

Méthodes de Développement : Extreme Programming

Claire Mouton - CREATIS, Lyon, France
claire.mouton@creatis.insa-lyon.fr

Gestion de projet logiciel

- Penser à l'avance l'architecture du code (UML, ...)
- Penser à l'évolution du code lors de son développement (cycle en V, ...)
- Méthode agile (Xtreme programming, ...)
- Méthode d'intégration continue (tests de non-régression, tests unitaires, ...)

Unified Modeling Language (UML)

Adopter une méthode de développement

Unified Modeling Language (UML)



Dessins qui résument les dépendances, la structure ou les fonctionnalités du programme

Faire le lien entre la théorie et la pratique

« Je faisais ce design pattern sans le savoir! »

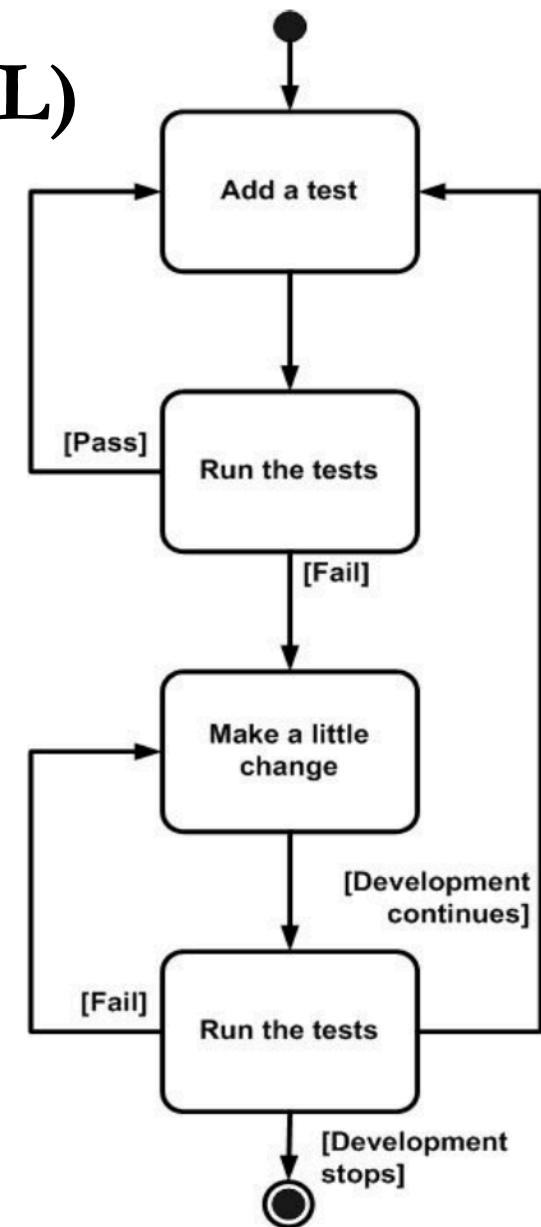
Conception dirigée par les modèles : un langage graphique pour modéliser

- pour mettre à plat le problème, réfléchir sur le programme en amont
- pour être compris par les autres
- pour comprendre les autres

Adopter une méthode de développement

Unified Modeling Language (UML)

Les diagrammes de machines à état :



