UTC504 – Systèmes d'Information et Bases de Données

Introduction

Sébastien Fourestier

2022

Bibliographie

- Pascal André, Alain Vally : *Conception des systèmes d'information*
- Jacqes Printz : *Le genie logiciel*
- Florence Petit : *Cycle de vie des systèmes informatiques*
- František Kardoš : Conception de systèmes d'information
- Pierre Gérard : UML, Diagrammes de classe
- `www.editions-eni.fr` : *Cycle en spirale*

Correctifs

- Ce cours est disponible sous licence libre sur ce dépôt github : `https://github.com/sfourestier/enseignement`

→ Voici pouvez :

  - L'améliorer en proposant des *Pull requests*
  - Partager autour de points pouvant être améliorés en créant des tickets (*Issues*)

Plan

Introduction

- Petits systèmes autonomes :
  - Développement aisé et court
- Accroissement des problèmes traités et diversité des domaines d'application de l'informatique

→ Systèmes complexes :

- Période de développement et durées de vie allongées
- Le développement est investissement durable

→ Rentabiliser le développement et les efforts produits

Le génie logiciel (1)

Définition du génie logiciel au Journal officiel du 19 février 1984 :

> *« l'ensemble des activités de conception et de mise en œuvre des produits et des procédures tendant à rationaliser la production du logiciel et son suivi »*

Le génie logiciel (2)

Définition adaptée de Jacques Printz :

> *« l'ensemble des moyens techniques, industriels, industriels et humains qu'il faut réunir pour spécifier, construire, distribuer et maintenir des logiciels de qualité »*

Le génie logiciel (3)

Premiers critères de qualité :

- Sûrs :
  - Réagissent de façon déterministe aux sollicitations
- Conviviaux :
  - Adaptés aux capacités des usagers
- Évolutifs :
  - S'adaptent aux nouveaux besoins
- Économiques :
  - Réalisent l'optimum entre le service rendu et les coûts de développement/maintenance

Système d'information (1)

Définition de Wikipédia :

> *« Un système d'information (SI) est un ensemble organisé de ressources qui permet de collecter, stocker, traiter et distribuer de l'information. »*

Système d'information (2)

Définition adaptée de C. Rolland :

- Un Système d'information est un objet artificiel greffé sur un objet naturel (organisation, processus industriel, commande embarquée, etc.)
- Il est conçu pour mémoriser un ensemble d'images de l'objet réel à différents moments de sa vie
- Ces images doivent être accessibles par les partenaires de l'organisation pour décider des actions à entreprendre

Classification des SI

1. SI de gestion :
   - Beaucoup de données : consultation, mise à jour
   - Possibilité d'accès distant (réseau)
   - Ex : gestion de clientèle, du stock, etc.
2. Le calcul scientifique :
   - Beaucoup de calculs, peu de données
   - Critère important : rapidité de traitement
   - Ex : simulation, météo, imagerie
3. L'informatique temps réel :
   - Critère important : réactivité du SI
   - Informatique embarquée, contrôle de processus (fabrication, surveillance), réseaux informatiques
   - Ex : pilotage auto d'un avion, centrale nucléaire

Plan

Modélisation et développement

Pour développer des SI, on utilise des modèles :

- Modèle :
  - Interprétation par son utilisateur de l'idée qu'il se fait d'une situation
- Développement :
  - Le développement est une suite de modèles de plus en plus précise (avec de moins en moins d'éléments laissés libres)

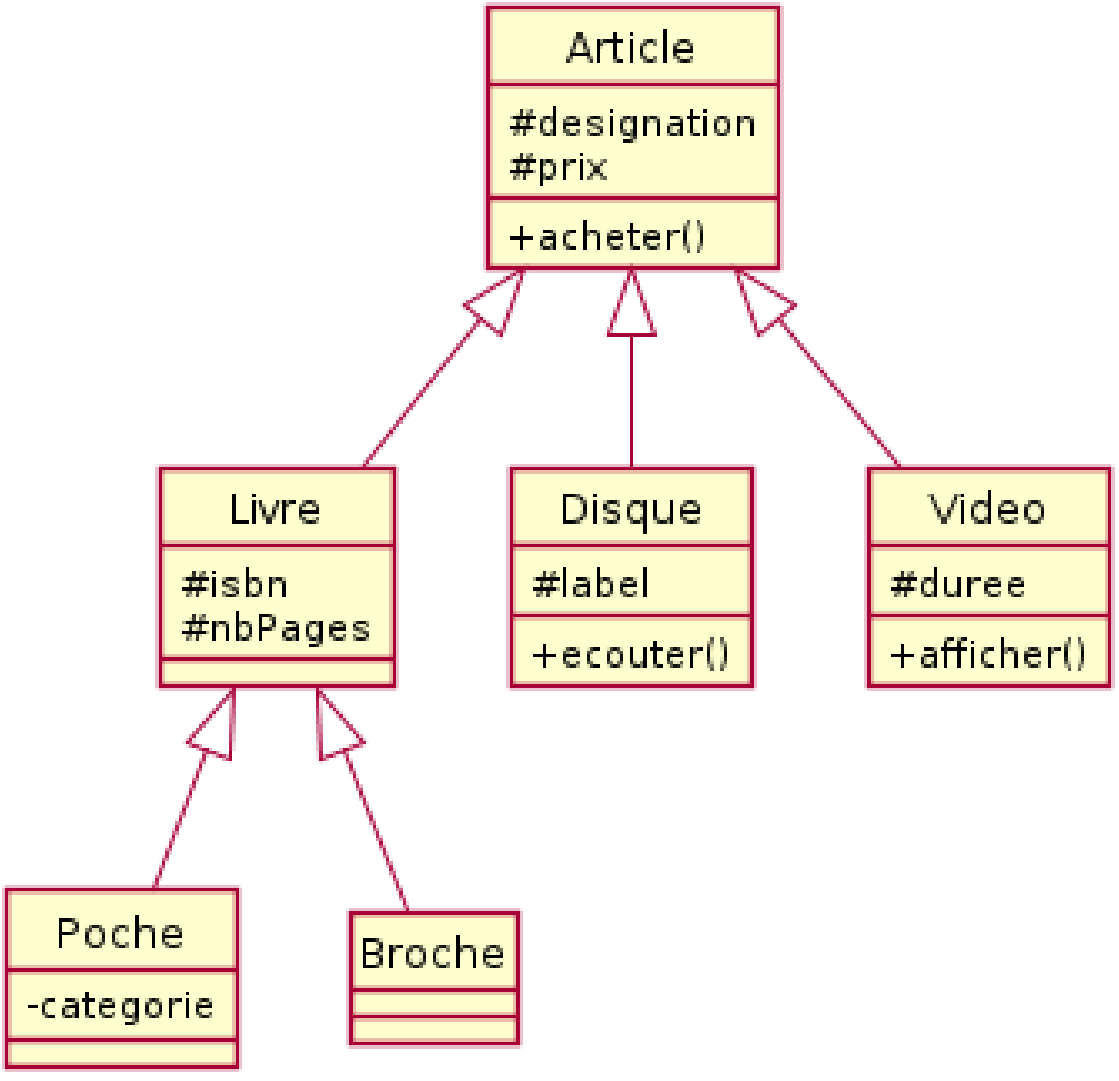
  $\rightarrow$ Lorsque tous les éléments ont été fixés, le SI est implémenté

