Intégration continue

Anne Cadiou

Laboratoire de Mécanique des Fluides et d'Acoustique

Informatique scientifique pour le calcul
École doctorale
2016-2017

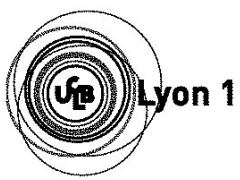


Table des matières

Table des matières

Définition

(*d'après Wikipédia*)

L'**intégration continue** est un ensemble de pratiques utilisées en génie logiciel consistant à vérifier à chaque modification de code source que le résultat des modifications ne produit **pas de régression** dans l'application développée. Le concept a pour la première fois été mentionné par Grady Booc et se réfère généralement à la pratique de l'**extreme programming**. Le principal but de cette pratique est de détecter les problèmes d'intégration au plus tôt lors du développement. De plus, elle permet d'**automatiser l'exécution des suites de tests** et de voir l'évolution du développement du logiciel.

À quoi ça sert ?

assurer la qualité des développements
s'appuie sur

- des outils de versionnement
- des tests unitaires
- des tests de non-régression
- des outils de compilation du codes
- des outils d'exécution et de reporting du code

au moyen de

- scripts
- serveurs

Table des matières

Outils

avec des scripts

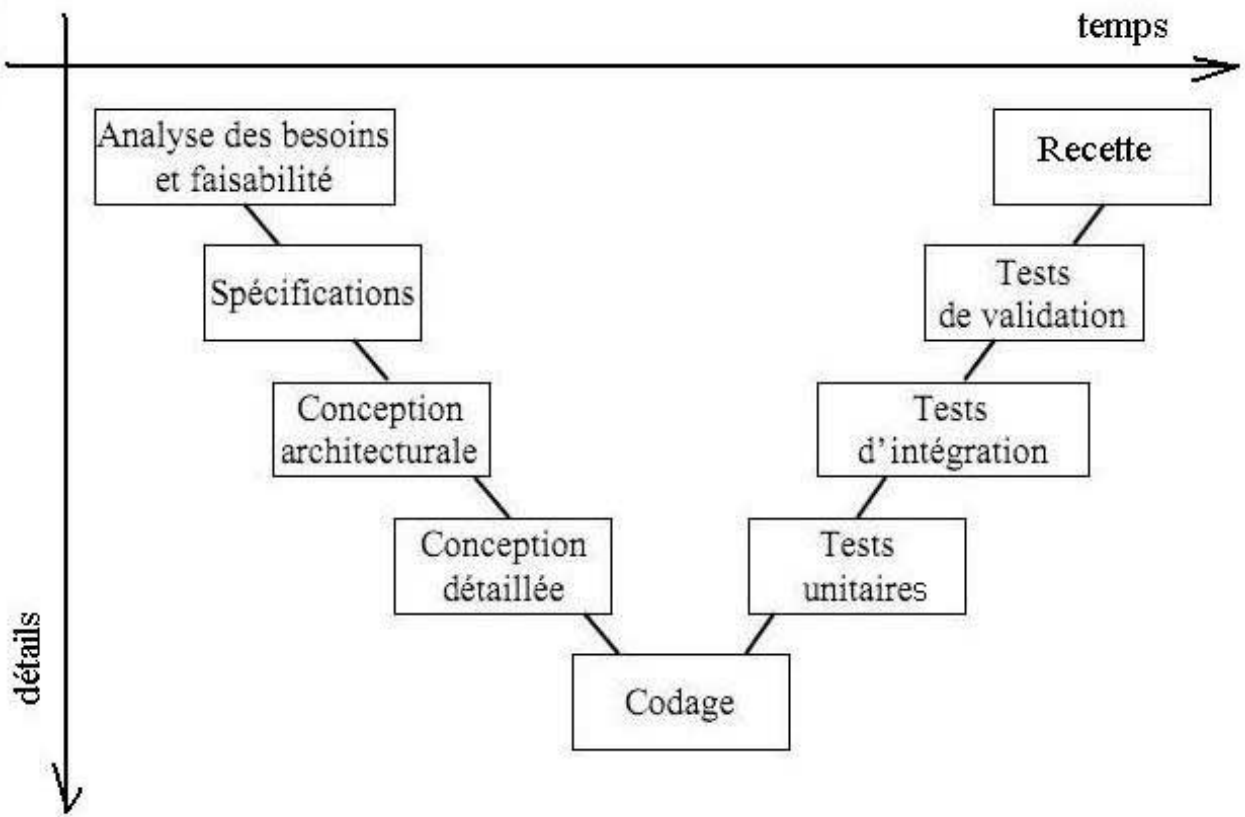
- Makefile, CMake, CTest, bash, matlab, python, cron, ...

avec des serveurs

- Jenkins (ex. Hudson)
- Strider-CD
- TeamCity
- CruiseControl
- Gitlab-CI
- CDash ...

Principes

- Méthodes prédictives,
  Cycle en V (*d'après wikipédia*)



Principes

- Méthodes agiles,
  Extreme Programming (XP)

  - *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

$\Rightarrow$ adapté aux développements de calcul scientifique pour la recherche

Développement piloté par les tests

- Définition et développement d'un test (unitaire)
- Développement de la fonctionnalité associées dans le code
- Tests (de validation et de non-régression)

$\Rightarrow$ écriture des tests avant et en même temps que le code

$\Rightarrow$ adaptation continue

En pratique

Prérequis

- partage du code source via un gestionnaire de version
- intégration quotidienne des modifications par les développeurs
- développement de tests d'intégration de l'application

Avantages

- test immédiat des modifications
  - détection rapide de codes incompatibles ou manquants
  - facilite le débogage
- assure la non-régression
- automatise le reporting

Mise en oeuvre