

Les principes Lean et agiles appliqués à la gouvernance
du développement de logiciels et de systèmes

Livre blanc

Mars 2009

Rational software



Lean Thinking et Gouvernance du développement.

*Scott W. Ambler, Leader de la Communauté de développement Agile,
Rational Methods Group, IBM Software Group*

Per Kroll, Responsable Méthodes, Rational Software, IBM Software Group

Sommaire

3	Application du Lean Thinking à la gouvernance de développement logiciel	15	Pratiques organisationnelles
4	Définition de la gouvernance	15	Adéquation entre la stratégie de Ressources Humaines et les valeurs IT
5	Principes d'efficacité de la gouvernance IT	15	Alignement des valeurs IT avec les parties prenantes
5	Processus	17	Pratiques pour les processus de développement
5	Cycle de vie des artefacts	17	Développement itératif
5	Risques	19	Étapes basées sur les risques
6	Adéquation	20	Adaptation des processus
6	Comportement	21	Amélioration continue
6	Déploiement	22	Conformité intégrée
6	Automatisation	23	Pratiques pour les mesures
6	Principes du développement logiciel Lean	23	Mesures simples et appropriées
7	Élimination du gaspillage	24	Contrôle continu des projets
7	Qualité intégrée	25	Pratiques applicables en matière de rôles et de responsabilités
7	Création de connaissances	25	Encouragement à l'autonomie d'organisation des équipes
8	Report d'engagement	27	Adéquation de la structure d'équipe à l'architecture
8	Livraison rapide	28	Pratiques applicables aux stratégies et aux standards
8	Respect des personnes	28	Environnement de la gestion du cycle de vie intégré
8	Optimisation globale	29	Actifs de l'entreprise valorisés
9	Catégories de gouvernance de développement	30	Architectures flexibles
11	Pratiques et principes	31	Conclusion
11	Entité de gouvernance pragmatique		
12	Livraison de programme échelonnée		
14	Pipeline des projets pilotés par les métiers		
14	Développement basé sur scénarios		

Points clés

Une gouvernance efficace se focalise sur la motivation et sur la facilitation.

Lean Thinking appliqué à la gouvernance du développement logiciel

Comme les équipes de développement adoptent de plus en plus les méthodes de développement Agile, des problèmes se posent au niveau des approches traditionnelles en matière de gouvernance IT. Ces approches telles que COBIT (Control Objectives for Information and Related Technology), PMI (Project Management Institute) avec OPM3 (Organizational Project Management Maturity Model)– sont souvent trop lourdes en pratique pour les équipes de développement qui souhaitent adopter davantage de pratiques Agiles ou Lean (concept d’optimisation des processus par l’élimination des gaspillages).

En effet, une gouvernance efficace de développement Lean ne s’obtient pas au moyen de commandes ou de contrôle. Il s’agit plutôt de la mise en application de comportements et de pratiques appropriés, via des techniques collaboratives. Il est bien plus efficace de motiver les personnes à exécuter les tâches appropriées que de les forcer à le faire.

L’approche de gouvernance de développement Lean permet d’adapter les philosophies de développement logiciel Agile et Lean à la gouvernance IT traditionnelle, à partir d’un cadre léger et collaboratif, qui reflètent la réalité des organisations IT modernes.

Ce livre blanc commence par une présentation des principes d’une gouvernance IT efficace et présente leur rapport avec un sous ensemble de pratiques plus spécifiques liées à la gouvernance de développement. Il présente ensuite les principes du développement logiciel Lean. Le coeur de ce livre blanc est consacré à la description de 18 pratiques qui définissent une approche Lean en matière de gouvernance de projets de développement logiciel au sein de votre structure IT. Ces pratiques indiquent comment intégrer la gouvernance aux outils, aux processus et aux recommandations de développement, afin de faciliter la conduite de projets de livraison de logiciels et de systèmes. Elles amènent le Lean Thinking, qui a révolutionné la gestion de la chaîne de production (Supply Chain Management), au domaine de la gouvernance du développement de logiciels et de systèmes.

Points clés

La bonne nouvelle, c'est que vous suivez sans doute déjà certaines de ces pratiques. La mauvaise nouvelle, c'est que votre entreprise doit sans doute consacrer plusieurs années à l'adoption de chaque pratique, et qu'il n'y a aucun ordre d'implémentation établi. Toutefois, ces pratiques sont synergiques et peuvent être adoptées de façon incrémentale. Chaque nouvelle pratique renforce votre capacité à implémenter efficacement les pratiques suivantes.

Définition de la gouvernance

Dans le domaine de l'IT et de l'ingénierie système, la gouvernance permet d'établir des chaînes de responsabilités, d'autorité et de communication qui supportent les objectifs de l'entreprise et sa stratégie globale. Elle établit également des mesures, des règles, des standards et des mécanismes de contrôle qui permettront aux différents intervenants de s'acquitter de leurs rôles et responsabilités avec efficacité.¹ Ceci est réalisé, en équilibrant les risques et le retour sur investissement, en mettant en place des processus et des pratiques efficaces, en définissant les directions et les objectifs du département, et enfin en déterminant les rôles des différentes personnes au sein de ce département.

Gouvernance et gestion sont deux notions distinctes : la gouvernance s'intéresse à l'organisation de l'extérieur, et la considère comme un système qui doit posséder la structure et les processus appropriés dans le but de fournir un flux de valeur. La gestion, quant à elle, adopte une vision interne et vérifie que la structure et les processus ont été mis en place efficacement.

La gouvernance de développement représente un sous-ensemble important de la gouvernance d'ingénierie système et IT, couvrant la conduite de projets de développement de logiciels et de systèmes. Basée sur l'expérience importante des équipes de développement Agile, IBM croit que les entreprises ayant adopté les cadres COBIT, OPM3 ou d'autres frameworks similaires, peuvent gagner en allégeant leur approche à la gouvernance des projets de développement sans rien sacrifier aux contrôles essentiels et audits nécessaires.

Il est possible d'alléger la pression pesant sur les équipes de développement sans pour autant perdre le contrôle.

Points clés

Les principes de la gouvernance IT sont le fondement de la gouvernance du développement.

Pour vous aider à améliorer votre cadre de gouvernance, IBM a développé une palette entière de pratiques qui favorisent l'adoption d'un développement Lean. Un grand nombre de ces pratiques s'applique également à la gouvernance IT et Ingénierie Systèmes au sens large ; toutefois, ce livre blanc traite spécifiquement de la gouvernance des projets de développement Agile et Lean.

Principes de Gouvernance IT efficace

En mai 2007, dans l'édition d' IBM Rational Edge,² Murray Cantor and John D. Sanders décrivaient sept principes qui guident la gouvernance IT. La gouvernance du développement est un sous-ensemble de la gouvernance IT : par conséquent, ces principes sont directement applicables à la gouvernance de développement Lean. Ils concernent les processus, le cycle de vie des artefacts, les risques, la pertinence, le comportement, le déploiement et l'automatisation.

Processus

La gouvernance est un processus qui s'applique aux processus ayant besoin d'être gouvernés. Des règles et des standards sont appliqués aux processus de développement ; des décisions sont mises en application au sein des processus, et ces derniers sont mesurés et contrôlés.

Cycle de vie des artefacts

Les cycles de vie des artefacts produits par les processus gouvernés guident la solution de gouvernance. Une partie de la gouvernance du développement logiciel consiste à bien comprendre les artefacts produits par les équipes de projets (tests exécutables et logiciels en fonctionnement, par exemple), puis à surveiller l'évolution de ces artefacts tout au long du développement.

Risques

Les mesures et les contrôles doivent être ajustés en fonction du niveau de risque. Lorsque le risque est faible, vous pouvez vous focaliser sur la mesure des activités de l'équipe. Inversement, lorsque le risque est élevé, vous devez focaliser vos efforts pour aider l'équipe à diminuer les risques encourus.

Points clés

Pertinence

Les besoins de l'entreprise déterminent le niveau et le style de la gouvernance adoptée. Par exemple, les équipes de développement qui créent des logiciels critiques pour la vie doivent adopter une approche de gouvernance plus stricte que celles qui sont chargées du développement d'un site Web d'information.

Comportement

La solution de gouvernance détermine le comportement organisationnel. D'un point de vue de la gouvernance du développement, l'objectif consiste à créer un environnement de développement favorisant l'attente des objectifs de l'entreprise. Les maximes « Ce que vous vendez, c'est votre organisation » et « Vous obtenez ce que vous mesurez » motivent les entreprises à créer un programme efficace de gouvernance de développement.

Déploiement

La solution de gouvernance doit être mise en œuvre de façon incrémentale. Vous ne pouvez pas mettre en œuvre en une seule fois toutes les pratiques décrites dans ce livre blanc. Vous devez commencer par sélectionner celles qui répondent à vos problèmes immédiats, avant d'adopter les autres.

