


Chapitre 1

Processus Unifié



Naissance du besoin de modéliser

❖ Un modèle :

- est une abstraction du réel à informatiser.
- utilise un ensemble de concepts et un ensemble de règles d'utilisation :
 - Les concepts fournissent le moyen de représenter les entités du monde réel et les relations qui existent entre elles.
 - Les règles sont définies sur les concepts du modèle et régularisent leurs utilisations.

Remarque:

Pour assister le concepteur afin d'élaborer le(s) modèle(s) on a besoin d'une méthode.

Naissance du besoin de modéliser

❖ Une méthode :

- Guide la transition du réel à sa perception et de cette dernière vers l'implantation.
- Composée de :
 - + Un ensemble d'étapes successives : une démarche.
 - + Un ensemble de modèles exprimant des points de vue différents.
 - + Un ensemble de concepts (et leurs règles d'utilisation) permettant la représentation de ces modèles.

Pourquoi une méthodologie / Processus?

- ▶ Les techniques (diagrammes d'UML) de développement de système doivent être organisées si elles doivent fonctionner ensemble
- ▶ UML même ne contient rien qui aide à prendre cette décision

Pourquoi une méthodologie / Processus?

- ▶ L'organisation des tâches n'est pas contenue dans les techniques
- ▶ Elle doit être décrite à un plus haut niveau d'abstraction

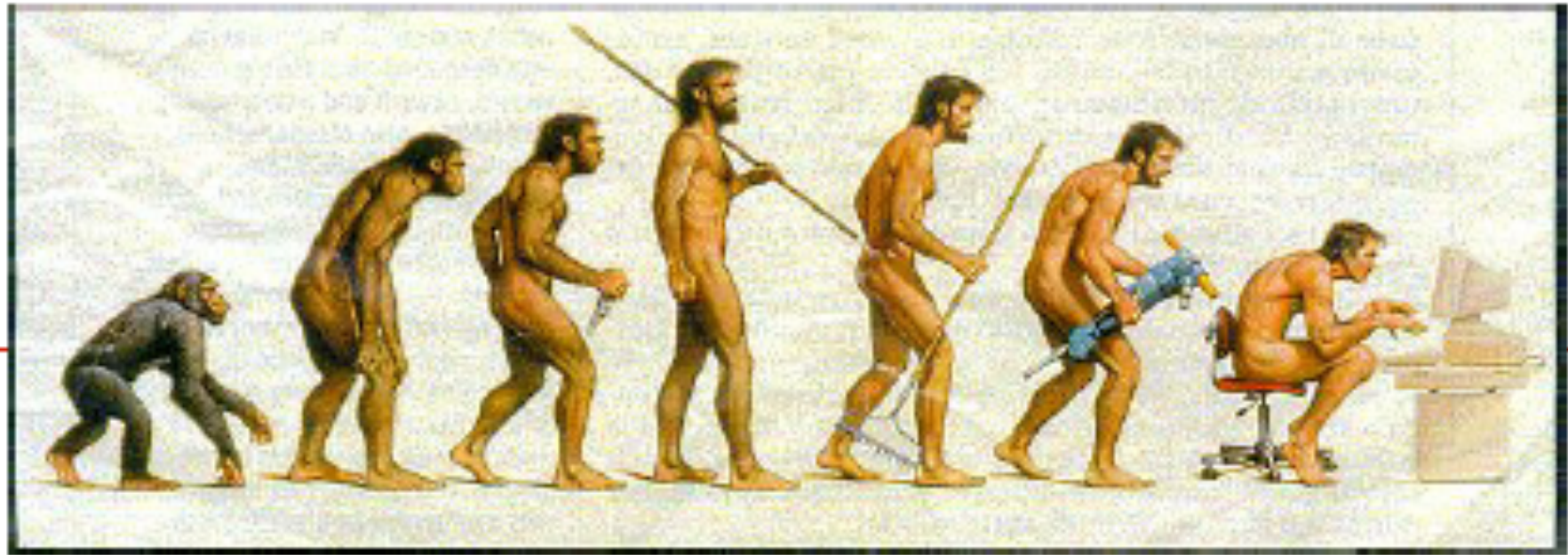
Pourquoi utiliser une méthodologie / Processus?

- ▶ De nombreux avantages sont avancés
 - ▶ Aide à produire un produit de meilleure qualité
 - ▶ Logiciel mieux documenté
 - ▶ Plus acceptable par l'utilisateur
 - ▶ Plus facile à maintenir/entretenir
 - ▶ Logiciel plus homogène
 - ▶ Aide à assurer que les spécifications des utilisateurs sont suivies
 - ▶ Aide le manager du projet à contrôler le projet
 - ▶ Réduit les coûts de développement
 - ▶ Encourage la communication entre les personnes

UML n'est qu'un langage

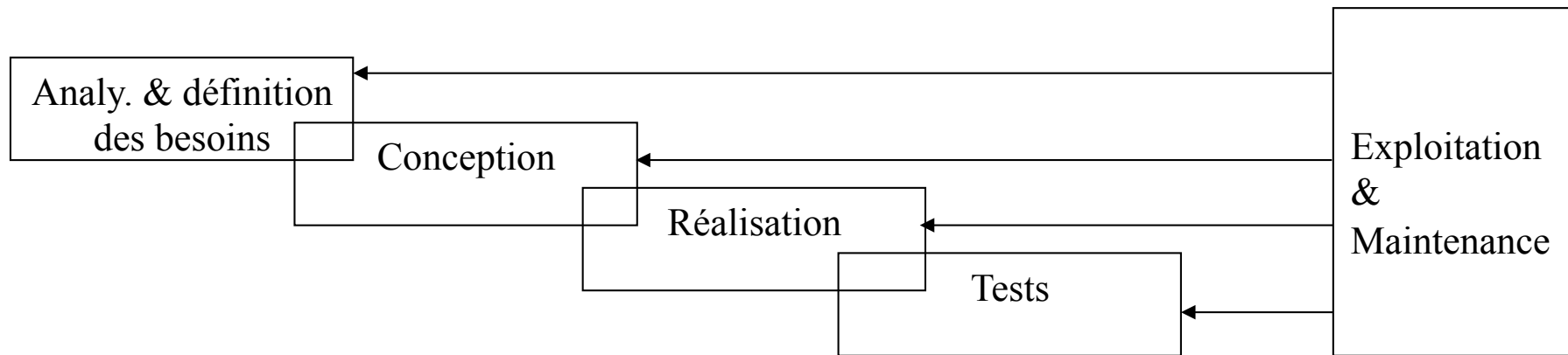
- ▶ Bien qu'issu d'une concertation visant à produire le processus Unifié, UML n'est qu'un langage de modélisation et non une méthodologie à part entière.
- ▶ UML définit des notations utiles dans toutes les étapes du développement d'un logiciel, de l'analyse des besoins à la livraison de celui-ci, mais il ne propose aucune démarche spécifique pour mener ce développement à terme.
- ▶ En ce sens, UML peut a priori être utilisé dans le cadre de l'utilisation de toute méthode de développements (par objets).

Evolution des méthodes de conduite de projet



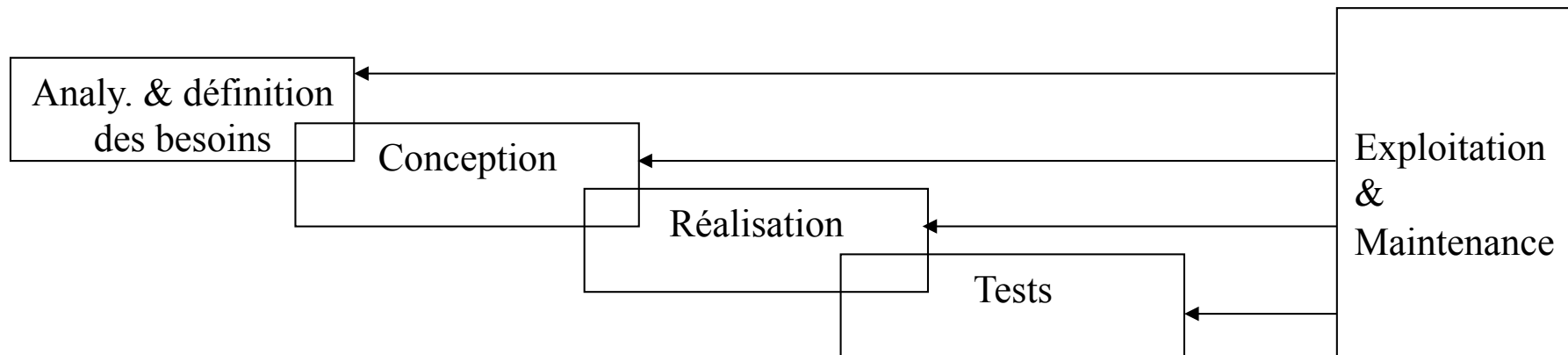
Approches Classiques dites Séquentielles

- **Toutes s'intersectent dans les étapes suivantes :**



Approches Classiques dites Séquentielles

- **Toutes s'intersectent dans les étapes suivantes :**



- ne sont pas forcément appelées de la même manière par toutes les méthodes
- **Avec les tests, deux étapes peuvent exister :**
 1. la vérification du système : « Est-ce que nous construisons bien le système ? »
 2. la validation du système : « Est-ce que nous construisons le bon système ? »

Activités des méthodes de conduite de projets

- ▶ Analyse et définition des besoins : Etude du métier du client, étude des besoins des utilisateurs, formulation du CDC sous une forme exploitable en conception
- ▶ Conception : Définition de l'architecture logicielle, définition du comportement de l'application
- ▶ Réalisation : Implémentation
- ▶ Tests
- ▶ Exploitation et maintenance

Analyse des besoins

- ▶ **L'analyse des besoins** définit les services du système, ses contraintes et ses buts, en consultant les utilisateurs du système. Une étude d'opportunité peut être menée pour savoir si le système est réalisable et donner une approximation de la rentabilité de ce système.
- ▶ Synonymes : analyse préalable, *user requirements analysis*.

Analyse

- ▶ **l'analyse** est la construction d'un modèle (une spécification) du système à partir de l'analyse des besoins. À partir de ce modèle on peut proposer plusieurs scénarii et réaliser une étude de faisabilité.
- ▶ Synonymes : spécification des besoins, analyse préalable,
- ▶ *analysis, user requirement specification, requirements engineering.*

Conception

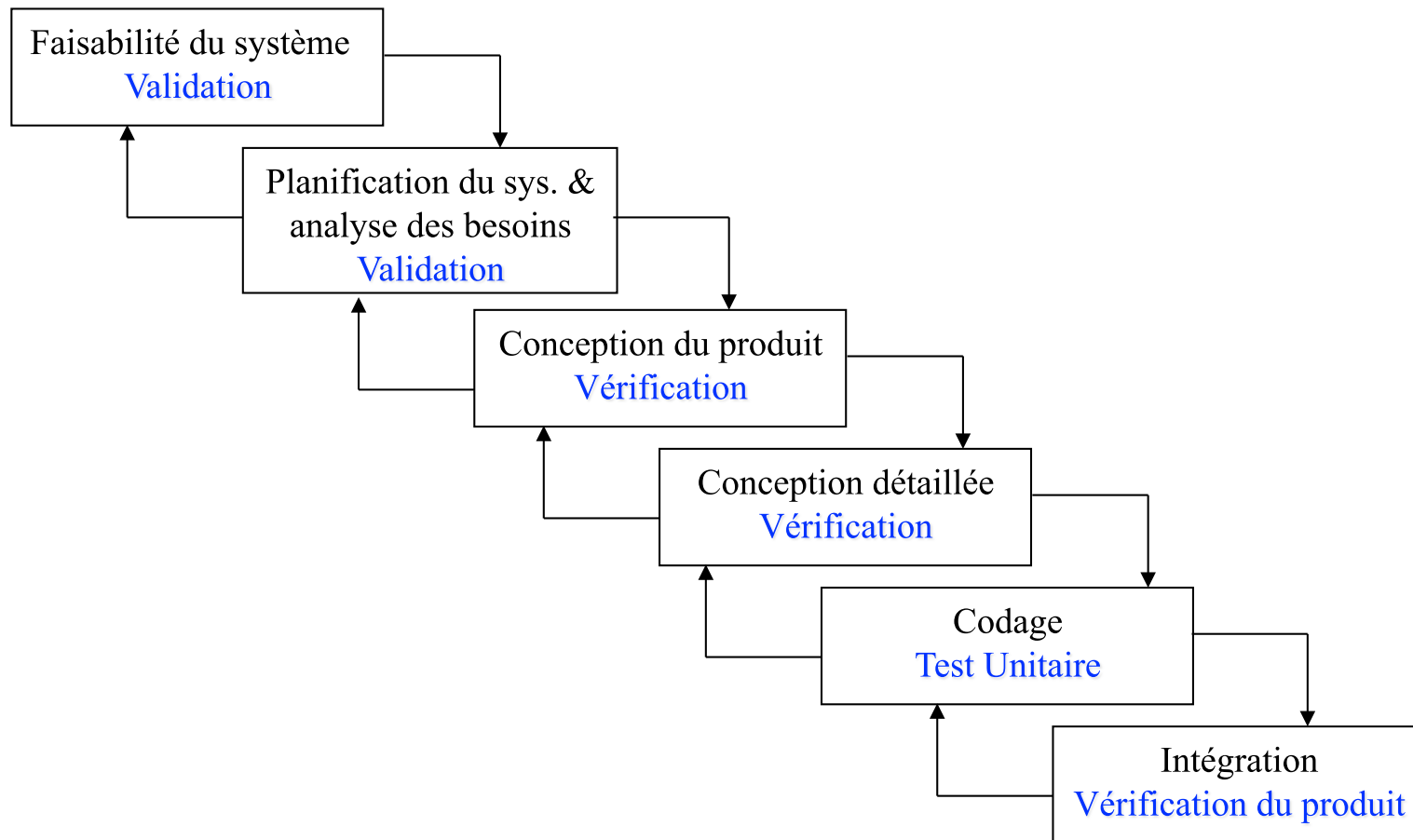
- ▶ la **conception** est une proposition de solution au problème spécifié dans l'analyse.
- ▶ Définit la solution retenue par prise en compte des caractéristiques logiques d'usage du futur système d'information et des moyens de réalisation, humains, techniques et organisationnels.
- ▶ On distingue souvent :
 - ▶ la conception système : a pour objectif de donner l'architecture globale du systèmes (i.e. les différentes parties)
 - ▶ la conception détaillée : décrit chaque partie du système.

