux complexités auxquelles les équipes doivent faire face.

Le modèle ASM (voir la Figure 1) définit trois catégories de processus :

1. **Le développement agile stratégique :** Les méthodes agiles stratégiques - Scrum, Extreme Programming et Agile Modeling - sont principalement tournées vers les activités orientées construction. Elles se caractérisent par des cycles de vie centrés sur la valeur - développement régulier de logiciels de haute qualité prêts à livrer par une équipe privilégiant la collaboration et l'organisation autonome. Ces équipes intègrent souvent un nombre réduit de personnes (< 15) qui travaillent sur le même site et développent des logiciels simples.

2. **Disciplined Agile Delivery**[1] **:**
   Ces méthodes - y compris l'infrastructure de processus DAD décrite dans le présent document et Harmony/ESW - concernent la totalité du cycle de vie (du début du projet à la production). En cas de besoin, ces méthodes apportent des techniques de gouvernance dans le but d'équilibrer l'organisation et de mêler une vision basée sur les risques à l'approche basée sur la valeur ajoutée. Cela renforce les chances de succès des projets. Tout comme les méthodes agiles stratégiques, ces méthodes concernent principalement de petites équipes travaillant sur le même site et développant des solutions simples.

3. **Agility@Scale :** Il s'agit de Disciplined Agile Delivery, à laquelle s'applique un ou plusieurs facteurs d'évolution. Les facteurs qui peuvent intervenir ici sont les suivants : taille de l'équipe, répartition géographique, répartition organisationnelle de l'entreprise, mise en conformité réglementaire, complexité culturelle ou organisationnelle, complexité technique et disciplines d'entreprise (architecture, réutilisation, gestion du portefeuille, par exemple).

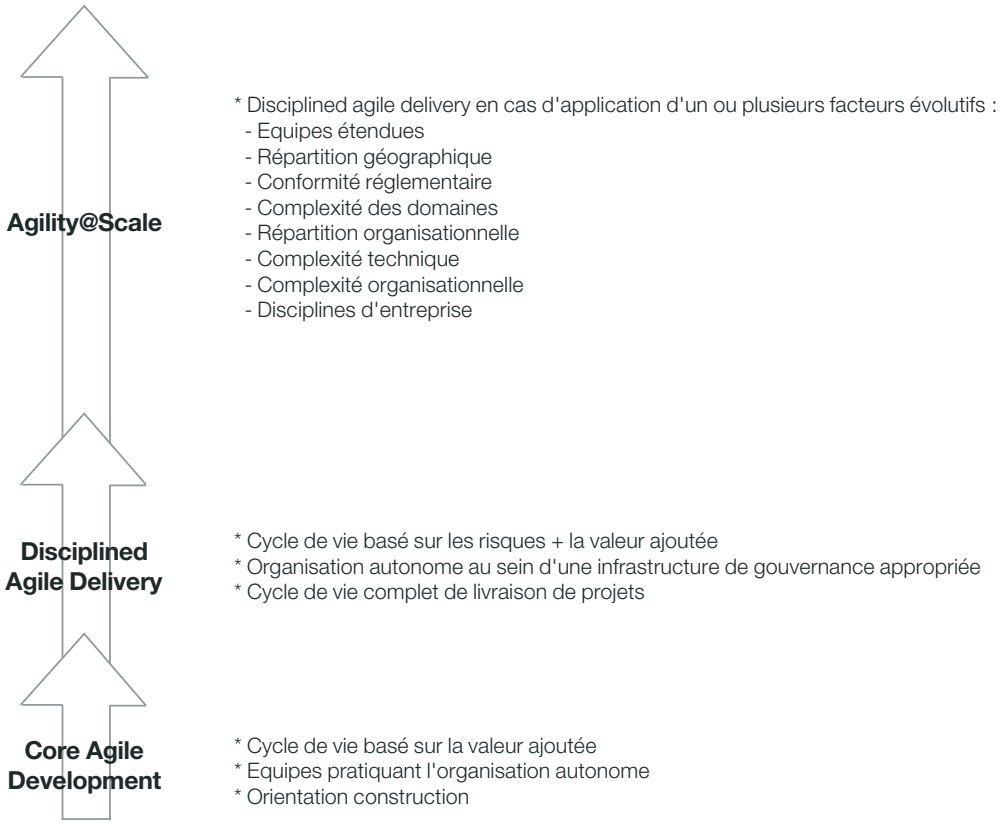


Figure 1 : Le modèle ASM (Agile Scaling Model)

Le présent document décrit l'infrastructure de processus DAD. Dans la plupart des cas, nous partons sur une hypothèse d'équipe restreinte (< 15 personnes) travaillant sur le même site ou sur des sites proches (même bâtiment, par exemple) et développant une solution relativement simple.

Qu'est-ce que l'infrastructure de processus DAD ?

Nous allons commencer par une définition :

"L'infrastructure de processus DAD (Disciplined Agile Delivery) est une approche hybride du développement et de la livraison de solutions informatiques privilégiant les personnes et orientée apprentissages. Son cycle de vie associant risque + valeur ajoutée est basé sur les objectifs et sur les besoins de l'entreprise".

D'après cette définition, nous voyons que plusieurs caractéristiques importantes se dégagent déjà au niveau de l'infrastructure de processus DAD. Il s'agit des caractéristiques suivantes :

- Priorité aux équipes
- Orientées apprentissage
- Agile
- Hybride
- Orientées solutions informatiques
- Cycle de vie de livraison des projets basé sur les objectifs
- Fondées sur les risques et la valeur ajoutée
- Connaissance de l'entreprise.

Pour mieux comprendre l'infrastructure DAD, explorons chacune de ces caractéristiques de façon plus détaillée.

Tout d'abord, examinons la priorité accordée aux équipes :

Alistair Cockburn fait référence aux personnes comme étant des “composants non linéaires et de premier ordre” au sein du processus de développement logiciel. Son observation, qui s'appuie sur une vaste expérience en ethnographie, est la suivante : les personnes et plus particulièrement leur mode de collaboration constituent les principales caractéristiques du succès des projets informatiques. Cette philosophie, qui se reflète dans la première affirmation de valeur du document intitulé ‘The Agile Manifesto’, imprègne l'infrastructure DAD.
Les membres des équipes DAD doivent appliquer une auto-discipline, une organisation autonome et se respecter les uns les autres. L'infrastructure de processus DAD permet d'améliorer l'efficacité des équipes grâce à un ensemble de conseils fournis, mais elle n'impose aucune procédure.

L'approche traditionnelle de transmissions d'informations formelles (composées principalement de documents) entre les différentes disciplines (exigences, analyses, conception, test et développement) crée des goulots d'étranglement , et constitue une perte importante de temps et d'argent. Les transmissions d'informations entre les personnes créent des malentendus et sont source d'erreurs ; dans le développement de logiciels Lean, elles sont décrites comme l'une des sept sources de gaspillage. Lorsque nous créons un document, nous ne documenterons pas l’intégralité de notre compréhension de ce que nous décrivons et inévitablement, certaines connaissances sont laissées de côté ; ce sont des connaissances tacites non transmises. Il est facile de se rendre compte qu'après de nombreuses transmissions, les livrables n'ont généralement plus grand chose à voir avec l'intention d'origine. Au sein d'un environnement agile, les frontières entre disciplines et les transmissions d'informations doivent être minimisées dans l'intérêt de l'équipe qui, dans cette approche, représente plus qu'un groupe de spécialistes.

Dans une approche DAD, nous mettons l'accent sur la stratégie des équipes intersectorielles qui se composent de personnes provenant de différents domaines. Il ne doit pas y avoir de hiérarchie au sein de l'équipe ; par ailleurs, les membres de l'équipe sont encouragés à varier leurs compétences et à effectuer des tâches appartenant à différentes disciplines (hors de leur propre discipline). Les connaissances acquises dans d’autres disciplines optimisent l'utilisation des ressources et diminuent les besoins de documentations et transmissions d'informations formelles.