Automatisation

La technologie rend la solution de gouvernance plus puissante et non obstrusive. Plus vous intégrez ou automatisez vos pratiques de gouvernance de développement (via l'automatisation et la culture d'entreprise), plus elles ont de chances d'être suivies.

***Tirer les leçons de la fabrication
Lean permet d'optimiser la
gouvernance IT.***

Principes du développement logiciel Lean

Dans un ouvrage intitulé *Implementing Lean Software Development*,³ Mary et Tom Poppendieck montrent comment les sept principes de la fabrication Lean peuvent être appliqués pour optimiser le flux de valeur IT dans son ensemble. Ces principes contiennent des méthodes pratiques et mesurables de transformation des processus de livraison de logiciels.

Points clés

L'intégration de la gouvernance aux processus permet aux développeurs de rationaliser l'exécution des actions.

Elimination des gaspillages

Le Lean Thinking consiste à considérer comme gaspillage toute activité n'ajoutant pas directement de la valeur au produit fini. Dans le domaine du développement logiciel, les trois plus grandes sources de gaspillage sont l'ajout de la fonctionnalité supplémentaire, le changement continu du périmètre et des frontières organisationnelles. Les frontières organisationnelles sont responsables d'une augmentation des coûts de l'ordre de 25 %, en raison des zones tampon qui ralentissent les temps de réponse et interfèrent avec les échanges de communications. Il est primordial que les équipes de développement soient autorisées à s'organiser et à travailler d'une façon qui reflète les tâches à accomplir, et non plus selon la définition des rôles fonctionnels des différents membres.

Qualité intégrée

Mary et Tom Poppendiecks ont fait une observation très simple : si vous détectez régulièrement des problèmes lors de votre processus de vérification, cela signifie que votre processus peut être défectueux. Lorsque vous notez régulièrement que vos développeurs réalisent des choses qu'ils ne sont pas censés effectuer, ou au contraire qu'ils n'effectuent pas ce qu'ils devraient faire, votre approche de la gouvernance est à revoir. Toutefois, il ne faut pas faire de la gouvernance un ensemble d'activités venant s'ajouter à votre processus logiciel. La bonne stratégie consiste plutôt à intégrer la gouvernance à votre processus, en la rendant la plus simple possible, pour aider les développeurs à réaliser les tâches appropriées.

Création de compétences

La planification est utile, mais l'apprentissage est essentielle. Votre objectif est de favoriser les stratégies comme le développement itératif, susceptible d'aider les équipes à identifier ce dont les parties prenantes ont vraiment besoin, puis à agir en fonction de ces besoins. Il est important également de disposer d'un ensemble réutilisable de standards et de recommandations facilement modifiable, qui permette de répondre aux besoins spécifiques des projets. De plus, l'obtention de retours cohérent et dans les délais est également importante, tant au sein de l'équipe qu'au niveau du programme, via le contrôle continu de mesures simples et adaptées.

Points clés

Pour pouvoir livrer plus rapidement des systèmes de grande qualité, vous devez adopter un développement itératif et limiter le travail de chaque équipe à sa capacité.

Report d'engagement

Il n'est pas nécessaire de commencer le développement d'un logiciel par la définition d'un cahier des charges exhaustif. Vous pouvez soutenir efficacement les activités de l'entreprise par des architectures flexibles, tolérantes aux changements, et en reportant les décisions irréversibles le plus tard possible dans le projet. Souvent, le report de décisions nécessite la capacité d'associer les scénarios aux fonctionnalités développées dans les applications au sein de multiples projets.

Livraison rapide

Il est possible de livrer rapidement des systèmes de grande qualité. En limitant le travail d'une équipe à sa capacité propre, vous pouvez établir un workflow fiable et répétitif. Une stratégie de gouvernance efficace n'exige pas que les équipes réalisent davantage de tâches qu'elles ne peuvent en faire, mais elle leur demande de s'organiser et de déterminer ce qu'elles sont capables d'accomplir. Au niveau de l'entreprise, il est important que les programmes puissent générer une valeur métier selon un rythme défini par les projets les plus rapides, et non par les projets les plus lents.

Respect des personnes

Mary et Tom Poppendiecks observent également que l'engagement et l'implication du personnel présentent des atouts de taille. La conséquence : la nécessité d'une stratégie au niveau des ressources humaines qui consiste à rendre plus autonomes les équipes ITs plutôt qu'à les contrôler.

Optimisation de l'ensemble

Si vous souhaitez qu'une gouvernance efficace s'applique à vos efforts de développement, vous devez raisonner globalement. Vous avez besoin de comprendre les processus métiers que chaque projet individuel soutient - qui parcourent souvent plusieurs systèmes. Vous devez gérer les programmes de systèmes interdépendants, afin de livrer un produit complet aux parties prenantes. Les mesures prises doivent permettre de déterminer la façon dont vous générez de la valeur métier, car c'est la raison d'être de votre département IT.

Points clés

Contrairement à l'orientation explicite de contrôle de la gouvernance traditionnelle, la gouvernance Lean utilise des stratégies collaboratives pour motiver implicitement les équipes.

Catégories de gouvernance de développement

La gouvernance traditionnelle utilise souvent des stratégies à base de commandes et de contrôles. Ces stratégies consistent à gérer et à diriger les équipes de développement, à l'aide de passerelles et de déclencheurs qui tentent de mettre en œuvre les règles et d'empêcher les violations. Cette approche, bien qu'efficace dans certaines situations, peut s'avérer totalement inopérante pour de nombreuses entreprises. En effet, elle consiste principalement à concevoir le framework de la gouvernance et à gérer l'effort correspondant, mais en pratique, peu de choses sont faites.

En revanche, la gouvernance Lean s'appuie sur des stratégies collaboratives qui autonomisent et motivent implicitement les membres des équipes. Par exemple, l'approche traditionnelle en matière de recommandations de codage comprend leur création et mise en application via des contrôles formels, ainsi qu'une correction du code incorrect. En revanche, l'approche Lean comprend la rédaction des recommandations de façon collaborative avec les programmeurs, l'explication de leur importance à toutes les personnes concernées et la fourniture des outils et du support simplifiant l'élaboration du code par les développeurs, en utilisant les guidelines de façon continue.

Points clés

Les pratiques de gouvernance Lean alignés avec les six catégories de gouvernance IT.

La Figure 1 illustre les relations entre les pratiques de gouvernance Lean. Elle montre leur adéquation avec les six principales catégories de gouvernance IT : mission et principes, organisation, processus, mesures, rôles et responsabilités, règles et standards.

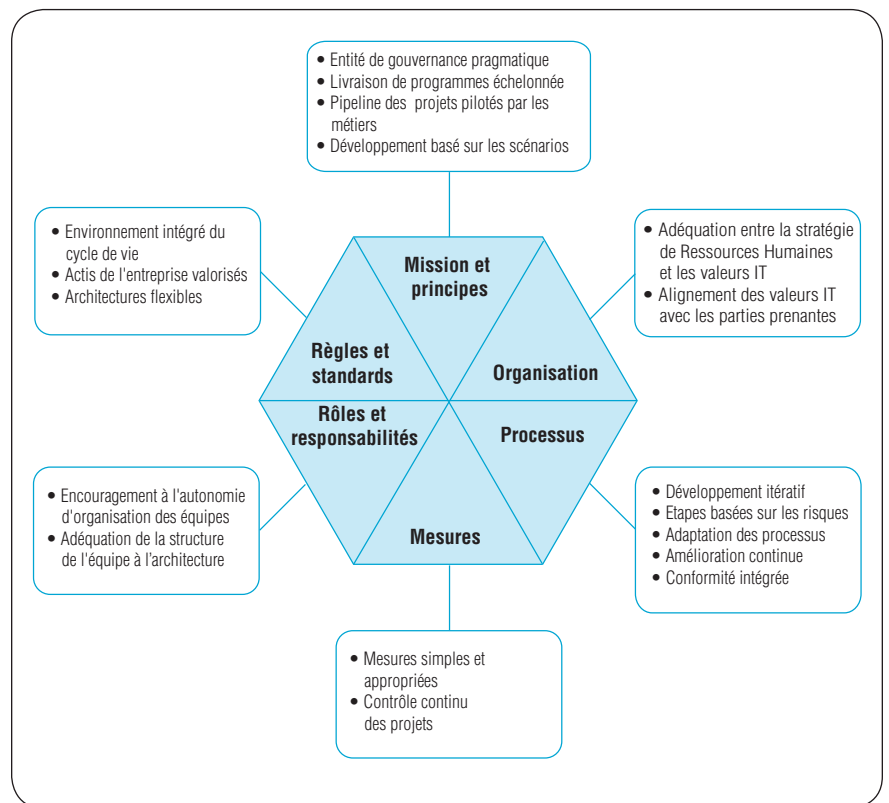
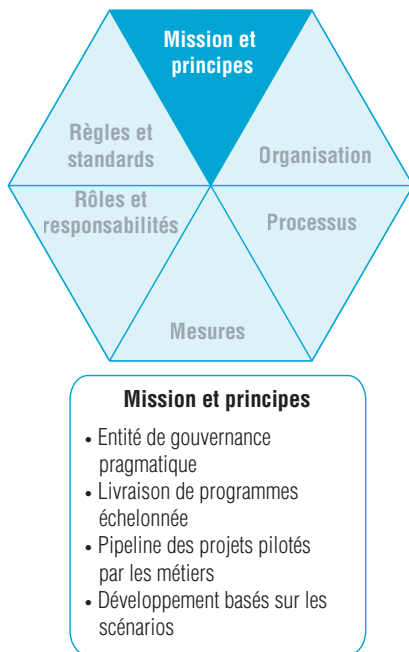


Figure 1 : les 18 pratiques de gouvernance Lean de développement logiciel sont en adéquation avec les 6 catégories de gouvernance.