Réalisation et tests

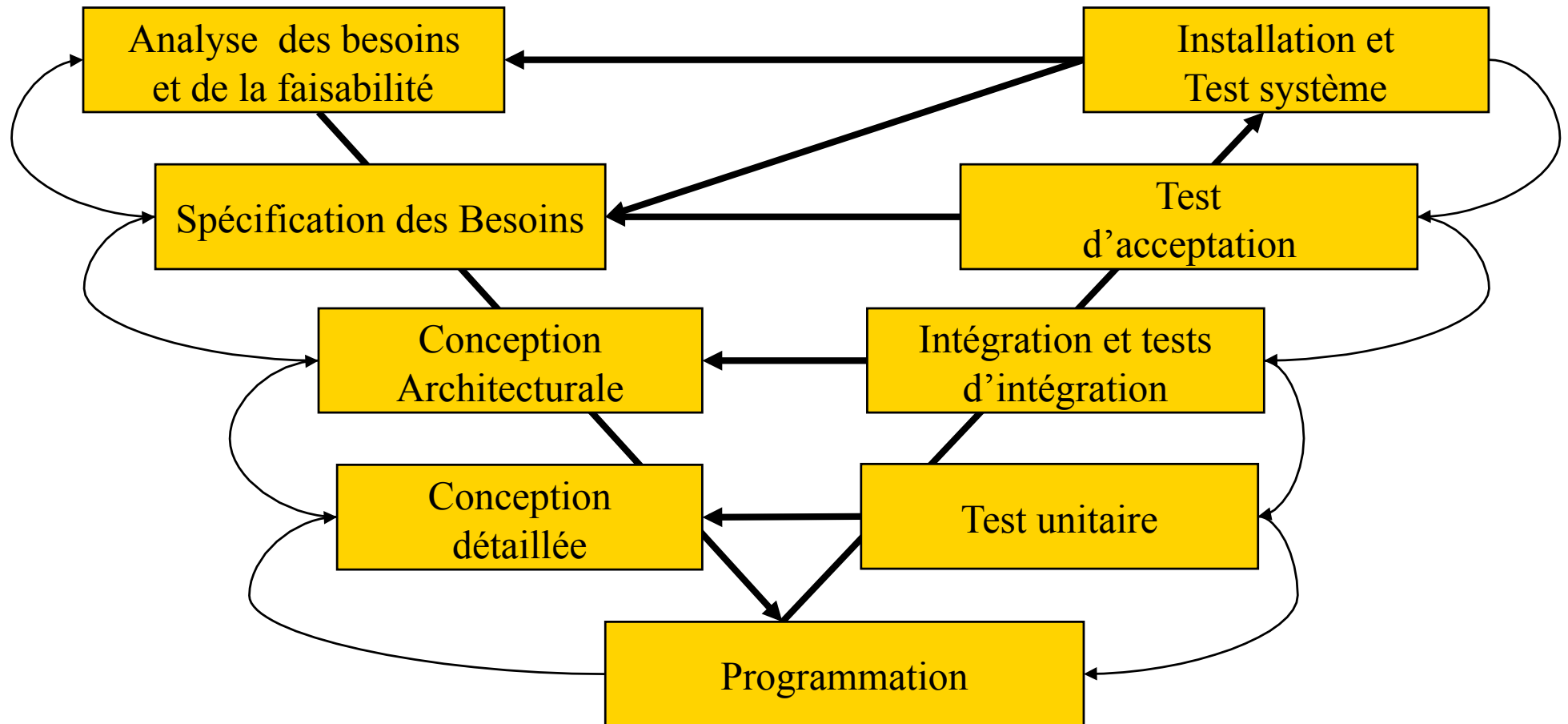
- ▶ La **réalisation** et le **test** produisent la solution exécutable en termes de programmes.
- ▶ Pour les logiciels complexes, on distingue l'implantation des différentes parties et leur validation par des tests unitaires et l'implantation du système complet par **intégration** des parties et tests systèmes.
- ▶ On vérifie ainsi que les spécifications des besoins sont satisfaites.

Processus en cascades

► **Schématiquement :**



Processus en V



Le Processus Unifié

Le Processus Unifié

- ▶ Les auteurs principaux d'UML, I. Jacobson, G. Booch et J. Rumbaugh, proposent une démarche de développement pouvant être facilement associée à UML : le Processus Unifié (Unified Software Development Process - 1999).
- ▶ L'essentiel du Processus Unifié provient des méthodes Objectory, Booch, et OMT, auxquelles s'ajoutent quelques apports issus des travaux d'élaboration d'UML.
- ▶ Le Processus Unifié ne fait pas partie intégrante du standard UML.

le Processus Unifié ou “Unified Software Development Process” (USDP)

- ▶ Processus du domaine public pour le développement Orienté-Objet de logiciel

Le Processus Unifié

- ▶ **Le Processus Unifié est :**
 - ▶ guidé par les use-cases,
 - ▶ centré sur l'architecture,
 - ▶ itératif et incrémental.

Guidé par les use-cases

- ▶ L'objectif d'un système logiciel est de rendre service à ses utilisateurs. Pour réussir la mise au point d'un système, il importe, par conséquent, de bien comprendre les désirs et les besoins de ses futurs utilisateurs.
- ▶ Un cas d'utilisation est une fonctionnalité du système produisant un résultat satisfaisant pour l'utilisateur.
- ▶ Les cas d'utilisation saisissent les besoins fonctionnels et leur ensemble forme le modèle des cas d'utilisation qui décrit les fonctionnalités complètes du système.

Centré sur l'architecture

- ▶ Le rôle de l'architecture logicielle est comparable à celle que joue l'architecte dans la construction d'un bâtiment. Le bâtiment est envisagé de différents points de vue: structure, services, conduite de chauffage, plomberie, etc.
- ▶ Ce regard multiple dessine une image complète du bâtiment avant le début de la construction. De la même façon, l'architecture d'un système logiciel peut être décrite comme les différentes vues du système qui doit être construit.

Itératif et incrémental

- ▶ Le développement d'un produit logiciel destiné à la commercialisation est une vaste opération qui peut s'étendre sur plusieurs mois, voire sur une année ou plus.
- ▶ Il n'est pas inutile de découper le travail en plusieurs parties qui sont autant de mini-projets (Concept systémique de système et sous-systèmes). Chacun d'eux représente une itération qui donne lieu à un incrément.
- ▶ Les itérations désignent des étapes de l'enchaînement d'activités, tandis que les incréments correspondent à des stades de développement du produit.

Les “4” P

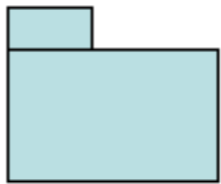
- ▶ Personne
- ▶ Processus
- ▶ Projet
- ▶ Produit

Personne

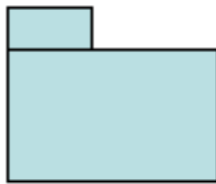
- ▶ Diverses personnes sont impliquées dans le développement d'un produit logiciel tout au long de son cycle de vie.
- ▶ Certains financent le produit, tandis que d'autres le planifient, le développent, le gèrent, le testent , l'utilisent ou en bénéficient.
- ▶ Le processus guidant le développement doit, par conséquent, être orienté vers les personnes, c'est-à-dire convenir parfaitement à ses utilisateurs.

Le projet n'est pas uniquement du Code

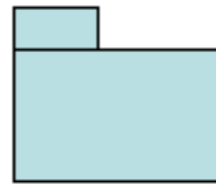
– *Modèles*



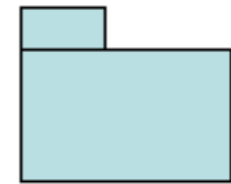
Modèle des
cas d'utilisation



Modèle
d'analyse



Modèle de
conception



Modèle de
déploiement



Modèle
d'implémentation



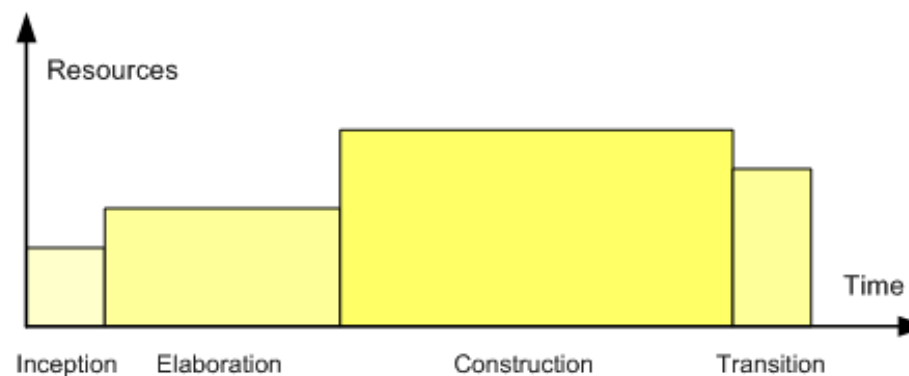
Modèle
des tests

Processus

- ▶ Un processus de développement définit les activités nécessaires à la transformation des besoins des utilisateurs en un ensemble cohérent d'artefacts (modèles) constituant un produit logiciel et, plus tard, par la transformation des évolutions de ces besoins en un ensemble d'artefacts tout aussi cohérent

Le Processus Unifié

- ▶ Les itérations peuvent être regroupées en 4 phases :
 - ▶ création (objectifs du projet, et coûts),
 - ▶ élaboration (besoins),
 - ▶ construction (obtention du produit quasi finalisé),
 - ▶ transition (mise au point et livraison).



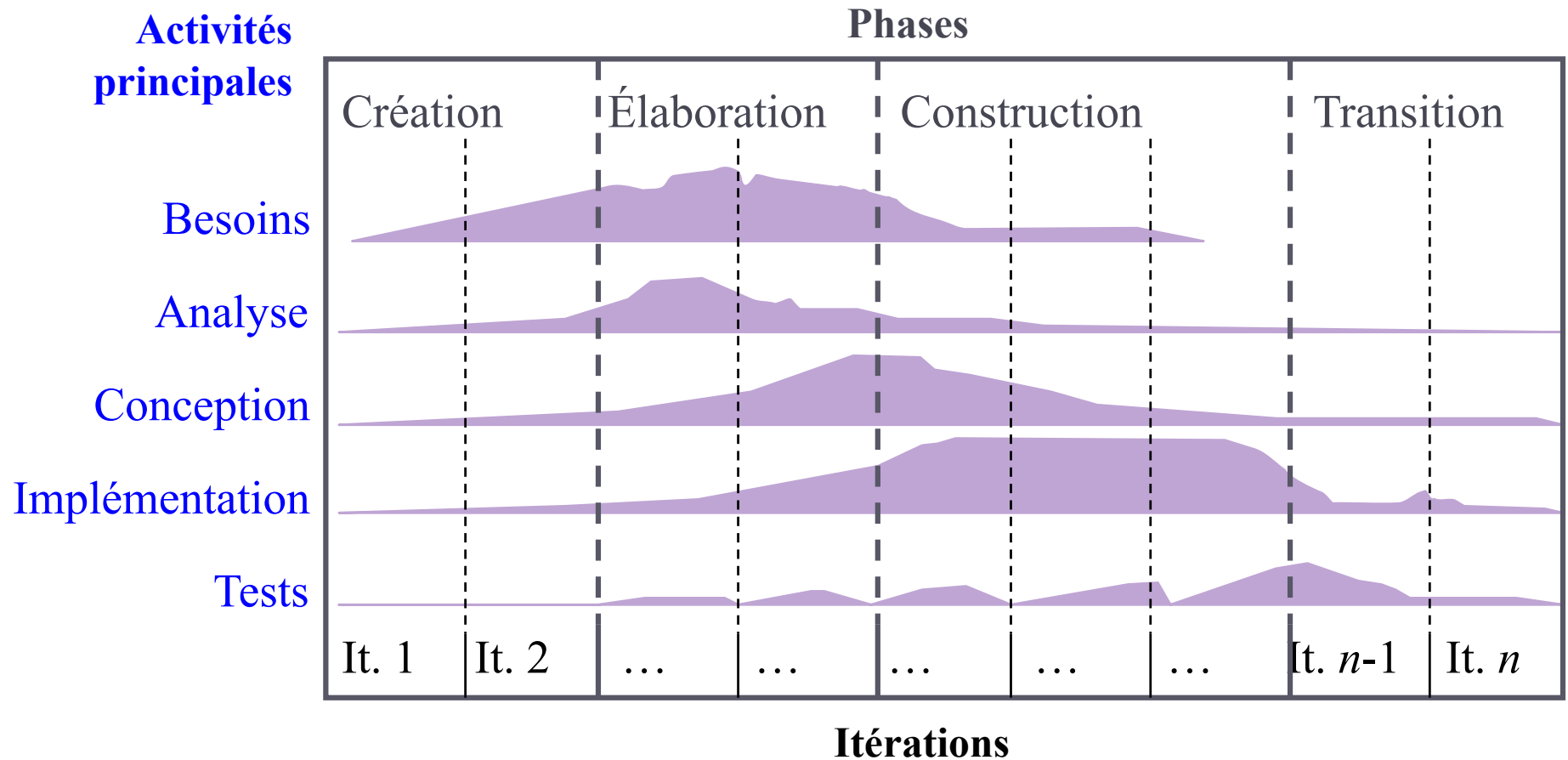
Profil d'un projet typique montrant les tailles relatives des quatre phases

