

# Méthodes pour l'informatisation - compléments

---

## Méthodes de développement des Systèmes d'Information

Christine Plumejeaud

Doctorante Informatique UJF

**CNAM - centre de Grenoble**

# Plan

---

1. Rationaliser le développement d'un SI ?
  - Définition d'un Système d'Information (SI)
  - Constats et remèdes
2. Quelques méthodes de développement
  - Historique
  - Cycles de développement
3. Un choix : les méthodes agiles
  - RUP : *Rational Unified Process*
  - XP : *eXtrem Programming*

# Définition d'un système d'information (SI)

---

## 1. Qu'est-ce qu'un **Système d'Information** ?

«Un ensemble d'éléments reliés par un ensemble de relations »

*J.Lesourne 1976*

Un ensemble formé de : (*C. Rolland, 1988*)

- Une collection de **données** donnant une représentation partielle, arbitraire d'aspects pertinents de la réalité
- Une collection de **règles** qui traduisent le fonctionnement de l'organisation
- Un ensemble de **procédés** pour l'acquisition, la mémorisation, la restitution, la recherche, et la communication des renseignements
- Un ensemble de **ressources humaines** et de **moyens techniques**

# Définition d'un système d'information (SI)

---

## **Un Système d'Information**

un système organisé de ressources, de personnes et de structures qui évolue dans une organisation et dont le comportement coordonné vise à atteindre un but commun.

## **Exemple**

Gestion des réparations dans un garage automobile,  
Gestion de la scolarité,  
Gestion des vols dans une compagnie aérienne,  
Etc.

# Définition d'un système d'information (SI)

---

Les SI servent à :

- Stocker et restaurer l'information
- Faire des calculs
- Aider à communiquer
- Ordonnancer et contrôler des tâches
- Etc.

# Différents types de SI

---

- Systèmes opérationnels (TPS): assistent et/ou contrôlent l'exécution des opérations de gestion. Usine de voitures.
- Systèmes de pilotage d'aide à la décision (DSS) : aident les décideurs (exemple : boursiers) dans leur décisions.
- Système d'information de gestion (MIS) : mise en place d'indicateurs permettant de suivre en permanence les principaux résultats des secteurs d'activité d'une entreprise.
- Systèmes Experts : capitalise l'expertise de professionnels au service de novices (SIG pour les risques naturels)
- Systèmes collaboratifs : système qui favorise le travail en groupe, comme les système de gestion de configuration (CVS)
- Systèmes bureautiques : offrent un ensemble de fonctionnalités bureautique (traitement de texte, impression, courrier, etc).

# Développer un SI

---

Le développement d'un SI est :

Un processus de *changement*, dirigé par un *groupe* de développeurs qui opère en prenant en compte un système, dans un *environnement* et ce pour atteindre des *objectifs* fixés par les acteurs.

# Processus de changement

---

## Ce processus de changement

- agit sur la structure
- est dirigé par des objectifs
- renferme une dimension sociale
- souvent incertain
- nécessite des moyens
- est spécifique au problème



# Le constat

## 1995 - la crise du logiciel

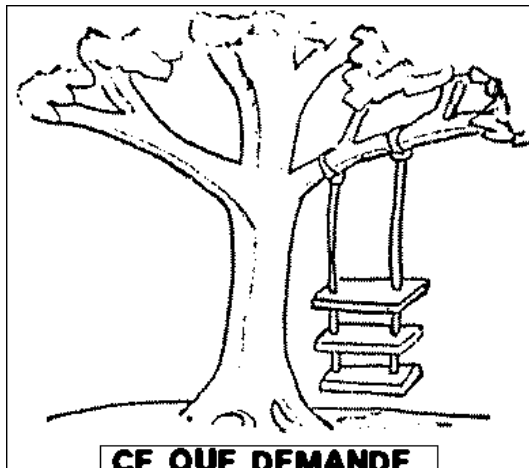
Statut des logiciels	Proportion
Abandonnés ou jamais utilisés	31%
Coûts largement dépassés	53%
Réalisés suivant les plans	16%