Pratiques pour missions et principes

IBM identifie quatre pratiques Lean⁴ dans la catégorie de gouvernance pratiques pour missions et principes.

Entité de gouvernance pragmatique

Un programme de gouvernance ne s'exécute pas lui-même, mais un groupe de personnes est chargé de l'exécuter : l'entité de gouvernance. La façon dont cette entité s'organise et se comporte représente un facteur clé d'efficacité globale du programme de gouvernance.

Pour soutenir les pratiques de développement Lean, l'entité de gouvernance pragmatique doit respecter les personnes en se focalisant d'abord sur l'autonomisation des professionnels de l'IT, puis en les contrôlant et en les gérant. Pour cela, elle crée un environnement dans lequel les personnes peuvent être efficaces en pratique, et pas uniquement en s'appuyant sur des théories managériales. Cet environnement favorise l'adoption de stratégies, procédures et pratiques adaptées aux situations spécifiques et donnent aux équipes l'accès aux ressources dont elles ont besoin (y compris un accès direct aux parties prenantes) ; cette entité dispense conseils, support et mentorat aux équipes qui ont dévié par rapport aux résultats attendus.

Grâce à l'aide de l'entité de gouvernance pragmatique, les équipes ITs pourront plus facilement se conformer au programme de gouvernance, car ce sera plus simple pour elles. Les différentes personnes concernées pourront faire des processus et des stratégies une réalité, afin d'aider l'entreprise à atteindre ses objectifs. L'alternative sera un environnement dans lequel les équipes font tout ce qui est nécessaire pour se conformer à la structure de gouvernance « commande et contrôle ». A l'issue des revues, elles reprennent alors la gestion du projet de la façon qui leur semble la plus adaptée ; cela génère deux réalités différentes : la situation réelle et la situation présentée hors de l'équipe.

Points clés

Pour éviter tout état d'esprit conflictuel, il est important d'inclure l'organisation gouvernée dans l'entité centrale de gouvernance.

Une approche efficace de développement Lean repose sur la création d'une petite équipe centrale (souvent appelée centre de compétences de gouvernance), qui est étendue à des membres des organisations IT gouvernées et des organisations métiers appropriées. Inclure ces représentants dans cette entité de gouvernance évite de subir un état d'esprit conflictuel et permet de vérifier que la gouvernance est appropriée et possible à mettre en œuvre. Dans ce cas, ce qui importe le plus, c'est de choisir les bonnes personnes. Très souvent, les volontaires sont les personnes que vous souhaitez le moins affecter à cette mission.

Livraison de programmes échelonnée

La livraison de programmes échelonnée vous permet d'optimiser la totalité du programme, tout en terminant rapidement vos projets. Les programmes, qui sont des regroupements de projets liés, doivent être délivrés de façon incrémentale. Plutôt que de conserver une livraison dans l'attente d'un sous-projet, chaque sous-projet doit s'engager sur une date de livraison prédéfinie. Si le sous-projet n'est pas prêt pour cette date, il est affecté à la prochaine livraison, ce qui minimise l'impact sur les clients du programme. C'est un peu comme avec les horaires de train. Si le projet rate un train, il attend le suivant. Certes, en raison des dépendances entre les différents projets, il peut arriver qu'un projet non livré entraîne plusieurs retards.

Comme l'illustre la Figure 2, l'utilisation d'un projet de contrôle permet de coordonner l'exécution du programme, tout en conservant une bonne souplesse d'exécution des projets individuels au sein de ce projet de contrôle. Les itérations jouent ici un rôle fondamental, car elles offrent des points d'ancrage stables, qui permettent aux projets d'être intégrés et validés.

Points clés

Les différents projets d'un programme doivent être aussi autonomes que possible.

Les avantages d'une livraison de programme échelonnée sont nombreux : Premièrement, le fait de regrouper les projets en fonction d'objectifs métier et de les gérer comme un programme vous permet d'atteindre plus facilement ces objectifs. Deuxièmement, l'utilisation d'un projet de contrôle sert de plan de gouvernance pour le programme, qui peut ainsi être tourné vers la réduction des risques et des écarts, et vers la création de valeur. Troisièmement, grâce au fractionnement d'un programme potentiellement étendu, vous pouvez délivrer de la valeur vis-à-vis des sous-objectifs métiers, de façon incrémentale. Enfin, l'exécution semi-indépendante de projets permet d'accroître l'efficacité du développement : chaque projet bénéficie de toute la souplesse tactique requise pour augmenter la productivité.

La bonne stratégie consiste à gérer les programmes avec les projets à faible couplage via l'exécution d'un projet de contrôle selon les quatre phases de la méthodologie IBM Rational® Unified Process® (IBM RUP®): inception, élaboration, construction et transition.⁵ Idéalement, une approche évolutive doit être adoptée pour les différents projets (instanciation de RUP ou d'autres processus tels que Scrum ou OpenUP, par exemple).

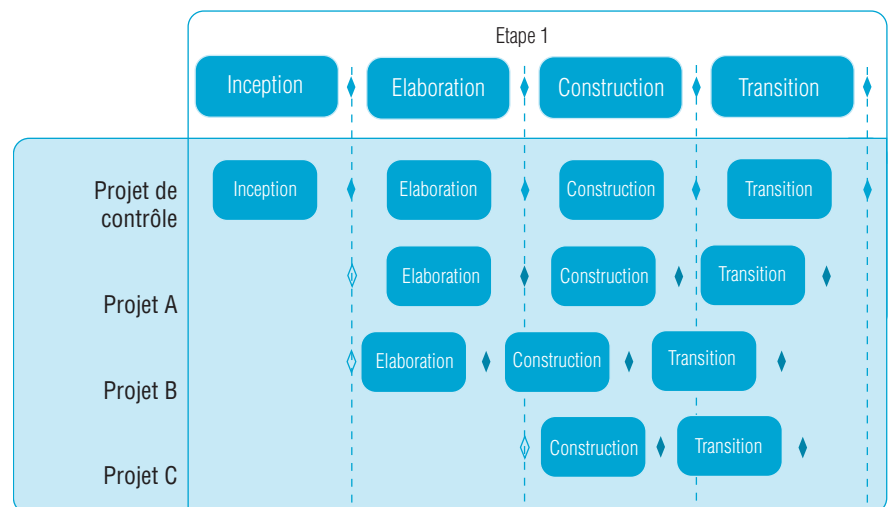


Figure 2 : comme l'illustrent Bittner et Spence dans l'ouvrage intitulé *Managing Iterative Software Development Projects*, la diminution des risques et la création de valeur sont réalisées par la gestion des étapes d'un programme via le projet de contrôle.

Points clés

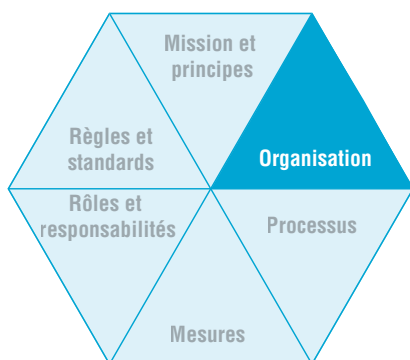
Un système doit refléter les besoins complets d'une entreprise, et pas uniquement le périmètre défini au sens strict.