Méthodes de développement

Plusieurs types de processus de développement :

- Méthodes cartésiennes : « Diviser pour régner »
  - On découpe en sous-éléments, on résout, on rassemble
  - Ex : approche Objet
  - Processus de développement :
    - Basé sur les fonctionnalités
- Méthodes systémiques : vision globale
  - Compréhension des éléments et leurs relations
  - Ex : Bases de données, Merise
  - Processus de développement :
    - Approche conceptuelle, par niveaux d'abstraction

Exemple approche Objet



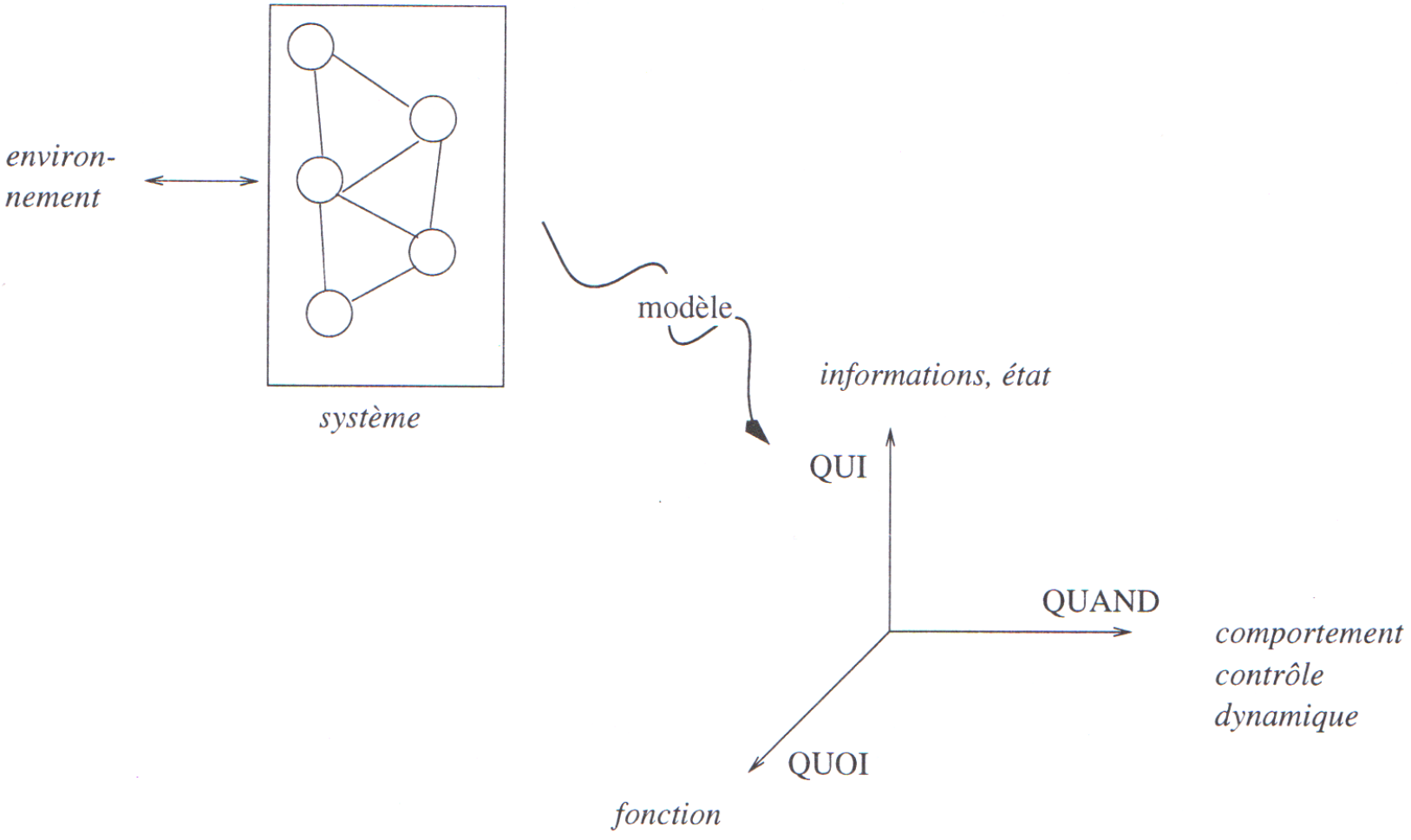
Exemple approche Merise

Trois axes de modélisation (1)

Un système d'information comprend 3 aspects :

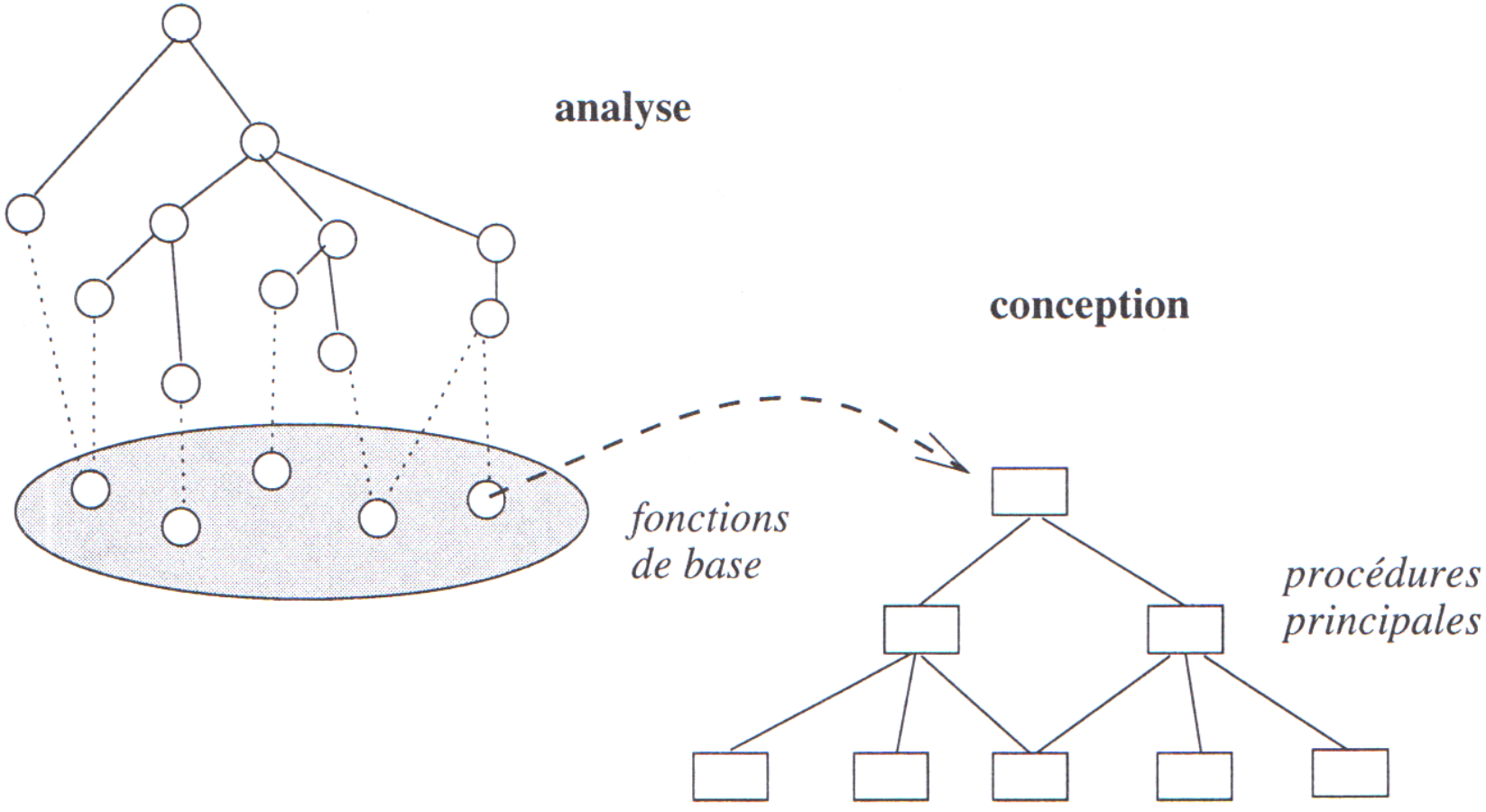
1. Données :
   - Ce que le système manipule
2. Comportement dynamique :
   - Comment s'enchaînent les événements
3. Comportement fonctionnel :
   - Quelles sont ses fonctionnalités