## Répartition de l'activité

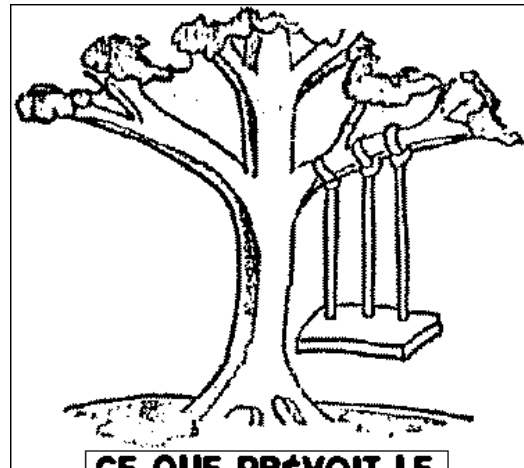
Etapes	Coût
Développement initial	40%
Maintenance	60%

Analyse et conception	40%
Mise en oeuvre	20%
Tests et validation	40%

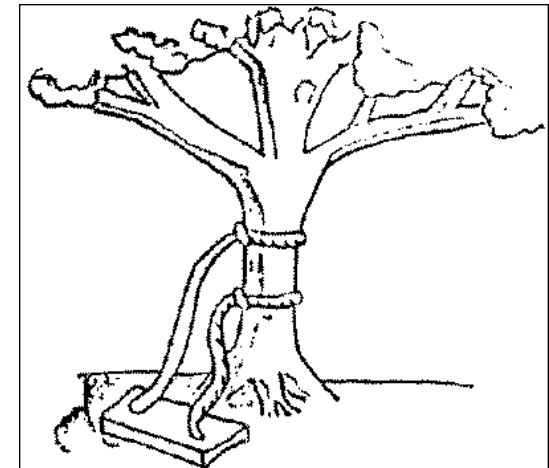
# Petite illustration



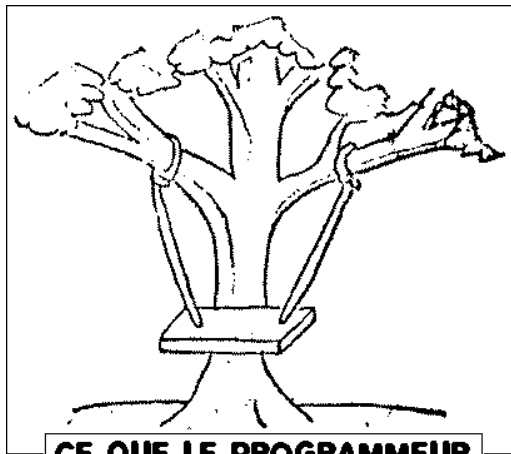
**CE QUE DEMANDE  
LE CLIENT**



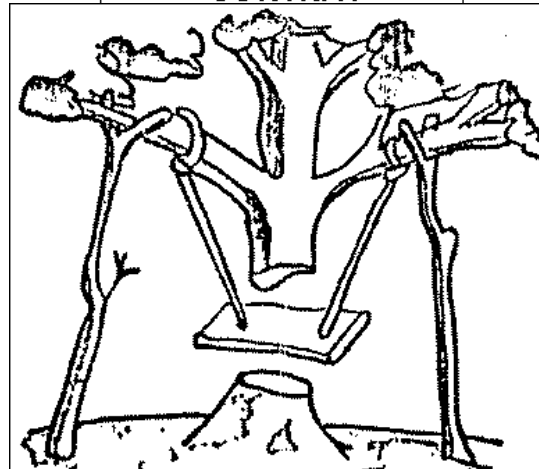
**CE QUE PRÉVOIT LE  
CONTRAT**



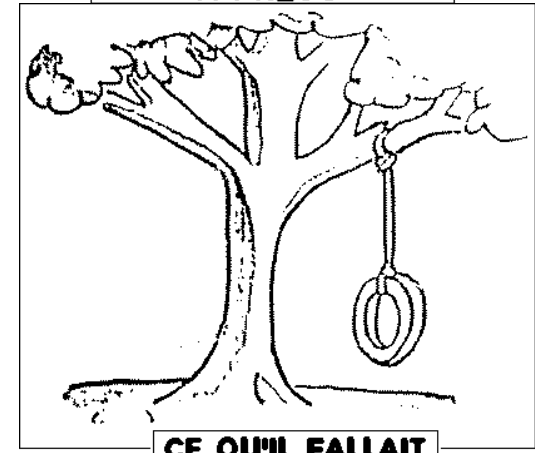
**CE QUE L'ANALYSTE  
A PRÉVU**



**CE QUE LE PROGRAMMEUR  
A ÉCRIT**



**CE QUE LA MISE AU  
POINT A FAIT**



**CE QU'IL FALLAIT**

# Rationaliser

L'art de produire de bons logiciels au meilleur rapport qualité prix.