Pipeline de projets piloté par les métiers

En matière de projets ITs, la demande dépasse systématiquement les ressources disponibles. Le pipeline des projets pilotés par les métiers maximise la valeur métier des investissements de développement en permettant aux organisations de prioriser et d'optimiser des projets alignés avec les objectifs métiers. Pour cela, vous pouvez adopter des stratégies de gestion de portefeuilles telles que les tableaux de pilotage (scorecard), qui permettent d'évaluer chaque projet à l'aide de paramètres identifiés comme des mesures de la valeur métier. Le Lean Thinking encourage les entreprises à limiter les paramètres à cinq et à s'appuyer sur les scores comme un mécanisme d'aide aux décisions métiers afin de prioriser plutôt que d'utiliser les mesures au sens strict.

Développement basé sur scénarios

Le développement basé sur les scénarios permet de bénéficier du contexte métier dans lequel les projets de développement deviennent efficaces. La vue d'ensemble ne peut pas être défini sans une compréhension de ses parties, et inversement. Si vous ne connaissez pas l'impact des différentes parties d'un projet sur la solution dans son ensemble, vous vous enlisez par la construction des composantes qui ne s'assembleront pas correctement. Pour comprendre le fonctionnement du métier d'entreprise, vous pouvez identifier des scénarios d'utilisation à la fois au niveau de l'entreprise et du projet, en utilisant des approches telles que les flux de cas d'utilisation⁷, ou encore des fils vert (green threads)⁸ Ces techniques permettent aux équipes de projet d'avoir une vue d'ensemble et de comprendre comment les différentes parties s'inscrivent dans cette vue. Par ailleurs, elles optimisent la collaboration des équipes et servent de mécanisme de contrôle, car elles attirent l'attention des développeurs sur les intégrations système.



Organisation

- Adéquation entre la stratégie des Ressources Humaines et les valeurs IT
- Alignement des valeurs IT avec les parties prenantes

Pratiques organisationnelles

Deux pratiques Lean favorisent particulièrement le climat organisationnel approprié afin de soutenir le développement agile.⁹ L'application de ces pratiques vous aidera à optimiser l'utilisation de l'une de vos ressources stratégiques : vos employés.

Adéquation entre la stratégie des Ressources Humaines et les valeurs IT

Embaucher un personnel technique, le faire évoluer et le garder au sein de l'entreprise nécessite des stratégies différentes de celles qui s'appliquent au personnel des autres secteurs que l'IT. Pour récompenser les bonnes performances, vous devez recourir à des mesures incitatives adaptées à leur état d'esprit. Par exemple, de nombreux professionnels du secteur IT souhaitent élargir leurs compétences techniques pour pouvoir participer à des projets plus complexes qui les motivent davantage. En revanche, ils sont souvent peu intéressés par la gestion de grandes équipes. Vous risquez de perdre des professionnels qualifiés si les seuls postes à pourvoir sont des postes de management. L'application d'une stratégie de Ressources Humaines efficace peut vous permettre d'augmenter votre pool de ressources et de conserver votre personnel qualifié.

Alignement des valeurs IT avec les parties prenantes

Si vos pratiques métier ne sont pas adaptées, vous risquez de faire échouer facilement vos projets de développement logiciel. Par exemple, de nombreuses entreprises s'attachent à obtenir une estimation budgétaire précise en début de projet de développement logiciel. Cette approche n'est pas réaliste, parce que les besoins évoluent tout au long de la durée de vie d'un projet¹⁰, et parce qu'il y a de nombreuses incertitudes qui peuvent conduire à des écarts par rapport à votre estimation.

Points clés

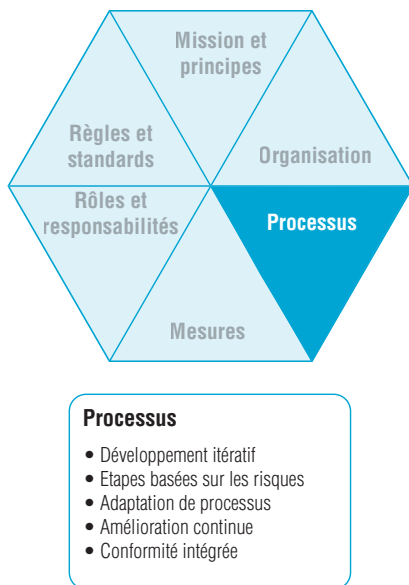
Les estimations précises des coûts, en début de projet, sont irréalistes.

Les parties prenantes doivent jouer un rôle actif au sein des équipes de projets de développement.

Outre le fait qu'elle manque de réalisme, la demande d'une estimation précise en début de projet encourage les équipes de développement à adopter des pratiques risquées (définition détaillée des exigences à un stade précoce du projet, par exemple).¹¹ Il est attendu que des contraintes de coût et de calendrier pèsent sur les équipes de développement logiciel, mais la manière dont celles-ci s'appliquent doit être le reflet de la réalité du cycle de développement et laisser aux équipes de développement la part de souplesse dont elles ont besoin pour rester efficaces.¹²

En résumé, les parties prenantes doivent adopter des stratégies réalistes en matière d'utilisation des ressources ITs (financement des projets, documentation des exigences, niveau d'implication des parties prenantes au sein des équipes de projets de développement logiciel).

Le Lean Thinking encourage les parties prenantes à être des participants actifs au sein des équipes de développement. Les professionnels de l'IT doivent être responsables de la formation des parties prenantes en ce qui concerne les des fondamentaux de développement moderne de logiciel, afin qu'elles comprennent bien les différents choix qui s'offrent à elles et les implications de leurs décisions. De leur côté, les professionnels de l'IT doivent apprendre les bases de gestion de la valeur métier, afin d'avertir que les coûts risquent de mettre en danger la valeur métier attendu d'un projet. Les avantages de ces pratiques comprennent la probabilité accrue de succès du projet et l'amélioration des aspects économiques du développement Logiciel grâce à des décisions plus avisées.



Pratiques pour les processus de développement

Cinq pratiques Lean favorisent l'adoption de stratégies de conduite de projet efficaces.¹³

Développement itératif

En adoptant une approche de développement itératif, vos projets sont organisés selon une séquence d'itérations courtes (parfois appelées « sprints »¹⁴). Chaque itération possède une durée bien déterminée calendrier, ainsi qu'un ensemble bien défini d'objectifs. L'objectif de chaque itération est la production de code complet, sans défaut et fonctionnant correctement.

Lors de chaque itération, vous vous appuyez sur l'expérience des itérations précédentes pour faire évoluer les exigences, l'analyse, la conception, l'implémentation et les tests jusqu'à ce que le produit final soit terminé (voir la Figure 3). Cette approche présente plusieurs avantages clés. Tout d'abord, la longueur de chaque itération est fixe, mais pas son périmètre. En cas d'écart par rapport à l'estimation initiale de l'effort à fournir (en raison d'événements ou de difficultés imprévisibles), c'est le périmètre de l'itération qui s'incline et qui doit être réévalué au cours de la prochaine itération. Ensuite, la définition de « produit final » et la notion de « complétude » peuvent évoluer considérablement au cours de l'exécution du projet, lorsque les parties prenantes et les personnes chargées de la mise en œuvre se mettent d'accord sur le fait que cela satisfait mieux les objectifs métiers. Chaque itération du projet augmente la compréhension partagée des exigences. Enfin, le respect des durées des itérations accélère les prises de décisions et oblige à se focaliser sur ce qui est important - la livraison dans les délais impartis de logiciels qui fonctionnent. Il est à noter que la livraison n'est pas seulement réalisée au niveau de la production IT. Elle concerne parfois un environnement de test. La livraison dans les délais impartis de logiciels qui fonctionnent permet d'augmenter le nombre de feedbacks, car les parties prenantes peuvent facilement se rendre compte si l'IT a bien compris ce qui leur a été demandé. Elle favorise également la gouvernance basée sur les faits, car un logiciel qui fonctionne correctement est une représentation concrète du travail accompli par une équipe (tandis que les artefacts secondaires, tels que plans ou spécifications, ne sont que les prévisions de ce qu'une équipe peut livrer à un moment donné).