En ce sens, les méthodes agiles renforcent les rôles principaux basés sur une grande variété de compétences au détriment des rôles basés uniquement sur un ensemble de compétences. Dans l'approche DAD, les cinq rôles principaux sont les suivants :

1. **Intervenant :** Une personne qui est matériellement impactée par le résultat de la solution est considérée comme un intervenant. Nous voyons que cela va au-delà de l'utilisateur final : en effet, un intervenant peut être un utilisateur (direct ou indirect), un responsable d'utilisateurs, un responsable expérimenté, un membre du directoire, le propriétaire du budget, un membre du service d'assistance, les auditeurs, le responsable de votre programme/portefeuille, les développeurs travaillant sur d'autres systèmes (en interaction avec le système en cours de développement) ou encore les professionnels de maintenance potentiellement affectés par le développement et/ou le déploiement d'un projet logiciel.

2. **Responsable produit :** Le responsable produit est la personne qui se fait le "porte-parole du client" dans l'équipe. Il fait état des besoins et des souhaits de la communauté à l'équipe de livraison agile. A ce titre, elle clarifie les informations concernant la solution et est chargée de la gestion de la liste des priorités que l'équipe mettra en œuvre pour livrer la solution. Elle ne peut pas répondre à toutes les questions, mais elle est chargée de suivre les réponses dans un délai raisonnable afin que l'équipe reste concentrée sur ses tâches. Le fait que cette personne puisse travailler en étroite collaboration avec l'équipe pour répondre aux questions relatives aux éléments de travail mis en œuvre permet de diminuer considérablement le besoin de documentation. Chaque équipe DAD (ou sous-équipe pour les vastes programmes organisés en équipe d'équipes) possède son propre responsable produit. Le deuxième objectif d'un responsable produit consiste à représenter le travail de l'équipe agile face à la communauté des intervenants sur le projet. Cela signifie organiser des démonstrations de la solution aux moments clés de l'évolution et communiquer l'état d'avancement du projet aux principaux intervenants.

3. **Membre de l'équipe :** Le membre de l'équipe se concentre sur la mise en place d'une solution à l'attention des intervenants projet. Les membres de l'équipe exécutent ainsi les tests, l'analyse, l'architecture, la conception, la programmation, la planification, l'estimation et d'autres activités requises tout au long du projet. Remarque : chaque membre de l'équipe n'est pas obligé de posséder l'ensemble de ces compétences, mais quelques-unes au minimum ; il doit tenter d'en acquérir d'autres petit à petit. Dans les méthodes agiles stratégiques, les membres de l'équipe sont parfois appelés développeurs ou tout simplement programmeurs. Toutefois, dans l'approche DAD, nous savons bien que chaque membre de l'équipe n'est pas nécessairement chargé de l'écriture du code.

4. **Chef d'équipe :** Le chef d'équipe est en quelque sorte le coach qui aide l'équipe à rester concentrée sur la livraison et sur l'atteinte des objectifs/engagements itératifs vis-à-vis du responsable produit. Il agit en tant que leader, facilitant la communication, encourageant l'équipe à optimiser les processus en toute autonomie, mettant à la disposition de l'équipe les ressources dont elle a besoin et éliminant le plus rapidement possible tout obstacle quant à la bonne marche des projets de l'équipe.

5. **Responsable architecture :** Le responsable architecture prend les décisions relatives à l'architecture et facilite la création et l'évolution de la conception globale de la solution. Au niveau d'un projet, l'architecture est une source importante de risque ; c'est pourquoi une personne doit veiller à ce que ce risque soit limité. Remarque : le responsable architecture ne dirige pas l'architecture de la solution, mais la formulation de cette architecture. Sur les petits projets, il est fréquent que le chef d'équipe soit également le responsable architecture.

Vous remarquerez qu'au sein de l'infrastructure de processus DAD, les rôles de testeur et d'analyste métier ne sont pas primordiaux. Ce qui est privilégié, c'est la polyvalence des différents membres de l'équipe. Par exemple, un membre de l'équipe spécialisé en exécution de tests peut proposer son aide

pour l'élaboration des exigences ou encore pour occuper temporairement le rôle de Scrum Master (chef d'équipe). Cela n'implique pas que chacun soit un expert dans tous les domaines, mais tout simplement que l'équipe doit parvenir à couvrir toutes les compétences requises et doit donc aller chercher ces compétences en cas de besoin. Les membres de l'équipe doivent en quelque sorte être des "spécialistes généralistes" (spécialistes dans une ou plusieurs disciplines, tout en possédant des connaissances générales dans d'autres disciplines). Plus important encore : les "spécialistes généralistes" souhaitent collaborer étroitement avec les autres personnes, partager leurs compétences et leur expérience, et bénéficier des compétences des personnes avec lesquelles ils travaillent. Une équipe composée de spécialistes généralistes nécessite peu de transmissions de données entre les personnes et affiche une bonne collaboration, car les personnes ont une juste appréciation des compétences et priorités des différentes disciplines informatiques. Elles peuvent s'attacher aux besoins plutôt qu'au contenu des spécialisations.

Les équipes DAD doivent présenter les caractéristiques suivantes :

- Auto-discipline - engagement uniquement à hauteur du travail réalisable et exécution de ce travail le plus efficacement possible
- Organisation autonome - estimation/planification de leur travail, puis collaboration itérative
- Connaissance personnelle - identification des éléments qui fonctionnent bien d'un point de vue personnel, analyse et ajustement en conséquence.

Bien que les personnes soient les principaux facteurs de succès des projets informatiques, il ne suffit pas, dans la plupart des cas, de constituer une bonne équipe et de la laisser se débrouiller. Dans ce cas, l'équipe est confrontée à plusieurs risques (trop de temps et d'efforts consacrés au développement de ses propres processus et pratiques, identification des mauvais processus et des mauvaises pratiques sans détecter les bons, adaptation inefficace de ces processus et pratiques). En d'autres termes, les personnes ne sont pas les seuls facteurs de succès. L'infrastructure de processus DAD permet de dispenser des conseils cohérents et éprouvés que les équipes agiles pourront utiliser pour limiter (voire éviter) les risques décrits ci-dessus.

Orientées apprentissage

Dans les années qui ont suivi la parution du document 'The Agile Manifesto', nous avons découvert que les entreprises les plus efficaces étaient celles qui facilitaient l'apprentissage au sein de leur environnement. Cet environnement axé sur l'apprentissage couvre trois aspects principaux. Le premier est l'apprentissage de domaines (l'exploration et l'identification des besoins des intervenants, puis, étape fondamentale, l'aide à la réalisation). Le deuxième est l'apprentissage visant à améliorer les processus au niveau individuel, de l'équipe et de l'entreprise. Le troisième est l'apprentissage technique qui s'attache à comprendre comment travailler efficacement avec les outils et les technologies utilisés pour le développement de la solution.

L'infrastructure de processus DAD propose différentes stratégies dédiées à l'apprentissage de domaines - modélisation des exigences initiales, livraison incrémentale d'une solution potentiellement déployable et participation active des différents intervenants tout au long du cycle de vie. Dans le cadre d'une infrastructure de processus DAD, il est recommandé d'adopter une approche rétrospective dans laquelle l'équipe identifie de façon explicite les améliorations de processus potentielles, définit une stratégie agile commune et effectue un suivi continu de ces améliorations. Au sein d'IBM Software Group, nous avons constaté que les équipes agiles qui ont adopté cette approche ont amélioré leur productivité ; celles qui ont assuré le suivi de la mise en œuvre de leurs stratégies d'amélioration ont connu un succès encore plus marqué. L'apprentissage technique est souvent naturel pour les professionnels de l'informatique, car ils aiment découvrir et utiliser de nouveaux outils, de nouvelles techniques et technologies. Mais cela peut être à double tranchant : ils apprennent de nouveaux concepts techniques, mais risquent par la même occasion de ne pas consacrer suffisamment de temps à la maîtrise d'une stratégie avant de passer à la suivante ou bien d'abandonner une technologie bien adaptée tout simplement parce qu'ils veulent faire quelque chose de nouveau.