Trois axes de modélisation (2)

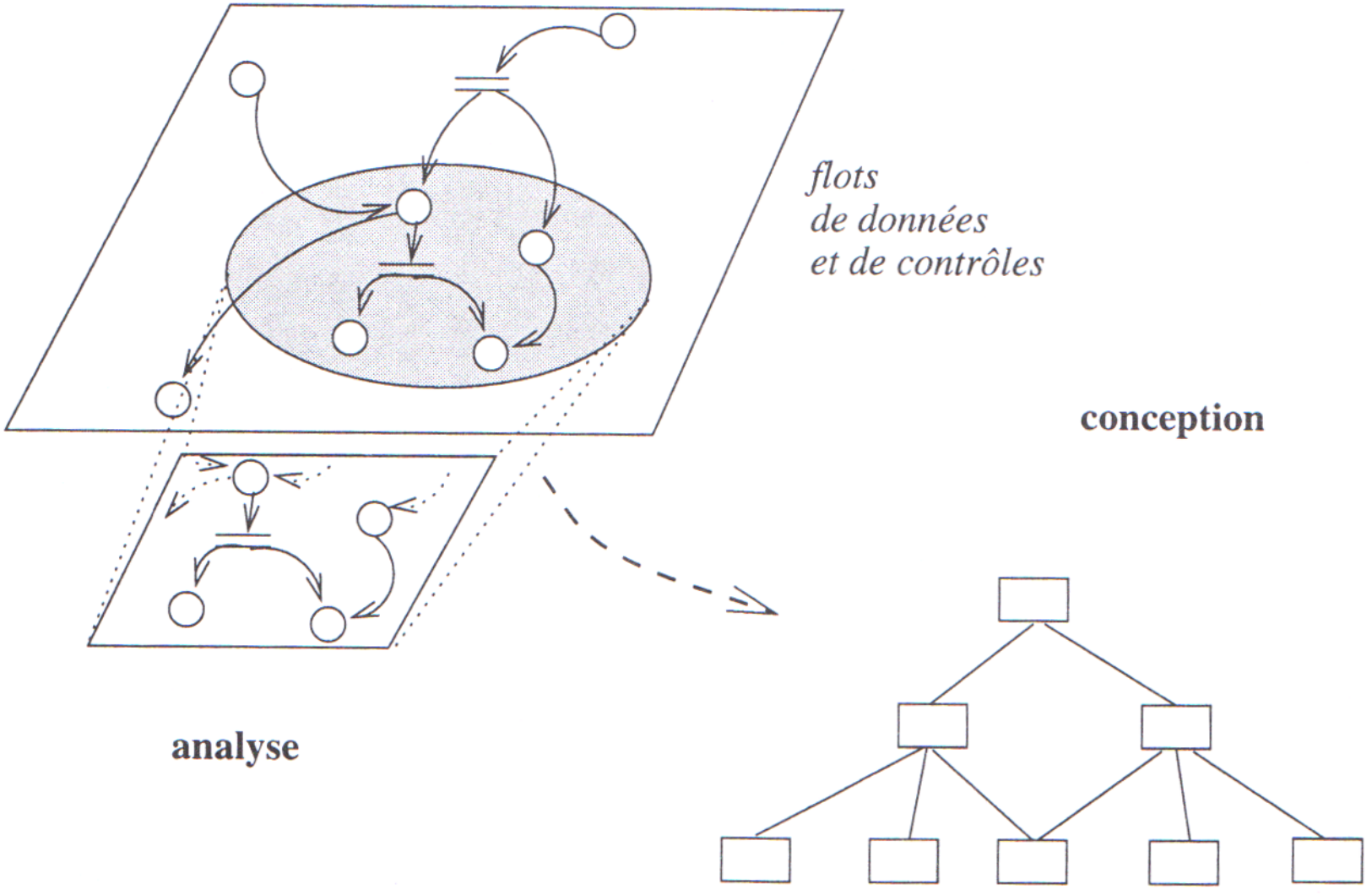


→ Trois courants de méthodes d'analyse/conception

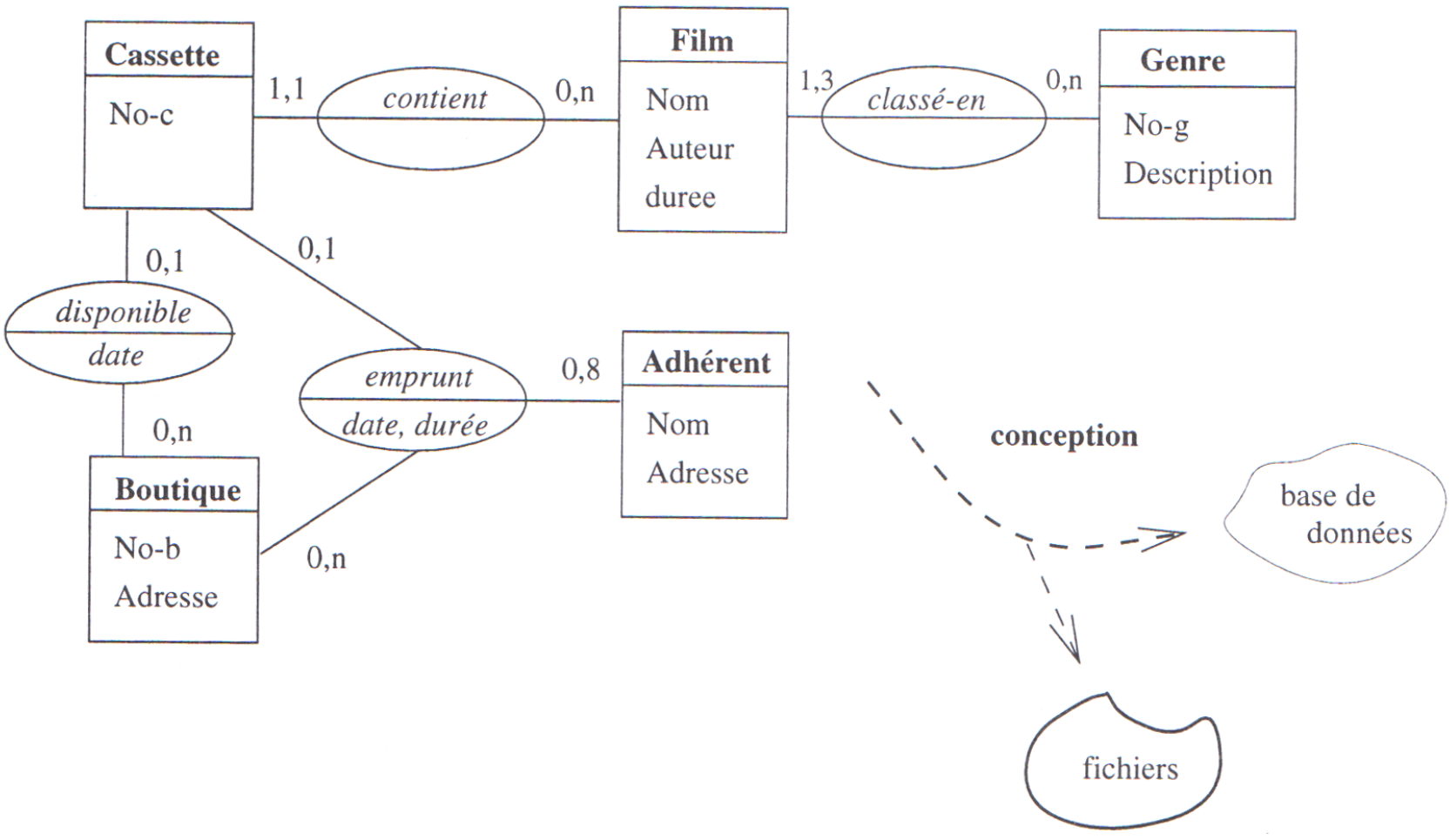
Approche fonctionnelle



Approche flots de données



Approche modèle de données



Plan

Étapes de développement des logiciels

1. Analyse des besoins :
   - Objectif du logiciel
2. Analyse :
   - Expression des besoins
3. Conception :
   - Proposition de solutions
4. Réalisation et Test :
   - Production du programme
5. Installation :
   - Mise en place dans l'organisation
6. Maintenance :
   - Adaptation du logiciel et corrections

Étape 1/6 : Analyse des besoins

Objectif du logiciel :

- Phase initiale du développement
- Description et une évaluation globale des besoins
- Prend en compte :
  - Les aspects économiques
  - Les risques
  - La compétitivité
  - L'organisation globale du projet

Étape 2/6 : Analyse

Expression des besoins :

- Écriture des fonctions que le logiciel doit effectuer et le contexte (conditions d'exploitation, qualité requise, etc.)
- Fait abstraction de la façon dont les fonctionnalités seront réalisée

Étape 3/6 : Conception

- Définir de façon très précise :
  - Les fonctionnalités
  - L'architecture du logiciel
- À partir :
  - Des besoins exprimés
  - Des contraintes générales définies dans les deux première phases
- La spécification peut être :
  - Informelle, en langage naturel