---

## Critères de qualités

- Externes :
  - Fiable : conforme aux spécifications
  - Robuste : ne plante pas
  - Efficace : espace mémoire, temps d'exécution
  - Convivial : facile et agréable à utiliser
  - Documenté : manuel utilisateur, tutoriel, cours
- Internes :
  - Documenté : documentation de conception
  - Lisible : l'écriture respecte les conventions de programmation
  - Portable : autre machine, autre système d'exploitation (durée vie allongée)
  - Extensible : possibilité d'ajouter d'autres fonctionnalités
  - Réutilisable : certaines parties peuvent être récupérées

# Développement et environnement

---

L'environnement = { tout ce qui est extérieur au système et au groupe de développement }

Il regroupe un ensemble de conditions et de facteurs qui ont un *impact* sur le processus de développement. Il se compose d'une multitude de sous environnements :

- des facteurs économiques
- un environnement technique (outils, méthodes, langage)
- légal (législation)
- humain (acteurs : commercial, chef, développeur, client, consultant, ...)

# Développement et développeurs

---

Les développeurs :

## 1. Une organisation formelle

- Rôles : architecte, testeur, développeur, commercial
- Assignation de tâches/responsabilités
- Niveaux d'autorité

## 2. Une organisation informelle

- Rapport de force
- Relations personnelles
- Niveau d'engagement/d'implication personnel

# Développement et objectifs

---

## Les objectifs

- implicites/explicites
- imposés/négociés
- contradictoires/complémentaires
- conformes à la législation
- réalistes/trop ambitieux

# Développement et acteurs

---

## Les acteurs

- **Internes** : utilisateurs, dirigeants, unités organisationnelles
- **Externes** : clients, instances gouvernementales, associations, éditeurs de logiciels, détenteurs de technologies, etc.
  - sont guidés par des intérêts et des objectifs
  - peuvent être impliqués dans le développement en tant que membres
  - peuvent avoir des droits sur le système et ses propriétés

**Le développement est la combinaison de facteurs technologiques, sociaux, psychologiques et culturels**

# Comment concevoir un SI ?

---

Concevoir un SI c'est :

- Résoudre un problème
- Décomposer le problème
- Le modéliser

La modélisation est une activité guidée par *le bon sens*. Modéliser :

1. Abstraire
2. Organiser
3. Formaliser



BESOIN DE MODÈLES  
ET DE MÉTHODES



# Les modèles de développement

---

## 1. Le modèle en cascade

Succession de phases distinctes : spécification, conception, réalisation, test, etc.

