INF2015 – Développement de logiciels dans un environnement Agile

eXtreme Programming

Jacques Berger

© 2012-2013 Jacques Berger

Objectifs

Présenter la méthode eXtreme Programming

© 2012-2013 Jacques Berger

Prérequis

Aucun

© 2012-2013 Jacques Berger

XP

eXtreme Programming

Méthode conçue par Kent Beck en 1999

© 2012-2013 Jacques Berger

XP

Un ensemble de valeurs, de principes et de pratiques pour améliorer la qualité des logiciels et la productivité des développeurs

Objectif : Développer des logiciels remarquables

© 2012-2013 Jacques Berger

XP

XP est une méthode qu'on peut adopter petit-à-petit, selon nos besoins et selon la maturité de l'équipe (ou l'entreprise)

Les pratiques sont bonnes seules mais sont optimales lorsqu'utilisées toutes ensembles

© 2012-2013 Jacques Berger

XP

XP pousse à l'extrême des pratiques simples et efficaces

Les pratiques de XP favorisent l'adaptation aux changements

© 2012-2013 Jacques Berger

Valeurs

La communication

Une équipe doit communiquer beaucoup. Un développeur moyen qui communique bien vaut mieux qu'un super-développeur solitaire. La communication règle beaucoup de problèmes humains.

© 2012-2013 Jacques Berger

Valeurs

La simplicité

On préconise les solutions simples. On essaie toujours la solution qui semble la plus simple.

© 2012-2013 Jacques Berger

Valeurs

Le feedback

On veut du feedback le plus tôt possible, le plus souvent possible, sur tout (logiciel, code, documentation, etc.)

© 2012-2013 Jacques Berger

Valeurs

Courage

Dire les choses comme elles sont; écarter les solutions qui ne fonctionnent pas; rechercher des solutions plus simples; rechercher du feedback significatif.

© 2012-2013 Jacques Berger

Valeurs

Le respect

Respecter son travail et le travail des autres; rien ne peut sauver un projet si un membre de l'équipe s'en fout.

© 2012-2013 Jacques Berger

Principes

Humanité

Les développeurs sont des personnes. Il faut les traiter comme des personnes. Leurs besoins : sécurité; accomplissement; appartenance; croissance; intimité.

© 2012-2013 Jacques Berger

Principes

Économique

Ça coûte de l'argent. Poser des gestes qui ont de la valeur; rencontrer les objectifs de l'entreprise. Produire de la valeur pour l'entreprise.

© 2012-2013 Jacques Berger

Principes

Bénéfice mutuel

Chaque activité doit profiter à toutes les parties prenantes.

© 2012-2013 Jacques Berger

Principes

Auto-similitude

Lorsqu'une solution fonctionne, on peut l'utiliser dans un nouveau contexte ou à un différent niveau.

© 2012-2013 Jacques Berger

Principes

Amélioration

Rien n'est parfait mais on peut perfectionner nos processus, notre code, notre design, nos méthodes de travail, etc.

© 2012-2013 Jacques Berger

Principes

Diversité

La diversité des compétences dans une équipe est une force. Plusieurs perspectives sur un même problème permet de trouver la solution la plus simple.

© 2012-2013 Jacques Berger

Principes

Réflexion

Faire son travail, mais également réfléchir sur le fonctionnement et les échecs. Ne pas cacher ses erreurs mais apprendre d'elles.

© 2012-2013 Jacques Berger

Principes

Flux

Un flux continu d'activités plutôt que des phases distinctes.

© 2012-2013 Jacques Berger

Principes

Opportunité

Les problèmes sont des opportunités pour améliorer le logiciel.

© 2012-2013 Jacques Berger

Principes

Redondance

Une pratique pour améliorer la qualité, c'est bien. Plusieurs pratiques pour améliorer la qualité, c'est beaucoup plus fiable.

© 2012-2013 Jacques Berger

Principes

L'échec

Ne pas avoir peur d'échouer. Il y a 4 solutions possibles, on essaie les 4. Si les 4 échouent, l'apprentissage acquis en chemin nous aidera à trouver une solution.

© 2012-2013 Jacques Berger

Principes

Qualité

On ne peut pas diminuer la qualité. Un projet n'ira pas plus vite si on sacrifie la qualité. Ce n'est pas négociable.