De nombreuses stratégies générales permettent d'améliorer la capacité d'apprentissage. Vous pouvez améliorer la collaboration entre les personnes, ce qui favorise les apprentissages mutuels

(or, la collaboration signale l'agilité). Vous pouvez également recourir à des formations, à un mentorat ou à un coaching, car ce sont également des stratégies d'apprentissage efficaces. Une méthode moins intuitive consiste à mettre l'accent sur les compétences fondamentales plutôt que sur les compétences spécialisées, c'est-à-dire valoriser les spécialistes généralistes.

Les entreprises novatrices se font les ardents défenseurs des opportunités d'apprentissages hors des spécialisations habituelles (et des opportunités d'application correspondantes). Si vous avez déjà entendu parler de développement logiciel agile, ces stratégies ne vous sont sans doute pas inconnues, mais là où l'infrastructure de processus DAD pousse l'apprentissage encore plus loin, c'est dans le domaine de l'approche globale vis-à-vis de l'entreprise. En effet, les méthodes agiles stratégiques telles que Scrum et Extreme Programming sont généralement orientées projet, tandis que les équipes DAD exploitent l'écosystème de l'entreprise dans laquelle elles interviennent et l'optimisent. Par conséquent, elles doivent s'appuyer sur l'expérience des autres équipes agiles et prendre le temps de partager leurs propres expériences. L'implication est la suivante : votre département informatique doit investir dans une technologie de socialisation de l'expérience d'apprentissage entre équipes. En 2005, IBM Software Group a mis en place différents forums de discussion internes et différents wikis, ainsi qu'un centre de compétences (ce que certaines entreprises appellent les centres d'excellence) pour renforcer les pratiques d'apprentissage agile. Quelques années plus tard, IBM Software Group a adopté une stratégie Web 2.0 basée sur IBM®Lotus®Connections afin de développer l'apprentissage au sein de l'entreprise.

Agilité

L'infrastructure de processus DAD adhère aux valeurs et aux principes du document 'The Agile Manifesto', et vise à les optimiser. Les équipes qui suivent des processus itératifs ou agiles ont démontré leur capacité à fournir des résultats de meilleure qualité, à générer des retours sur investissement plus élevés, à améliorer la satisfaction des intervenants et à raccourcir leurs délais de livraison par rapport à une équipe adoptant une approche traditionnelle ou ad-hoc (absence de définition de processus). L'application de techniques telles que l'intégration continue, les tests de régression, le développement avec tests anticipés et les refontes permettent d'atteindre des niveaux élevés de qualité. L'augmentation du retour sur investissement provient du ciblage sur les activités à forte valeur ajoutée grâce à la mise en place de priorités, à l'automatisation des tâches informatiques pénibles, à l'organisation autonome, à une collaboration étroite et, en général, grâce à une façon de travailler plus intelligente (et non pas plus intensive). L'augmentation de la satisfaction des intervenants est liée à leur participation active, à la livraison incrémentale, à chaque itération, d'une solution potentiellement déployable et à leur modification en cours de projet des exigences.

Infrastructure de processus hybrides

L'approche DAD correspond à la formulation de nombreuses stratégies et pratiques élaborées à partir des méthodes de développement agile les plus répandues ou d'autres sources. L'infrastructure de processus DAD étend le cycle de vie de construction Scrum afin d'appréhender la totalité du cycle de vie de livraison, avec adoption des stratégies de développement agile et Lean. De nombreuses pratiques suggérées par cette approche font régulièrement l'objet de discussions au sein de la communauté agile (intégration continue, réunions quotidiennes de coordination, refonte, etc.) alors que, pour une raison ou une autre, cela n'est pas le cas concernant certaines pratiques avancées qui sont appliquées par une majorité d'entreprises. Ces pratiques avancées sont, entre autres, la modélisation des exigenceset de l'architecture initiales, ainsi que les tests de fin de cycle de vie.

L'infrastructure DAD est hybride : elle adopte et adapte des stratégies issues de différentes sources. Au sein des entreprises, nous avons souvent assisté à l'adoption de l'infrastructure de processus Scrum, puis à l'adaptation d'idées provenant d'autres sources, ce qui implique des efforts considérables. Cette stratégie semble être une bonne idée, surtout si vous êtes spécialisé dans le

conseil en adoption de processus agiles, mais vous allez constater que les entreprises ont tendance à adapter l'infrastructure Scrum de la même façon. Dans ce cas, pourquoi ne pas utiliser dès le départ une infrastructure de processus robuste avec ces tâches communes déjà réalisées ? L'infrastructure de processus DAD adopte des stratégies issues des méthodes suivantes :

1. **Scrum :** L'infrastructure Scrum est orientée conduite de projets et gestion des exigences (sous certains aspects spécifiques pour cette dernière). L'infrastructure DAD, quant à elle, adopte et adapte de nombreuses idées provenant de Scrum (utilisation d'une série d'éléments de travail par ordre de priorité, représentation des intervenants du projet par le responsable produit, production à chaque itération d'une solution potentiellement déployable). Toutefois, DAD a abandonné la plupart de la terminologie Scrum (sans parler du Scrum Master, scrum signifie mêlée de rugby en anglais et on se lance dans des échappées ou sprints et on ressort blessé des mêlées) à l'exception du terme "responsable produit".

2. **Extreme programming (XP) :** XP constitue une importante source de pratiques de développement pour DAD (intégration continue, refonte, développement basé sur les tests, propriété collective, entre autres).

3. **Agile Modeling (AM) :** Comme son nom l'indique, AM est la source des pratiques DAD en matière de modélisation et de documentation. Cela inclut la modélisation des exigences et de l'architecture, la modélisation des itérations, la documentation continue et les séances de groupe pour les modèles JIT (Just In Time).

4. **Unified Process (UP) :** L'infrastructure DAD s'inspire des instanciations agiles d'UP (OpenUP et Agile Unified Process, ou AUP) pour de nombreuses stratégies de gouvernance. Cela inclut notamment des stratégies telles que les étapes allégées et les phases explicites. Comme dans UP, nous accordons également de l'importance de la mise à l'épreuve de l'architecture dès les premières itérations et à la diminution de tous les types de risque dès le départ du cycle de vie.

5. **Agile Data (AD) :** Comme son nom l'indique, AD est une source de pratiques agiles relatives aux bases de données (refonte et test de bases de données, et modélisation agile de données). AD représente également une source importante de stratégies agiles d'entreprise. Les équipes agiles peuvent collaborer efficacement avec les architectes d'entreprise et avec les administrateurs de données.

6. **Kanban :** L'infrastructure DAD adopte deux concepts stratégiques issus de Kanban qui est une infrastructure Lean : la limitation du volume de travail en cours et les travaux de visualisation. Ces concepts viennent s'ajouter aux sept principes de développement logiciel Lean (éliminer les gaspillages, favoriser la qualité intrinsèque, créer de la connaissance, reporter la décision, livrer rapidement, respecter les personnes et améliorer le système).

Des solutions plutôt que des logiciels

L'approche DAD est axée sur la fourniture de solutions plutôt que sur la production de logiciels et c'est précisément à ce niveau que réside la véritable valeur ajoutée pour les intervenants. Les professionnels du secteur informatique vont bien au-delà du simple développement de logiciels. Les logiciels sont bien sûr prépondérants, mais pour répondre aux besoins des intervenants, nous sommes souvent amenés à mettre à niveau ou à remplacer des composants matériels, à changer les processus opérationnels/métier suivis par les intervenants, voire à modifier la structure organisationnelle dans laquelle ces intervenants opèrent. Cette évolution exige que votre entreprise remédie à certains problèmes soulevés dans le document 'The Agile Manifesto'. Nous soutenons totalement ce document, mais ses auteurs d'origine étaient en majorité des développeurs logiciels, des consultants en développement logiciel ou encore des personnes des deux catégories.