## 2. Le développement évolutif

Les phases s'enchevêtrent

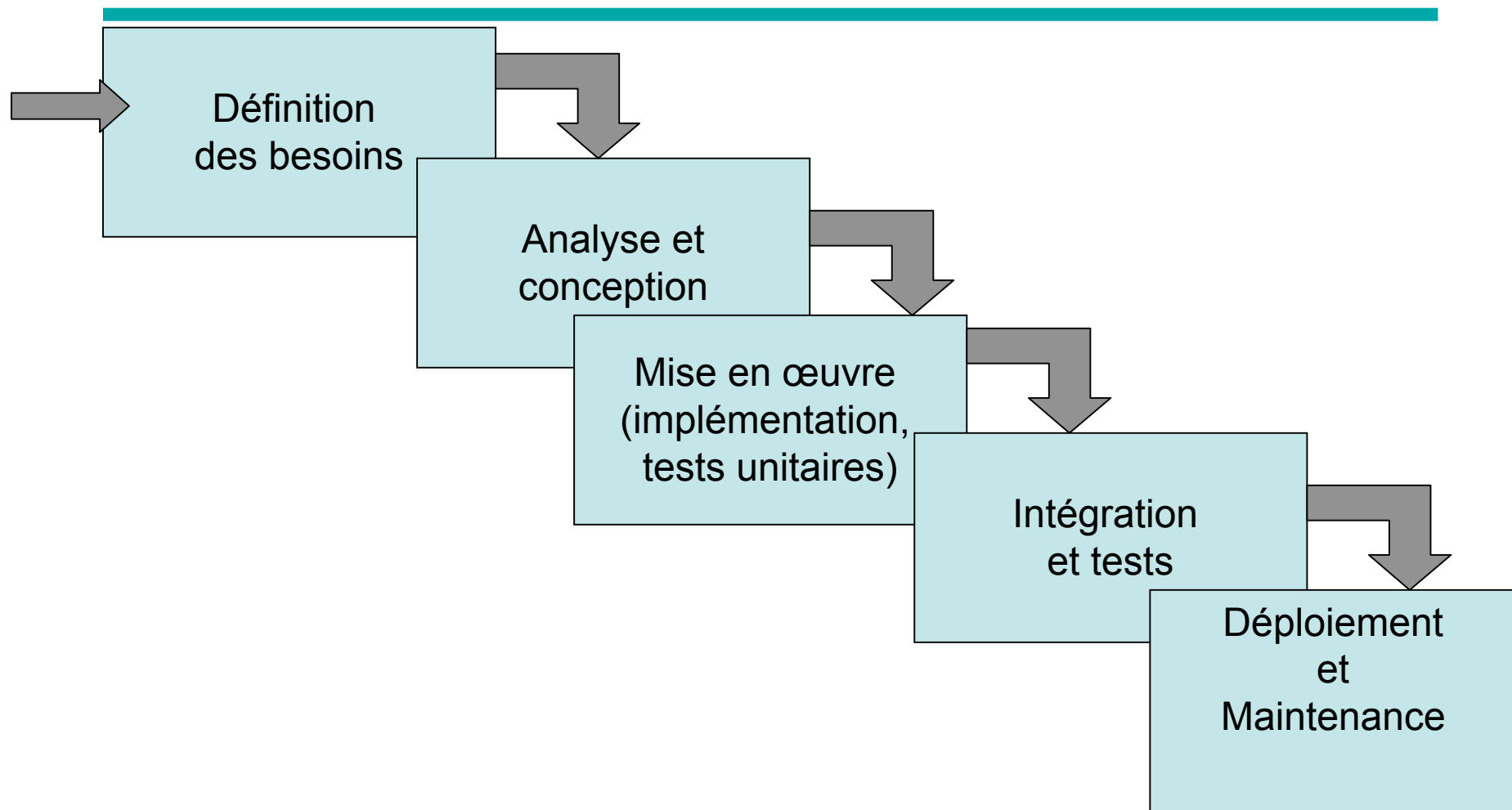
## 3. Le développement formel des systèmes

Un modèle mathématique formel modélise le système, puis il est ensuite transformé en code

## 4. Développement par la réutilisation

Développement par l'assemblage de composants réutilisables

# Le modèle en cascade



## Inconvénients du modèle en cascade

---

1. Processus séquentiel et peu réactif
2. Rigidité du découpage du projet en phases
3. Manque de réactivité aux changements des besoins des utilisateurs
4. Nécessite la connaissance et la compréhension des besoins très tôt dans le processus de développement