Le Processus Unifié

- ▶ Chaque itération consiste à enchaîner 5 activités principales :
 - ▶ Expression des Besoins,
 - ▶ Analyse,
 - ▶ Conception,
 - ▶ Implémentation,
 - ▶ Tests.
- ▶ Différentes activités plus spécifiques peuvent s'enchaîner dans chacune de ces 5 activités principales. Elles sont effectuées par différents travailleurs.

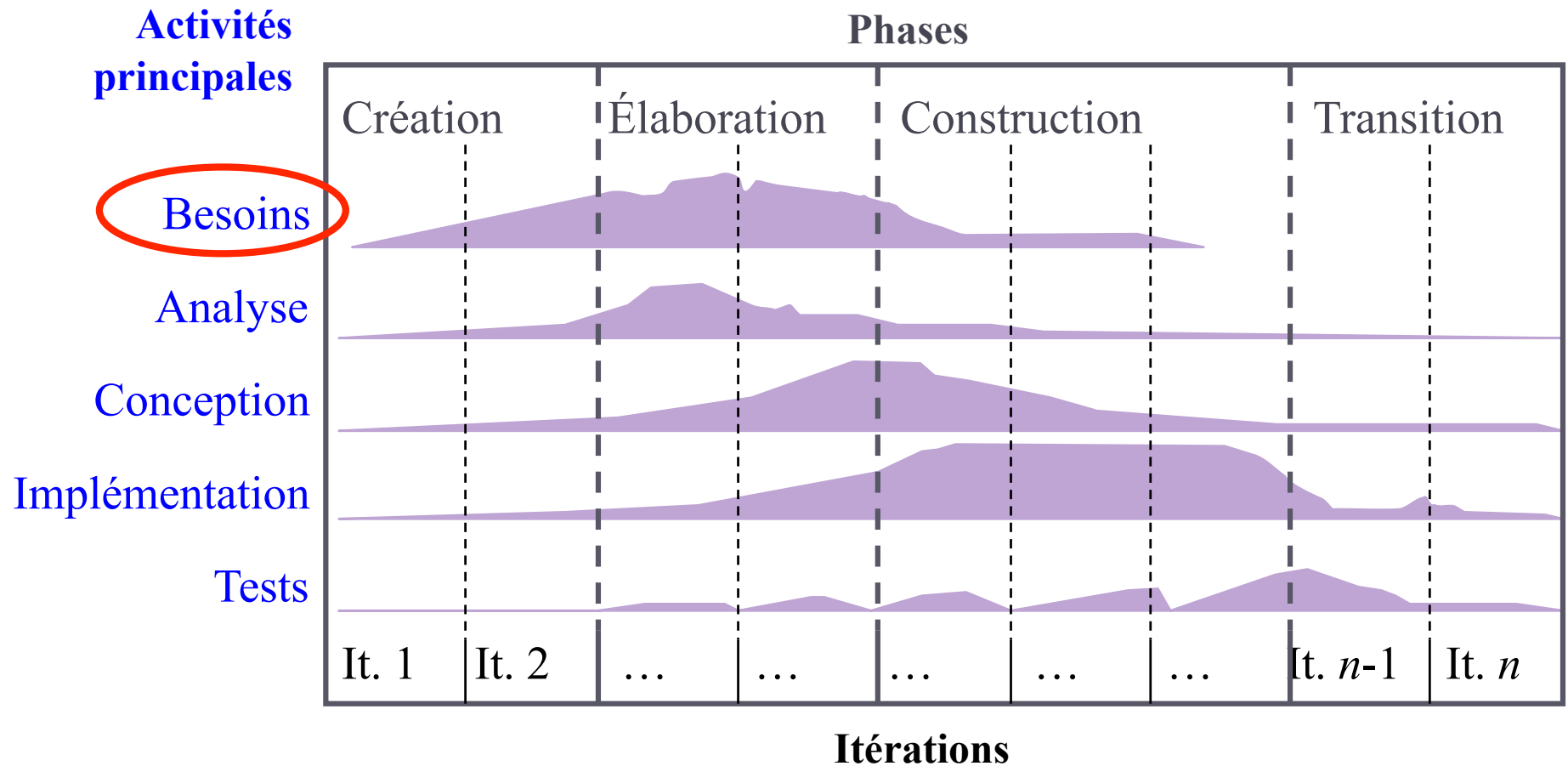


Organisation en phases, itérations et activités



Répartition du travail en phases, itérations et activités principales

Organisation en phases, itérations et activités



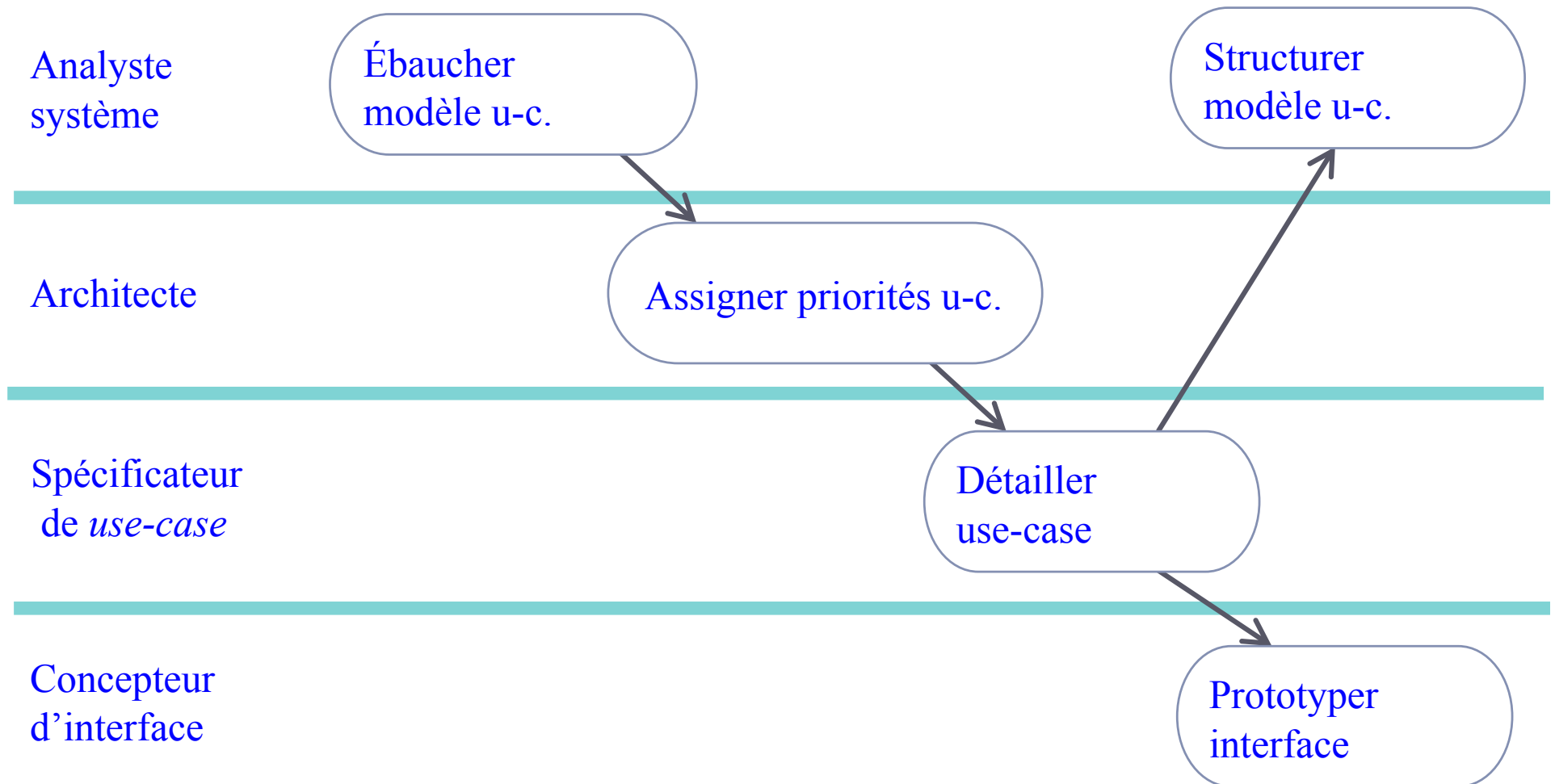
Répartition du travail en phases, itérations et activités principales

Activité- Analyse des besoins

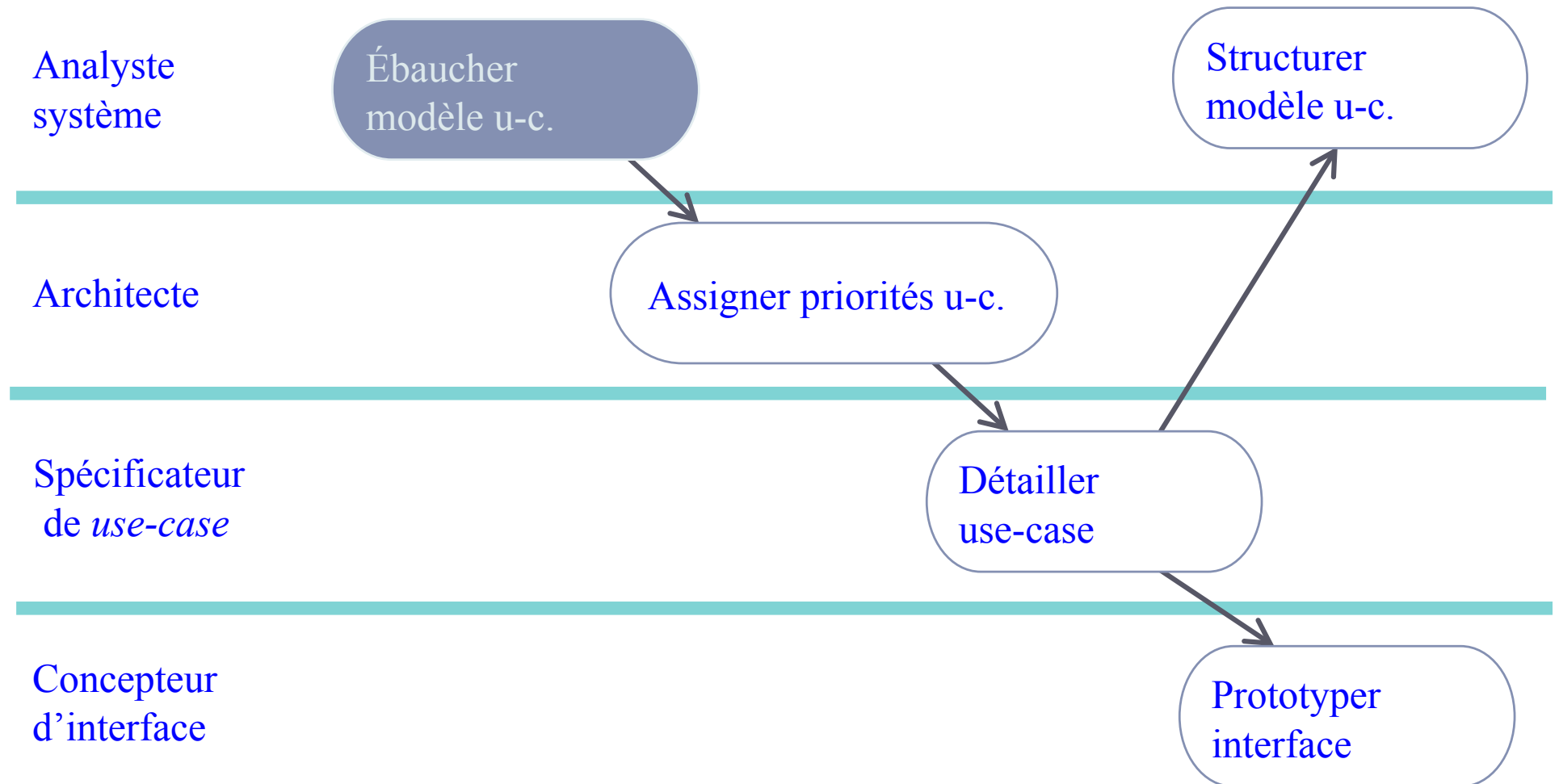
Activités de l'analyse des besoins

- ▶ L'analyse des besoins regroupe les activités suivantes :
 - ▶ établir une ébauche du modèle des use-cases,
 - ▶ assigner des priorités aux use-cases,
 - ▶ détailler chaque use-case,
 - ▶ prototyper l'interface utilisateur,
 - ▶ structurer le modèle des use-cases.