Bien sûr, ce document est orienté développement logiciel, car c'est l'une des nombreuses expertises comprises dans le cycle de vie de livraison de projets logiciels. Les infrastructures de processus DAD mettent l'accent de façon explicite sur les activités visant à résoudre les problèmes liés à l'expérience des utilisateurs, aux bases de données, aux processus métier et à la documentation. Cela va donc au-delà du développement logiciel.

Cycle de vie de livraison des projets basé sur les objectifs

DAD englobe la totalité du cycle de vie des projets (lancement, construction jusqu'à la mise en production de la solution). Nous voyons bien que chaque itération est différente. Les projets évoluent et les priorités changent tout au long du cycle de vie. Pour clarifier ce point, nous divisons le projet en phases avec des étapes allégées afin de nous assurer que nous sommes concentrés sur les points essentiels au moment opportun (modélisation initiale, modélisation de l'architecture, gestion du risque, planification du déploiement, etc.). En cela, notre approche diffère de celle des méthodes agiles mainstream qui sont généralement tournées vers les aspects de construction du cycle de vie. Dans la plupart des cas, aucune instruction claire n'est donnée concernant les méthodes de lancement et de distribution, et leur intégration dans le cycle de vie global.

A maintes reprises, lorsque nous avons collaboré avec des équipes qui avaient adopté l'infrastructure Scrum, nous avons constaté que l'adaptation du cycle de vie Scrum (voir l'illustration dans la Figure 2) ressemblait fortement à un cycle de vie de projet DAD.[2] Ce cycle de vie comprend plusieurs caractéristiques principales :

1. **Il s'agit d'un cycle de vie de livraison :** Le cycle de vie DAD étend le cycle de vie de construction Scrum et englobe explicitement la totalité du cycle de vie de livraison (du lancement d'un projet à la mise en production de la solution ou à sa sortie sur le marché).

2. **Les étapes sont explicites :** Le cycle de vie DAD est organisé en 3 étapes distinctes désignées par un intitulé et s'inspirant des rythmes agiles 3C.

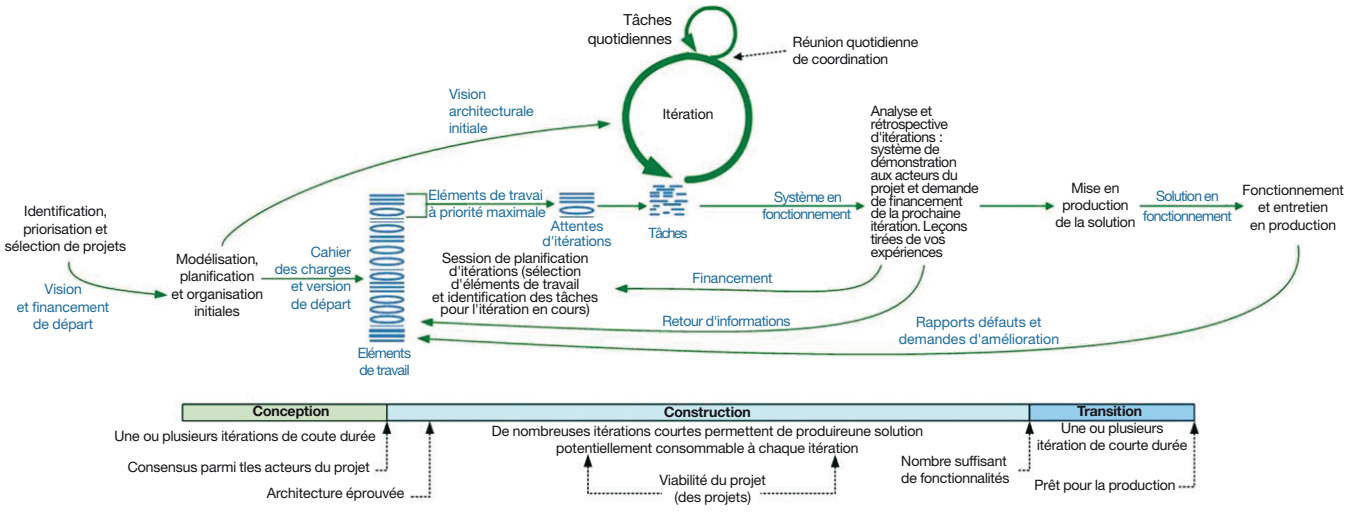


Figure 2 : Cycle de vie de l'infrastructure DAD (Disciplined Agile Delivery)

Objectifs de la phase de conception	Objectifs des itérations de la phase de construction	Objectifs de la phase de transition
- Identification de la vision du projet - Accord des acteurs du projet sur la vision - Identification de la stratégie technique, du cahier des charges et du plan de projet de départ - Formation de l'équipe initiale - Financement - Identification des risques	- Production d'une solution potentiellement consommable - Réponse à l'évolution des besoins des acteurs du projet - Avancement vers la version déployable - Conservation ou amélioration des niveaux de qualité existants - Supprimer les risques les plus élevés	- Garantir que la solution est à l'état Prêt pour la production - Garantir que les acteurs du projet sont prêts à recevoir la solution - Mise en production de la solution

Objectifs constants

- Respect de la mission du projet
- Evolution des compétences des membres de l'équipe
- Optimisation de l'infrastructure existante
- Amélioration des processus et de l'environnement de l'équipe
- Utilisation de l'infrastructure existante

Table 1 : Objectifs atteints lors de l'exécution d'un projet DAD

3. **Le cycle de vie de livraison apparaît en contexte :** Le cycle de vie DAD tient compte du fait que les activités d'identification et de sélection de projets ont lieu longtemps avant le lancement officiel (parfois même plusieurs années avant). La solution créée par l'équipe DAD sera mise en fonctionnement et bénéficiera d'un support une fois mise en production, c'est pourquoi les retours d'informations provenant de personnes qui ont utilisé les versions précédentes de la solution sont importants.

4. **Les étapes sont explicites :** Dans le cadre de l'approche DAD, les étapes constituent une stratégie importante de gouvernance et de réduction des risques.

L'une des difficultés à décrire une infrastructure de processus est la suivante : vous devez fournir suffisamment d'instructions pour une bonne compréhension, mais sans aller trop loin. Nous avons aidé différentes entreprises à améliorer leurs processus logiciels et au vu de cette expérience, il nous est apparu qu'on peut passer d'un extrême à l'autre. D'un côté, on peut trouver des descriptions de processus très détaillées (comme IBM Rational® Unified Process ou RUP), et de l'autre, des descriptions très succinctes (Scrum en est un parfait exemple). Avec RUP, la difficulté réside dans le fait que de nombreuses équipes n'ont pas les compétences pour en faire une adaptation appropriée, ce qui entraîne souvent un surcroît de travail. On rencontre la situation inverse avec Scrum : les équipes ne savent pas comment personnaliser l'infrastructure et consacrent beaucoup d'efforts à concevoir ou à apprendre à nouveau certaines techniques pour traiter les nombreux problèmes que Scrum ne couvre pas. Dans une situation comme dans l'autre, cela aurait pu être simplifié si un juste équilibre avait été trouvé entre ces deux extrêmes.

Pour répondre à cela, l'infrastructure DAD est basée sur des objectifs, comme l'illustre le tableau 1. Bien sûr, ces objectifs peuvent être traités de différentes façons ; par conséquent, en dresser simplement la liste n'apporte que peu de valeur ajoutée. Notre expérience est la suivante : cette approche basée sur les objectifs offre un cadre adapté aux équipes en matière de recommandations, tout en apportant une souplesse suffisante pour que les équipes soient capables d'adapter le processus au contexte dans lequel elles se trouvent.Le tableau 1 ne contient pas

la liste complète des objectifs destinés à votre équipe. On pourrait citer également les objectifs personnels des membres de l'équipe (objectifs en matière d'apprentissages spécifiques, de contribution à un travail intéressant, de rémunération et de reconnaissance). Par ailleurs, certains objectifs sont rattachés au projet en lui-même pour certains intervenants.