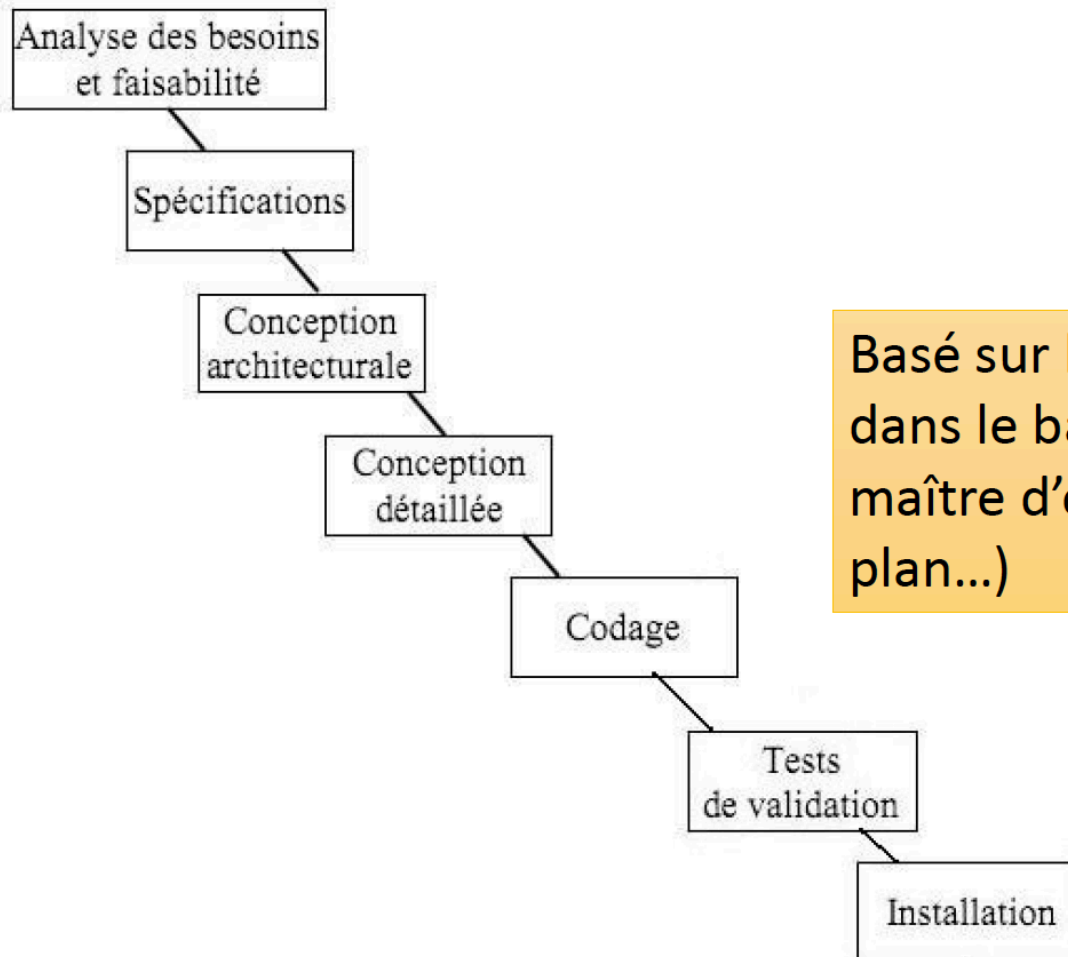
Copyright 2003 Scott W. Ambler

Les cycles du développement logiciel

Adopter une méthode de développement

Les cycles du développement logiciel

waterfall model (1970 Winston W. Royce)



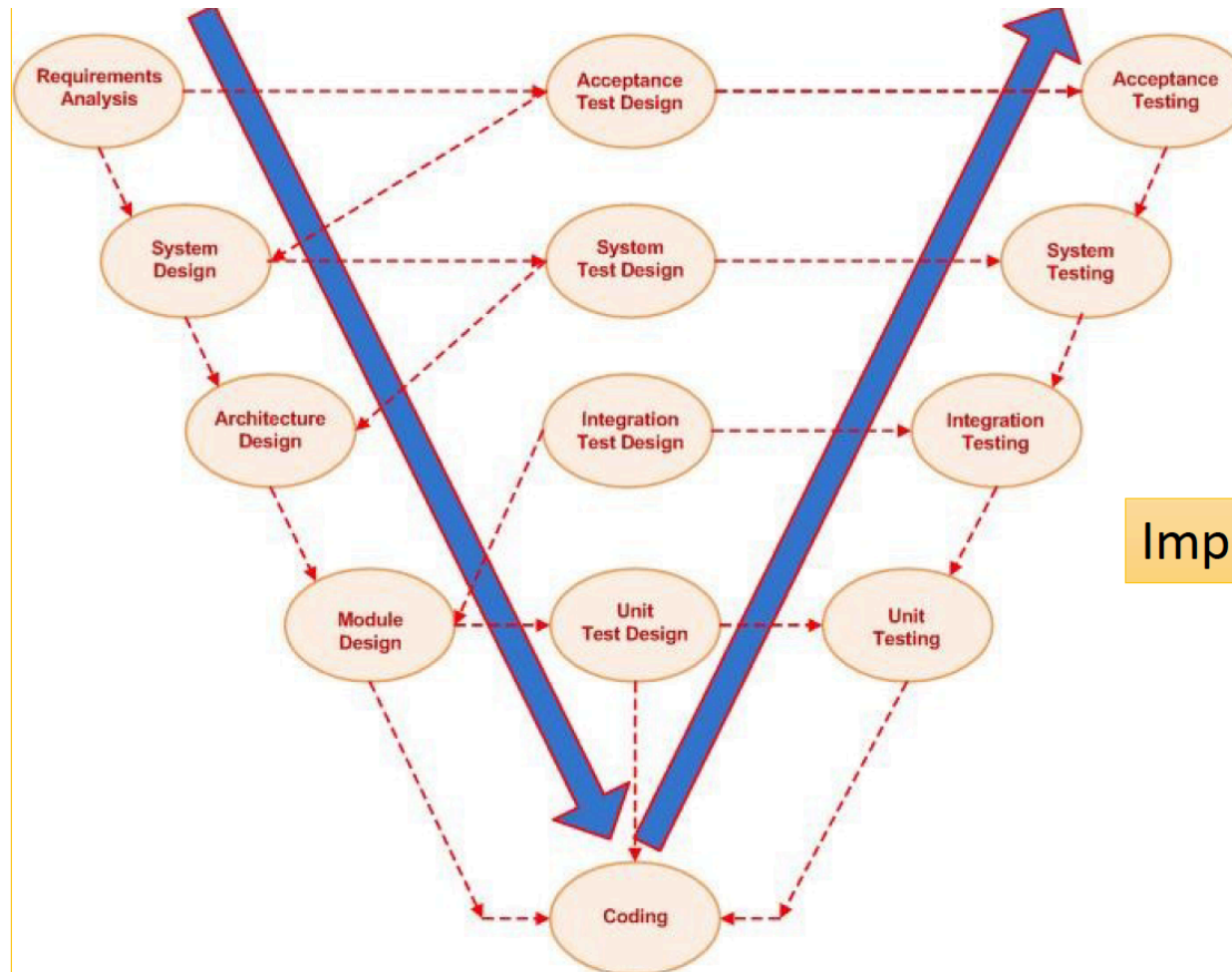
Basé sur les méthodes appliquées dans le bâtiment (maître d'œuvre / maître d'ouvrage, importance du plan...)

