

LIFAPCD

CONCEPTION ET DEVELOPPEMENT D'APPLICATIONS

Université Claude Bernard Lyon 1
Licence Informatique 2^{ème} année
Nicolas Pronost

Conception et gestion de projet

Bilan des UE

- LIFAPI : Algorithmique et programmation
 - notions de bases, fonctions, paramètres, etc.
- LIFAPSD : Programmation et TDA
 - modélisation en classe, structuration des données et algorithmes classiques
- LIFAPCD : Développement d'un projet
 - expérience et mise en application des notions acquises dans les UE précédentes (surtout I et SD)
- LIFAPC : Algorithme et complexité
 - évaluer les qualités et les défauts des algorithmes

Définition de projet

- **Projet**

- ensemble des **actions** à entreprendre
- afin de répondre à un **besoin**
- défini dans des **délais** fixés (début, fin)
- mobilisant des **ressources** identifiées (humaines et matérielles)
- possédant un **coût** : budgétisation
- où l'on appelle «**livrables**» les résultats attendus du projet

- Maître d'ouvrage (demandeur du projet) ≠ maître d'œuvre (réalisateur du projet)

Problèmes classiques en développement logiciel

- On passe son temps à réinventer la roue
 - Perte de temps, perte d'argent
- Génération après génération, les informaticiens refont les mêmes erreurs
 - Transmission du savoir-faire non optimale
- La maintenance de la plupart des logiciels est difficile à faire
 - Manque de standards
- Presque tous les développements logiciels coûtent chers

Problèmes classiques en développement logiciel

- Les logiciels requièrent de la flexibilité car les technologies évoluent rapidement
 - Langages
 - Intergiciels (en anglais, *middleware*)
 - Composants
 - Protocoles
- La plupart des applications écrites aujourd'hui devront survivre aux changements technologiques de demain

Logiciel de très grande taille (LTGT)

Tentative de définition

- Quantitativement
 - C: > 100 000 lignes de code
 - Java/C++: > 1 000 classes
- Qualitativement
 - Impossible d'avoir tout le code en tête
 - Nombreux développeurs
 - Longue durée de vie
 - Investissement élevé (argent, années-hommes)
 - Coût du *reengineering* prohibitif

Exemples de type de LTGT

- Scientifique
 - Beaucoup de calculs, peu d'utilisateurs
 - Centralisé ou fortement distribué
 - Fortran, C/C++
 - modèles météorologiques et astronomiques
- E-business
 - Systèmes distribués
 - Très grand nombre de clients
 - Java, C#
 - serveur d'application e-business