Examinons les différentes phases DAD pour mieux comprendre le contenu de l'infrastructure de processus DAD.[3]

Phase de démarrage

Avant d'élaborer ou d'acheter une solution, il est conseillé de consacrer du temps à l'identification des objectifs du projet. Dans le cadre des méthodes traditionnelles, beaucoup de temps et d'efforts sont consacrés à la planification des projets. Dans le cadre des méthodes agiles, on considère qu'il n'est pas nécessaire d'aller trop loin dans les détails dès le début, car on possède peu d'informations sur ce qui est véritablement requis et réalisable compte tenu des contraintes de temps et de budget. Les méthodes agiles mainstream recommandent de limiter les efforts consacrés à la planification initiale. Elles suggèrent de commencer et de faire le point en cours de projet. D'un point de vue objectif, disons que certaines méthodes agiles ont une itération de planification courte, appelée "Sprint 0" dans Scrum, et "Planning Game" dans Extreme Programming, dont les délais moyens sont d'environ 3,9 semaines.[4] Dans DAD, nous considérons qu'il faut définir correctement le cap avant d'entrer à pleine vitesse dans le projet, ce qui équivaut en général à un délai de quelques jours ou de quelques semaines. La Figure 3 présente les activités potentielles qui peuvent avoir lieu pendant la phase de démarrage. Cette phase se termine une fois que l'équipe a développé une vision claire de la version qui a été acceptée par les différents intervenants ayant également validé le reste du projet (ou au minimum l'étape suivante).

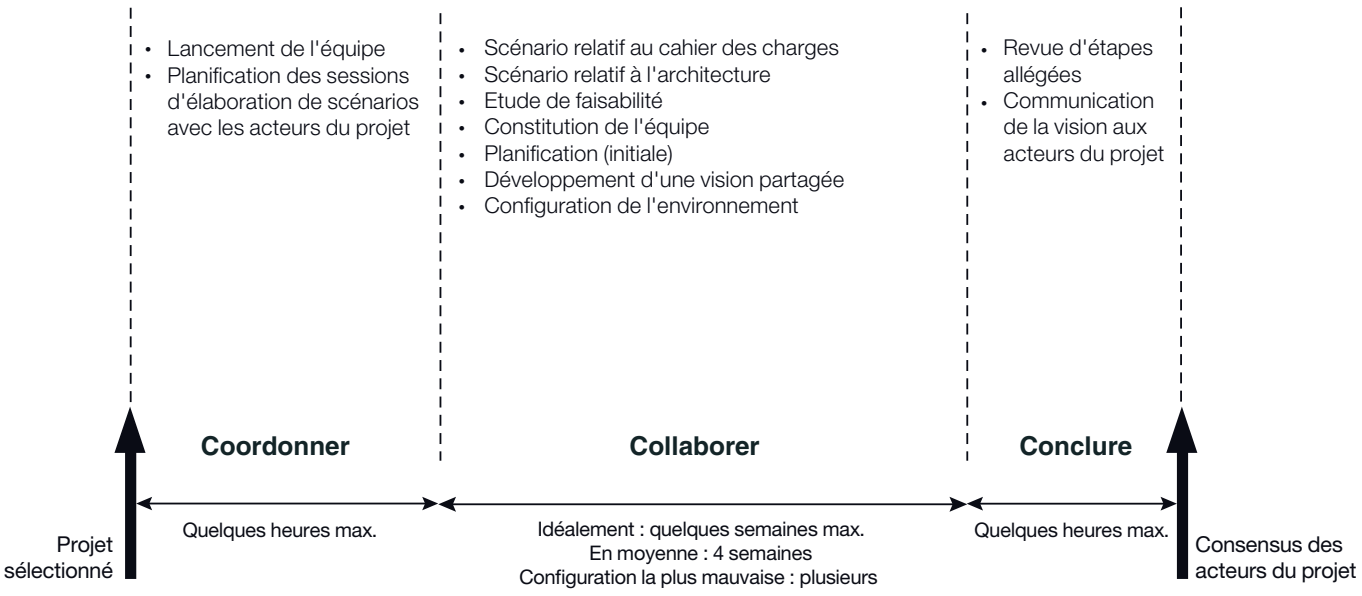


Figure 3 : Présentation de la phase de conception

Phase de construction

Dans DAD, la phase de construction représente la période de création des fonctionnalités requises. Le calendrier est divisé en plusieurs itérations limitées dans le temps. Ces itérations (activités potentielles présentées dans la Figure 4) doivent être de même durée pour un projet donné (une à quatre semaines, généralement deux ou quatre semaines) et ne se chevauchent pas dans la plupart des cas. A la fin de chaque itération, un incrément démontrable de solution potentiellement déployable est produit et un test de régression a lieu. La phase de construction se termine lorsqu'il y a un nombre suffisant de fonctionnalités pour justifier le coût de la transition (version MMR, c'est-à-dire version minimale commercialisable) et lorsque ce nombre est acceptable aux yeux des intervenants.

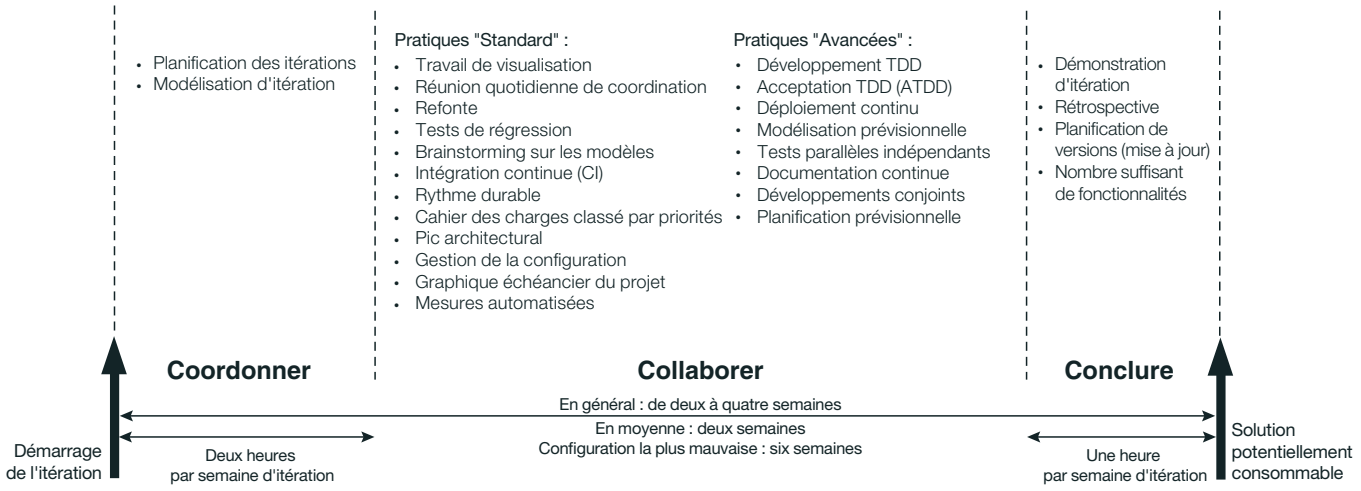
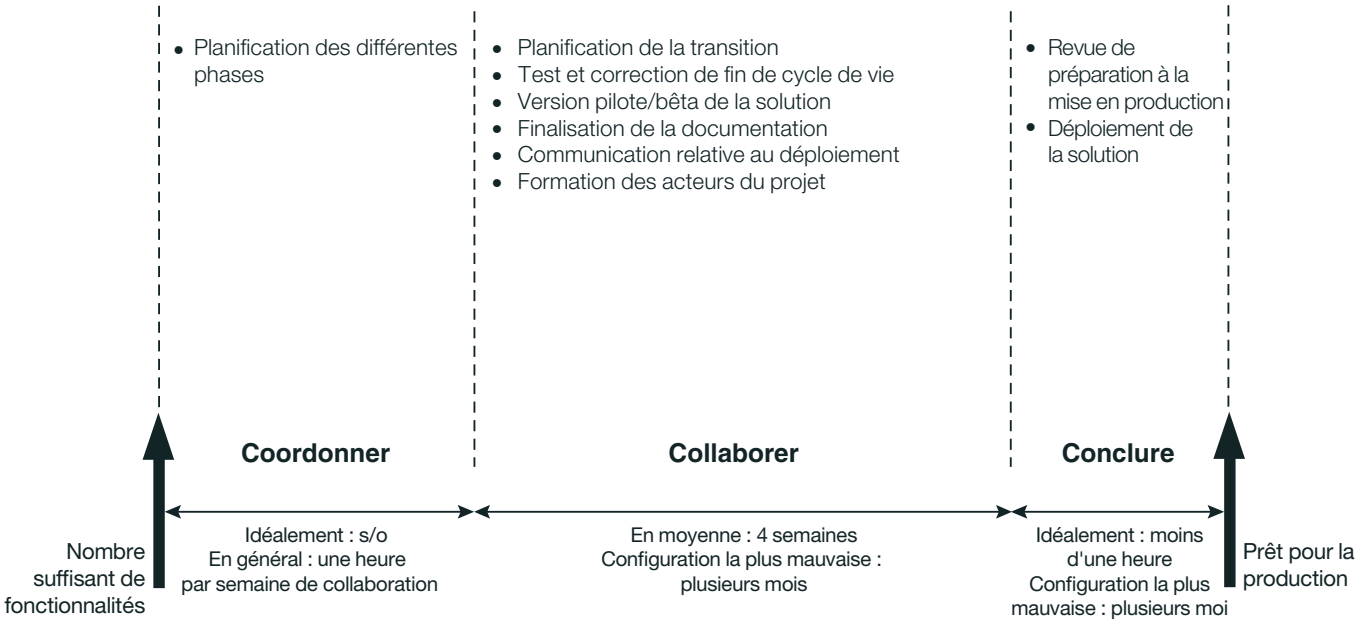


