Cursus SALESFORCE

M2I - Formation - 2024

Donjon Audrey

m2iformation.fr

Les Méthodes agiles

Donjon Audrey
Formatrice Frontend

Contact :

www.linkedin.com/in/audrey-djn
audrey._
audrey-donjon

Sommaire

La gestion de projet

La gestion de projet

1. La gestion de projet
2. Importance de la gestion de projet
3. Méthode traditionnelle de gestion de projet
4. Exemple d'application de méthode
5. Les limites de la méthode linéaire

La gestion de projet

La **gestion de projet** est l'application de **connaissance**, **compétences**, **outils** et **techniques** pour **répondre aux exigences d'un projet**. Le projet peut prendre plusieurs formes selon l'objectif visé, il est conçue pour produire un résultat unique que ce soit un produit, un service ou un résultat spécifique.

La **gestion de projet** va donc impliquer de la **planification**, de **l'organisation**, de la **coordination** et de **contrôler** toutes les activités nécessaires pour atteindre les objectifs du projet dans les délais impartis et le budget prévu.

Importance de la gestion de projet

Dans le monde des affaires et de la tech, la gestion de projet est cruciale pour assurer le succès des objectifs. Elle permet de :

- ❑ **Définir clairement les objectifs :** Identifier ce qui doit être accompli et les critères de succès
- ❑ **Gérer les ressources :** Utiliser efficacement les ressources humaines, financières et matérielles
- ❑ **Respecter les délais et le budget :** Planifier et contrôler les coûts et les calendriers
- ❑ **Communiquer efficacement :** Faciliter la communication entre les membres de l'équipe et les parties prenantes
- ❑ **Gérer les risques :** Identifier, analyser et atténuer les risques potentiels

Méthode traditionnelle de gestion de projet

L'une des méthodes connues existant bien avant l'arrivé des méthodes agiles est la **méthode en cascade**. La méthodologie **Waterfall** « **en cascade** » est considéré comme l'approche de référence à la gestion de projet.

Cette méthode est de **nature linéaire**, elle décompose un projet en **une série de phase** où **chaque phase doit être complétée** avant de **passer à la suivante**. **Les principales étapes** sont :

- **Initiation** : cartographie et analyse des exigences et besoins liés à l'application, au produit ou au projet à livrer, consolidation des résultats dans un cahier des charges qui devra servir de document de référence au cours de la réalisation du projet.
- **Planification** : développer un plan détaillé pour atteindre les objectifs du projet et définir les ressources nécessaires.
- **Exécution** : mettre en œuvre le plan et réaliser le travail nécessaire pour atteindre les objectifs du projet.
- **Suivi et contrôle** : implication de bêta-testeurs et de professionnels du contrôle qualité pour détecter, signaler et corriger d'éventuels problèmes ou bugs.
- **Clôture** : déploiement et mise en service du livrable achevé, prestations d'assistance et de maintenance.

Exemple d’un projet web avec la méthode Waterfall

Projet de développement d'un site web pour une petite entreprise qui souhaite vendre ses produits en ligne :

1. **Initiation** : définir le but du projet / Identifier les parties prenantes / Obtenir l'approbation initiale.
   - Rédaction d'un document expliquant les bénéfices attendus, les coûts et le retour sur investissement (ROI).
   - Identification des fonctionnalités clés du site web, comme le catalogue de produits, le panier d'achat, et le système de paiement.
   - Document formel décrivant les objectifs, les parties prenantes, les responsabilités, les délais, et le budget.
   - Présentation du business case et de la charte de projet aux sponsors et parties prenantes pour approbation.
2. **Planification** : développer un plan détaillé pour atteindre les objectifs du projet / Définir les ressources nécessaires.
   - Réalisation de réunions avec les parties prenantes pour recueillir et documenter les exigences détaillées.
   - Création d'un document de spécifications détaillant les fonctionnalités du site web.
   - Définition de l'architecture du site web, incluant les bases de données, les serveurs, et les technologies utilisées.
   - Développement d'un plan de projet incluant le calendrier, le budget, les ressources, et les jalons.
   - Identification des risques potentiels et planification des réponses appropriées.
3. **Exécution** : mettre en œuvre le plan et réaliser le travail nécessaire pour atteindre les objectifs du projet.
   - Création des pages web, design UI/UX, et intégration des éléments graphiques.
   - Développement des fonctionnalités côté serveur, base de données, et API.
   - Intégration des systèmes tiers comme les passerelles de paiement, les services de livraison, etc.
   - Test des composants individuels du site web pour s'assurer qu'ils fonctionnent correctement.
   - Réunions régulières pour suivre l'avancement du projet et résoudre les problèmes.