  - À l'aide de diagrammes
  - Sur les 3 axes évoqués précédemment

Étape 4/6 : Réalisation et Test

- Phase de programmation
- Tests qui prouvent la logique du programme
  - Tests unitaires :
    - Validation des parties du logiciel (fonctions, modules)
  - Tests d'intégration :
    - Validation de plusieurs parties du logiciel utilisées ensemble
- Possibilité tests statistiques :
  - Nombre de pannes observées sur une durée donnée, etc.

$\rightarrow$ Les tests sont indispensables pour faciliter la maintenance à long terme

Étape 5/6 : Installation

Mise en place dans l'organisation, s'assurer que :

- La procédure d'installation fonctionne conformément aux exigences
- Les formations sont en place
- Le support technique est opérationnel

Étape 6/6 : Exploitation et maintenance

- Exploitation :
  - Mise à disposition du logiciel auprès de tous les utilisateurs
- Cette phase peut être très longue
- Maintenance :
  - Correction des erreurs détectées

Plan

Cycles de vie

Cycle de vie :

- Organisation de ces étapes

On distingue trois catégories :

1. Les modèles linéaires :
   - Étapes réalisées tour à tour
   - Ex : en cascade, en V
2. Les modèles itératifs :
   - Développement incrémental, évaluation des risques
   - Ex : à spirale, les méthodes agiles
3. Les modèles contractuels :
   - Suite de contrats entre client et fournisseurs
   - Ex : méthodes formelles