Méthodes de conception

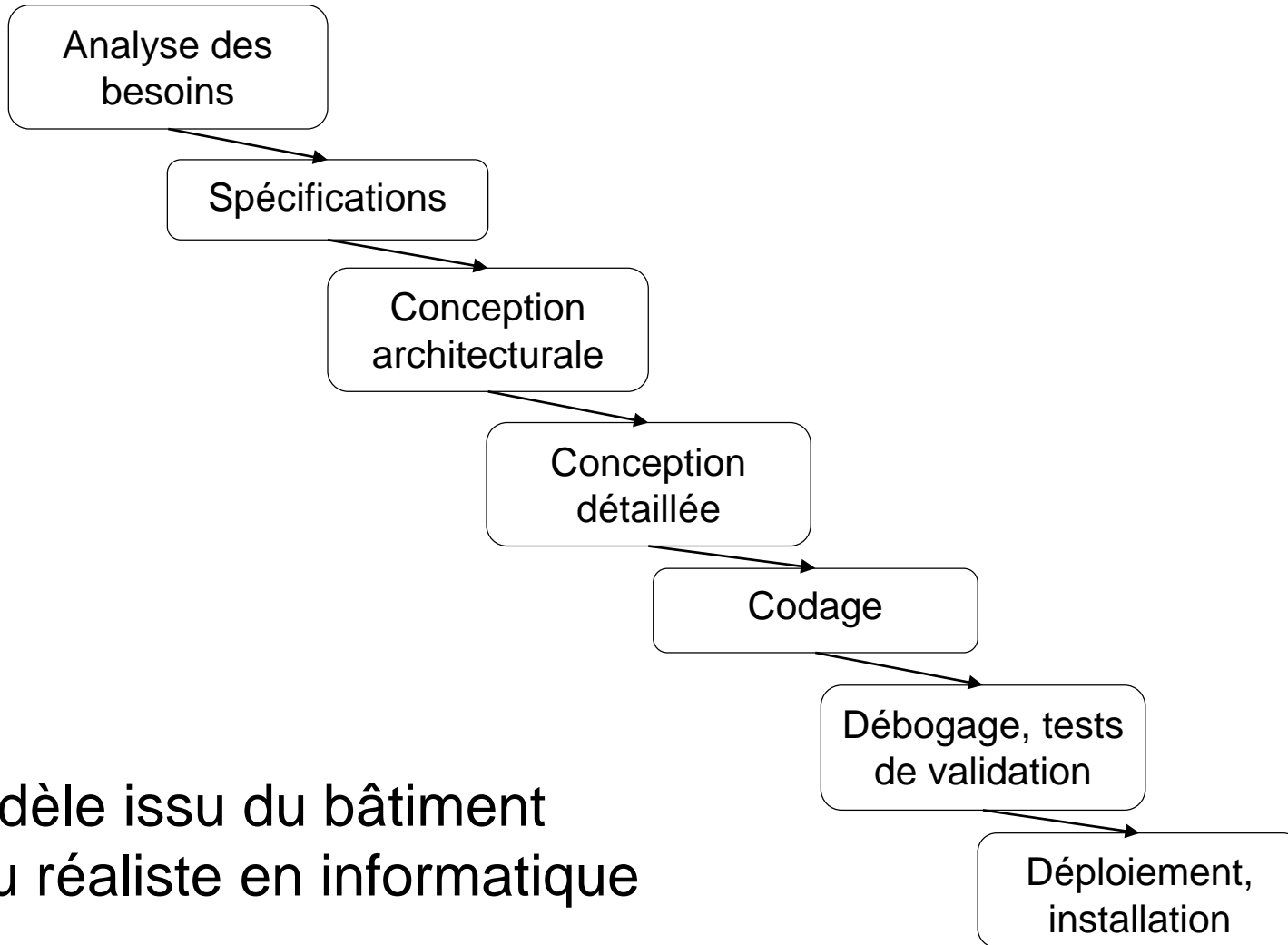
Cycle de développement

- Cycle en cascade
- Cycle itératif
- Cycle en V
- Cycle en spirale
- Méthode AGILE
- Extreme programming, Scrum et autres

Cycle en cascade

- Analyse des besoins (quoi), spécification
- Conception architecturale, de haut niveau (comment)
- Conception détaillée (comment en détail)
- Codage
- Débogage (proc. d'assurance qualité)
- Déploiement
- Tests/Utilisation