Points clés

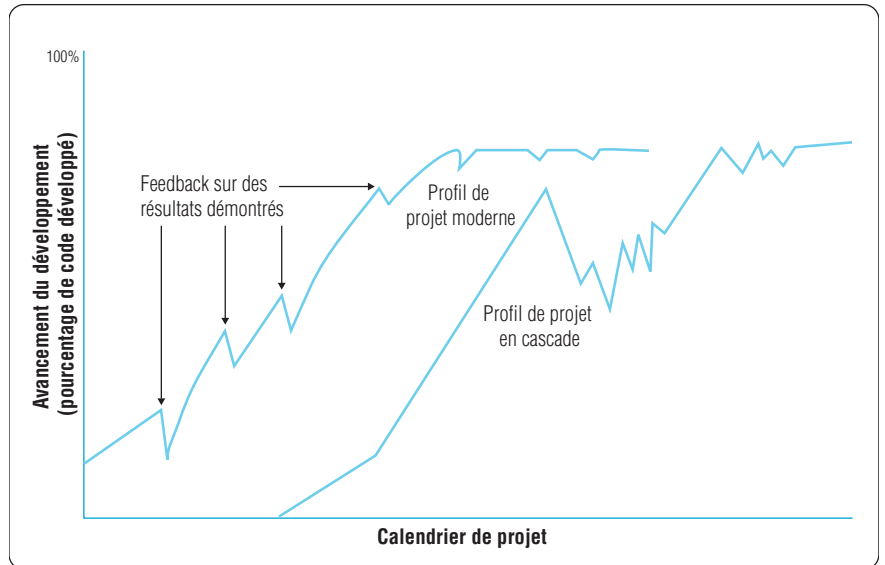


Figure 3 : grâce aux démonstrations fréquentes et aux feedback des parties prenantes, l'apport de corrections mineures au début du cycle d'un projet itératif (moderne) mène plus rapidement vers le succès qu'un projet en cascade.

Les itérations courtes et bornées dans le temps réduisent le risque et augmentent la gouvernance basée sur les faits.

Plus important encore, une approche de projet itératif augmente vos capacités à construire des systèmes qui répondent aux besoins changeants des parties prenantes. IBM recommande d'appliquer par défaut des itérations d'une durée de quatre semaines.¹⁵ Si vous vous rendez compte qu'une autre durée est plus appropriée dans le contexte de votre projet vous l'appliquez pour refléter ce contexte. Toutefois, la longueur des itérations doit être la plus courte possible, car cela réduit le risque d'alourdir vos processus de livraison de logiciels avec de la bureaucratie inutile. Les itérations doivent se suivre sans temps mort : Toutes les activités (de l'évaluation de l'état d'avancement à la planification, en passant par l'exécution de l'itération suivante) font partie intégrante d'une itération, et ne constituent pas des processus externes superposés.

Points clés

L'implémentation des fonctionnalités par ordre de priorité diminue le risque, car elle permet de valider l'architecture à travers le code qui fonctionne, à une phase précoce du projet.

Étapes basées sur les risques

Le développement itératif est plus efficace lorsque vous l'associez à un équilibre délibéré entre la diminution rapide des risques et la création de valeur en amont, grâce à l'application d'étapes basées sur les risques. Cela signifie que au moment de priorisation du travail pour chaque itération, vous choisissez d'implémenter les fonctionnalités qui représentent les plus forts risques métiers – risques organisationnels, risques techniques, tout en délivrant la valeur la plus importante. Ces deux objectifs (risque le plus élevé et valeur la plus forte) ne sont généralement pas en adéquation : c'est pourquoi l'approche d'étapes basées sur les risques force un choix délibéré entre la génération d'une valeur maximale et la réduction rapide des risques. Les deux sont essentiels pour le succès du projet (voir la Figure 4). Il importe donc de disposer de points de contrôle appropriés.

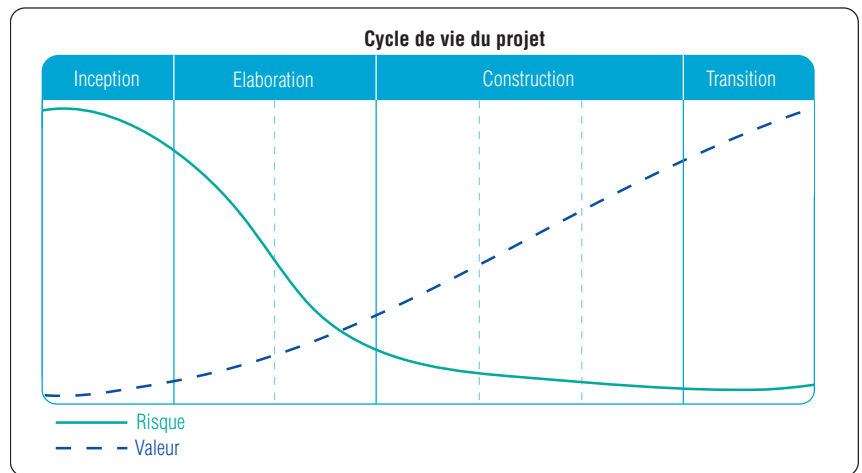


Figure 4: réduction des risques (courbe en bleu clair) et génération de valeur (courbe en pointillés bleus) pendant la durée de vie du projet

Points clés

Les quatre phases RUP se terminent par un bilan formel, qui s'attache à l'évaluation des éléments livrables vis-à-vis de la réduction des risques et de la création de valeur. Par exemple, à la fin de la phase d'élaboration RUP, vous souhaitez éliminer autant que possible les risques techniques et produire une architecture stable. L'équipe doit démontrer qu'elle a élaboré une architecture exécutable à travers quelques scénarios pouvant être exécutés, et avec une liste reflétant la réduction de nombreux risques, y compris techniques. La diminution des risques doit être équilibré avec une démonstration de la valeur du code exécutable, ce qui permet de mettre en évidence l'avancement réalisé par l'équipe. IBM recommande d'utiliser les étapes RUP suivantes : inception, élaboration, construction et transition.¹⁶ Vous pouvez adopter des stratégies similaires avec d'autres processus agiles tels que Extreme Programming (XP), OpenUP, ou encore Scrum.

Les étapes basées sur les risques permettent aux parties prenantes d'avoir un meilleur contrôle et une meilleure vue d'ensemble des projets, et de favoriser en amont la création de valeur. Elles permettent également de réduire les risques d'échec et d'augmenter la productivité, en supprimant très tôt les risques techniques (ce qui diminue d'autant le risque global lié au projet).

L'obtention de résultats répétitifs est beaucoup plus souhaitable que le processus répétable.

Adaptation des processus

Tous les projets ne sont pas créés de la même façon, c'est pourquoi il est essentiel d'adapter le processus de développement aux besoins du projet. Une équipe chargée du développement d'un site Web ne travaille pas de la même façon qu'une équipe chargée du développement d'un datawarehouse. Une équipe de 5 personnes ne travaille pas de la même façon qu'une équipe de 50. Une équipe développant un système critique pour la vie ne travaillera pas non plus de la même façon qu'une équipe développant une application métier. La question n'est pas de savoir si travailler avec davantage de processus est mieux ou moins bien que travailler avec moins de processus. Il s'agit en réalité d'adapter le niveau de formalisme, de précision et de contrôle d'un projet à plusieurs facteurs (taille et répartition des équipes, niveau de contraintes externes, phase du projet, besoin de traçabilité et d'auditabilité, et particulièrement niveau de risque associé).

Points clés

La gouvernance Lean encourage les équipes à agir au niveau des améliorations de processus, tout au long du cycle de vie du projet - et pas seulement de les identifier à la fin du projet.

Un projet doit également adapter le formalisme des processus à la phase concernée du cycle de vie. Le début d'un projet s'accompagne généralement d'une grande incertitude, et vous voulez fortement encourager la créativité pour développer une application qui répond aux besoins métier. En règle générale, plus le processus est lourd, plus la créativité diminue ; c'est pourquoi vous devez utiliser un processus plus léger en début de projet, lorsque l'incertitude est omniprésente. Par ailleurs, un peu plus tard dans le projet, vous recherchez généralement à appliquer davantage de contrôle (comme le gel des fonctionnalités ou le comité de contrôle du changement) afin d'éliminer l'imprévisibilité et les risques associés à l'introduction tardive de défauts.

Amélioration continue

Les entreprises qui adoptent le Lean Thinking cherchent à améliorer continuellement leurs processus. IBM recommande d'effectuer une évaluation (une rétrospective, par exemple),¹⁷ à la fin de chaque itération et de chaque projet, afin d'en tirer les leçons et d'utiliser les connaissances acquises pour améliorer les processus. Le cadre de processus RUP (qui contient des processus prêts à l'emploi adaptés aux besoins des projets) constitue une véritable richesse de méthodes prêtes à être utilisées.

Le concept fondamental sur lequel s'appuie l'amélioration continue est simple : améliorez votre façon de travailler, chaque fois que l'opportunité se présente. Le vieil adage qui consiste à dire que « Vous devez apprendre quelque chose de nouveau tous les jours » est vrai. De plus, cette pratique va encore plus loin, en vous recommandant d'agir sur ce que vous apprenez, pour augmenter votre efficacité globale.