Cycle en cascade

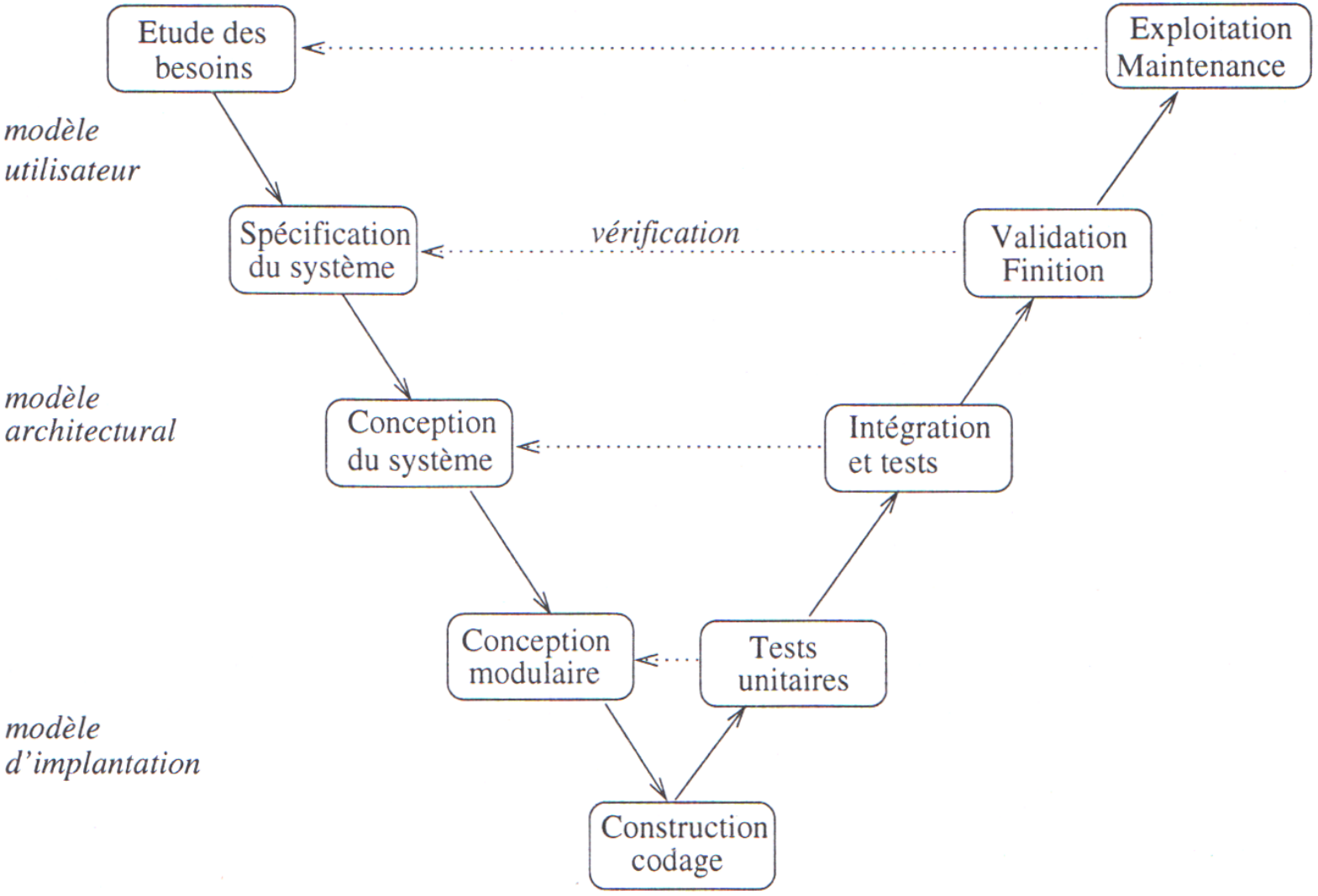
- Les différentes phases sont réalisées tour à tour
- On commence la suivante une fois la précédente achevée
- En cas d'erreur, on revient à la précédente

Cycle de vie en V (1)

On insiste sur :

- Une séparation entre :
  - La construction des diverses spécifications
  - leur validation a posteriori
    (tests unitaires, tests d'intégration, etc.)
- Le niveau d'abstraction :
  - Utilisateur
  - Architecture
  - Implémentation

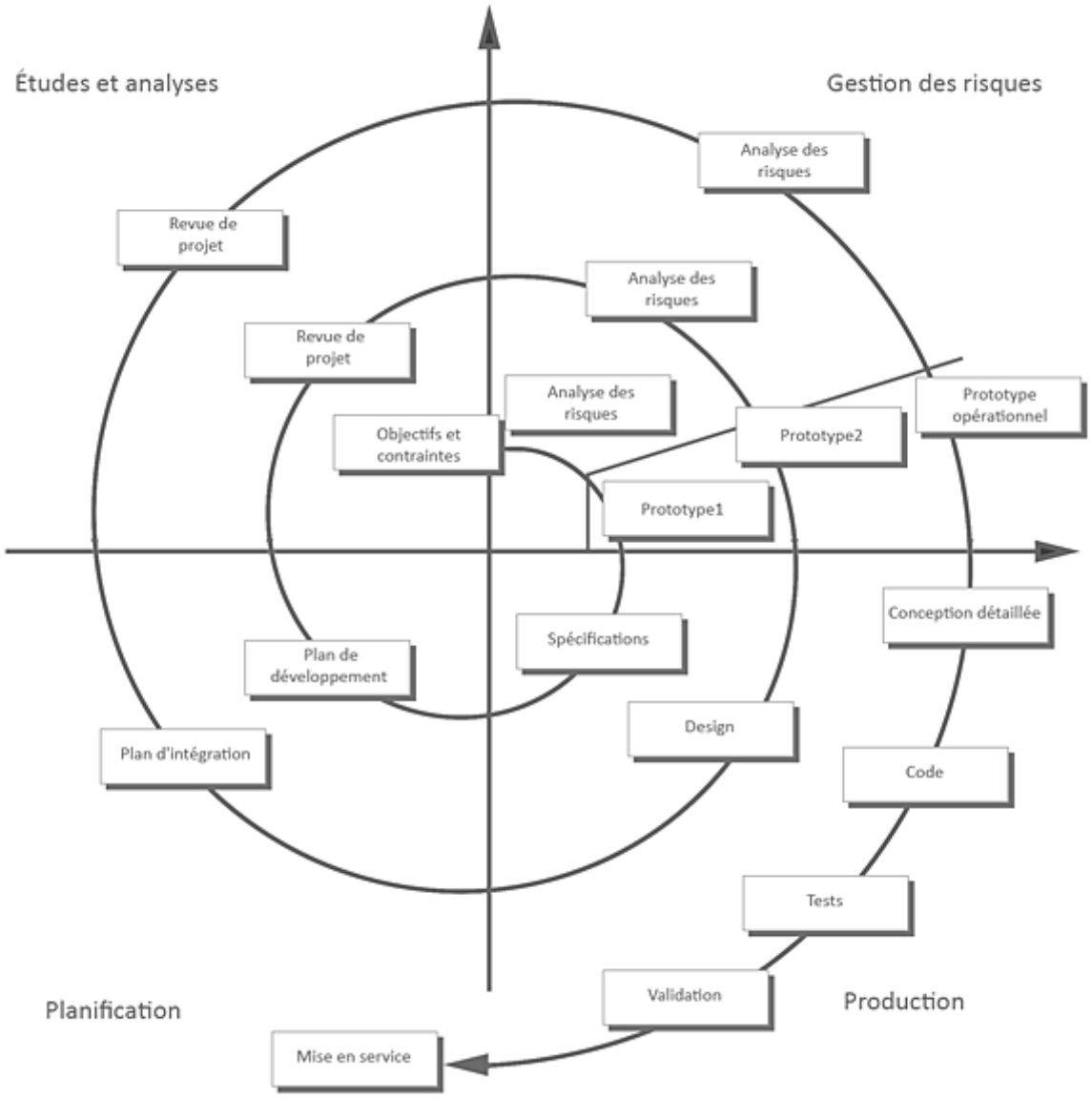
Cycle de vie en V (2)



Modèle en spirale (1)

- 1988 : Boehm
- Prise en compte des risques
  - Actions pour éviter les risques
- 1 cycle :
  1. Analyse
  2. Développement du prototype
  3. Essai du prototype
- Dernier cycle : produit fini

Modèle en spirale (2)



Méthodes agiles

- Cycle de développement court
- Grande réactivité :
  - Acceptation du changement
- Équipe communicante plus importante que les moyens et les outils
- Application plus importante que la documentation
- Collaboration :
  - Client impliqué en feed-back continu

Plan

Qualité du logiciel

Qualités les plus importantes :

- Validité
  - Réaliser exactement les tâches définies par la spécification
- Robustesse
  - Fonctionne même dans des conditions anormales
- Extensibilité
  - Facilité d'adaptation du logiciel aux changements de spécification
- Réutilisabilité
  - Une partie ou le tout peut être réutilisé pour de nouvelles applications
- Compatibilité
  - Le parties peuvent être combinés