©Christophe.moustier

Adopter une méthode de développement

Cycle en V

Vision en un bloc



V model,
années 80

Importance des tests

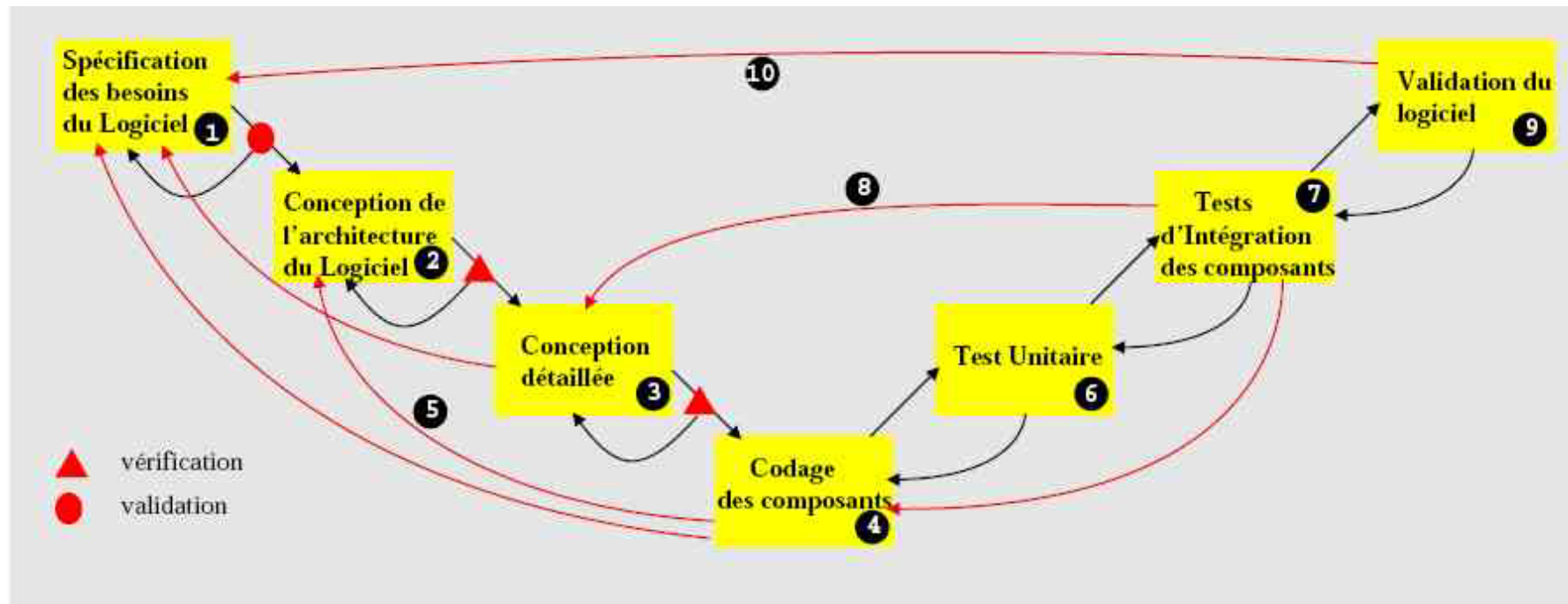
<http://www.coleyconsulting.co.uk/v-model.htm>