Points clés

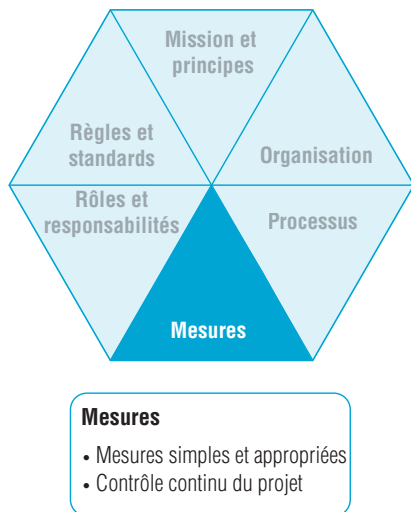
Une gouvernance Lean intègre la mise en conformité à la culture de l'entreprise, et d'automatise, dans la mesure du possible, les tâches et les activités correspondantes.

Pour identifier les améliorations potentielles de vos processus de développement logiciel au cours de l'exécution de vos projets, vous avez le choix entre différentes méthodes : par exemple, vous pouvez inclure des sessions informelles d'amélioration, ou mettre des boîtes à idées à l'attention de vos équipes, ou encore faire un site Web dédié. Vous pouvez également encourager la réflexion de chacun et mettre à la disposition de vos équipes un processus éditable (peut-être via un wiki). L'une des stratégies efficaces consiste à planifier deux heures à la fin de chaque itération consacrées à une rétrospective informelle. La pratique de l'amélioration continue vous permet d'augmenter vos connaissances au fur et à mesure et permet à l'équipe de savoir où elle va.

Conformité intégrée

Plus la mise en conformité est simple, plus les professionnels de l'IT parviendront réellement à la mettre en œuvre. Par conséquent, dans la mesure du possible, vous devez automatiser la mise en conformité à la réglementation, à la politique de l'entreprise et aux règles de conduite. Quand cette automatisation n'est pas possible, la mise en conformité doit faire partie de votre culture d'entreprise et être intégrée dans le travail quotidien, et non s'inscrire dans un ensemble de tâches séparées, exécutées en fin de projet. Pour que le thème de la conformité fasse partie de votre culture d'entreprise, vos employés doivent comprendre pourquoi c'est important et quels sont les principes sous-jacents qui la composent. Si la mise en conformité suppose un surcroît de travail important, et si ce travail est perçu comme étant onéreux ou arbitraire par les personnes qui en ont la charge, vos efforts de mise en conformité risquent de ne pas être appliqués par les équipes de développement.

Les avantages d'une conformité intégrée sont les suivants : diminution du coût total de la mise en conformité, réduction de la réticence des équipes de développement et niveaux de conformité plus élevés qu'avec les approches traditionnelles. IBM recommande de définir une solution minimale pouvant être intégrée à vos processus et à vos outils, sur la base des réglementations correspondantes. La solution consiste à affecter les personnes les plus adaptées à l'interprétation des réglementations et à la création des recommandations que devront suivre les équipes de développement. Si vous affectez ces tâches à des bureaucrates, la solution sera de nature bureaucratique.



Pratiques pour les mesures

Dans la catégorie de Mesures de la gouvernance de développement, IBM recommande deux pratiques Lean qui renforcent les prises de décisions à l'aide de définition d'objectifs et de mesures incitatives.¹⁸

Mesures simples et appropriées

Des mesures simples et appropriées fournissent toutes les informations nécessaires pour comprendre l'état dans lequel vous êtes et prendre les actions correctives, si nécessaire. Malheureusement, les entreprises n'utilisent pour la plupart aucune mesure, - ce qui signifie qu'elles progressent en aveugle - ou au contraire les utilisent exagérément. Elles collectent un si grand nombre de mesures qu'elles sont noyées sous les informations.

Les mesures efficaces sont appropriées car elles sont utilisées largement dans votre entreprise, et elles fournissent aux cadres et dirigeants l'information dont ils ont besoin pour prendre les mesures appropriées en temps voulu. Lorsque cela est bien fait, les mesures simples et appropriées favorisent la gouvernance basée sur les faits. Si elles sont en plus automatisées, la gouvernance est facilitée. Et lorsqu'elles servent à examiner les tendances, elles permettent de réaliser une gouvernance proactive.

Il est préférable de commencer par des mesures qui déterminent la valeur générée par les équipes, la qualité livrée par le projet et le coût induit. Ces mesures sont utiles non seulement pour déterminer l'état actuel de vos projets, mais aussi pour évaluer les écarts par rapport aux attentes initiales. Les attentes sont définies par rapport aux études commerciales en fonction de : durée, dépenses et utilisation des ressources. Elles peuvent être mises à jour périodiquement au cours du projet. Souvenez-vous que la règle fondamentale de toutes les mesures métier (mesurez ce que vous souhaitez réellement contrôler) s'applique également aux mesures de projets de développement. Par exemple, si vous souhaitez améliorer la qualité du code, vous ne devez pas mesurer le nombre de bogues corrigés.

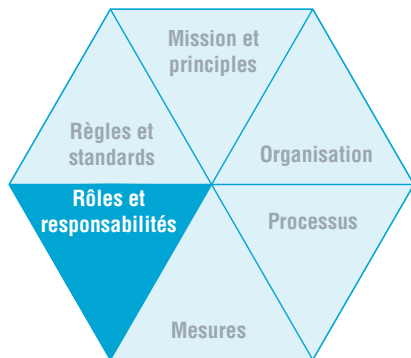
Points clés

Mesurez la valeur générée, la qualité produite et les coûts induits pour identifier les secteurs qui nécessitent une attention accrue et des actions correctives.

Mais vous devez mesurer le nombre de bogues restants. Lorsque l'objectif réel est difficile à mesurer, il peut être tentant de le remplacer par un élément substitut quantifié qui semble plus simple pour effectuer le suivi d'un objectif réel. Par exemple, votre objectif réel peut être d'améliorer le retour sur investissement global ; or, vous mesurez à la place les coûts. L'expérience a prouvé qu'inévitablement, nous avons tendance à nous focaliser sur ce qui était réellement mesuré. Au fil du temps, la mesure de substitution deviendra de plus en plus déconnectée et perdra de sa fiabilité en tant que mesure d'objectif réel. Pire encore, plus les récompenses ou les pénalités associées sont importantes, plus cette divergence se fera sentir rapidement. Il est préférable de mesurer, même approximativement, l'objectif réel plutôt que de mesurer avec précision un élément de substitution.

Contrôle continu du projet

Vous pouvez contrôler régulièrement le bon déroulement de vos projets ITs via la collecte automatisée de mesures, les revues d'étapes de projet, l'analyse postmortem, voire les informations informelles. Le contrôle continu des projets, associé au développement itératif, permet d'obtenir une gouvernance basée sur les faits grâce aux mesures précises et actualisées du code livré, comparé aux évaluations des spécifications. Ce contrôle permet de détecter les problèmes à un stade précoce et d'entreprendre les actions correctives plus tôt et ainsi de réaliser une gouvernance plus efficace en prenant les bonnes mesures. IBM conseille de commencer par la capture automatique et l'affichage de vos mesures via le logiciel de suivi de projets . Toutefois, ne tentez pas de faire une gestion à l'aide de chiffres uniquement. Lorsqu'un projet semble s'écarter des prévisions, vous devez en parler à l'équipe chargée du projet, afin de déterminer ce qui se passe réellement et d'identifier si les membres de l'équipe ont besoin de votre aide.



Rôles et responsabilités

- Encouragement à l'autonomie des équipes
- Adéquation de la structure de l'équipe à l'architecture

Pratiques applicables en matière de rôles et de responsabilités

Les deux pratiques Lean de la catégorie gouvernance des Rôles et responsabilités se focalisent sur la définition du développement en clarifiant qui est responsable, qui a l'autorité et quelle est la structure hiérarchique.¹⁹

Encouragement de l'autonomie d'organisation des équipes

Le premier atout du développement logiciel Agile est la préférence donnée aux personnes et aux interactions par rapport aux processus et aux outils.²⁰

En matière de gouvernance de développement Lean, la priorité devrait être l'encouragement des professionnels du secteur IT à créer des logiciels de grande qualité et la promotion de la collaboration efficace entre les membres des équipes, et non le contrôle direct de ces personnes.

Une équipe autonome peut allouer le travail dans le cadre de la structure de gouvernance dans laquelle elle se trouve. Elle assume la responsabilité de ses méthodes de travail. Les membres de l'équipe choisissent leurs activités, chacun s'engage à réaliser son travail et une coordination est régulièrement effectuée. Il s'agit d'une approche participative des prises de décisions : chacun a la possibilité de participer et de prêter attention au processus de prise de décision.

Points clés

Les personnes les plus aptes à planifier le travail sont celles qui le font.