Enchaînement des activités de l'analyse des besoins



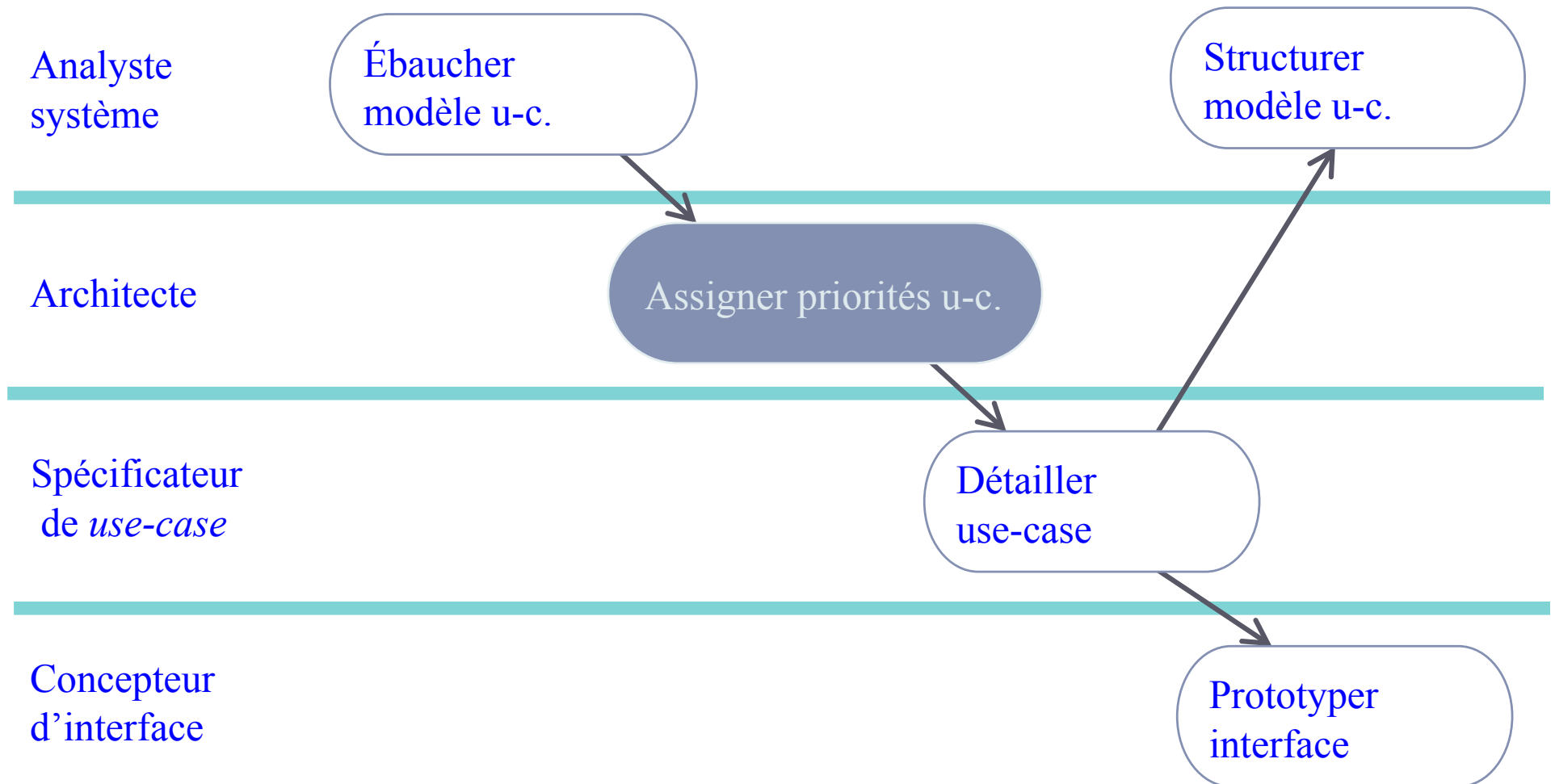
Enchaînement des activités de l'analyse des besoins



Ébauche du modèle des use-cases

- ▶ L'analyste système est responsable de cette activité.
- ▶ A partir notamment de la liste des fonctionnalités, elle consiste à :
 - ▶ identifier les acteurs,
 - ▶ identifier les use-cases,
 - ▶ décrire brièvement chaque use-case,
 - ▶ fournir un diagramme global de use-cases,
 - ▶ élaborer un glossaire.

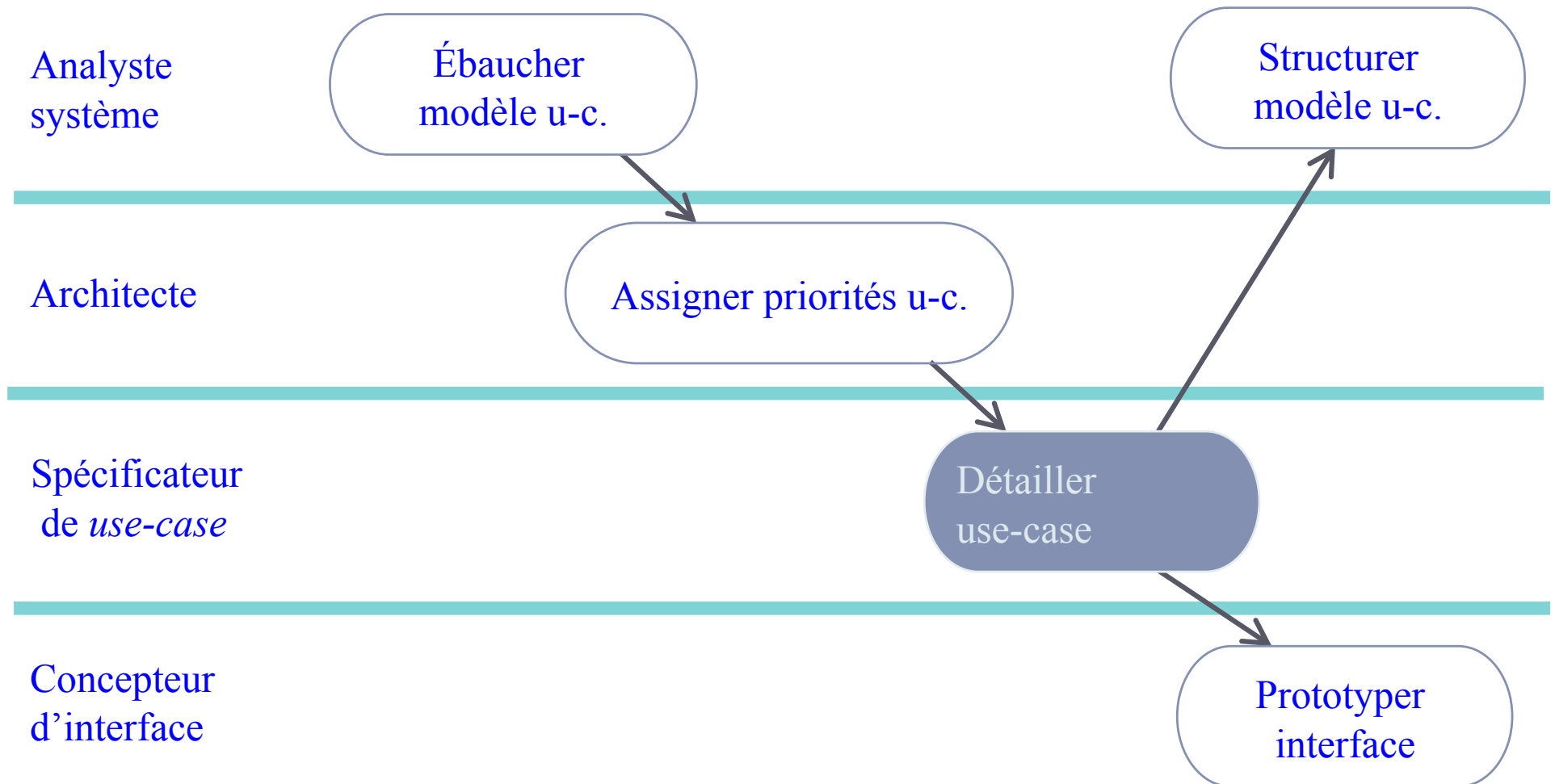
Enchaînement des activités de l'analyse des besoins



Priorité des use-cases

- ▶ Cette activité fait suite à l'ébauche du modèle des use-cases.
- ▶ L'architecte établit un ordre de priorité pour la réalisation (conception, implémentation, ...) des use-cases dans les itérations ultérieures.

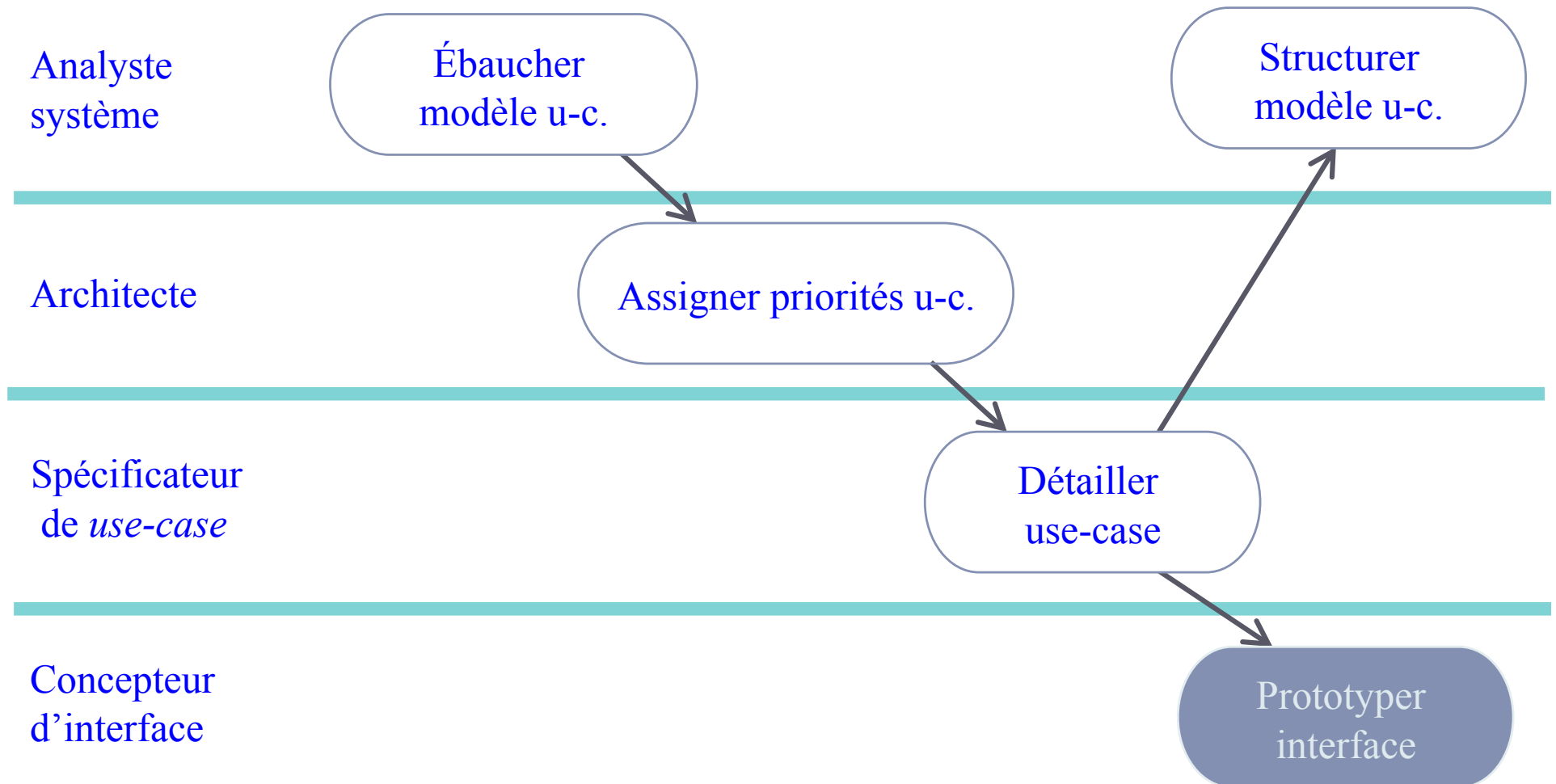
Enchaînement des activités de l'analyse des besoins



Description détaillée des use-cases

- ▶ Cette activité fait suite à l'établissement des priorités sur les use-cases.
- ▶ Un responsable est nommé pour la description détaillée de chaque use-case, il doit :
 - ▶ détailler la description sommaire fournie par l'analyste système, par des descriptions textuelles, des diagrammes de séquence ou de collaboration;
 - ▶ rechercher les déroulements anormaux et exceptionnels possibles et les décrire.