Figure 4 : Présentation de l'itération de construction

Phase de transition

La phase de transition traite de la mise en production du système (ou de la mise sur le marché dans le cas d'un produit de consommation). Comme vous pouvez le voir dans la Figure 5, la transition ne consiste pas uniquement à copier des fichiers sur un serveur. Le temps et les efforts consacrés à la phase de transition varient d'un projet à l'autre. Les logiciels "sous emballage" comprennent la fabrication et la distribution des logiciels et de leur documentation. Les systèmes internes sont généralement plus simples à déployer que les systèmes externes. Les systèmes à visibilité élevée peuvent nécessiter de nombreux tests de versions bêta réalisés par de petits groupes avant que la version finale ne soit distribuée à une plus grande échelle. La sortie d'un nouveau système peut nécessiter à la fois l'achat et la configuration de matériel, tandis que la mise à jour d'un système existant peut nécessiter des conversions de données et une coordination importante avec la communauté des utilisateurs. Chaque projet est différent. La phase de transition se termine lorsque les différents intervenants sont prêts et que le système est totalement déployé .



Présentation de la phase de transition

Connaissance de l'entreprise

Les équipes DAD, comme les autres équipes, travaillent dans l'écosystème de votre entreprise et tentent de tirer profit des opportunités qui se présentent. Pour emprunter une expression du secteur de l'environnement, ces équipes disciplinées "agissent localement et pensent globalement". Cela implique une collaboration étroite avec les équipes et les personnes suivantes : architectes d'entreprise et ingénieurs détachés pour l'utilisation et l'amélioration5 de l'infrastructure technique existante et planifiée ; architectes métier et responsables de portefeuilles pour l'adaptation à l'écosystème existant ; responsables de l'entreprise pour la gestion appropriée des différentes équipes ; administrateurs de données pour l'accès aux sources de données et l'amélioration de ces sources ; membres du service d'assistance au développement informatique pour l'interprétation et le suivi des conseils en matière d'informatique (codage, interface utilisateur, sécurité, conventions applicables aux données, etc.). En d'autres termes, les équipes DAD doivent adopter ce que Mark appelle un état d'esprit d'entreprise global.

A l'exception des start-up, les équipes de livraison de projets agiles ne travaillent pas en vase clos. Il existe souvent des systèmes en production : votre solution ne doit pas les affecter, mais elle doit tirer profit des fonctionnalités et données qui se trouvent en production. Il est fréquent que d'autres équipes travaillent en parallèle et vous pouvez bénéficier de leur travail (et inversement). Une vision commune peut exister au sein de votre entreprise et votre équipe doit s'y conformer. Même si cela ne vous semble pas évident, il peut y avoir une stratégie de gouvernance en place, ce qui permet d'optimiser les réalisations de votre équipe. Dans une politique d'autonomie en matière de discipline, la connaissance de l'entreprise est un facteur primordial, car en tant que professionnel, vous devez mettre en œuvre ce qui est bon pour l'entreprise, et pas uniquement ce qui est intéressant pour vous.

Malheureusement, ce n'est pas toujours le cas. En effet, certains "professionnels" de l'informatique choisissent parfois d'utiliser de nouvelles technologies (et même de les mettre en place au sein des solutions qu'ils produisent) pour développer leur CV plutôt que parce qu'elles sont les plus appropriées pour les projets en cours. Ils peuvent parfois décider de créer un outil en partant de zéro, d'utiliser de nouveaux outils de développement ou encore de créer de nouvelles sources de données alors que de très bonnes sources existent déjà au sein de l'entreprise. Nous pouvons (et devons) faire mieux, et voici comment :

1. **Utilisation des ressources de l'entreprise :** Vous pouvez (ou devriez) trouver de nombreuses ressources d'entreprise que vous pouvez utiliser et faire évoluer. Cela comprend les instructions de développement classiques (normes de codage, conventions de données, mesures de sécurité et normes d'interfaces utilisateur). Les ressources d'entreprise ne se limitent pas aux normes. Si votre société utilise une approche orientée architecture disciplinée pour l'élaboration des logiciels d'entreprise, vous disposerez d'une bibliothèque de plus en plus volumineuse de composants à base de services que vous pourrez réutiliser et optimiser dans le cadre de toutes les solutions actuelles et futures. Les équipes DAD s'efforcent d'utiliser une infrastructure commune (par exemple, pour utiliser chaque fois que cela est possible les technologies et sources de données approuvées, et pour optimiser leurs travaux en fonction de la vision d'avenir de votre infrastructure). Pour cela, les équipes DAD collaborent avec des professionnels (architectes, modélisateurs métier ou administrateurs de données, personnel d'exécution et ingénieurs détachés) tout au long du cycle de vie et plus particulièrement pendant l'élaboration des modélisations de la phase de démarrage. L'utilisation des ressources de l'entreprise renforce la cohérence et donc la simplicité en matière de maintenance. Elle diminue également les coûts et les délais de développement, ainsi que les coûts opérationnels.

2. **Optimisation de l'écosystème de l'entreprise :** La solution fournie par une équipe DAD doit au minimum s'inscrire dans l'écosystème organisationnel6 existant - c'est-à-dire les systèmes et les processus métier sous-jacents. Dans le meilleur des cas, elle doit optimiser cet écosystème. Pour cela, la première étape consiste à utiliser chaque fois que cela est possible les ressources existantes de l'entreprise, comme nous l'avons indiqué plus haut. Les équipes DAD collaborent étroitement avec le personnel d'exécution et d'assistance tout au long du cycle de vie, et tout particulièrement à l'approche de la mise en production pour s'assurer de la bonne compréhension de l'état actuel et des objectifs de l'écosystème de l'entreprise. Les équipes DAD peuvent, dans beaucoup de cas, s'appuyer sur une équipe de test indépendante qui se charge notamment des tests d'intégration en production, garantissant ainsi que la solution fonctionne dans l'environnement de production dans lequel elle se trouvera lors du déploiement.
3. **Surveillance ouverte et honnête :** Bien que les approches agiles soient basées sur la confiance, les stratégies de gouvernance intelligente sont basées sur une approche de confiance assortie de vérifications suivies de conseils. L'un des aspects importants de toute gouvernance appropriée est la surveillance des équipes de projet sous différents angles. Une bonne stratégie consiste à participer à la réunion de coordination quotidienne (solution recommandée par la communauté Scrum) pour suivre l'avancement du projet. Toutefois, même si nous y sommes favorables, son application n'est pas très efficace car les hauts responsables chargés de la gouvernance ont souvent un planning très chargé et ne s'occupent pas exclusivement d'une seuleéquipe. Scott Ambler a pu le constater dans le cadre de son enquête 'How Agile Are You?' menée en 2010. Une autre approche extrêmement efficace consiste à utiliser des outils instrumentés et intégrés (et plus spécialement les produits Jazz™tels que les logiciels IBM Rational Team Concert™) qui permettent de générer des indicateurs en temps réel et de les afficher dans les tableaux de bord des projets. Pour consulter un exemple de tableau de bord utilisé par l'équipe Jazz elle-même, rendez-vous sur le site jazz.net (stratégie commerciale ouverte). Une troisième stratégie consiste à suivre un cycle de vie basé sur le risque (voir la section suivante) et divisé en étapes explicites. Cela permet de disposer d'informations en retour cohérentes et régulières sur l'avancement du projet qui peuvent être transmises aux parties concernées.