Exemple d’un projet web avec la métode Waterfall

4. **Suivi et contrôle : implication de bêta-testeurs et de professionnels du contrôle qualité pour détecter, signaler et corriger d’éventuels problèmes ou bugs.**
    - Comparer l'avancement réel aux prévisions et ajuster le plan de projet si nécessaire.
    - Traiter les demandes de changement et évaluer leur impact sur le projet.
    - Générer des rapports réguliers pour informer les parties prenantes de l'avancement du projet.
    - S'assurer que le travail accompli répond aux normes de qualité définies.

5. **Clôture : déploiement et mise en service du livrable achevé, prestations d’assistance et de maintenance.**
    - Impliquer les utilisateurs finaux pour tester le site web et valider qu'il répond à leurs besoins.
    - Mettre le site web en ligne et le rendre accessible aux utilisateurs.
    - Former les utilisateurs à l'utilisation du nouveau site web.
    - Fournir la documentation complète du projet, incluant le manuel utilisateur, les guides de maintenance, etc.
    - Réaliser une rétrospective pour évaluer les succès et les leçons apprises.
    - Archiver les documents du projet et libérer les ressources.

Les limites de la méthode linéaire

Cette approche est bien adaptée aux projets où les **exigences** sont **claires** et **peu susceptibles de changer**.

Par contre elle peut être **moins efficace** dans des **environnements dynamiques** où les besoins peuvent **évoluer au fil du temps**.

Dans les **limites** que peut avoir cette méthode nous avons :

- **Sa rigidité :** il est difficile d'apporter des modifications une fois qu'une phase est terminée
- **De longs cycles de développement :** les projets peuvent prendre beaucoup de temps avant de produire des résultats tangibles
- **Une communication limité :** interaction limitée entre les différentes phases du projet
- **Réactivité réduite :** Moins de flexibilité pour s'adapter aux changements imprévus

Introduction aux méthodes agiles

Introduction aux méthodes agiles

1. Origines des méthodes agiles
2. Que sont les méthodes agiles
3. Les méthodologies agiles
4. Le manifeste agile
5. Les 12 principes du Manifeste Agile

Origines des méthodes agiles

L'origine de la **méthode Agile** remonte au **années 1990** pour répondre aux **limitations des approches traditionnelles** de gestion de projets comme la méthode en **cascade**.

Vis-à-vis des approches traditionnelles qui étaient assez **rigides**, **lentes** et **moins efficaces** conduisant souvent à des **retards** et à une **insatisfaction des clients**, les **méthodes agiles** ont alors émergés.

Que sont les méthodes agiles

Contrairement aux **méthodes linéaires**, les **méthodes agiles** s'adaptent mieux aux changements et favorise la collaboration pour satisfaire les besoins évolutifs des clients.