Cycle en cascade

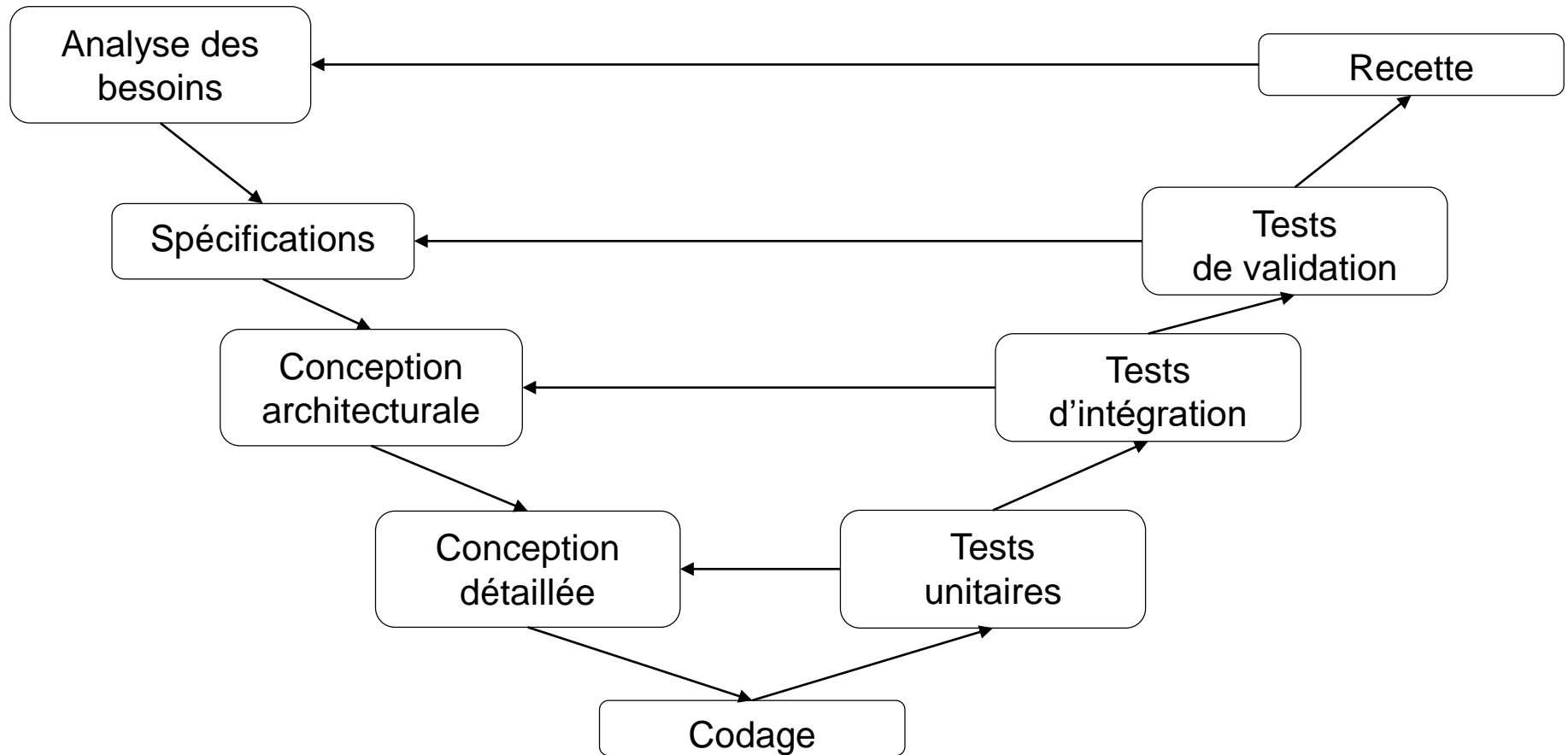


Modèle issu du bâtiment
Peu réaliste en informatique

Cycle itératif

1. Analyse des besoins (quoi), spécification
 2. Conception architecturale, de haut niveau (comment)
 3. Conception détaillée (comment en détail)
 4. Codage
 5. Débogage (proc. d'assurance qualité)
 6. Déploiement
 7. Tests/Utilisation → Version 1.0
- on recommence en 1 pour la version 2.0, etc.

Cycle en V



Cascade, itératif ou en V

Problème : globalement il n'y a que la version finale qui tourne

Cycle en spirale

- Cycle => version beta 1 (v0.1)
- Cycle => version beta 2 (v0.2)
- ...
- Cycle => version 1.0



Par l'implémentation de versions successives, le cycle recommence en proposant un produit de plus en plus complet et robuste

La méthode que nous allons utiliser en l'appliquant à l'informatique → AGILE

Méthode *AGILE*

- Suis un cycle en spirale
 - Méthode de développement informatique
 - Impliquant au maximum le demandeur (client)
-
- permet une grande réactivité à ses demandes
 - plus pragmatiques que les méthodes traditionnelles
 - visent la satisfaction réelle du besoin du client, et non d'un contrat établi préalablement

Méthode *AGILE*

- Valeurs
 - **L'équipe**
 - La communication est une notion fondamentale dans l'équipe
 - + Une équipe de développeurs moyens qui communique
 - Une équipe de très bons développeurs sans communication
 - **L'application**
 - Il est vital que l'application fonctionne
 - Documenter le code et mettre à jour la documentation existante
 - **La collaboration**
 - Le client doit collaborer avec l'équipe et fournir un feed-back continu
 - **L'acceptation du changement**
 - La planification et la structure du logiciel doivent être flexibles
 - Permettre l'évolution de la demande du client tout au long du projet

Méthode *AGILE*

Principes (une douzaine)

- Satisfaire le client en livrant tôt et régulièrement des logiciels utiles
- Le changement est bienvenu
- Les artistes et les développeurs doivent collaborer quotidiennement au projet
- Bâissez le projet autour de personnes motivées
- La méthode la plus efficace pour transmettre l'information est une conversation en face à face
- Un logiciel fonctionnel est la meilleure unité de mesure de la progression du projet
- Les processus agiles promeuvent un rythme de développement soutenable
- La simplicité - l'art de maximiser la quantité de travail à ne pas faire - est essentielle
- Les meilleures architectures, spécifications et conceptions sont issues d'équipes qui s'auto-organisent
- À intervalle régulier, l'équipe réfléchit aux moyens de devenir plus efficace
- ...

Premier bilan

- Que le cycle soit long ou court, ces méthodes se basent toutes sur des documents formalisés
 - Même si, avec les méthodes « agiles », la forme du document a moins d'importance que le bon fonctionnement du logiciel
 - L'oral ne suffit pas ! Exemple: changement de développeurs en cours de développement
- Une équipe de développement produit
 - Le cahier des charges avec des diagrammes de classes
 - Le code géré par une base de données de code (ex. git)
 - La documentation du code (ex. doxygen)

Vous ferez pareil!