Adopter une méthode de développement

Cycle en V

Vision en un bloc

En réalité, le cycle de développement se déroule ainsi :



Extreme Programming

Extreme Programming

Agile Manifesto, signé par 17 personnalités du Génie Logiciel en 2001 :

- Un processus de développement
- Un état d'esprit
- Un ensemble de bonnes pratiques

→ Adapté aux collaborations ingénieur / chercheur pour lesquels les besoins sont fluctuants

Adopter une méthode de développement

Manifeste pour le développement Agile de logiciels

Nous découvrons comment mieux développer des logiciels par la pratique et en aidant les autres à le faire. Ces expériences nous ont amenés à valoriser :

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

Nous reconnaissons la valeur des seconds éléments, mais privilégions les premiers.

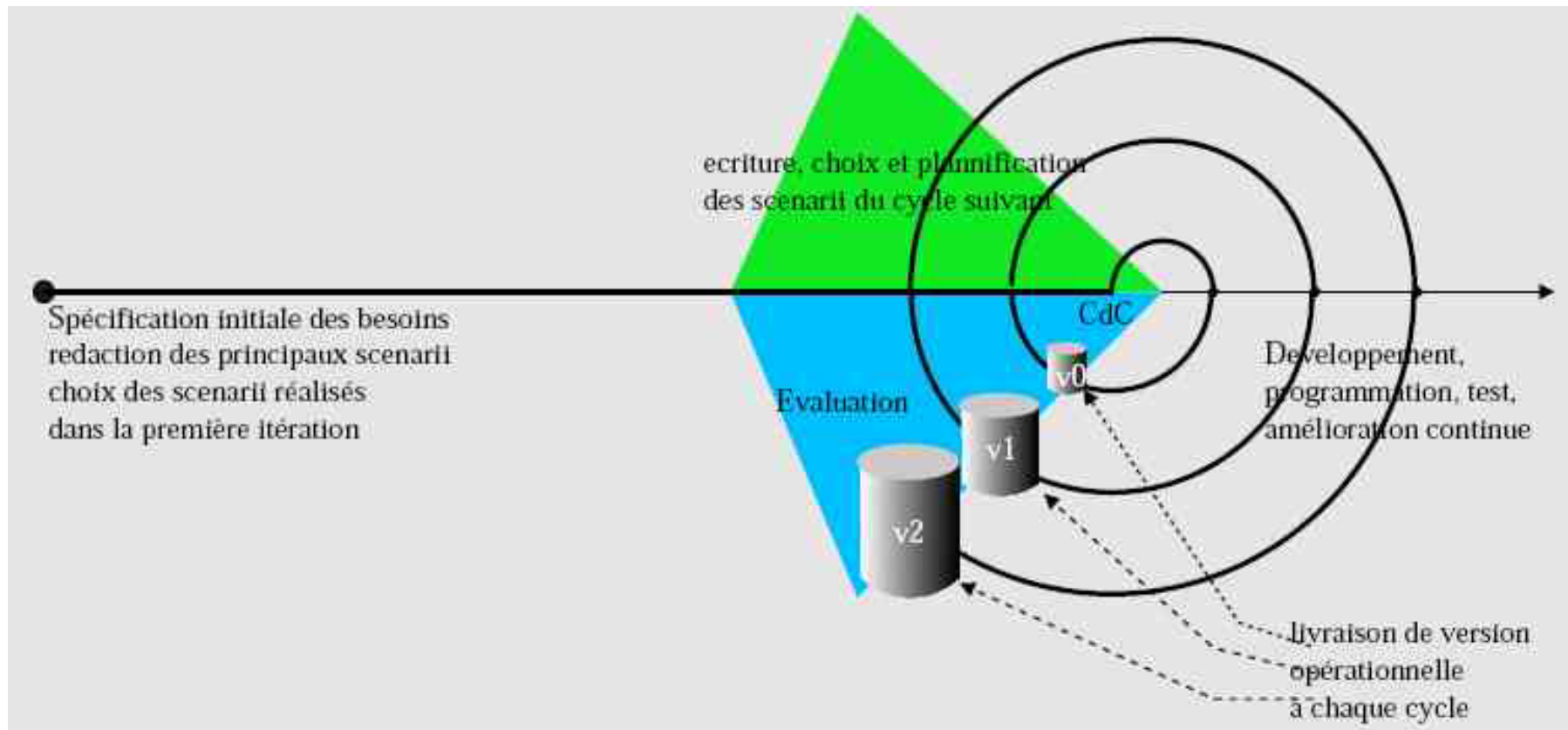
Agilité
1990 ... 2000

<http://agilemanifesto.org/iso/fr/>

-relation entre
individus
-adaptation

Adopter une méthode de développement

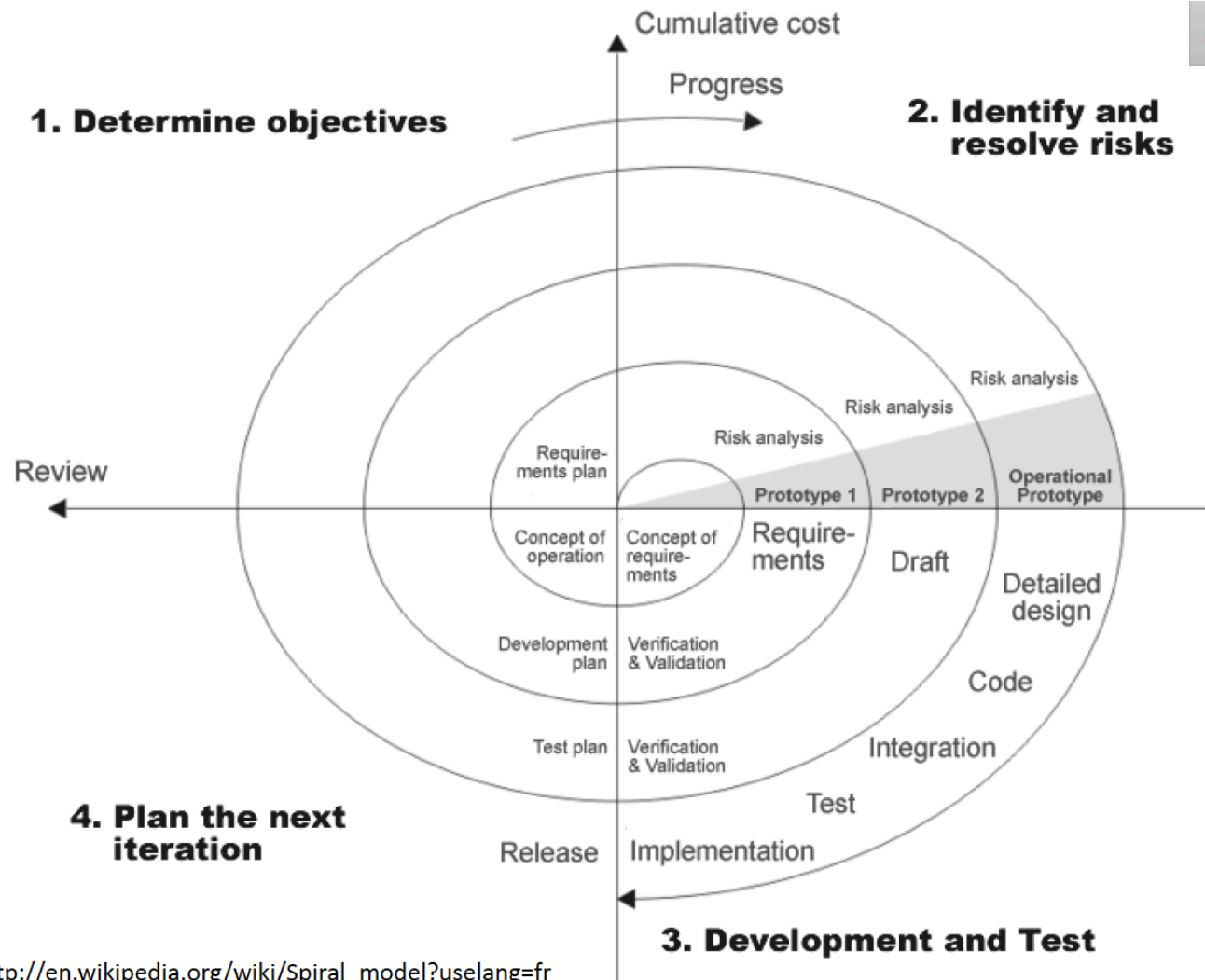
Extreme Programming



Adopter une méthode de développement

Extreme Programming

Spiral, Barry Boehm 1986



Itératif, prototypes, risques

http://en.wikipedia.org/wiki/Spiral_model?uselang=fr