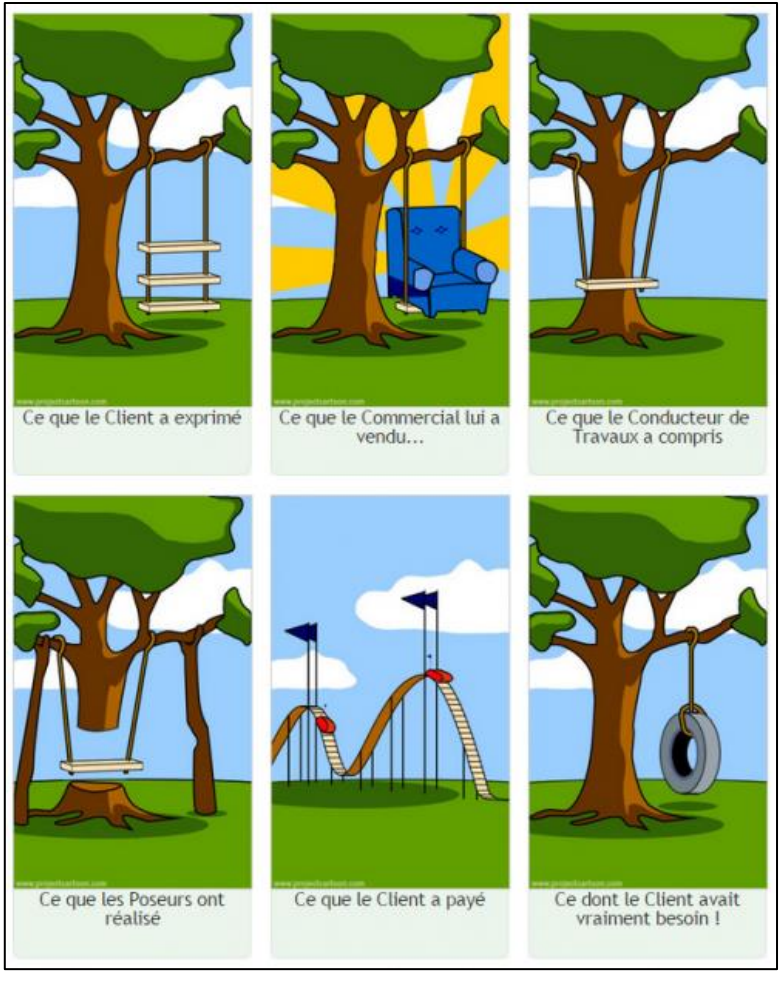
Méthodes linéaires vs agiles

Voici un mini-jeu illustrant l'intérêt de la méthode agile par rapport aux méthodes linéaires :

https://www.boxuk.com/battleships/

Le problème des **méthodes linéaires** tel que celui en **cascade**, c'est le **manque de flexibilité** et de **communication**.

Les **méthodes agiles** s'**adaptent mieux aux changements** et **favorise la collaboration** pour satisfaire les besoins évolutifs des clients.



Les méthodologies de développement Agile

Dans les **années 1990** et au **début des années 2000**, plusieurs méthodologies de développement Agile ont émergé, telles que **Scrum**, **Extreme Programming** (XP) et **Feature-Driven Development** (FDD).

Bien qu'elles aient été développées indépendamment, ces méthodologie partageaient des principes communs, tels que la **flexibilité**, **l'adaptabilité** et la **collaboration** étroite avec le client.

Sont considérés comme étant des méthodes agiles, les approches qui s'inscrivent dans la philosophie du **manifeste pour le développement agile de logiciels**.

Le manifeste Agile

En **février 2001**, **17 développeurs (experts en dev logiciel)** se sont réunis à Snowbird, dans l'Utah, aux États-Unis pour discuter de **nouvelles approches de développement logiciel**. De cette réunion est né le **Manifeste Agile**, un **document fondateur** qui a **formalisé les valeurs** et **principes des méthodes agiles.**

Les **quatre valeurs principales** sur lequel est fondé le Manifeste Agile sont :

- Les **individus** et les **interactions** plutôt que les processus et les outils
- Des **logiciels opérationnels** plutôt que de la documentation exhaustive
- La **collaboration avec les clients** plutôt que la négociation contractuelle
- L'**adaptation au changement** plutôt que le suivi d'un plan

Les 12 principes du Manifeste Agile

Le **Manifeste Agile** définit également **12 principes** pour **guider le développement logiciel Agile**.

Ces principes encouragent la **satisfaction** du **client**, la **collaboration**, l'**auto-organisation des équipes**, la **communication** face à face, la **simplicité** et la **réflexion** sur les **méthodes de travail** pour les **améliorer continuellement**.

Principe 01 : Satisfaction du client

Principe : Satisfaire le client par des livraisons rapides et continues de logiciels opérationnels.

Ce principe explique qu'il faut s'assurer que les **clients** soient **satisfait** en leur fournissant des **fonctionnalités utilisables dès que possible**.