# Le développement évolutif

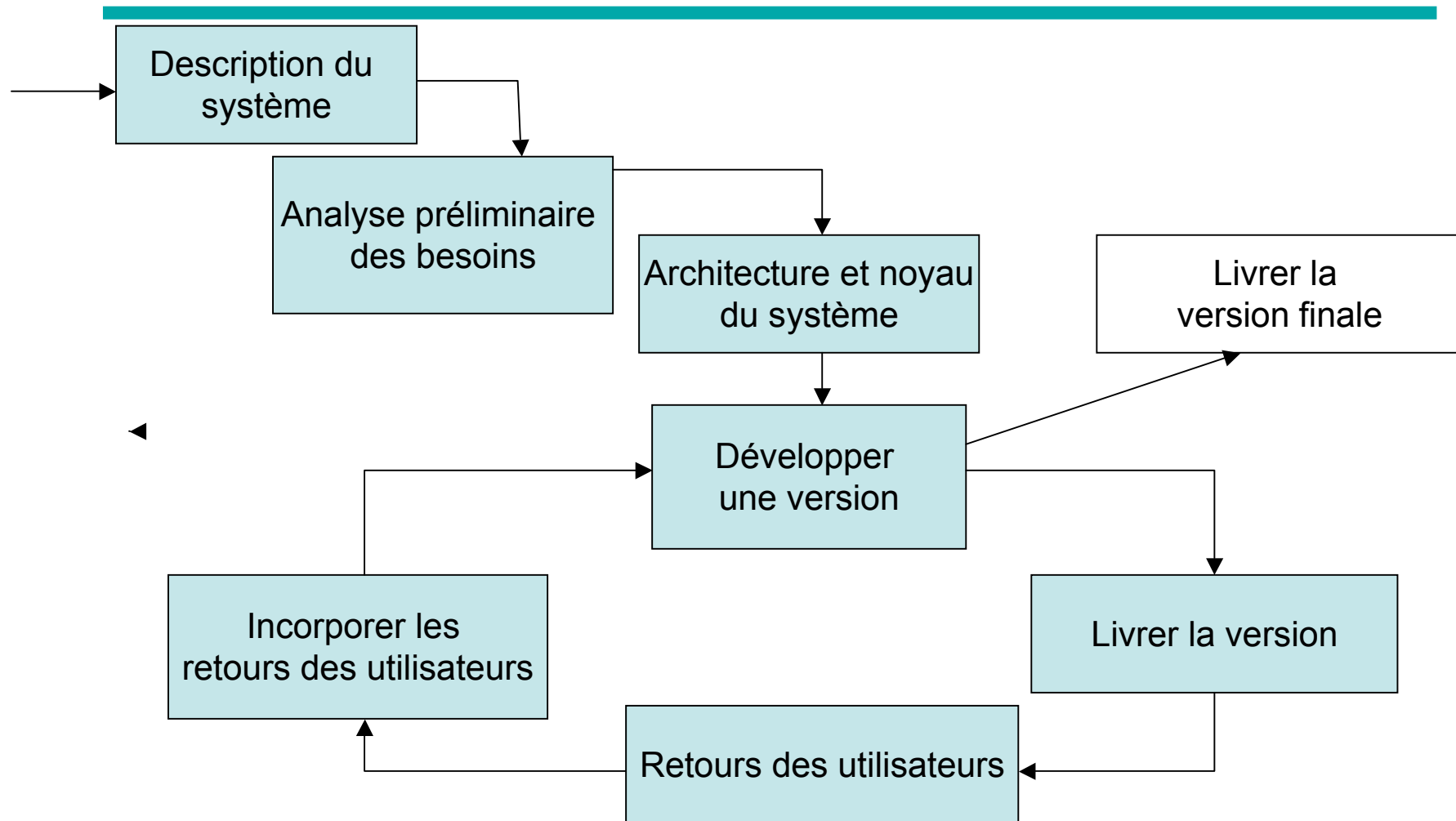
---

1. Le système est livré en versions successives
2. Le changement des besoins est pris en compte tout au long du processus de développement

## Avantages :

- Révéler les problèmes de façon précoce
- Obtenir rapidement un produit montrable au client
- Le client peut effectuer des retours sur la version livrée (et il précise ses besoins)
- Avoir plus de points de mesure pour apprécier l'avancement du projet