Fondées sur les risques et la valeur ajoutée

L'infrastructure de processus DAD adopte un cycle de vie basé sur les risques et sur la valeur ajoutée. Il s'agit d'une version allégée de la stratégie UP (Unified Process). Les équipes DAD font tout leur possible pour réduire les risques classiques liés aux projets. Cela passe, par exemple, par l'obtention d'un consensus des intervenants autour de la vision d'avenir et la validation de l'architecture en amont du cycle de vie. L'approche DAD inclut également des contrôles formels visant à vérifier la viabilité continue du projet, si le nombre de fonctionnalités est suffisant et si la solution est prête pour la production. Elle est également basée sur la valeur ajoutée, ce qui permet de réduire le risque lié à la livraison étant donné que les équipes DAD produisent régulièrement des solutions potentiellement déployables. Comme on l'entend parfois "Eliminez les risques avant qu'ils ne vous éliminent" : cette philosophie illustre parfaitement l'approche DAD. Le cycle de vie DAD basé sur les risques et la valeur ajoutée est une version plus élaborée du cycle de vie basé sur la valeur ajoutée commun aux méthodes telles que Scrum et Extreme Programming.

Avec un cycle de vie basé sur la valeur ajoutée, vous pouvez produire des logiciels (ou, plus précisément, des solutions avec une approche DAD) prêts à être livrés à chaque itération. Les fonctions livrées représentent celles des exigences qui ont le plus de valeur ajoutée aux yeux des intervenants. Avec un cycle de vie

basé sur les risques et sur la valeur ajoutée, ces deux paramètres sont prioritaires lorsque vous évaluez les fonctions. Cela permet de traiter les risques communs aux projets informatiques le plus en amont possible. Pour schématiser, les cycles de vie basés sur la valeur ajoutée sont confrontés à trois risques importants : 1) non livraison, 2) livraison des mauvaises fonctionnalités et 3) politiques résultant d'un manque de visibilité concernant les livrables. Le traitement de ces risques est un bon début, mais ce n'est pas suffisant.

L'approche DAD intègre les stratégies standard des méthodes de développement agile et étend leur portée afin de réduire les risques communs des livraisons de projets informatiques :

1. **Solutions potentiellement déployables :** Les équipes DAD produisent des solutions potentiellement déployables lors de chaque itération de construction, étendant la stratégie Scrum aux questions de convivialité et de production de solutions (et pas uniquement de logiciels). Cela diminue les risques à la livraison, car les différents intervenants peuvent choisir le moment le plus opportun de mise en production de la solution.
2. **Démonstrations d'itérations :** A la fin de chaque itération de construction, l'équipe doit fournir une version de démonstration aux principaux intervenants. L'objectif principal est d'obtenir leur avis et d'améliorer la solution produite tout en diminuant le risque lié aux fonctionnalités. Le deuxième objectif consiste à présenter l'état d'avancement du projet en exposant le travail réalisé, ce qui permet de diminuer le risque politique (en supposant que l'équipe travaille correctement).
3. **Participation active des intervenants :** L'idée de base est la suivante : les différents intervenants ou leurs représentants (par exemple, les responsables produits) fournissent des informations et prennent des décisions en temps opportun, mais ils peuvent également être activement impliqués dans l'effort de développement lui-même. Par exemple, ces intervenants peuvent participer activement à la modélisation lorsqu'elle est réalisée avec des outils collaboratifs tels que du papier et des tableaux blancs. L'implication étroite des intervenants tout au long de l'itération (et pas uniquement lors des démonstrations) permet de diminuer le risque lié à la livraison et aux fonctionnalités, grâce au retour d'informations dont l'équipe bénéficie.

Les stratégies agiles mainstream sont un bon début en matière de gestion du risque informatique, mais cela ne va pas plus loin. DAD, en revanche, repose sur une approche basée sur des étapes explicites et simplifiées pour réduire les risques. A chaque étape du processus, une évaluation explicite de la viabilité du projet est réalisée par les principaux intervenants et une décision est prise quant à la poursuite du projet. Ces étapes (mentionnées dans le cycle de vie DAD illustré par la Figure 2) sont les suivantes :

1. **Consensus des intervenants :** A la fin de la phase de démarrage, l'objectif de cette étape consiste à s'assurer que les intervenants sont parvenus à un consensus raisonnable au sujet de leur vision de la version à livrer. En parvenant à cet accord, on obtient une réduction considérable du risque lié à la livraison et aux fonctionnalités, même si à ce moment-là, peu d'investissements ont été réalisés concernant le développement d'une solution opérationnelle. A cette étape, comptez en moyenne une annulation de projets comprise entre 10 et 15 %.

2. **Architecture éprouvée :** Lors des premières itérations de la phase de construction, l'accent est mis sur la réduction maximale des risques et des incertitudes liés au projet. Le risque peut être lié à de nombreuses choses (incertitudes associées aux exigences, productivité de l'équipe, risque métier et planification du risque, par exemple). Cependant, à ce niveau, la plus grande partie du risque lié à un projet informatique est d'ordre technologique (risque lié à l'architecture, plus précisément). Bien que les modèles d'architecture de haut niveau créés pendant la phase de démarrage soient d'une grande utilité pour l'élaboration de l'architecture, la seule façon d'être absolument certain que l'architecture soit adaptée aux exigences consiste à la tester avec un code fonctionnel. Il s'agit là d'une coupe verticale dans les composants matériels et logiciels qui touchent tous les points de l'architecture de bout en bout. Dans UP, on parle de "couverture architecturale" et dans Extreme Programming, on parle de développement en "fil d'acier" ou en "balle traçante". Les équipes DAD, qui développent des logiciels afin de tester l'architecture, parviennent à diminuer fortement le risque technique et les incertitudes : elles les identifient puis traitent à un stade précoce du projet les dysfonctionnements constatés.

3. **Viabilité continue :** Dans Scrum, l'idée est la suivante : à la fin de chaque sprint (ou itération), les différents intervenants évaluent la viabilité du projet. En théorie, c'est une bonne idée, mais en pratique, cela se déroule différemment dans la plupart des cas. Ce problème vient peut-être du fait que les décisions comportent trop d'enjeux politiques pour qu'une décision soit prise avant que les choses n'aillent vraiment mal (et psychologiquement, les personnes ne se rendent peut-être pas compte des problèmes pendant les fenêtres étroites des itérations agiles). Vous devez donc organiser des analyses d'étapes ciblées pour examiner explicitement la viabilité du projet. Nous vous recommandons d'en organiser une par trimestre. Cette étape peut se produire plusieurs fois pendant la phase de construction d'une version longue ou ne pas se produire du tout pour les versions plus courtes.

4. **Nombre suffisant de fonctionnalités :** La phase de construction est achevée lorsqu'un nombre suffisant de fonctionnalités a été atteint afin de justifier les dépenses de mise en production de la solution. La solution doit correspondre aux critères d'acceptation convenus en début de projet ou s'en rapprocher suffisamment pour que les principaux problèmes de qualité puissent être résolus pendant la phase de transition.