L'autonomisation fonctionne bien lorsqu'elle est associée à un développement itératif, car l'approche itérative permet à l'équipe et aux parties prenantes de se mettre d'accord sur ce qui doit être livré lors de la prochaine itération. A la fin de l'itération, l'équipe et les parties prenantes évaluent ce qui a été fait, puis des mesures correctives sont prises. Bien que l'équipe exerce une responsabilité collective, la responsabilité finale (ainsi que les prises de décisions associées) incombent au Chef d'équipe. Les itérations représentent un mécanisme de contrôle qui permet aux équipes autonomes de rester sur la bonne voie. Les avantages de l'organisation autonome sont les suivants :

- *Motivation accrue, génératrice d'augmentation de la productivité.*
- *Les décisions sont prises au bon niveau de la structure organisationnelle car les équipes ont à la fois la responsabilité et le pouvoir de mener à bien leurs tâches.*
- *Moins de risques d'erreur, grâce à une amélioration de la communication et moins de transitions de produits entre les personnes.*
- *Davantage de possibilités d'élargissement des compétences.*

En bref, les équipes de projets doivent pouvoir organiser leur travail, leur environnement et leur approche globale de la façon qui leur convient pour atteindre les résultats de projet escomptés. Toutes les contraintes que subit l'équipe (recommandations organisationnelles , par exemple) doivent être décrites et négociées avec l'équipe tout au long du projet.

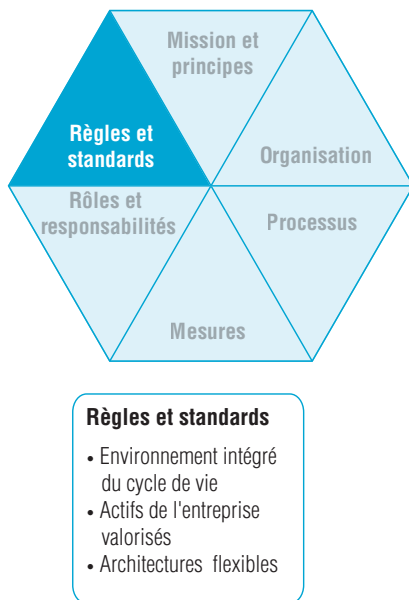
Points clés

Le logiciel créé par une équipe de projet reflète sa structure organisationnelle.

Adéquation de la structure d'équipe à l'architecture

Il est important de bien faire correspondre structure d'équipe et architecture du système créé. La loi Conway, définie par Melvin Conway à la fin des années 1960,²¹ indique que tout logiciel reflète la structure organisationnelle du groupe qui l'a produit (que l'on peut résumer par « Ce que vous délivrez, c'est votre organisation »).

Les gens travailleront d'une manière, qui reflète la façon dont ils sont organisés. En d'autres termes, il est fort probable qu'un groupe décentralisé produise un système dont l'architecture sera également décentralisée. Les forces et les faiblesses de la structure organisationnelle d'une équipe se reflètent inévitablement dans le système qu'elle produit. Par conséquent, si vous faites correspondre votre structure organisationnelle à l'architecture que vous souhaitez obtenir, vous pourrez rationaliser vos efforts de développement. Vous pouvez obtenir des itérations plus courtes et diminuer les risques liés au projet. Vous pouvez supprimer pratiquement tous les cycles d'échanges d'information, de revue et de refonte sans fin, entre les différents groupes. Et par la même occasion, vous pouvez améliorer la communication au sein de l'équipe. La bonne façon de faire est de s'organiser autour de l'architecture souhaitée, avec des équipes cross-fonctionnelles et travaillant au même endroit, responsables de chaque composant majeur.



Pratiques applicables aux stratégies et aux standards

Les trois pratiques Lean à la base de la gouvernance des règles et standards décrivent les recommandations spécifiques destinées à renforcer le fonctionnement homogène des différentes parties impliquées dans le développement.²²

Le premier atout du développement logiciel agile réside dans la préférence donnée aux personnes et aux interactions, plutôt qu'aux processus et aux outils. Cela n'implique pas que vous n'allez pas utiliser ni processus, ni outil ; cela veut simplement dire qu'ils doivent être secondaires par rapport aux personnes, et que les outils et processus doivent être le reflet de la façon dont les personnes travaillent, et pas inversement. Grâce à l'approche Lean, votre entreprise pourra adopter des stratégies et des standards qui permettront aux développeurs de collaborer efficacement, qui les inciteront à développer et à réutiliser l'infrastructure et les actifs existants et qui leur permettront de fournir un travail de grande qualité. En cas de conflit entre comportements humains et comportements des outils, c'est le comportement humain qui prévaut.

Environnement de la gestion du cycle de vie intégré

Le développement logiciel s'appuie sur une collaboration intensive entre les membres d'équipes dispersées, dont les heures de travail ou les fuseaux horaires divergent. La taille des développements rend également la collaboration difficile, car vous devez soutenir des interactions à grande échelle. Ajouté à ce challenge se trouvent la nécessité de surveillance par le management et de la comptabilité afin d'assurer la mise en conformité réglementaire. Les environnements de gestion du cycle de vie intégrés ont pour but de favoriser la collaboration pour un coût le plus faible possible - utilisant l'infrastructure pour la majorité, voire la totalité des autres pratiques de gouvernance.

Points clés

Grâce à des outils de développement efficaces, vous pouvez facilement collecter les informations d'audit et les mesures appropriées, de façon automatisée.

Les développeurs appliquent sans sourciller les standards et les recommandations simples à comprendre.

Les principaux composants d'un environnement intégré du cycle de vie incluent les outils de gestion de configuration logicielle, d'analyse, de conception, de construction, de test, d'assurance qualité, de gestion de processus, de gestion de projet et de gestion de portefeuille. Les environnements intégrés tels que la plateforme IBM Rational de livraison de logiciels (IBM Rational Software Delivery Platform) facilitent l'implémentation de presque toutes les pratiques de gouvernance identifiées. Ils permettent de diminuer le coût total de possession des environnements d'outils et favorisent la collaboration. Et les meilleurs environnements d'outils captureront automatiquement autant de données de mesures et d'audits que possible en freinant le moins possible le travail des développeurs.

Actifs de l'entreprise valorisés

Un actif IT est un composant logiciel, un service, un template, un model d'entreprise une architecture de référence, un exemple, un pattern, une recommandation ou un standard que l'organisation demande aux professionnels IT d'appliquer dans leur travail quotidien. Un actif IT valorisé est un actif que les professionnels IT veulent réellement utiliser, car ils la considèrent comme adaptée à leur travail et offrant une qualité suffisante. En d'autres termes, ils utilisent les actifs ITs disponibles pour accroître leur productivité.

Les développeurs respectent l'architecture de l'entreprise et les architectures de référence appropriées, car elles leur permettent de gagner du temps. Leurs systèmes appellent des services existants, et ils peuvent donc bénéficier d'une structure technique existante. Ce qui importe, c'est de faciliter la réutilisation des actifs, plutôt que de les recréer.

Les avantages organisationnels qui en découlent sont nombreux : cohérence et efficacité accrues, délais de mise sur le marché plus courts,

Points clés

Les développeurs réutilisent facilement les artefacts de qualité qui fournissent de la valeur.

Les architectures flexibles favorisent la flexibilité de l'entreprise.

meilleure communication et coûts de gouvernance réduits. Les services ITs doivent maintenir des règles de codage pour leurs principaux langages de développement, une bibliothèque de patterns courants qu'ils souhaitent utiliser dans leurs systèmes, des architectures de référence pour les aspects critiques de leur infrastructure technique, ainsi qu'une description de la vision de leur architecture d'entreprise.

Il doit exister des personnes en place pour maintenir, faire évoluer, acheter ou collecter des actifs ITs valorisés. Il est par ailleurs important de disposer de mesures permettant d'identifier et de limiter les freins à la réutilisation, tout comme de mesures indiquant la valeur des différents éléments concernés par cette réutilisation. Dans la mesure du possible, vous devez privilégier l'achat d'un actif IT à sa création, particulièrement si vous pouvez acquérir facilement quelque chose qui correspond bien à vos besoins : cela vous permet de vous décharger du fardeau que représente sa maintenance.

Architectures flexibles

L'un des objectifs de la gouvernance de développement logiciel Lean consiste à favoriser la flexibilité et la réactivité de votre métier, ce qui se traduit par le même besoin de flexibilité et de réactivité en matière d'organisation et d'évolution de vos systèmes ITs. La gouvernance Lean s'obtient via l'utilisation d'architectures flexibles, car celles-ci permettent aux équipes de développement de réagir avec efficacité aux besoins changeants du métier. A l'heure actuelle, les architectures les plus flexibles se retrouvent dans le domaine des systèmes ouverts, des orientations services et de l'adoption de techniques de développement Lean telles que : développement agile piloté par les modèles,²³ TDD (Test Driven Development) développement piloté par les tests,²⁴ et intégration continue.²⁵ Les architectures flexibles permettent de répondre aux besoins évolutifs des entreprises, de diminuer les délais de mise sur le marché, d'augmenter le retour sur investissement, de diminuer le coût total de possession et de limiter le risque technique. Par défaut, IBM recommande les architectures orientées services (SOAs), les technologies Open Source telles qu'Eclipse et les standards tels que Java™ Platform, Enterprise Edition (Java EE) et UML (Unified Modeling Language).