© 2012-2013 Jacques Berger

Principes

Petits pas de bébé

Faire des petits changements à la fois; des petites intégrations; des petites mises à jour, etc. Éviter les déploiements de type «Big Bang»

© 2012-2013 Jacques Berger

Pratiques

Les pratiques primaires

La base; les premières pratiques à adopter

© 2012-2013 Jacques Berger

Pratiques

S'asseoir ensemble

L'équipe doit travailler ensemble; aire ouverte; la proximité favorise la communication.

© 2012-2013 Jacques Berger

Pratiques

L'équipe complète

L'équipe contient toutes les compétences et perspectives nécessaires pour compléter le projet. Être engagé dans l'équipe; sentiment d'appartenance.

© 2012-2013 Jacques Berger

Pratiques

Espace de travail informatif

Avoir une idée de l'avancement du projet ou du fonctionnement de l'équipe en quelques secondes, uniquement en regardant l'espace de travail de l'équipe. Ex : tableaux et post-its

© 2012-2013 Jacques Berger

Pratiques

Travail sous tension

Maintenir une cadence raisonnable; ne pas brûler les membres de l'équipe; heures de travail limitées. Un développeur travaille mieux lorsqu'il est préparé, relax et bien reposé. Bien gérer le temps productif.

© 2012-2013 Jacques Berger

Pratiques

Programmer en binôme

2 développeurs sur une machine; deux têtes valent mieux qu'une. Une personne code, l'autre observe l'ensemble (design, erreurs, etc.)

© 2012-2013 Jacques Berger

Pratiques

Scénarios (stories)

Description d'une fonctionnalité visible par l'utilisateur final. On élabore les scénarios uniquement lorsque c'est nécessaire. On choisit les scénarios à implémenter.

© 2012-2013 Jacques Berger

Pratiques

Cycle hebdomadaire

Planifier le travail une semaine à la fois. Une réunion au début de la semaine où l'on discute : de la progression du dernier cycle; le client choisit les scénarios à implémenter durant la semaine; on divise les scénarios en tâches.

© 2012-2013 Jacques Berger

Pratiques

Cycle trimestriel

On planifie le travail un trimestre à la fois. Identifier les goulots d'étranglement; planifier les thèmes du trimestre; choisir des scénarios reliés au thème pour le trimestre.

© 2012-2013 Jacques Berger

Pratiques

Relâchement

Inclure des tâches mineures qu'on peut facilement laisser tomber en cas de retard. L'idée est d'avoir un certain jeu en cas de pépin. On peut toujours rajouter des scénarios et livrer plus que prévu au client si tout va bien.

© 2012-2013 Jacques Berger

Pratiques

Construction en 10 minutes

Automatiser le build. Son exécution ne devrait pas prendre plus de 10 minutes.

© 2012-2013 Jacques Berger

Pratiques

L'intégration continue

Plusieurs commits par jour pour chaque développeur. On intègre le travail de tous, au fur et à mesure.

© 2012-2013 Jacques Berger

Pratiques

Les tests d'abord

Rédiger les tests unitaires avant d'écrire le code.
Écrire le test, écrire le code, refactoring.

© 2012-2013 Jacques Berger

Pratiques

Conception incrémentale

Investir au quotidien dans le design. Effectuer fréquemment du refactoring.

© 2012-2013 Jacques Berger

Pratiques

Pratiques corollaires

D'autres pratiques de XP mais qu'il vaut mieux attendre que les pratiques primaires soient bien implantées avant de s'y lancer.

© 2012-2013 Jacques Berger

Pratiques

Pratiques corollaires
- Implication réelle du client
- Déploiement incrémental
- Continuité de l'équipe
- Réduire les équipes
- L'analyse des causes des défauts
- Code partagé (sans propriétaire)
- Ne maintenir que le code et les tests
- Une seule base de code
- Déploiement quotidien
- Pay-per-use

© 2012-2013 Jacques Berger

Changement

Introduire XP dans une entreprise peut entraîner beaucoup de changement. Ceci peut être difficile.

Les développeurs devront également s'y adapter et pourraient résister à certaines pratiques.

Ne brûlez pas d'étapes.

© 2012-2013 Jacques Berger

Objectifs

Nous voulons obtenir :

- La qualité
- La productivité

Ne jamais sacrifier la qualité pour obtenir la productivité. Visez toujours la qualité.

© 2012-2013 Jacques Berger

Liens

Extreme programming
http://www.extremeprogramming.org/

© 2012-2013 Jacques Berger