Adopter une méthode de développement

Extreme Programming

Développement piloté par les tests

Règles de codage

Conception simple

Livraisons frequentes

Travail en binome

Metaphore

Integration continue

Remaniement

Rythme durable

Client sur site

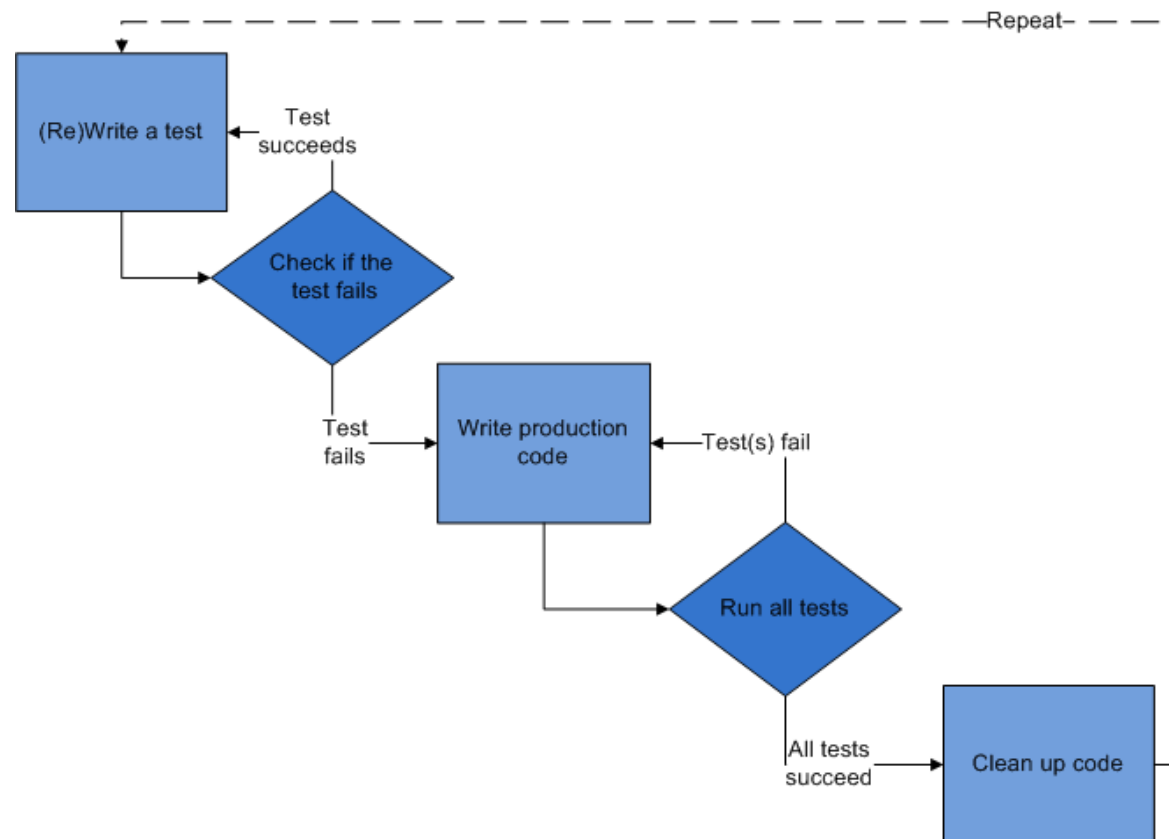
Responsabilite collective du code

Planification iterative



Adopter une méthode de développement

Extreme Programming : Développement piloté par les tests



Adopter une méthode de développement

Pratiques agiles (XP)



PROJET	CODE	EQUIPE
Livraisons fréquentes : l'équipe vise la mise en production rapide d'une version minimale du logiciel, puis elle fournit ensuite régulièrement de nouvelles livraisons en tenant compte des retours du client.	Conception simple : on ne développe rien qui ne soit utile tout de suite.	Programmation en binômes : les développeurs travaillent en binômes, ces binômes étant renouvelés fréquemment.
Planification itérative : un plan de développement est préparé au début du projet, puis il est revu et remanié tout au long du développement pour tenir compte de l'expérience acquise par le client et l'équipe de développement.	Tests unitaires : les développeurs mettent en place une batterie de tests structurels qui leur permettent de valider leur développement et de faire des modifications sans crainte (garantie de non régression).	Responsabilité collective du code : chaque développeur est susceptible de travailler sur n'importe quelle partie de l'application.
Client sur site : le client est intégré à l'équipe de développement pour répondre aux questions des développeurs et définir les tests fonctionnels.	Restructuration (refactoring) : le code est en permanence réorganisé pour rester aussi clair et simple que possible.	Rythme régulier : l'équipe adopte un rythme de travail qui lui permet de fournir un travail de qualité constant tout au long du projet.
Tests de recette : les testeurs mettent en place des tests fonctionnels qui vérifient que le logiciel répond aux exigences du client. Ces tests permettent une validation du cahier des charges par l'application livrée.	Intégration continue : l'intégration des nouveaux développements est faite de façon continue de manière à délivrer de la valeur de façon incrémentale.	Métaphore : les développeurs s'appuient sur une description commune du design.
Règles de codage : les développeurs se plient à des règles de codage strictes définies par l'équipe elle-même.		