# Développement évolutif



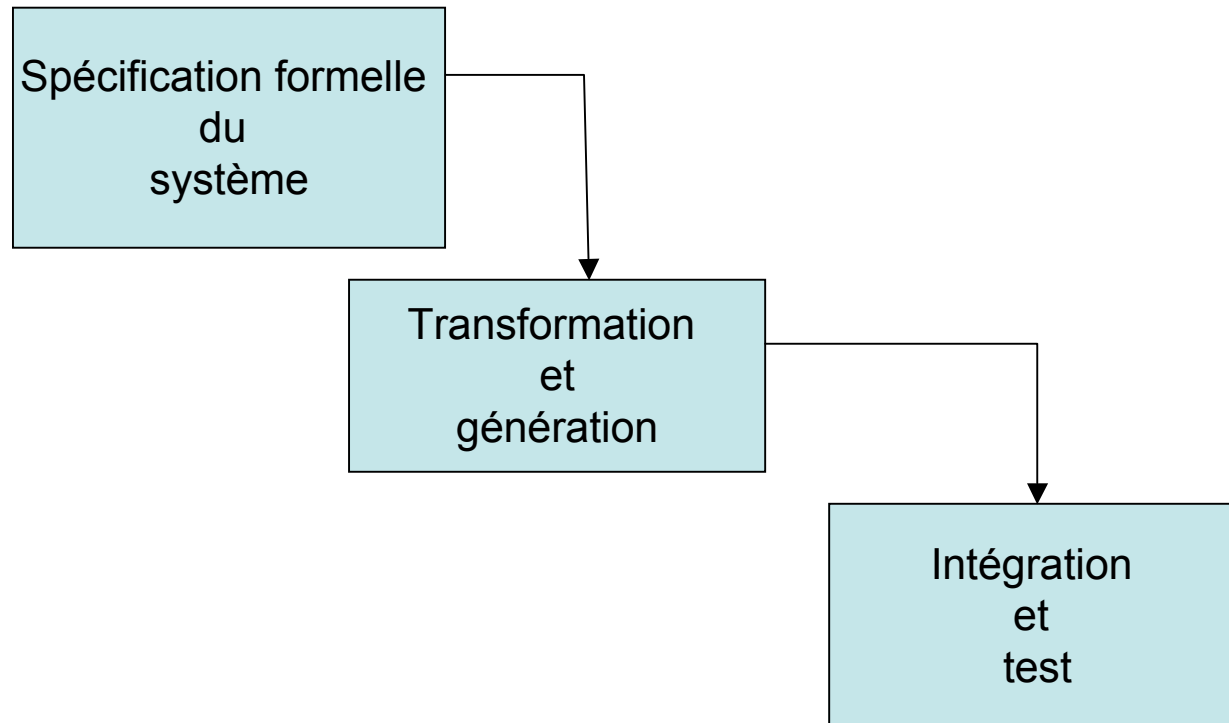
# Développement formel des systèmes

---

1. Fondé sur la transformation de spécifications mathématiques en code
2. Les transformations préservent la justesse des spécifications et assurent ainsi que le code généré est conforme aux spécifications
  - Utilisé dans le développement de *systems critiques* (métro 14 à Paris)
  - Cette approche nécessite des compétences dans la modélisation mathématique

# Développement formel des systèmes

---



# Développement par la réutilisation

---

1. Fondé sur la réutilisation de composants de code : COTS (*Commercial-Off-The -Shelf*)  
*Design **by** reuse*
2. Cette approche aspire à réduire les coûts et les délais en limitant les nouveaux développements
3. La génération de nouveaux composants nécessite l'intégration du besoin des développements futurs, dans les processus du développement (*design **for** reuse*)



# Processus itératifs

---

Consiste à itérer un ensemble de phases génériques

Deux approches sous-jacentes :