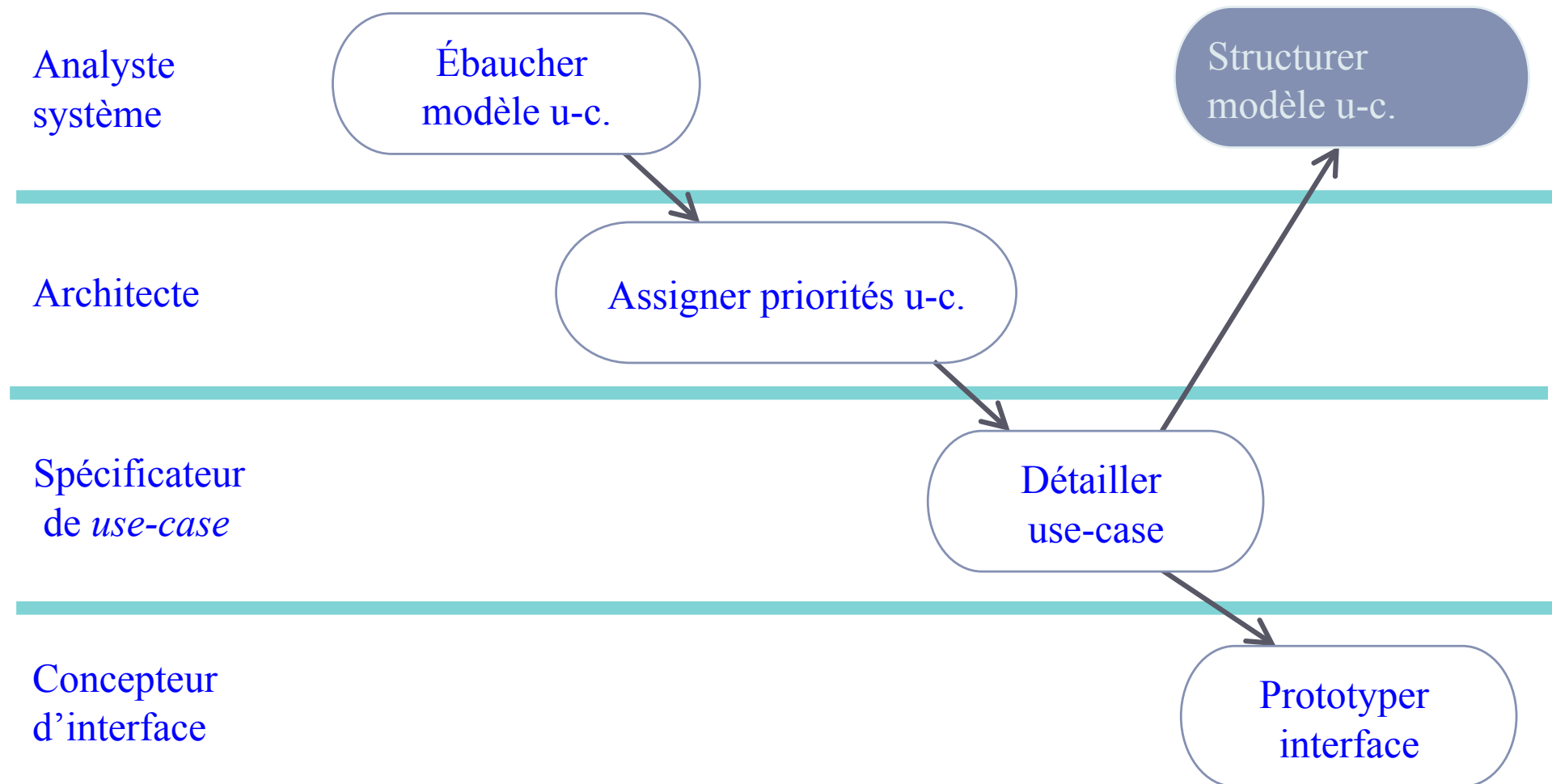
Enchaînement des activités de l'analyse des besoins



Prototypage de l'interface utilisateur

- ▶ Cette activité fait suite à la description détaillée des use-cases.
- ▶ Le concepteur d'interface fournit :
 - ▶ une conception logique de l'interface utilisateur (indépendante de la manière dont elle peut être implémentée),
 - ▶ des croquis de "pages-écrans" en accord avec cette conception logique.

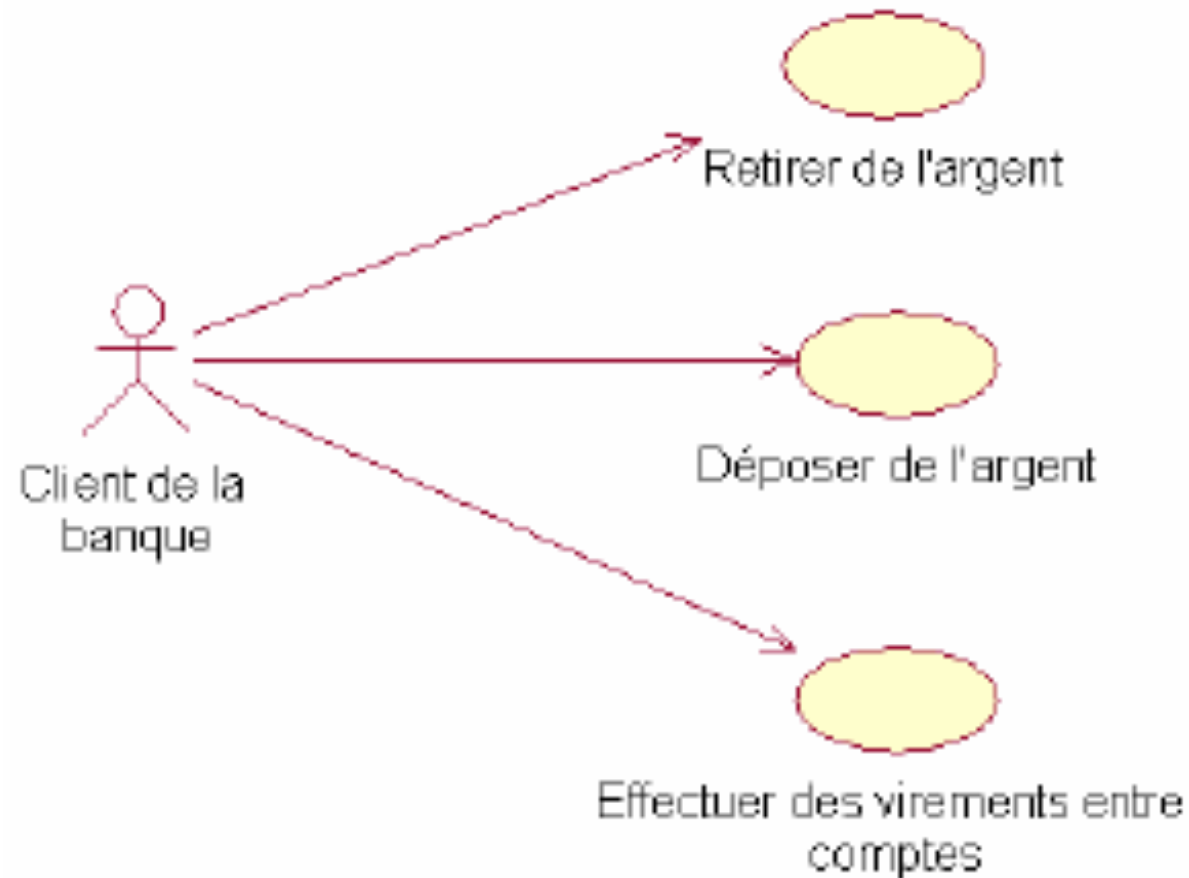
Enchaînement des activités de l'analyse des besoins



Structurer le modèle des use-cases

- ▶ Cette activité fait suite à la description détaillée des use-cases.
- ▶ L'analyste système améliore le modèle des use-cases, notamment en identifiant :
 - ▶ des descriptions similaires (généralisation),
 - ▶ des descriptions communes (inclusion),
 - ▶ des descriptions optionnelles (extension).

Modèle des use-cases

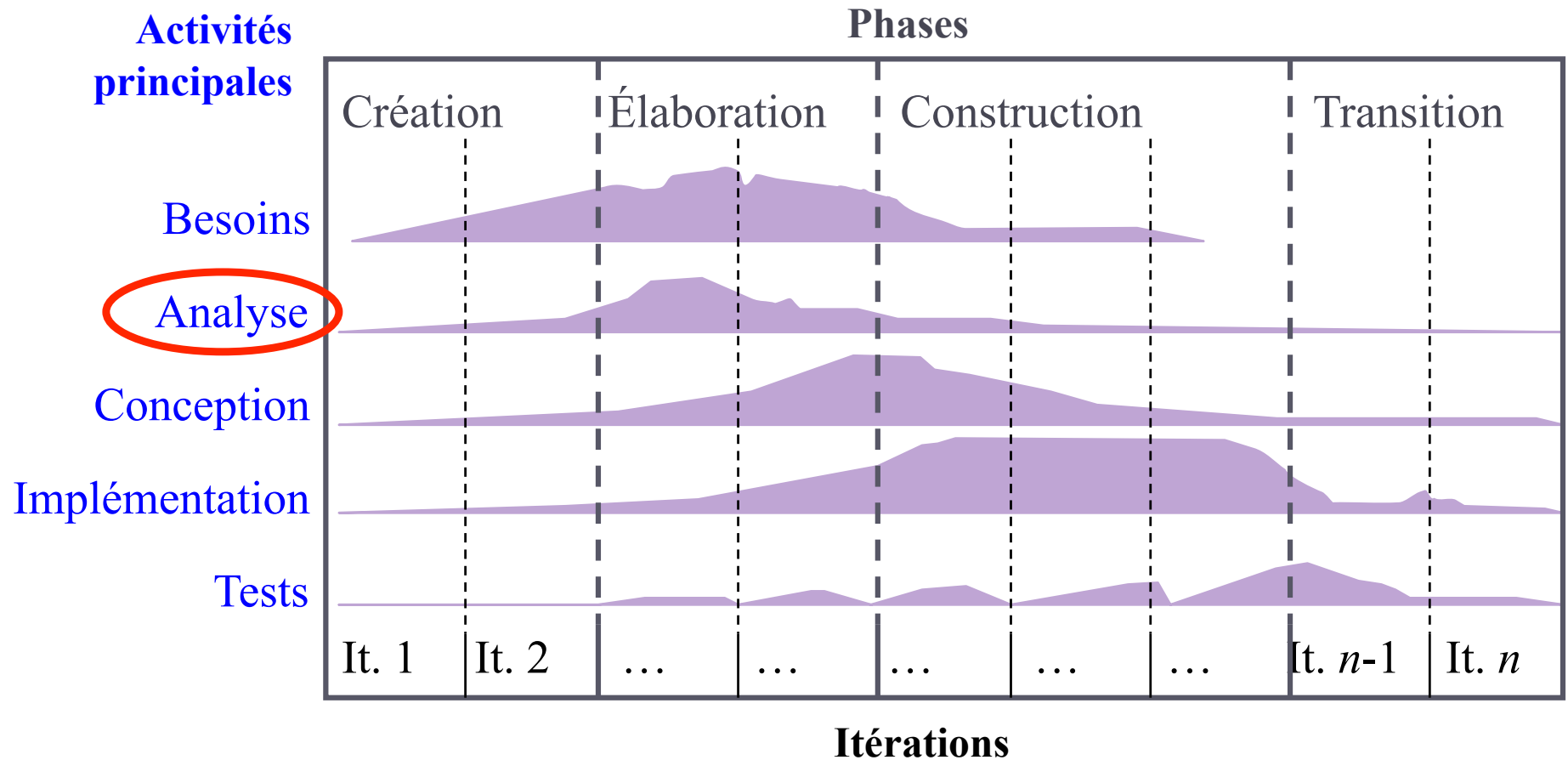


Modèle des use-cases

Cas d'utilisation « Retirer de l'argent »