Critères informatiques (1)

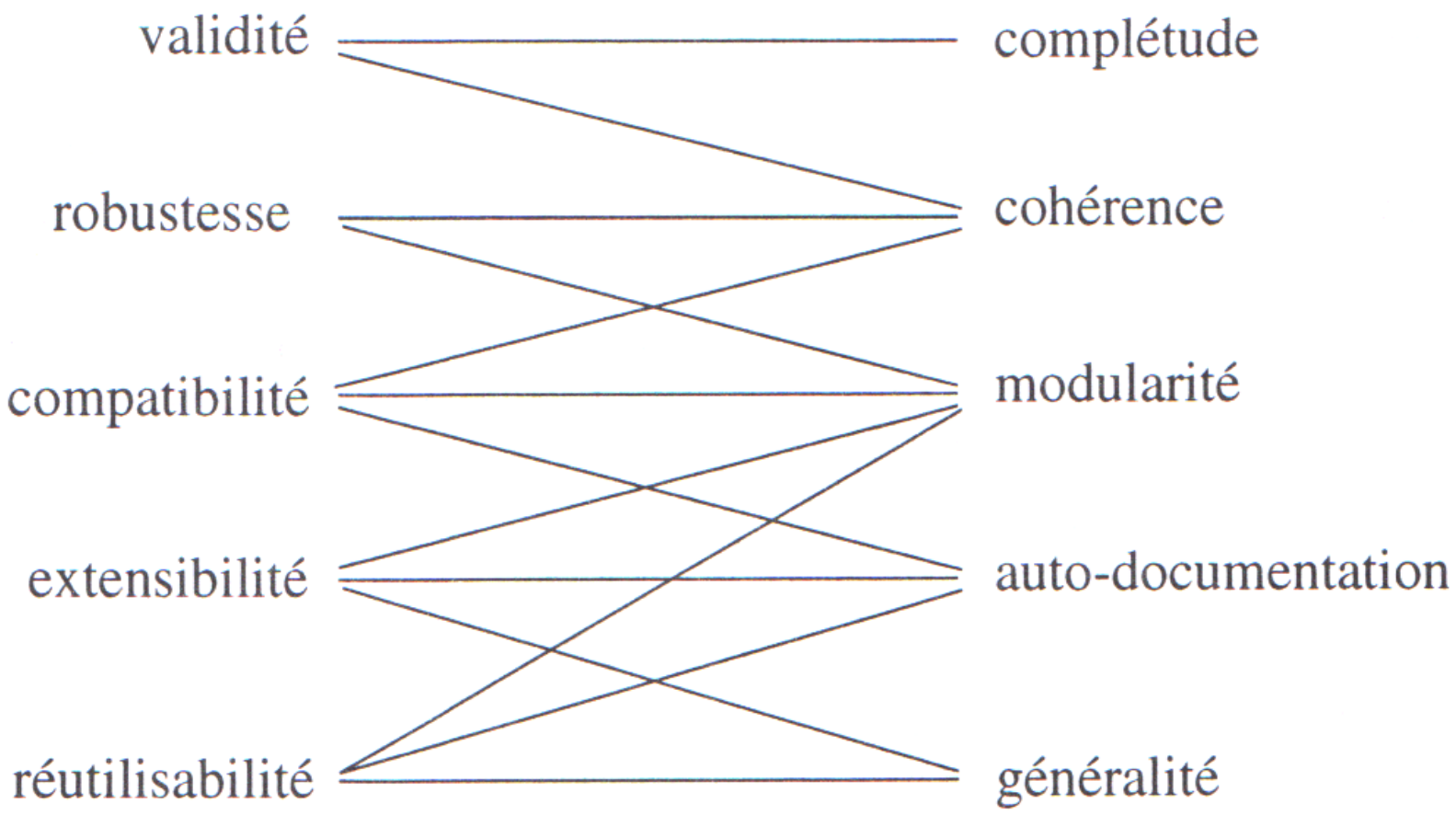
Critères permettant d'atteindre ces qualités :

- Modularité
  - décomposition en composants simples et indépendants
- Complétude
  - Degré d'implémentation des spécifications
- Cohérence
  - Possibilité retour étape de dév. précédente
  - Ex : remonter une erreur détectée en maintenance au niveau de l'implémentation

Critères informatiques (2)

- Généralité
  - Plage d'application potentielle des composants
- Auto-documentation ou lisibilité
  - Possibilité d'extraction de la documentation depuis les composants logiciels
  - Ex : nom de variables, docstrings, etc.

Liens qualités $\leftrightarrow$ critères info.



Qualité du proc. de développement (1)

Principales qualités :

- Sûreté
  - Minimise les retours arrière et validations régulières
- Terminaison
  - Obtention du produit en temps fini
- Rigueur
  - Étapes logiques, en accord avec les habitudes des développeurs
- Cohérence
  - Pas de duplication ou d'oublis

Qualité du proc. de développement (2)

- Souplesse
  - Adaptation à l'application à développer
- Accessibilité
  - Comprendre les choix effectués
- Rentabilité
  - Capitaliser l'expérience

Critères (1)

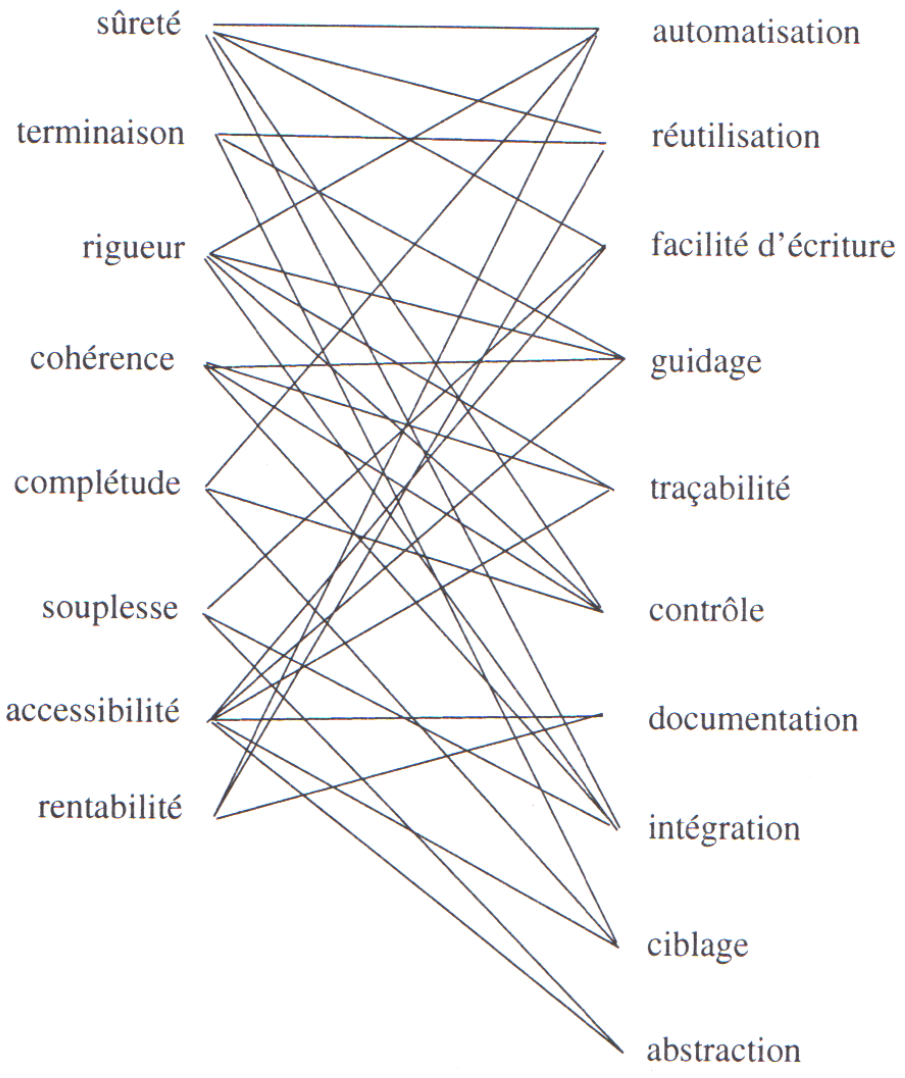
Critères permettant d'atteindre ces qualités :