5. **Prêt pour la production :** A la fin de la phase de transition, les principaux intervenants devront déterminer si la solution doit être mise en production. Au cours de cette étape, les intervenants métier acceptent le système et le considèrent comme satisfaisant, les personnes chargées du fonctionnement du système en production acceptent les procédures et documentations correspondantes, et les personnes chargées des services d'assistance en environnement de production acceptent également ces procédures et documentations.

Conclusion

Nous avons une bonne et une mauvaise nouvelle. La bonne nouvelle, c'est que les méthodes agiles offrent clairement de meilleurs résultats que les approches traditionnelles et les entreprises utilisent, pour la plupart, des techniques agiles ou envisagent d'en utiliser à courte échéance. Mais la mauvaise nouvelle, c'est que les méthodes agiles mainstream (notamment Scrum, Extreme Programming (XP) et Agile Modeling (AM)) n'offrent qu'une approche partielle en matière de fourniture de solutions informatiques. L'approche DAD est une infrastructure de processus hybrides qui rassemble des pratiques et des stratégies communes issues de ces méthodes et qui englobe la totalité du cycle de vie de livraison de projets. L'approche DAD accorde une plus grande priorité aux personnes en affirmant que ces dernières

et leurs interactions sont des facteurs décisifs de succès des projets informatiques. Cette approche met l'accent sur la connaissance de l'entreprise en incitant les équipes à utiliser et à optimiser l'écosystème organisationnel existant, à suivre les instructions de développement et à collaborer avec les équipes chargées de l'administration. Le cycle de vie DAD comprend des étapes explicites ayant pour objectif de diminuer les risques liés aux projets et d'accroître la visibilité externe des principaux problèmes afin d'aider les hauts responsables à mettre en place des mesures de gouvernance appropriées.

Pour plus d'informations

Pour en savoir plus sur les différents thèmes de ce document, consultez les références suivantes :

- **Le document intitulé 'The Agile Manifesto' :** Les quatre valeurs décrites dans ce document sont détaillées sur le site http://www.agilemanifesto.org/ et les douze principes sous-jacents sont détaillés sur le site http://www.agilemanifesto.org/principles.html.
- **Etudes menées sur l'approche agile :** Nous mentionnons dans l'ensemble de ce document plusieurs études. L'étude *'Agile Journal Survey'* se trouve sur le site http://www.agilejournal.com/. Les résultats du *DDJ (Dr. Dobb's Journal)* et d'Ambysoft sont publiés sur le site http://www.ambysoft.com/surveys/, y compris les données source d'origine, les questions posées et les présentations résumant l'analyse de Scott Ambler.
- Priorité aux équipes. L'article d'Alistair Cockburn intitulé "Characterizing people as nonlinear, first-order components in software development" se trouve sur le site http://alistair.cockburn.us/Characterizing+people+as+ non-linear%2c+first-order+components+in+software+development. Il y explique pourquoi les équipes sont déterminantes dans le succès des projets informatiques. Dans "Generalizing Specialists: Improving Your IT Skills" sur le site http://www.agilemodeling.com/essays/generalizingSpecialists.htm, Scott explique pourquoi il n'est pas nécessaire de constituer des équipes composées de membres sur-spécialisés.
- **Modèle ASM (Agile Scaling Model) :** Le modèle ASM est décrit en détails dans le livre blanc IBM intitulé "The Agile Scaling Model (ASM): Adapting Agile Methods for Complex Environments" que vous trouverez sur le site ftp://ftp.software.ibm.com/common/ssi/sa/wh/n/raw14204usen/RAW14204USEN.PDF
- **Développement logiciel Lean :** Pour plus d'informations sur le développement logiciel Lean, vous pouvez commencer par consulter le document de Mary et Tom Poppendieck intitulé *'Implementing Lean Software Development: From Concept to Cash'* (Addison Wesley, 2007).
- **Processus hybrides :** Dans "SDLC 3.0 : Beyond a Tacit Understanding of Agile" (Fourth Medium Press, 2010), Mark Kennaley fournit une synthèse de l'historique des processus logiciels et explique l'intérêt des processus hybrides qui réunissent les meilleures idées issues de différents courants en la matière au cours des décennies passées.

De plus, les offres de financement d'IBM Global Financing peuvent vous aider à gérer de façon efficace votre budget, à éviter que votre technologie ne devienne obsolète, à améliorer le coût total de possession et à optimiser votre retour sur investissement. Les services GARS (Global Asset Recovery Services) peuvent également vous aider à répondre à vos besoins en matière d'environnement grâce à de nouvelles solutions dont la consommation d'énergie est optimisée. Pour en savoir plus sur les solutions IBM Global Financing, consultez le site Web suivant : ibm.com/financing

À propos des auteurs

Scott W. Ambler est responsable de la méthodologie Agile et Lean pour IBM Rational ; il travaille avec des clients d'IBM dans le monde entier pour les aider à améliorer leurs processus logiciels. Outre la création de l'approche DAD (Disciplined Agile Delivery), il a créé les méthodologies suivantes : Agile Modeling (AM), Agile Data (AD), Agile Unified Process (AUP) et Enterprise Unified Process (EUP). Il a également créé le modèle ASM (Agile Scaling Model). Scott Ambler est le co-auteur de 19 livres *(Refactoring Databases*, *Agile Modeling*, *Agile Database Techniques*, *The Object Primer 3rd Edition et The Enterprise Unified Process)*. Il contribue depuis longtemps au magazine intitulé Dr. Dobb's Journal. Sa page personnelle est www.ambysoft.com.

En 2007, **Mark Lines** a participé à la création d'UPMentors. En tant que coach en approche de développement agile “discipliné”, il aide les entreprises à améliorer tous les aspects du développement logiciel. Il s’intéresse plus particulièrement à la diminution du gaspillage que l'on rencontre dans de nombreuses sociétés informatiques et propose des approches concrètes d'accélération de l'exécution et d'optimisation de la qualité grâce à l'application des techniques agiles et Lean. Mark Lines intervient fréquemment dans des émissions consacrées à l'informatique et écrit des articles dans de nombreux magazines spécialisés. Le site Web de sa société est www.UPMentors.com. Vous pouvez le contacter à l’adresse électronique suivante : Mark@UPMentors.com.

[1] Nous sommes désolés de la confusion induite par l'utilisation du même terme pour deux notions différentes, mais nous les avons distingués comme suit : approche disciplinée de livraison agile (en minuscules) fait référence à la catégorie et Disciplined Agile Delivery (en anglais) fait référence à l'infrastructure de processus.

[2] Dans cette version, nous utilisons le terme 'itération' au lieu du terme 'sprint' et 'éléments de travail' au lieu de 'backlog de produit'.

[3] Pour ceux d'entre vous qui sont familiarisés avec UP, nous avons adopté les mêmes noms de phases, mais nous avons combiné les phases UP suivantes : Elaboration et Construction. Nous avons conservé l'orientation Elaboration qui consiste à explorer puis à tester l'architecture avec du code fonctionnel dans le cadre d’une étape d'architecture éprouvée. Le fait que la phase Elaboration ne constitue plus une phase distincte permet de rationaliser le cycle de vie DAD.

[4] Résultats d'une enquête menée en 2009 par Ambysoft.

[5] Les équipes qui pratiquent l'approche de livraison agile disciplinée font tout leur possible pour diminuer le niveau de dette technique dans l'entreprise en suivant la maxime des campeurs et randonneurs du monde entier : Essayez de laisser ce monde un peu meilleur qu'il ne l'était quand vous y êtes venus.

[6] Nous sommes désolés d'avoir dû utiliser un jargon d’entreprise, mais "écosystème organisationnel" est un terme précis.

Compagnie IBM France
17 avenue de l'Europe
92275 Bois-Colombes Cedex

Produit aux États-Unis d'Amérique
Avril 2011

RAW14261-FRFR-00