1. Le Client de la banque *s'identifie*
2. Le Client de la banque *choisit le compte duquel il veut effectuer son retrait et spécifie le montant du retrait.*
3. *Le système déduit le montant du compte et délivre l'argent.*

Organisation en phases, itérations et activités



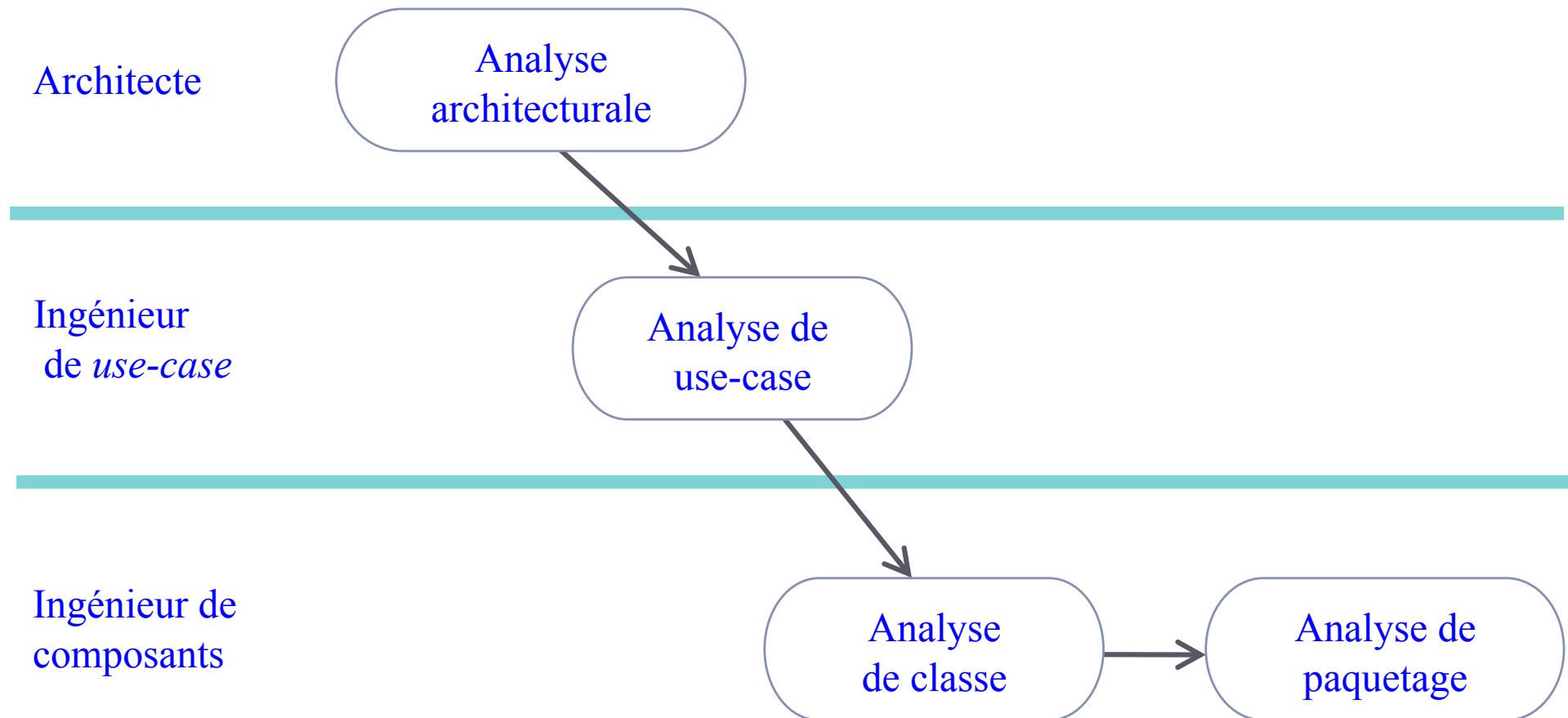
Répartition du travail en phases, itérations et activités principales

Activité - Analyse

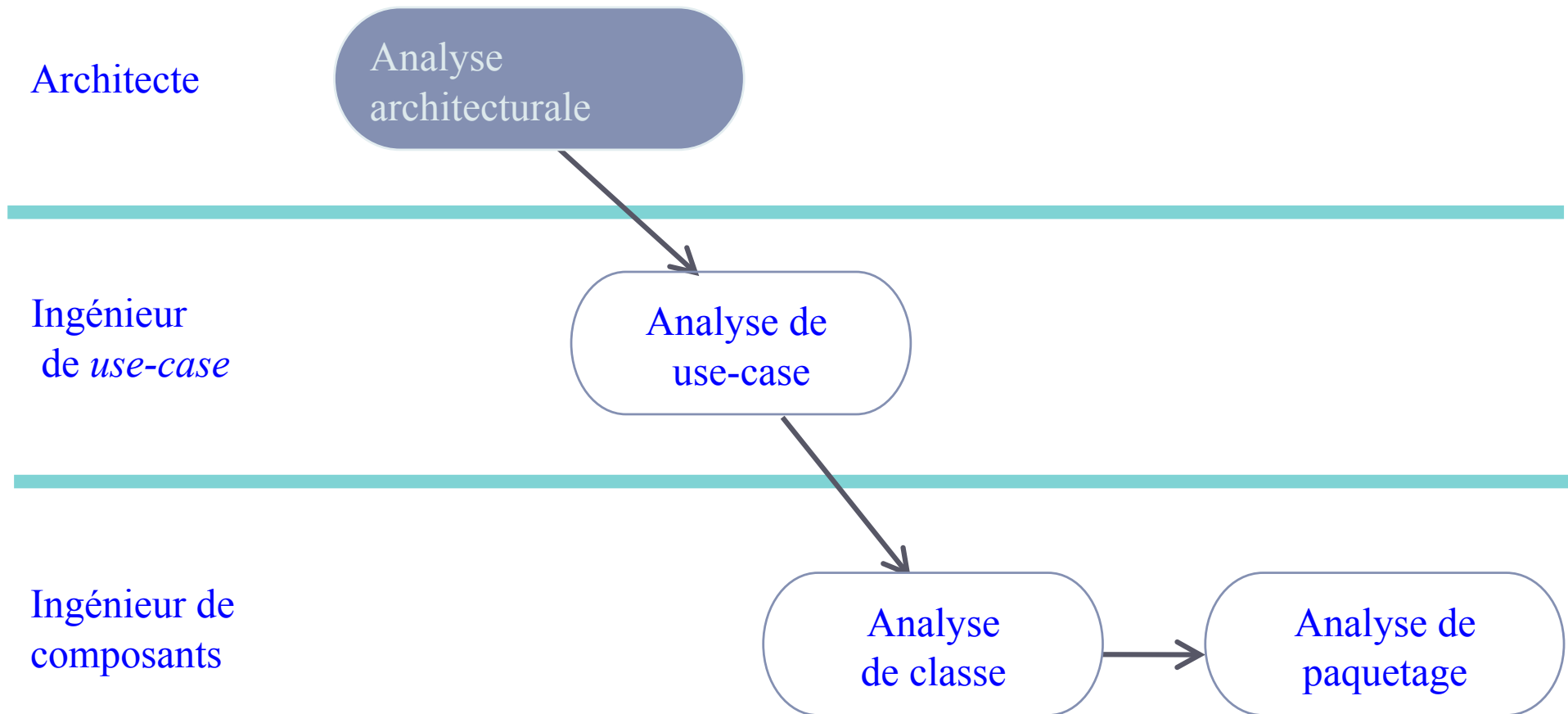
Activités de l'analyse

- ▶ L'analyse regroupe les activités suivantes :
 - ▶ analyse architecturale,
 - ▶ analyse de use-case,
 - ▶ analyse de classe,
 - ▶ analyse de paquetage.

Enchaînement des activités de l'analyse



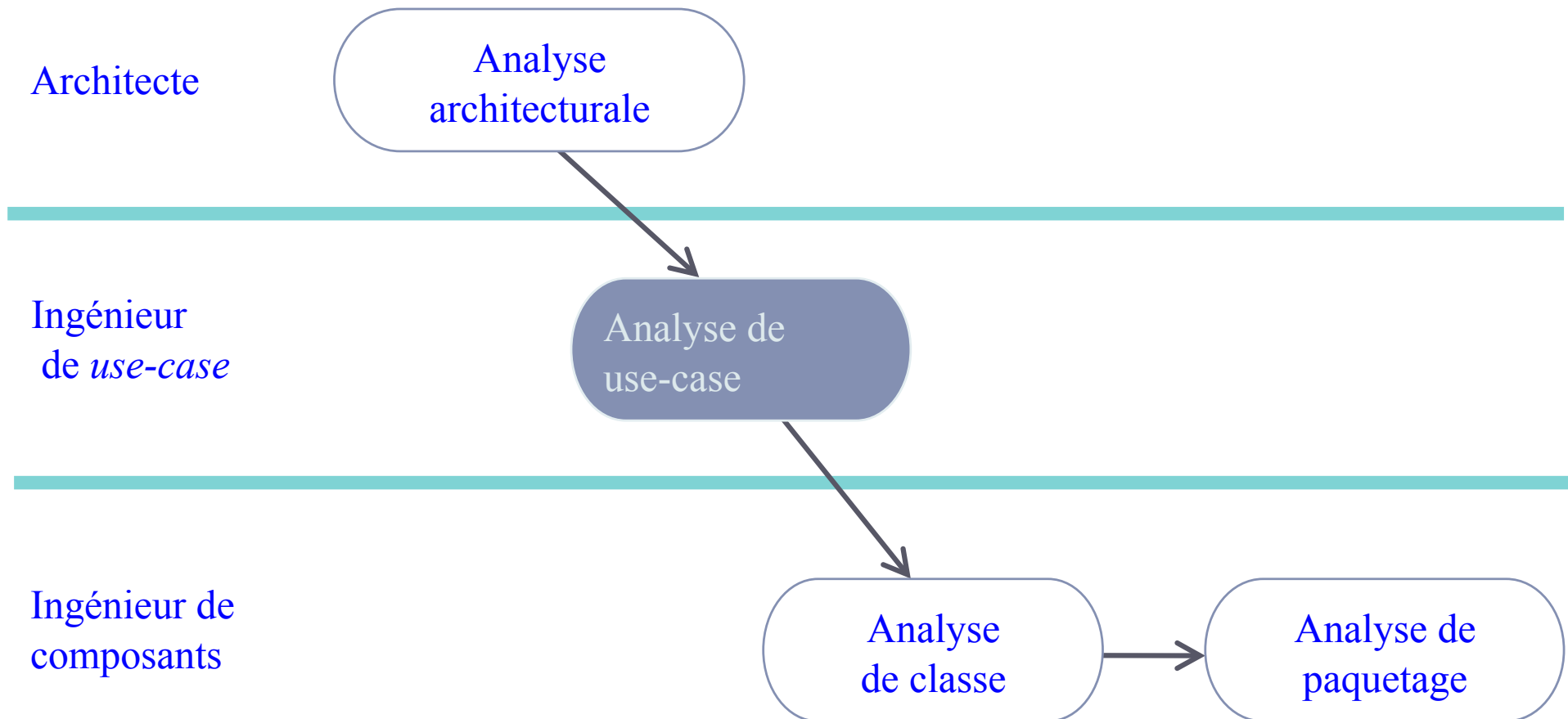
Enchaînement des activités de l'analyse



Analyse architecturale

- ▶ Cette activité est sous la responsabilité de l'architecte.
- ▶ A partir du modèle des use-cases, elle consiste à fournir :
 - ▶ une ébauche des classes d'analyse manifestes,
 - ▶ une ébauche des paquetages d'analyse
 - ▶ une première description de l'architecture (exigences).