Adopter une méthode de développement

Avantages

- fun
- Communication entre les acteurs améliorée
- Développements pertinents/besoins

Inconvénients

- Changement des méthodes de travail
 - Collaboration/synchronisation difficile avec les autres équipes qui n'utilisent pas cette méthode.
 - Bonne volonté, entente et communication de tous les participants impératif.
- Gros travail de rédaction pour les Clients
- Les développeurs doivent être autonomes
 - Refactoring

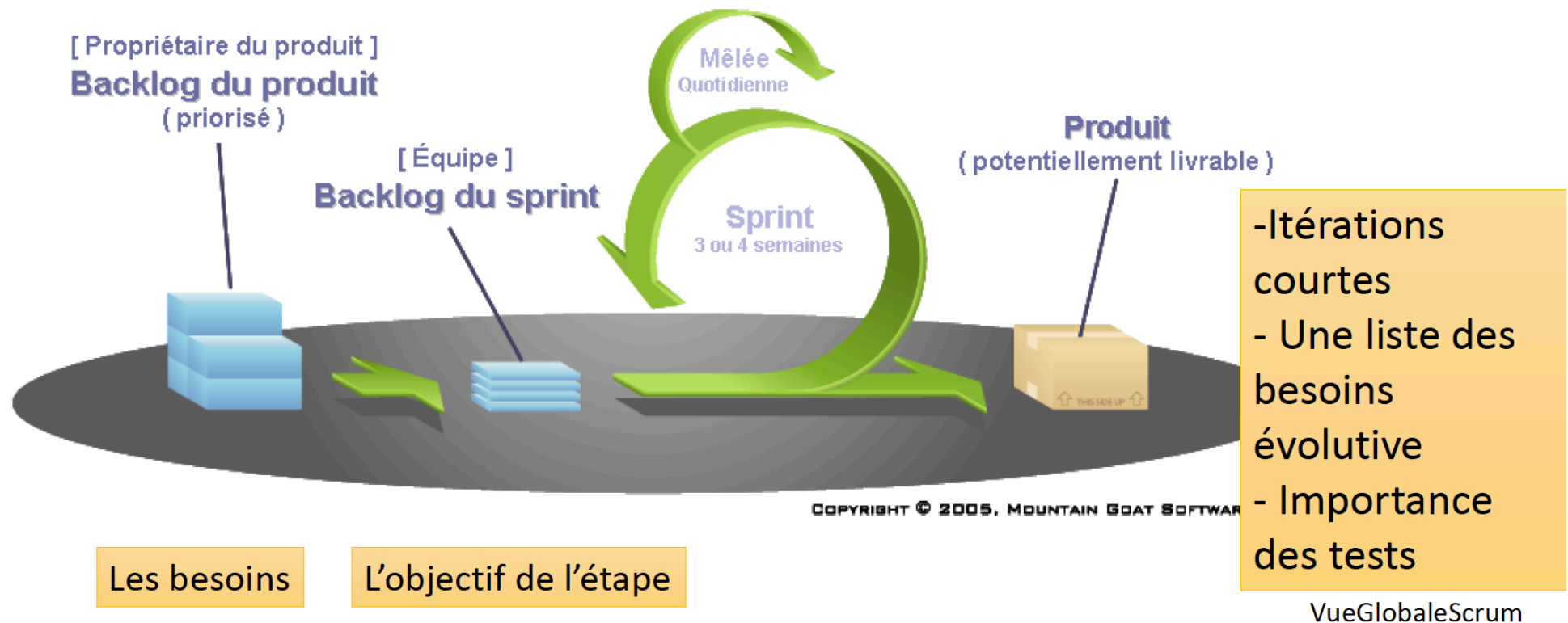
Agilité

1990 ... 2000

Agile: <http://www.journaldunet.com/developpeur/expert/56616/la-methode-agile---optimisation-de-la-relation--client---fournisseur.shtml>