Cahier des charges

- **Un cahier des charges**
 - un document visant à définir exhaustivement les spécifications de base d'un produit (même pour une version beta) ou d'un service à réaliser
 - il indique les modalités d'exécution
 - et les objectifs à atteindre et vise à bien cadrer une mission
 - le prestataire doit anticiper ou freiner les exigences du client (non-informaticien) le cas échéant

Le cahier des charges est un document contractuel, il fait office de contrat entre le client et le prestataire de service

Cahier des charges – Plan possible

- **Chapitre 1 - Présentation du projet**
 - Contexte, client, historique, etc.
- **Chapitre 2 - Description de la demande**
 - Définir les résultats que le projet doit atteindre
 - Définir les fonctionnalités du produit
- **Chapitre 3 - Contraintes**
 - Coût, durée de développement, budget, etc.
- **Chapitre 4 - Déroulement du projet**
 - Définir les grandes étapes du projet et les découper en tâches
 - Pour chaque tâche on doit
 - Définir les ressources nécessaires: humaines, matérielles, etc.
 - Expliquer ce qu'elle doit faire et comment
 - Définir un « livrable » sous une forme concrète, ex. code, document ou test
 - Planification et synchronisation des tâches dans le temps
- **Annexes**

Cahier des charges : exemple

- **Chapitre 1 – Présentation du projet**
 - Qui : SNCF
 - Contexte : S'occupe des trains en France depuis toujours, société publique, etc.
- **Chapitre 2 - Description de la demande**
 - Résultats visés : construire un train rapide entre Lyon et Milan pour tous et tout le temps
 - Fonctionnalités
 - Durée < 2h, donc vitesse > 300 km/h
 - Un train toutes les heures
 - Coût du billet < 50 euros par voyage
 - 2 classes de wagon
- **Chapitre 3 - Contraintes**
 - Coût : budget < 10M euros sur fond propre + 50M euros de l'état
 - Durée de développement : prêt dans 2 ans pour les JO de Milan

Cahier des charges : exemple

- **Chapitre 4 - Déroulement du projet**

- Liste des tâches

- Tâche 1

- Construction des rails
 - Livrable => les rails
 - Réalisée quand les N rails seront dans l'entrepôt X

- Tâche 2

- Pose des rails
 - Livrable => 600 km de rails entre Lyon et Milan
 - Réalisée quand rails posés en continu entre Lyon et Milan

- Tâche 3

- Conception de la locomotive
 - Livrable => plan de la locomotive
 - Réalisée quand le plan de la locomotive respecte les contraintes

...

Cahier des charges : exemple

- Tâche 4 : Construction de la locomotive, Livrable => 1 locomotive
 - Tâche 5 : Conception des wagons, Livrable => 2 plans de wagon
 - Tâche 6 : Construction des wagons ...
 - Tâche 7 :
 - Test de la locomotive à grande vitesse sur rail
 - Réalisée quand la locomotive aura une moyenne de 300 km/h
 - ... et cetera jusqu'à par exemple tâche 129
- Les tâches dans le temps => Diagramme de Gantt
- Planifications des tâches et des livrables : estimation du temps que prend chaque tâche + dépendance entre les tâches
 - Peut inclure des « milestones », point d'achèvement (qui ne donne pas forcément lieu à un livrable)

Cahier des charges : exemple

- Diagramme de Gantt

Tâche / Mois	1	2	3	4	5	6	7	8	9	10	11		24
1 : Construction des rails													
2 : Pose des rails													
3 : Conception locomotive													
4 : Construction locomotive													
5 : Conception wagons													
6 : Construction wagons													
7 : Test locomotive													
129 : Ouverture ligne													

Cahier des charges : exemple

- **Annexe : données financières, budget prévisionnel, subvention**

- **Projet sur 48 mois**

- **SNCF**

• Personnels présents + environnements	2 014 039,80 €
• Ingénieurs	2 056 500,00 €
	dont salaires 1 042 500,00 €
• Fonctionnement (hors personnel)	230 342,00 €
• Coût complet	5 000 515,48 €
• Aide demandé	1 072 475,68 €

- **Alstom**

• 10 techniciens, 10 ingénieurs	3 064 500,00 €
• Fonctionnement	3 010 096,00 €
• Equipement	9 060 876,00 €
• Coût complet	5 001 430,04 €
• Aide demandé à l'état	2 030 874,02 €

- **Coût complet** 100 001 945,52 €

- **Aide demandé** 4 003 349,70 €