Principe 02 : Accueillir le changement

Principe : Accueillir favorablement les changements de besoins, même tard dans le développement.

Ce principe explique qu'il faut être prêt à **adapter le produit** en fonction des nouvelles informations et des besoins changeants.

Principe 03 : Livraison fréquente

Principe : Livrer fréquemment des logiciels opérationnels, avec une préférence pour des cycles courts.

Ce principe explique qu'il faut fournir des **mises à jour régulières** du logiciel pour recevoir rapidement des retours et apporter des améliorations.

Principe 04 : Collaboration quotidienne

Principe : Les utilisateurs et les développeurs doivent travailler ensemble quotidiennement tout au long du projet.

Ce principe explique qu'il faut encourager une **communication constante** entre **l'équipe de développement** et **les utilisateurs** pour mieux comprendre les besoins et ajuster les fonctionnalités.

Principe 05 : Individus motivés

Principe : Construire les projets autour d'individus motivés. Leur fournir l'environnement et le soutien dont ils ont besoin et leur faire confiance pour accomplir le travail.

Ce principe explique qu'il faut **créer un environnement de travail favorable** où les développeurs se sentent valorisés et soutenus..

Principe 06 : Communication en face à face

Principe : La méthode la plus efficace pour transmettre de l'information à une équipe de développement est la conversation en face à face.

Ce principe explique qu'il faut **privilégier les discussions directes** plutôt que les emails ou les documents pour une meilleure compréhension et une communication plus rapide.

Principe 07 : logiciel opérationnel

Principe : Un logiciel opérationnel est la principale mesure de progrès.

Ce principe explique qu'il faut **évaluer l'avancement du projet** en fonction des **fonctionnalités réellement utilisables** plutôt qu'en fonction de documents ou de plans.

Principe 08 : Développement durable :

Principe : Les processus agiles favorisent un développement durable. Les sponsors, les développeurs et les utilisateurs devraient être capables de maintenir un rythme constant indéfiniment.

Ce principe explique qu'il faut adopter un **rythme de travail soutenable** qui évite le surmenage et permet une productivité à long terme.

Principe 09 : Excellence technique

Principe : Une attention continue à l'excellence technique et à une bonne conception renforce l'agilité.

Ce principe explique qu'il faut **maintenir un haut niveau de qualité technique** pour faciliter les modifications et les améliorations du logiciel.

Principe 10 : Simplicité

Principe : La **simplicité**, c'est-à-dire l'art de maximiser l’efficacité en éliminant le travail inutile est essentiel

Ce principe explique qu’il faut se concentrer sur les fonctionnalités vraiment nécessaires et **évitez les complexités inutiles**.

Principe 11 : Équipes auto-organisées

Principe : Les meilleures architectures, exigences et conceptions émergent d'équipes auto-organisées.

Ce principe explique qu'il faut donner à l'équipe la **liberté de s'organiser et de décider** comment accomplir le travail pour tirer parti de leur expertise collective.

Principe 12 : Réflexion et ajustement

Principe : À intervalles réguliers, l'équipe réfléchit aux moyens de devenir plus efficace, puis règle et ajuste son comportement en conséquence.

Ce principe explique qu'il faut prendre le temps de **revoir régulièrement les processus et les performances de l'équipe** pour identifier les améliorations possibles et les mettre en œuvre.

Les 12 principes du Manifeste Agile

En respectant ces **12 principes**, les équipes Agile sont en mesure de **créer un environnement de travail flexible**, **collaboratif** et **axé sur la satisfaction des besoins du client**.

Ces principes encouragent les équipes à s'adapter rapidement aux changements, à travailler en étroite collaboration et à améliorer continuellement leurs processus et résultats.

Exercice 01 : méthodes agiles vs méthodes traditionnelles

Prenez 10 minutes pour lister les différences clés entre les méthodes agiles et les méthodes traditionnelles.

Les différentes méthodologies Agiles

Principales méthodologies agiles

1. Méthodologie Scrum
2. Méthodologie Kanban
3. Extreme Programming (XP)
4. Lean Software Development
5. Comment choisir la bonne méthode ?