Adopter une méthode de développement

Scrum
2001, Ken Schwaber

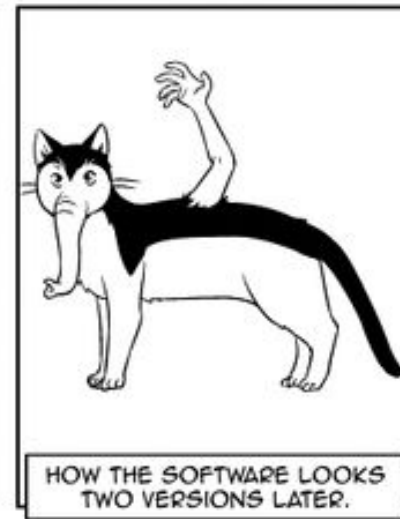
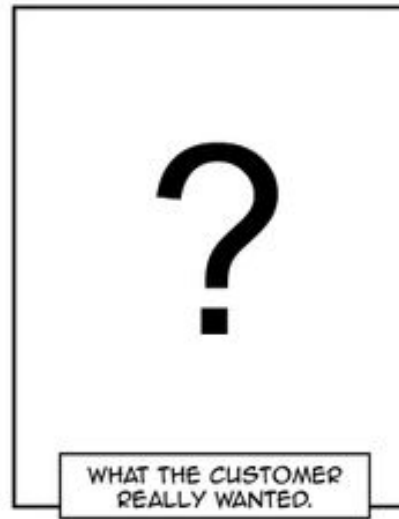
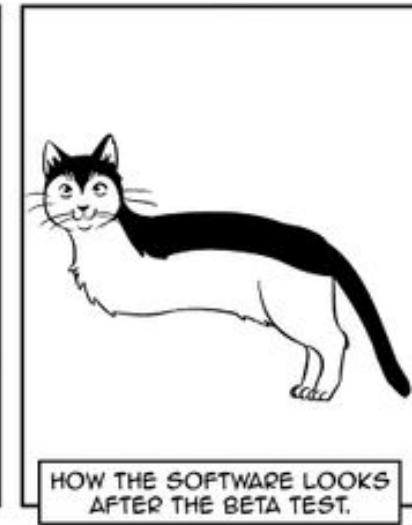
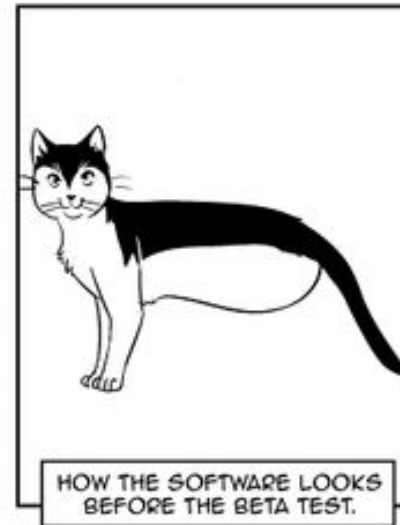
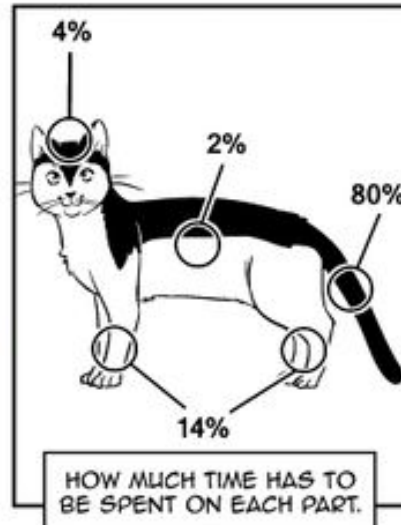
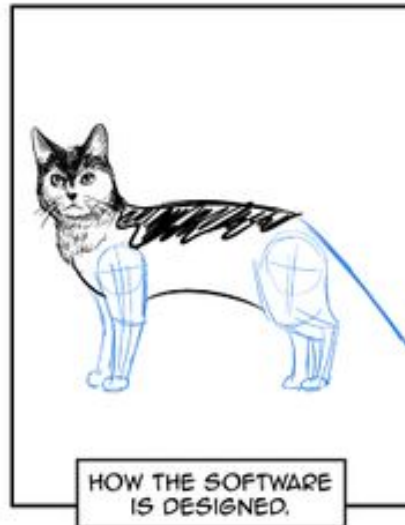


What software would we get
without any test and methodology?

What software would we get
without any test and methodology?

Your software would live its own odd life...

Richard's guide to software development



Sandra and Woo by Oliver Knörzer (writer) and Powree (artist) – www.sandraandwoo.com

Et vous, comment vous faites pour la conception et le développement de vos applications ?

Vos pratiques ?

Vos retours d'expérience ?

Des erreurs à ne pas faire / des conseils ?

Des compléments à ma présentation ?