- Le développement incrémental
- Le développement en spirale

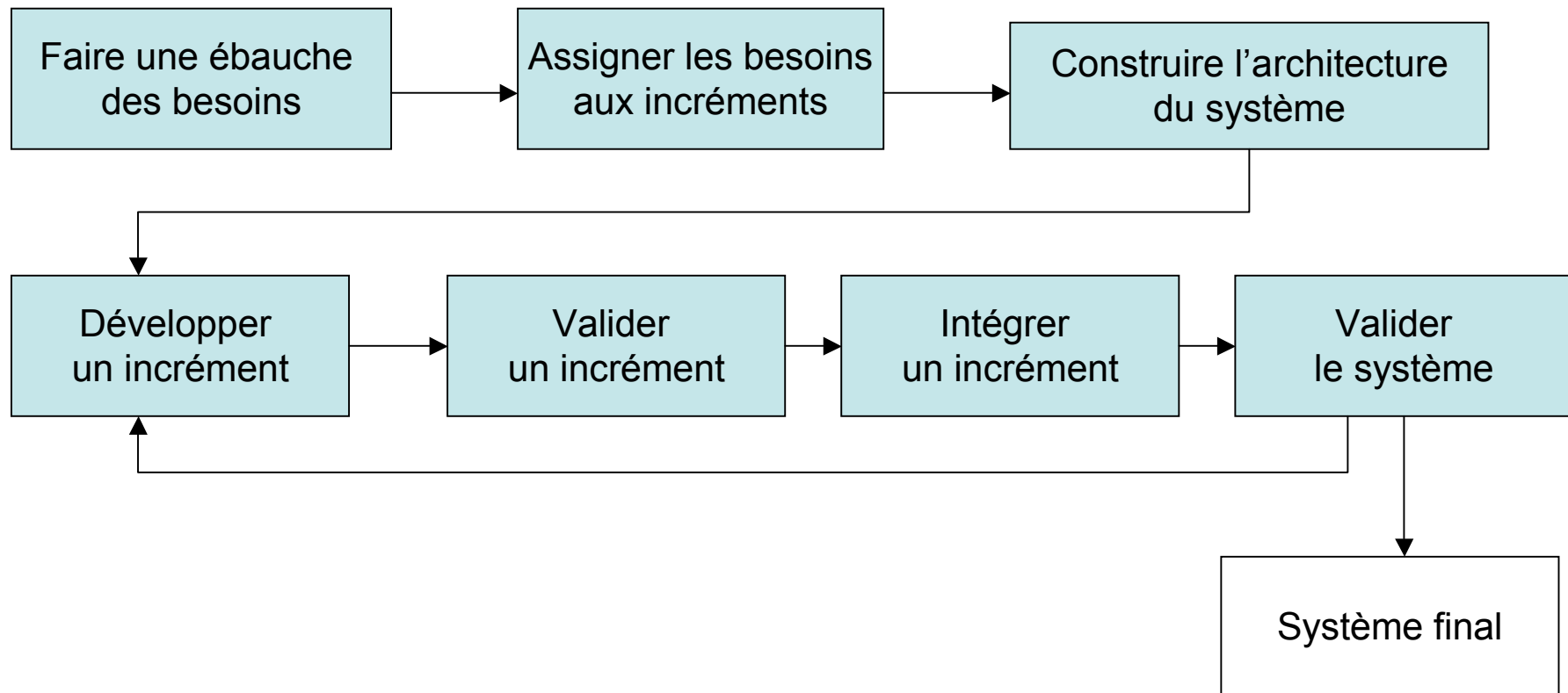
# Le développement incrémental

---

1. Le système n'est pas livré d'un bloc, mais sous formes d'*incréments* couvrant chacun une partie des fonctionnalités du système
2. Nécessite la définition de *priorités* sur l'ensemble des besoins. Les plus prioritaires sont l'objet des premiers incréments.
3. Il est indispensable de *disposer d'une vue d'ensemble* (architecture) du système afin de suivre l'évolution et l'intégration des incréments

# Le développement incrémental

---



## Avantages du développement incrémental

---

1. Disponibilité rapide de parties du système à livrer
2. Les prototypes des premiers incréments aident à découvrir les besoins pour les suivants
3. Minimisation du risque du projet
4. Les parties les plus prioritaires profitent de plus de test (elles sont revalidées à chaque nouvel incrément)

# Les méthode agiles

---

Elles correspondent à un cycle de vie itératif

Hypothèse : les *changements* sont inévitables en cours de vie du projet (besoins des utilisateurs, évolution de l'architecture, de la technologie, etc.) et doivent être pris en compte au plus tôt.

- Privilégier la livraison de fonctionnalités utiles
- Abandonner la production de documentation intermédiaire sans intérêt pour le client.

# Le processus unifié RUP

## *Rational Unified Process*

---

Développement itératif et incrémental basé sur les besoins des utilisateurs, centré sur l'architecture du logiciel.

### 4 phases (séquentielles)

- Etude d'opportunité : marché ? Concurrence ?
- Elaboration : définition de l'architecture après modélisation du domaine
- Construction : livraison par incréments du logiciel
- Transition : installation et formation des utilisateurs

# La méthode XP, *eXtreme Programming*

---

Nouvelle approche de développement *légère* fondée sur des itérations de livraisons courtes visant à produire un logiciel de *qualité* avec un minimum de documentation intermédiaire et règles.

## 4 règles essentielles:

- Communication (entre développeurs et avec les utilisateurs)
- Simplicité (conception et programmation simples et claires)
- Retour (prise en compte des remarques utilisateurs)
- Courage (reconnaissance de la complexité de l'activité de développement)

# Pratiques de la méthode XP

---

- Travail en équipe avec un représentant des utilisateurs
- Planning
- Petites livraisons
- Conception simple et restructuration du code (*code refactoring*)
- Programmation par binômes et propriété collective du code
- Développement piloté par les tests
- Intégration continue
- Métaphore = vision commune de l'ensemble
- Rythme supportable



# Usage des méthodes agiles

---

## 1. Quand les utiliser ?

- Si projet de moins de 10 personnes ET géographiquement proches
- Si les besoins des utilisateurs sont susceptibles d'évoluer très rapidement

## 2. Ne pas utiliser si :

- Le projet occupe plus de 20 développeurs
- Les équipes sont réparties sur des sites géographiques différents
- Pour les applications critiques (qui mettent en jeu des vies humaines : transport, médecine)
- Cultures militaires « *Command-and-control* »