Méthodologie Scrum

Scrum est l'une des méthodologies Agile les plus populaires largement adoptées pour la gestion de projets complexes et le développement logiciel.

Cette méthodologie a été créée en **1995**, elle a été développé par **Jeff Sutherland** et **Ken Schwaber**. Les deux créateurs ont présenté Scrum pour la première fois lors de la **conférence OOPSLA** (**O**bject-**O**riented **P**rogramming, **S**ystems, **L**anguages & **A**pplications) de 1995.

Le nom **Scrum** provient du **rugby**, où un "**scrum**" (ou "**mêlée**" en français) est une **formation spécifique** utilisée pour redémarrer le jeu après une infraction mineure. Dans ce contexte, les joueurs des **deux équipes se rassemblent** et **travaillent ensemble de manière coordonnée pour gagner la possession du ballon.**

Jeff Sutherland et Ken Schwaber, ont emprunté ce terme pour illustrer l'idée d'une **équipe** de développement **travaillant de manière cohésive** et **collaborative** pour **atteindre un objectif commun**. L'analogie avec le rugby **souligne l'importance** de la **collaboration**, de la **communication** et de l'**auto-organisation**, qui sont des principes fondamentaux de la **méthodologie Scrum**.

Méthodologie Scrum

Dans la méthodologie **Scrum**, il existe **trois rôles principaux** : le **Product Owner**, le **Scrum Master** et **l'équipe de développement.**

Il y également ce qu'on appel les **artéfacts** Scrum qui sont le **Product Backlog**, le **Sprint Backlog** et le **Product Increment**.
Les **artéfacts Scrum** sont des **outils** utilisés pour **planifier** et **suivre le travail** dans un projet **Scrum**.

Enfin il existe aussi les **événements Scrum** qui sont les **étapes clés** qui ont lieu tout au long du processus de développement Scrum pour planifier, suivre et améliorer le travail de l'équipe.

Il y a **5 événements dans Scrum** :

- ❑ Le sprint planning
- ❑ Le sprint
- ❑ Le daily scrum
- ❑ La sprint review
- ❑ La sprint retrospect

Méthodologie Kanban

Kanban est une **méthode de gestion du travail agile** visant à améliorer **l'efficacité** et la **flexibilité** des équipes. Elle utilise un **système visuel** pour gérer les tâches, généralement sous la forme de **tableaux** avec des **colonnes représentant les différentes étapes** du **flux de travail** (par exemple, "À faire", "En cours", "Terminé").

Kanban (mot japonais qui signifie « **carte visuelle** » ou « **panneau** ») a été développé par **Taiichi Ohno** chez **Toyota** dans les années **1940 et 1950**. **Kanban** a été initialement conçu dans le cadre du **système de production Toyota** pour améliorer **l'efficacité** de la chaîne d'assemblage automobile en utilisant un **système de cartes** pour signaler la nécessité de réapprovisionnement en matériaux.