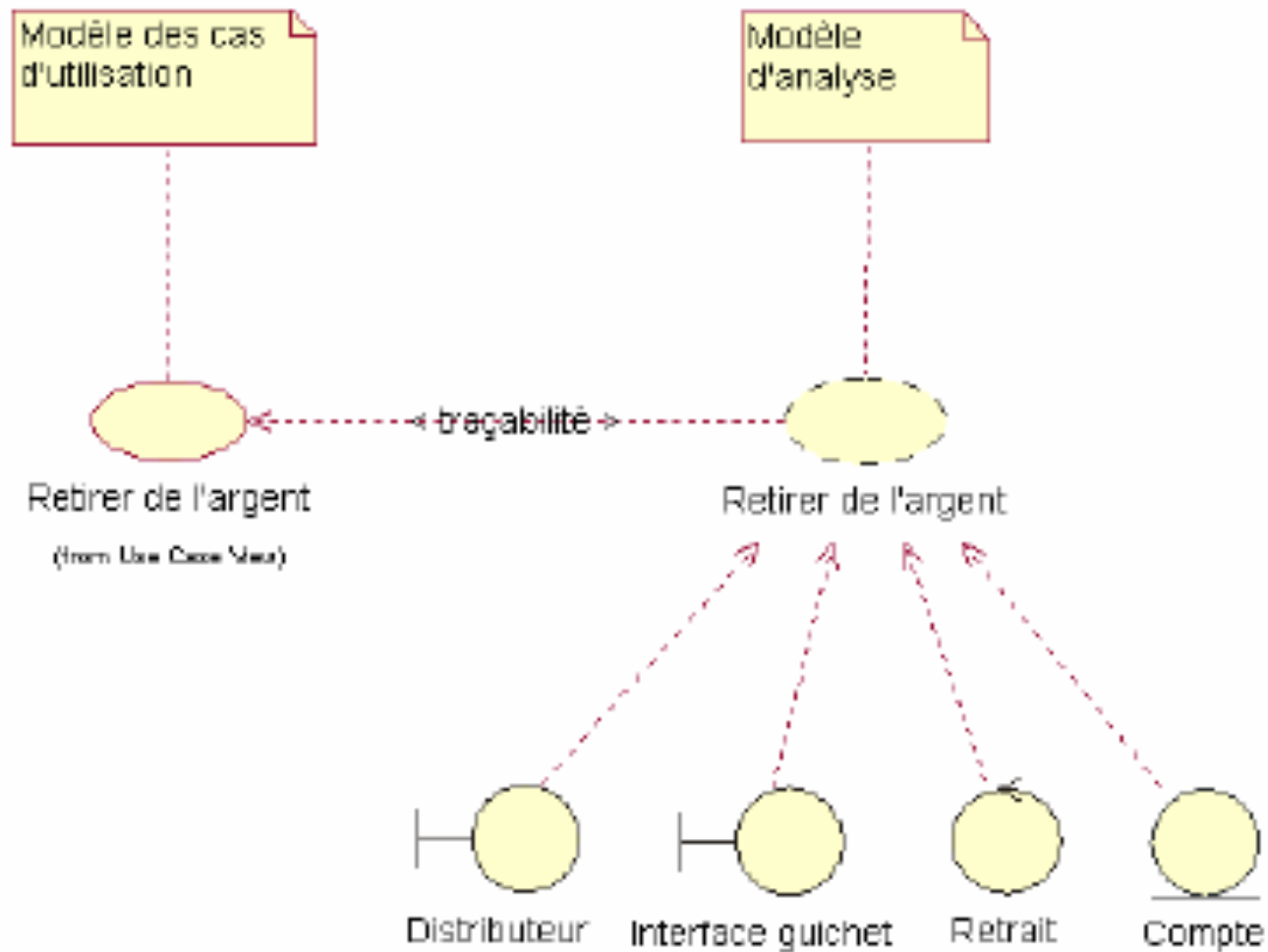
Enchaînement des activités de l'analyse



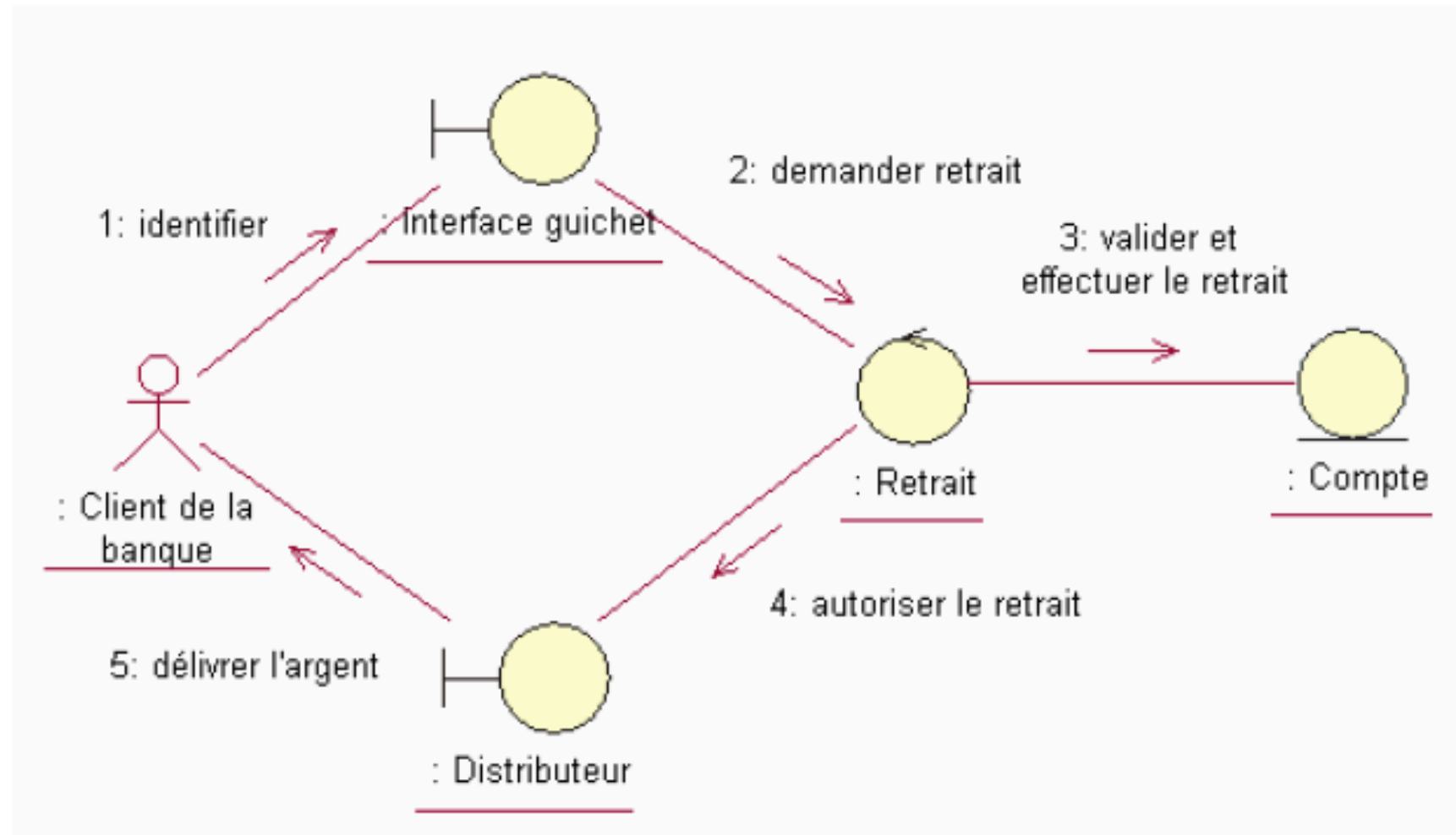
Analyse de use-case

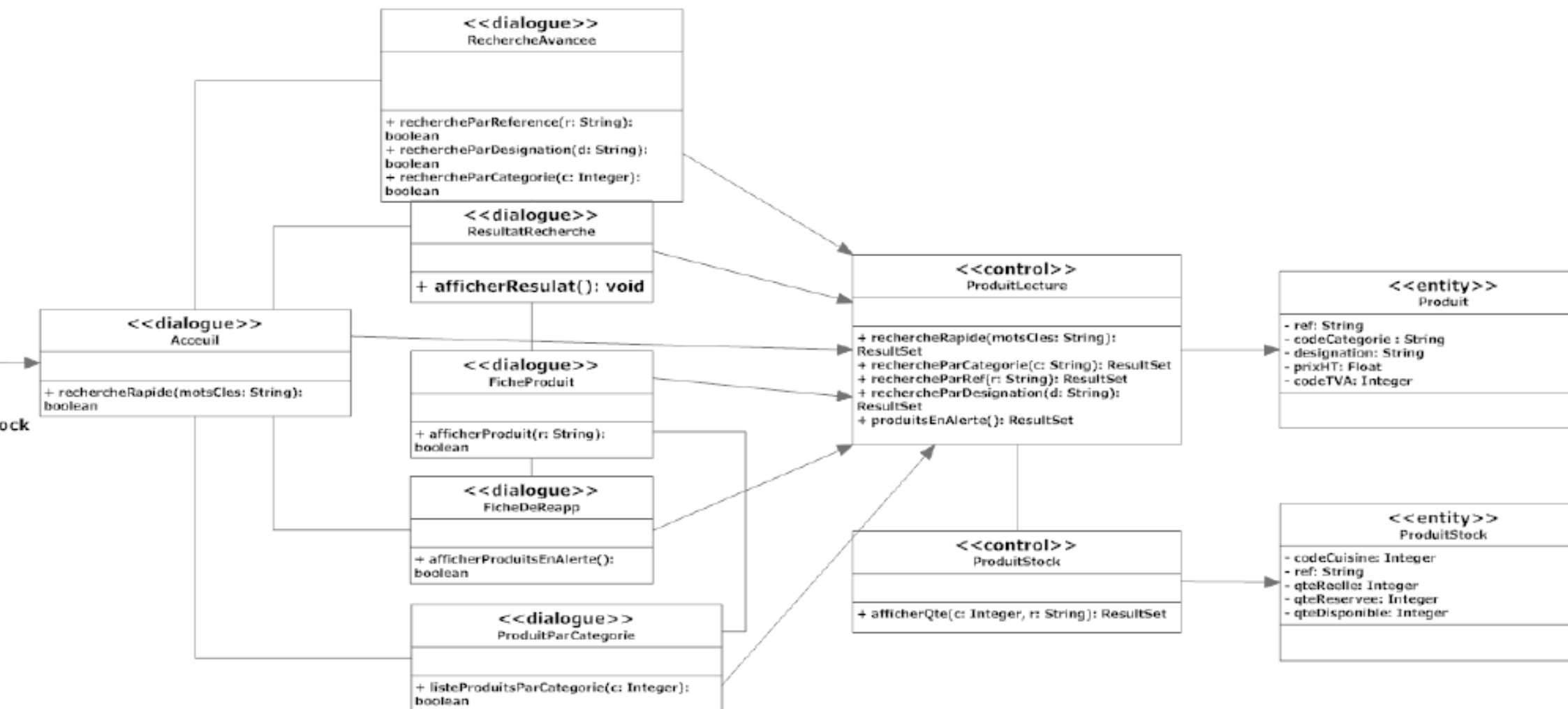
- ▶ Cette activité fait suite à l'analyse architecturale, elle est effectuée par un ingénieur de use-case.
- ▶ Cela consiste à identifier les classes d'analyse et les interactions nécessaires à la réalisation d'un use-case.
 - ▶ De nouvelles classes d'analyse peuvent être identifiées, ou bien on peut donner de nouveaux éléments de définition de classes déjà identifiées.
 - ▶ Les interactions peuvent être décrites par des diagrammes de description dynamique.