- Automatisation
  - Moins d'erreur
  - Plus vite
- Réutilisation
  - Réduit le coût
  - Augmente la sureté
- Facilité d'écriture
  - Modèles simples et naturels pour les développeurs
- Guidage
  - Opérations à réaliser pour obtenir un bon résultat
- Traçabilité
  - Vérification de la cohérence entre les modèles

Critères (2)

- Contrôle
  - Contrôle régulier
- Intégration
  - Cohérence entre modèles d'une même étape ou deux successives
- Documentation
  - Raisonnement et choix explicites
- Ciblage
  - Domaine d'application explicite
- Abstraction
  - Le raisonnement et la preuve doivent progressivement prendre en compte les concepts de programmation

Liens qualités ↔ critères

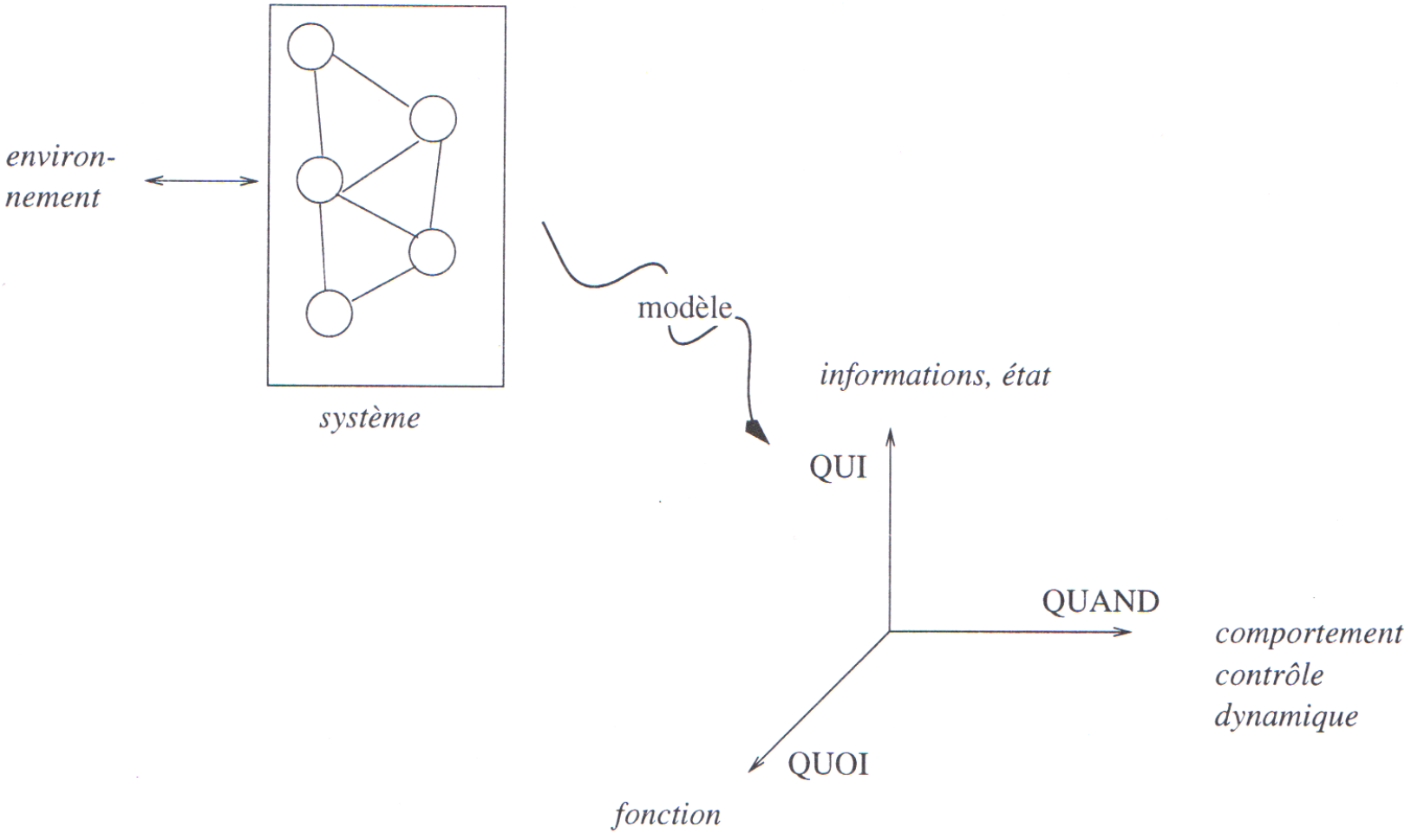


Plan

Plan de la suite du cours

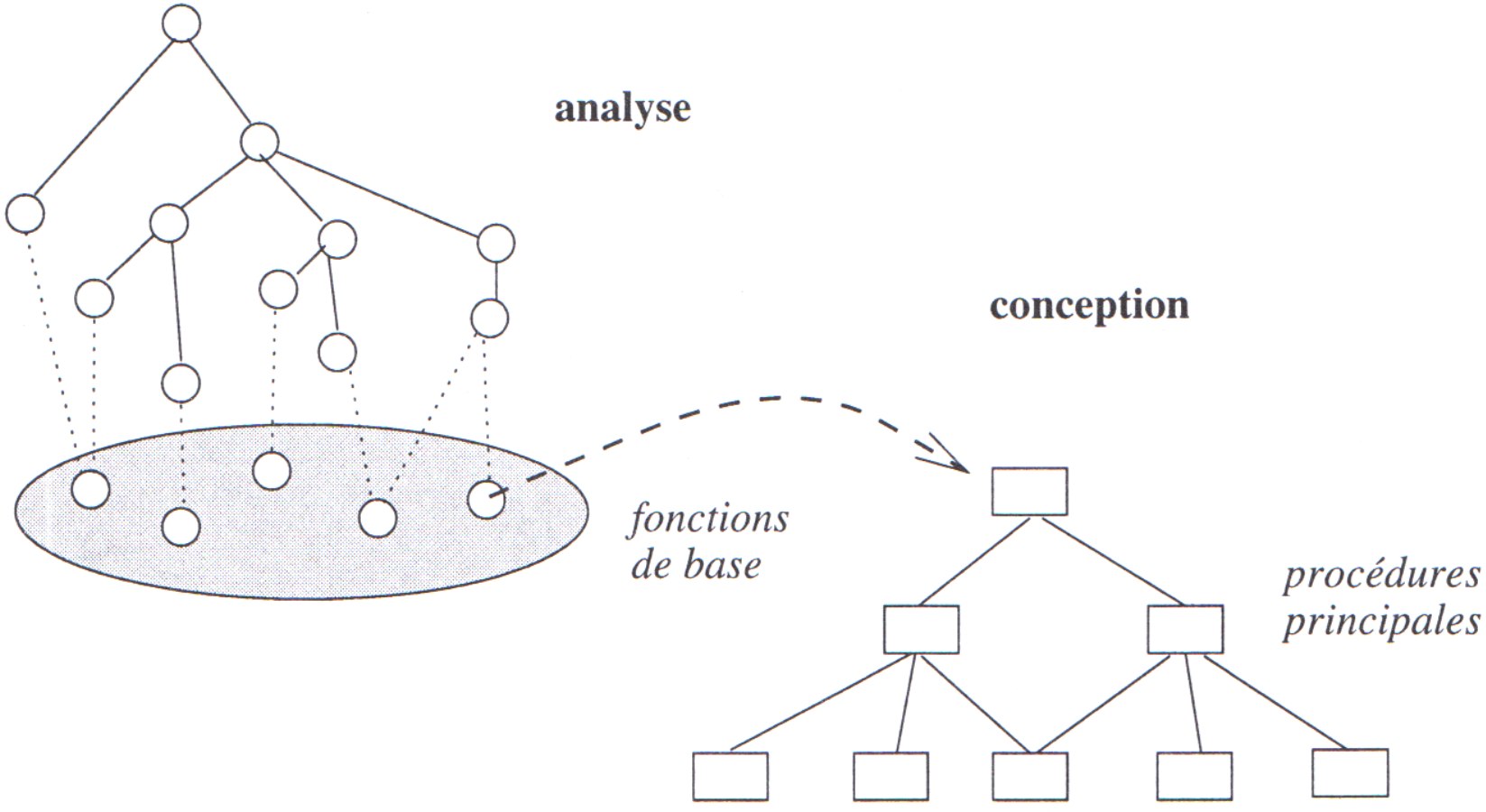
1. Merise
   - Méthode systémique : vision globale
   - Modélisation selon l'axe des données
   - Utilisation pour la modélisation des bases de données
2. Introduction à UML
   - Panel de diagrammes
     - Approche fonctionnelle, objet (méthode cartésiennes)
     - Flot de données
     - Possible : modèle de données
   - En détail : cas d'utilisation (modélisation fonctionnelle)
3. Méthodes agiles

Trois axes de modélisation

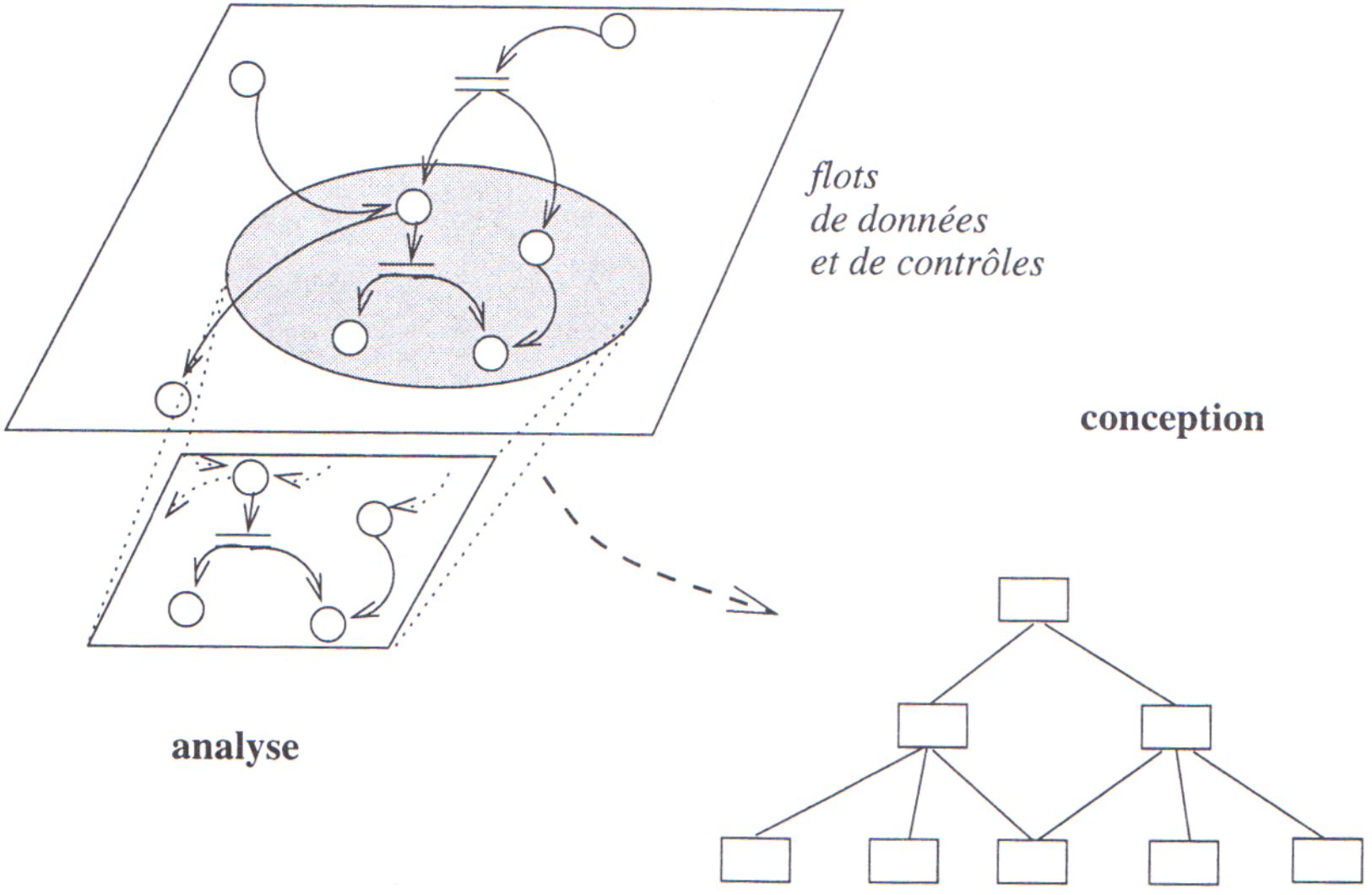


→ Trois courants de méthodes d'analyse/conception

Approche fonctionnelle



Approche flots de données



Approche modèle de données