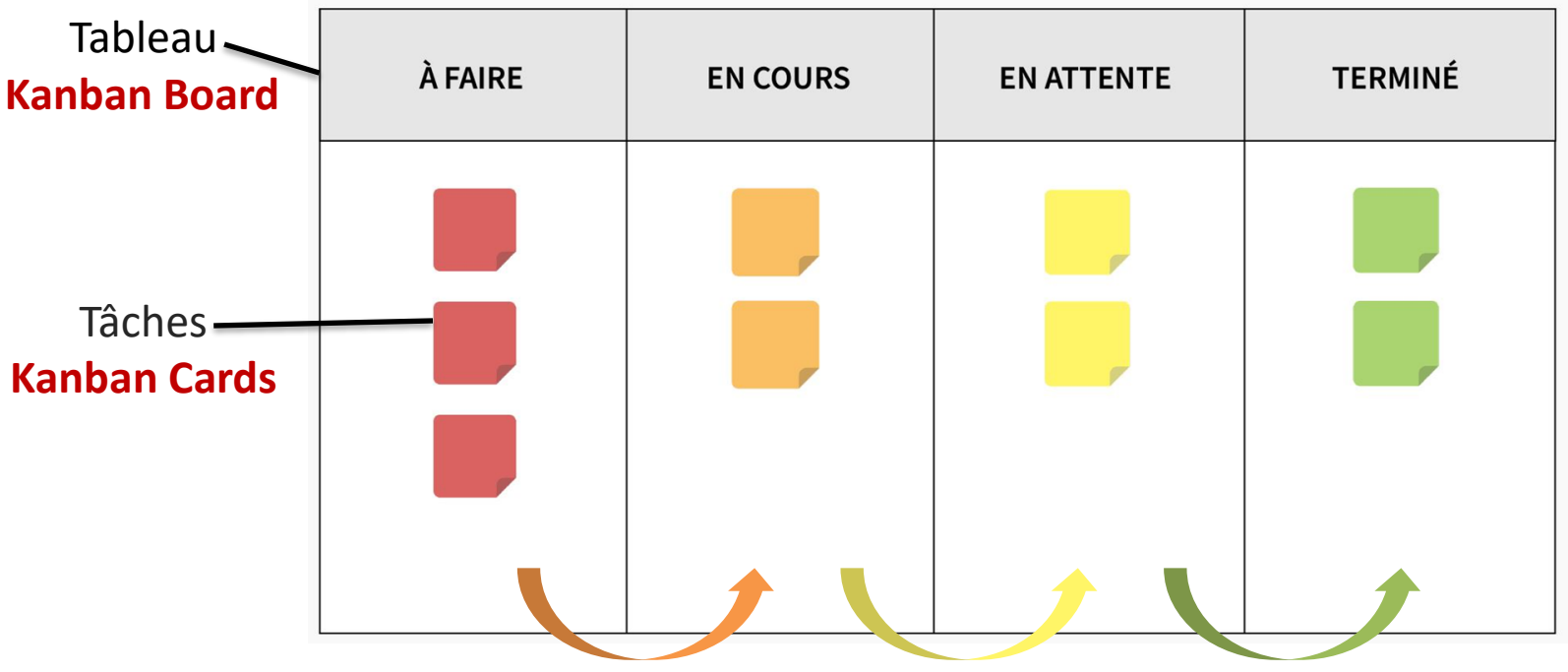
Il a été adapté aux environnements de développement logiciel et de gestion de projet à partir des années 2000.

Kanban est axé sur la **visualisation du flux de travail**, la **limitation du travail en cours** et **l'amélioration continue**.

Méthodologie Kanban : visualisation du travail

Dans les **principes de base de Kanban**, on utilise un **tableau** pour représenter visuellement le **flux de travail** et les **tâches**.

Quand une tâche avance, on la change de colonne.



Méthodologie Kanban : limitation du travail

Dans les **principes de base de Kanban**, on fixe des limites au nombre de tâches pouvant être en cours à un moment donné (**WIP**) pour éviter la surcharge.

Le but est d'encourager la **concentration** sur les **tâches en cours** avant d'en commencer de nouvelles. De ce fait nous avons une **réduction des goulots d'étranglement**, une **amélioration de la qualité du travail** et **une accélération du délai de livraison**.

Méthodologie Kanban : Amélioration continu

Dans les principes de base de Kanban, on recherche constamment des moyens d'améliorer les processus et les pratiques de travail par l'amélioration continue.

Kanban encourage les équipes à surveiller et à analyser constamment leur flux de travail pour identifier les opportunités d'amélioration.

Méthodologie Kanban : les avantages

Kanban met l'accent sur la **visualisation du flux de travail**, la **limitation du travail en cours** et **l'amélioration continue.**

Les équipes utilisent un tableau Kanban pour **gérer et suivre les éléments de travail**, en s'assurant que les priorités sont claires et que le travail progresse efficacement à travers le processus de développement.

Grande flexibilité et adaptabilité, ce qui permet aux équipes de **répondre rapidement aux changements** et **d'optimiser** constamment **leur performance**.

Kanban est particulièrement **adapté aux projets où les priorités peuvent changer fréquemment.**

Méthodologie Extreme Programming (XP)

XP est une approche Agile axée sur la **qualité du code** et **l'efficacité du développement**, qui met l'accent sur des pratiques comme la **programmation en binôme**, l'**intégration continue** et le **test automatisé**.