Points clés

Les pratiques de gouvernance Lean soutiennent les approches modernes en matière de développement.

Conclusion

Chaque structure IT possède un programme de gouvernance de développement en place, mais ce programme peut très bien ne pas être explicite, ou être inefficace. Les frameworks traditionnels de gouvernance de développement ont connu un succès limité, car les développeurs sont des intellectuels qui ne travaillent pas bien dans des approches de commande-et-contrôle. Pour être efficace, un programme de gouvernance doit refléter l'environnement réel dans lequel il est appliqué ; il doit aussi tenir compte des personnes concernées.

Une gouvernance réussie de développement s'attache à la mise en application de comportements et de meilleures pratiques, via des techniques collaboratives. Il est beaucoup plus efficace de motiver une personne à adopter le comportement souhaité que de la forcer à le faire. Les 18 pratiques de gouvernance de développement Lean qu'IBM a identifiées reflètent la réalité du développement logiciel actuel. Ces pratiques peuvent vous permettre d'incorporer la gouvernance à vos outils, processus et recommandations de développement. Elles prennent en charge les approches modernes de développement telles que : Lean Software Development, RUP, XP, OpenUP, Dynamic Systems Development Method (DSDM), Eclipse Way et Scrum.

La gouvernance de développement représente un sous-ensemble de la gouvernance IT et de l'ingénierie systèmes globale, mais d'une importance capitale. Grâce à la rationalisation du workflow réalisé par l'équipe, l'approche Lean peut vous aider à accroître la valeur de vos projets de développement.

Pour plus d'informations

Pour en savoir plus sur les outils Rational qui peuvent vous aider à appliquer des pratiques de gouvernance de développement Lean, contactez votre représentant IBM ou visitez le site :

ibm.com/rational/agile

ou:

ibm.com/rational/rmc

Notes

Nous tenons à remercier Cheri Bergeron, Lynn Brennan, Murray Cantor, Clay M. Nelson, Bob Noble, Ted Rivera et Carl Zetie pour les commentaires et les informations qu'ils ont bien voulu nous transmettre dans le cadre de ce livre blanc.

- 1 Kroll, Per; "Making Agile Mainstream: Crossing the Chasm"; *IBM Rational Edge*; www.ibm.com/developerworks/rational/library/mar07/kroll/; 2007.
- 2 Cantor, Murray et Sanders, John D.; "Operational IT Governance"; *IBM Rational Edge*; www-128.ibm.com/developerworks/rational/library/may07/cantor_sanders/index.html; Mai 2007.
- 3 Poppendieck, Mary et Poppendieck, Tom; *Implementing Lean Software Development: From Concept to Cash*; Addison-Wesley Professional; Boston; 2006.
- 4 Ambler, S. W. and Kroll, Per; "Best practices for lean development governance—Part 1: Principles and Organizations"; *IBM Rational Edge*; www.ibm.com/developerworks/rational/library/jun07/kroll/; 2007.
- 5 Kroll, P. et MacIsaac, B.; *Agility and Discipline Made Easy—Practices from OpenUP and RUP*; Addison-Wesley Professional; Pearson Education, Inc.; Boston; 2006.
- 6 Bittner, K. et Spence, I.; *Managing Iterative Software Development Projects*; Addison-Wesley Professional; Boston; 2006.
- 7 Cantor, M.; "Requirements Analysis and Design Part 3"; *IBM Rational Edge*; www.ibm.com/developerworks/rational/library/content/RationalEdge/oct03/m_rupse_mc.pdf; 2003.
- 8 Nightingale, J.; "Green Threads: Improving Cross-Product Integration"; *Dr. Dobb's Journal*; www.ddj.com/architect/196603524; 2006.
- 9 Ambler, S. W. et Kroll, P.; "Best practices for lean development governance—Part 1: Principles and Organizations"; *IBM Rational Edge*; www.ibm.com/developerworks/rational/library/jun07/kroll/; 2007.
- 10 Kroll, P. et Royce, W.; "Key Principles for Business-Driven Development"; *IBM Rational Edge*; www.ibm.com/developerworks/rational/library/oct05/kroll/; 2005.
- 11 Ambler, Scott W.; Examining the "Big Requirements Up Front (BRUF)" Approach; www.agilemodeling.com/essays/examiningBRUF.htm; 2005.
- 12 Ambler, S. W.; "Agile on a Fixed Budget"; *Dr. Dobb's Journal*; www.ddj.com/architect/201202925; 2007.
- 13 Ambler, S. W. et Kroll, Per; "Best practices for lean development governance—Part 2: Processes and Measures"; *IBM Rational Edge*; www.ibm.com/developerworks/rational/library/jul07/kroll_ambler/; 2007.
- 14 Agile Alliance; *The Agile Manifesto*; www.agilemanifesto.org; 2001.
- 15 Kroll, P. et Kruchten, P.; *Rational Unified Process Made Easy—A Practitioner's Guide to the RUP*; Addison-Wesley Professional; Pearson Education, Inc.; Boston; 2003.
- 16 Kroll, P. and MacIsaac, B.; *Agility and Discipline Made Easy—Practices from OpenUP and RUP*; Addison-Wesley Professional; Pearson Education, Inc.; Boston; 2006.
- 17 Kerth, Norman L.; *Project Retrospectives: A Handbook for Team Reviews*; Dorset House; New York; 2001.
- 18 Ambler, S. W. et Kroll, P.; "Best practices for lean development governance—Part 2: Processes and Measures"; *IBM Rational Edge*; www.ibm.com/developerworks/rational/library/jul07/kroll_ambler/; 2007.
- 19 Ambler, S. W. et Kroll, Per; "Best practices for lean development governance—Part 3: Roles and Policies"; *IBM Rational Edge*; www.ibm.com/developerworks/rational/library/aug07/ambler_kroll/; 2007.
- 20 Agile Alliance; *The Agile Manifesto*; www.agilemanifesto.org; 2001.
- 21 Conway, M. E.; "How Do Committees Invent?"; *Datamation Magazine*; www.melconway.com/research/committees.html; 1968.
- 22 Ambler, S. W. et Kroll, P.; "Best practices for lean development governance—Part 3: Roles and Policies"; *IBM Rational Edge*; www.ibm.com/developerworks/rational/library/aug07/ambler_kroll/; 2007.
- 23 Ambler, S. W.; *The Object Primer 3rd Edition: Agile Model Driven Development with UML 2*; Cambridge University Press; New York; 2005
- 24 Astels, D.; *Test-Driven Development: A Practical Guide*; Addison Wesley; Upper Saddle River, New Jersey; 2003.
- 25 IBM; Rational Build Forge; www-306.ibm.com/software/awdtools/buildforge/; 2007.



© Copyright IBM Corporation 2007

Compagnie IBM France
Tour Descartes - La Défense 5
2, avenue Gambetta
92066 Paris La Défense Cedex

Imprimé en France
03-09

Tous droits réservés

IBM, le logo IBM, Rational, Rational Unified Process et RUP sont des marques déposées d'International Business Machines Corporation aux Etats-Unis et/ou dans certains autres pays.

Java et toutes les marques incluant Java sont des marques de Sun Microsystems, Inc. aux Etats-Unis et/ou dans certains autres pays.

Les autres noms de société, de produit et de service peuvent appartenir à des tiers.

Les informations contenues dans la présente documentation sont fournies à des fins d'information uniquement. Même si tout a été mis en œuvre pour vérifier l'intégrité et l'exactitude des informations contenues dans la présente documentation, ces dernières sont fournies "en l'état", sans aucune garantie, explicite ou implicite. De plus, ces informations sont basées sur les plans et la stratégie de produits actuels d'IBM, lesquels sont sujets à modification par IBM sans préavis. IBM ne peut être tenu pour responsable de tout dommage émanant de l'utilisation de, ou sinon associée à la présente documentation ou toute autre documentation. Aucun élément présent dans cette documentation n'a pour objet, ni n'aura pour effet, de créer une quelconque garantie ou représentation de la part d'IBM (ou de ses fournisseurs ou concédants de licence) ou de modifier les conditions du contrat de licence en vigueur régissant l'utilisation des logiciels IBM.