Extreme Programming (XP) est née officiellement en **octobre 1999** avec la parution du livre « *Extreme Programming Explained* » de **Kent Beck**. Cette méthodo a été conçu en réponse aux **défis rencontrés lors du développement de logiciels**, tels que les **changements fréquents** des exigences et la **nécessité d'améliorer la qualité du code**. Il a été initialement mis en pratique dans un projet chez **Chrysler** (**C3 project**) pour **améliorer l'efficacité du développement logiciel**.

"**Extreme Programming**" vient de l'idée de **pousser les bonnes pratiques** de développement logiciel à l'**extrême**. Par exemple, si le **test de code** est une bonne pratique, XP préconise de **tester en continu**. Si la révision du code est bénéfique, XP pousse cette pratique à l'extrême en utilisant le **pair programming**.

Méthodologie XP : les cinq valeurs fondamentales

L'**extreme programming** repose sur cinq valeurs fondamentales :

Communication : Encourage une communication continue entre les membres de l'équipe et les parties prenantes.

Simplicité : Concevoir des solutions simples et éviter les complexités inutiles.

Feedback : Obtenir des retours réguliers des clients et des utilisateurs pour s'assurer que le produit répond à leurs besoins.

Courage : Prendre des décisions audacieuses pour améliorer le projet, même si cela signifie revoir ou refactorer le code existant.

Respect : Valoriser et respecter chaque membre de l'équipe et leurs contributions.

Méthodologie XP : pratiques clés

Voici quelques pratiques de cette méthode :

Pair Programming : Deux développeurs travaillent ensemble sur une même tâche, partageant un seul poste de travail.

Test-Driven Development (TDD) : Écrire des tests automatisés avant de coder les fonctionnalités.

Refactoring : Améliorer continuellement le code pour le rendre plus clair et plus maintenable.

Intégration Continue : Fusionner fréquemment le code développé et exécuter des tests automatisés pour détecter rapidement les problèmes.

Petites Versions Fréquentes : Livrer des versions opérationnelles du logiciel fréquemment pour obtenir des retours rapides.

Méthodologie XP : les avantages

Pour les **avantages** qu'offre cette méthode nous avons :

- Une communication clair avec les clients.
- Aucun travail de programmation inutile et un code clair à tout moment.
- Un logiciel stabilisé grâce à des tests continus.
- Une prévention des erreurs grâce au pair programming.
- des modifications qui peuvent être prises en charge à court terme.

Méthodologie Lean

La méthodologie **Lean**, développée par **Toyota** en sortie de la seconde guerre mondiale, vise à maximiser la valeur pour le client tout en minimisant le gaspillage.

Le terme "**Lean**" (qui signifie "**maigre**" ou "**dépouillé**") a été utilisé pour la première fois par **John Krafcik** en **1988** dans un article intitulé "**Triumph of the Lean Production System**". Il décrit un système de production où tout ce qui ne crée pas de valeur est éliminé, rendant ainsi le processus plus "maigre" et plus efficace.

Au cours des années qui suivront 2000, il y aura plusieurs déclinaisons dont le **lean software development** dans le domaine du **développement logiciel**.

Ce modèle met en place sept principes :

1. **Éliminer les gaspillages :** comme pour le ***lean***, le gaspillage est défini comme ce qui n'apporte pas de valeur au produit. La valeur étant définie du point de vue de l'utilisateur.
2. **Améliorer** l'apprentissage en Favorisant un environnement qui encourage l'apprentissage et l'amélioration continue.
3. **Retarder** l'engagement en retardant les décisions importantes jusqu'à ce que l'on dispose de toutes les informations nécessaires pour faire des choix plus éclairés.
4. **Livrer** aussi vite que possible en réduisant le temps entre le début et la livraison d'une fonctionnalité.
5. **Donner** le pouvoir à l'équipe en leur fournissant les ressources, les responsabilités et la liberté de prendre des décisions.
6. **Intégrer** la qualité dès la conception comme les revues de code, et les tests automatisés.
7. **Considérer** le produit dans sa globalité et ainsi optimiser l'ensemble du système.

Méthodologie Lean : avantages

Cette **méthodologie** de gestion de projet permet :

- ❑ **Une réduction** des gaspillages
- ❑ **Une amélioration** continue
- ❑ **Des décisions** retardée et éclairée
- ❑ **Des livraisons** rapides
- ❑ **Une autonomisation** de l'équipe
- ❑ **Une intégration** de la qualité dès la conception
- ❑ **Une vision** globale du produit

Comment choisir la bonne méthode ?

Chaque **méthodologie Agile** présente des **avantages** et des **inconvénients**, et leur efficacité **dépend des besoins spécifiques de chaque projet et équipe**.

Scrum : Idéal pour les projets qui nécessitent des **livraisons rapides** et **régulières**, ainsi qu'une **structure bien définie avec des rôles clairement définis**. Scrum est particulièrement adapté aux équipes qui sont nouvelles dans le développement Agile et qui ont besoin d'une structure pour les guider.

Kanban : Convient aux projets où les **priorités peuvent changer fréquemment** et où la **livraison continue des fonctionnalités** ou des **produits est importante**. Kanban est également utile pour les équipes qui cherchent à **améliorer leur flux de travail** et à **réduire les goulots d'étranglement**.

Comment choisir la bonne méthode ?

Extreme Programming (XP) : Idéal pour les projets où la **qualité du code** et la **capacité à s'adapter rapidement** aux changements sont essentielles. XP convient particulièrement aux équipes de développement qui ont déjà une certaine expérience avec les méthodologies Agile et qui cherchent à **améliorer leurs pratiques de développement et leur collaboration**.

Lean : Convient aux projets qui mettent l'accent sur la **maximisation de la valeur pour le client** et la **minimisation du gaspillage** dans les processus de développement et de gestion de projet.

Comment choisir la bonne méthode ?

En définitive, une **méthode Agile est rarement utilisée complètement seule** :

Il est tout à fait possible **de combiner plusieurs méthodes agiles** pour adapter les pratiques à l'équipe et au projet. En fait, de nombreuses équipes combinent des éléments de différentes méthodes, **en fonction de leurs besoins spécifiques** et de leurs **objectifs**.

L'essentiel est de **rester fidèle aux principes agiles** de **collaboration**, de **flexibilité**, **d'adaptabilité** et **d'amélioration continue** tout en adaptant les méthodes aux besoins.

Par exemple, certaines équipes peuvent adopter le cadre Scrum avec ses rôles et événements, tout en intégrant les pratiques de visualisation du flux de travail et de limitation du travail en cours de Kanban.

Exercice 02 : Choix de la méthodologie agile

Vous êtes chargé de conseiller une entreprise sur la méthodologie agile la plus adaptée pour différents projets de développement de sites web et de logiciels.

Pour chaque projet, identifiez la méthode agile la plus appropriée (Scrum, Kanban, Extreme Programming, Lean Software Development) et justifiez votre choix. Vous pouvez également combiner plusieurs méthodologies si cela est pertinent et justifié.

Exercice 02 : projet 01

Projet : Maintenance et Mise à Jour d'un Logiciel de Gestion des Ressources Humaines (GRH).

Objectif Principal : Améliorer la qualité et la performance d'un logiciel existant en déployant régulièrement des correctifs et des mises à jour.

Exercice 02 : projet 02

Projet : Optimisation des Processus de Développement d'un Site Web d'Actualités.

Objectif Principal : Identifier et éliminer les gaspillages dans le processus de développement pour réduire les coûts et améliorer l'efficacité globale.

Exercice 02 : projet 03

Projet : Développement d'un Nouveau Site E-commerce.

Objectif Principal : Lancer rapidement une plateforme e-commerce avec des fonctionnalités de base et ajouter des fonctionnalités avancées au fil du temps en fonction des retours des utilisateurs.

Exercice 02 : projet 04

Projet : Création d'un Logiciel de Gestion de Projet pour une Entreprise.

Objectif Principal : Développer un logiciel de gestion de projet robuste et scalable avec une forte collaboration entre les équipes de développement et les utilisateurs finaux.

À suivre

m2iformation.fr