Analyse de use-case

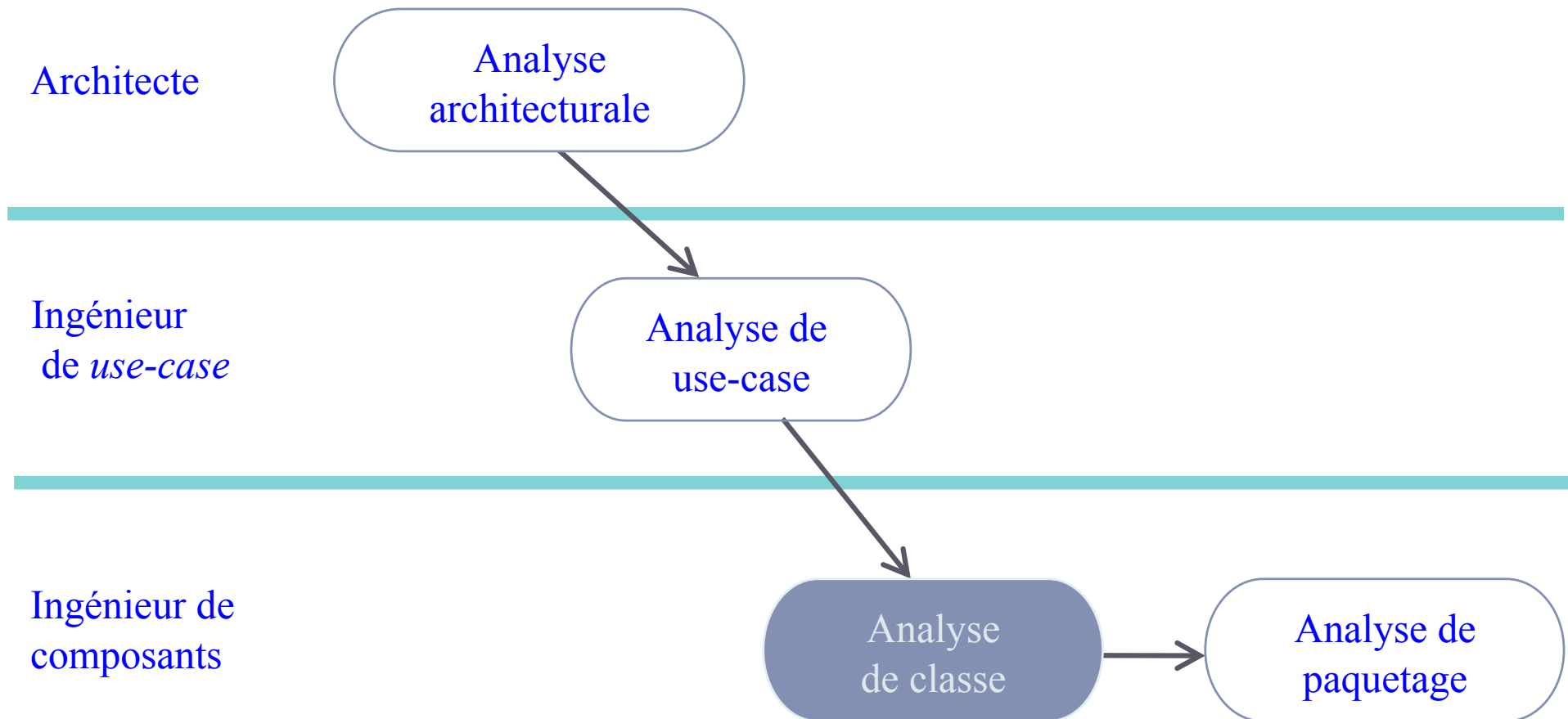


Analyse de use-case





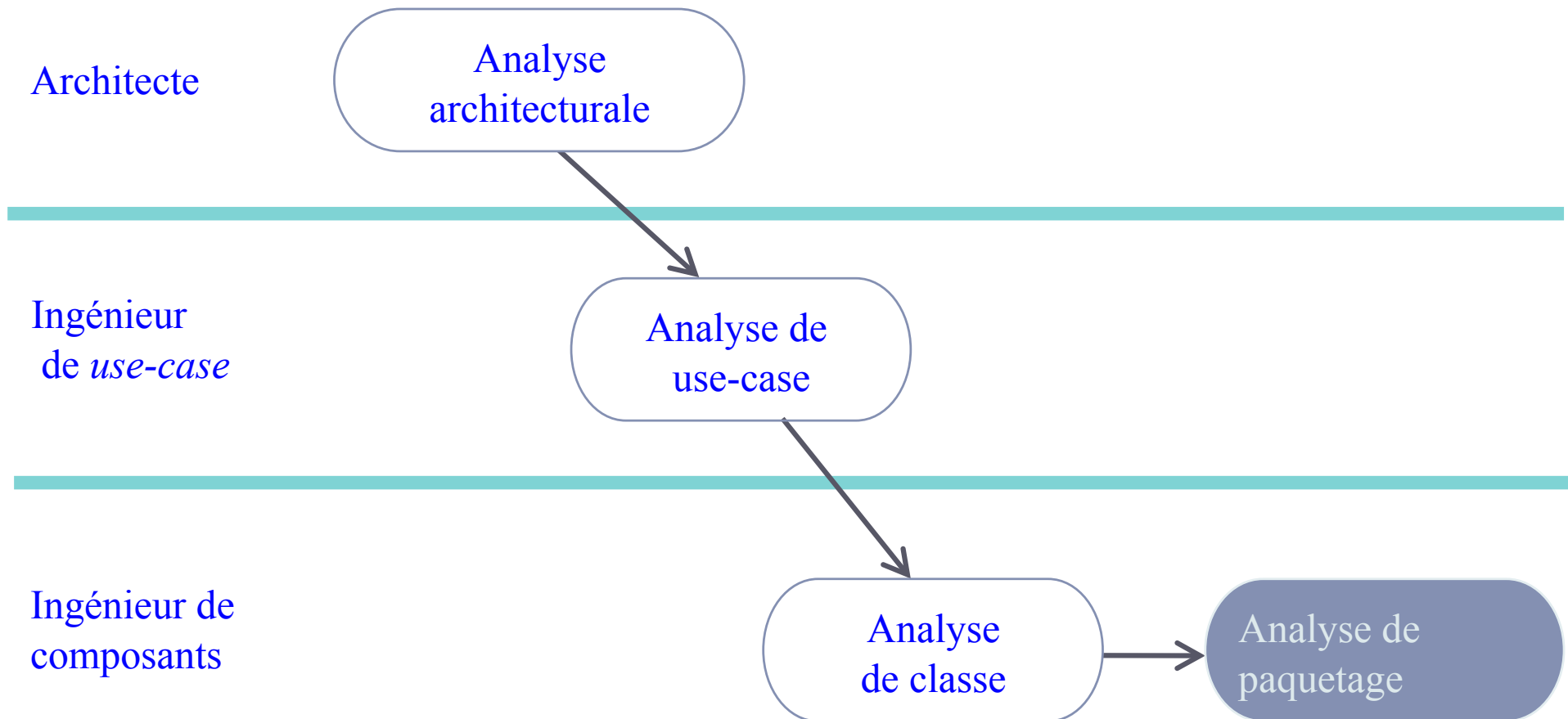
Enchaînement des activités de l'analyse



Analyse de classe

- ▶ Cette activité succède aux analyses de use-cases, elle est effectuée par un ingénieur de composants.
- ▶ Elle consiste à :
 - ▶ identifier les responsabilités d'une classe à partir de la synthèse des différents rôles qu'elle joue dans les use-cases (opérations),
 - ▶ identifier les attributs et les relations entre classes (associations, généralisation),
 - ▶ formuler éventuellement des exigences sur la conception (taille des attributs, ...).

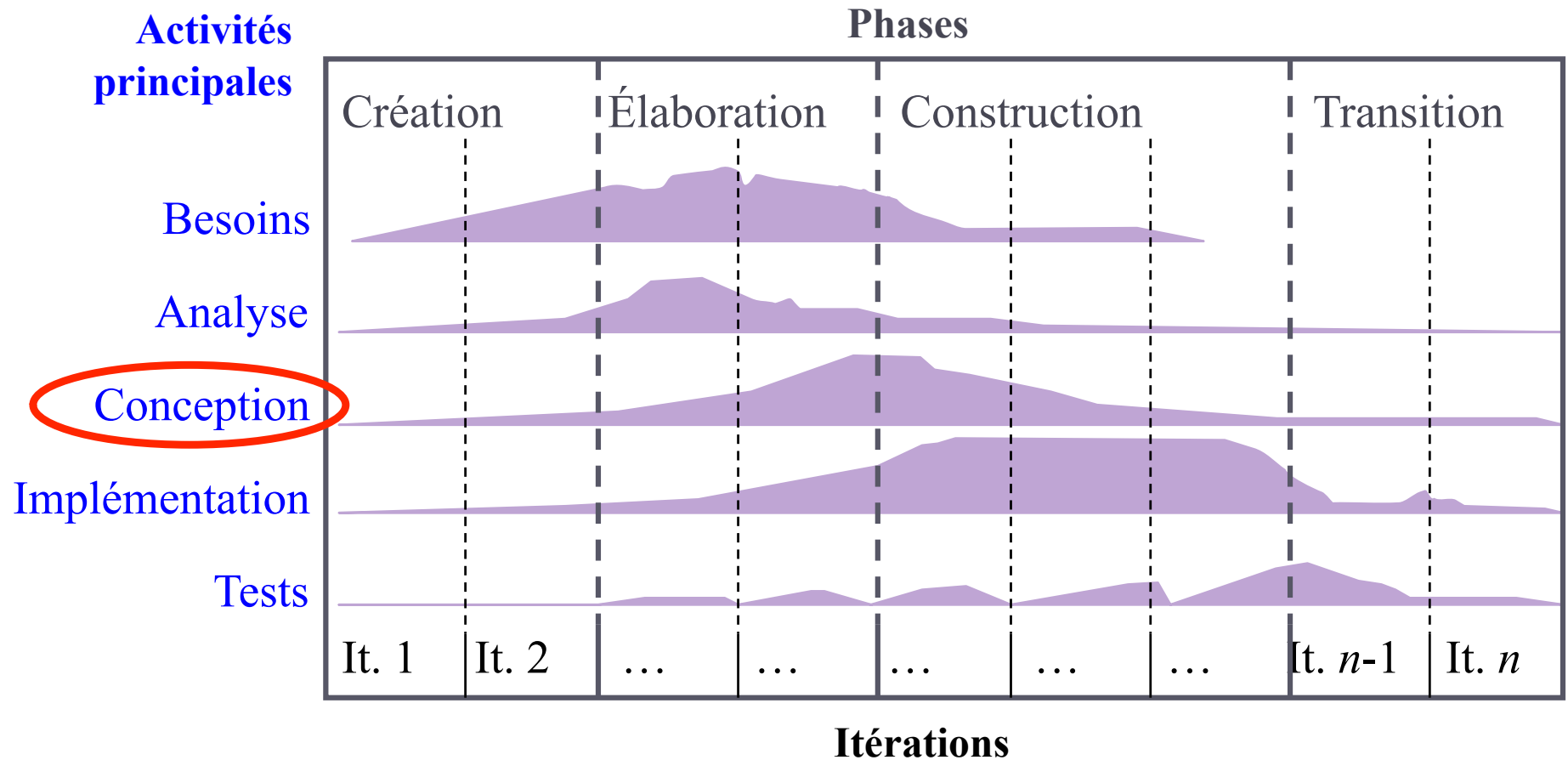
Enchaînement des activités de l'analyse



Analyse de paquetage

- ▶ Cette activité fait suite à l'analyse des classes, toutes deux sont effectuées par un ingénieur de composants.
- ▶ Cela consiste à améliorer la répartition des classes d'analyse en paquetages, ébauchées pendant l'analyse architecturale. Il faut :
 - ▶ garantir la cohésion des paquetages : des éléments fortement liés (forte cohésion)
 - ▶ décrire les dépendances entre paquetages et chercher à les diminuer (couplage faible)

Organisation en phases, itérations et activités



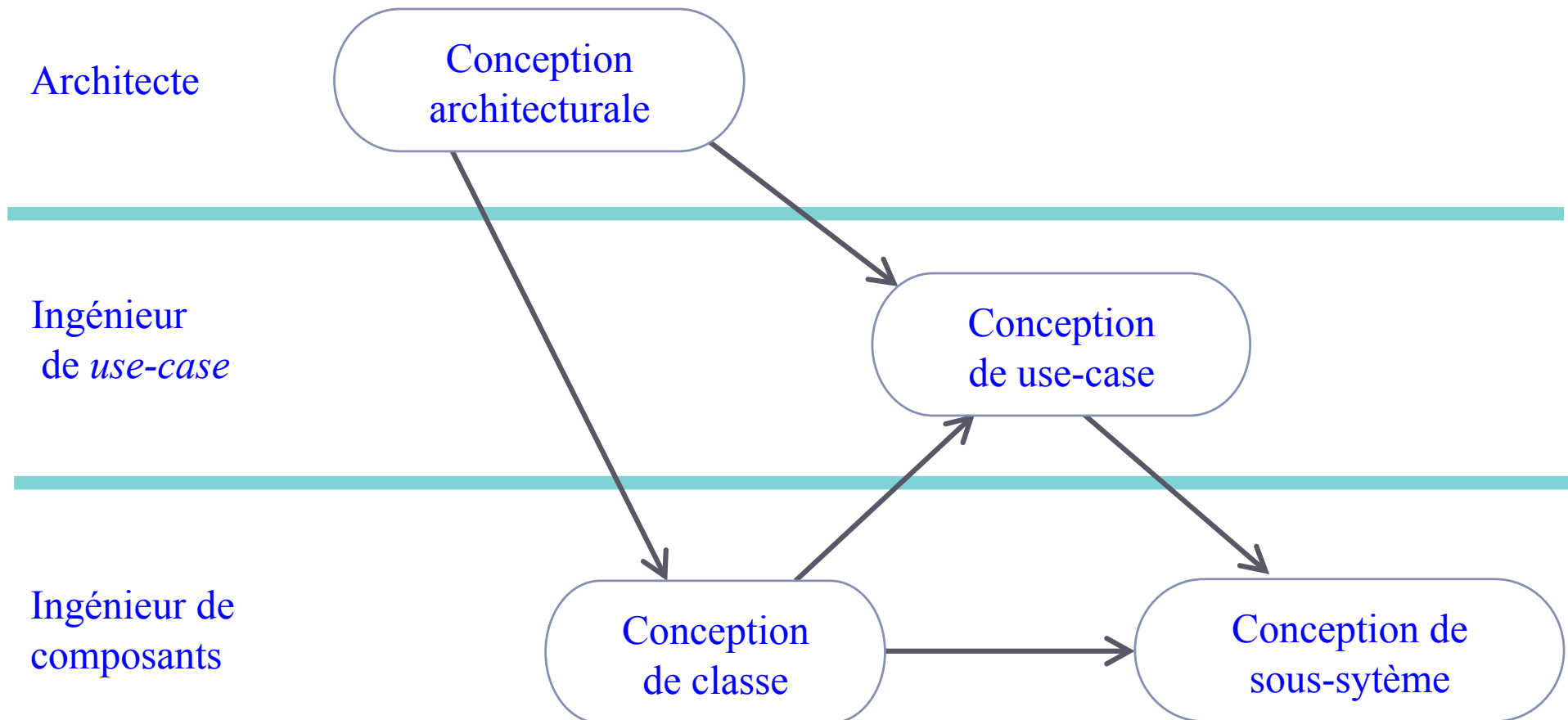
Répartition du travail en phases, itérations et activités principales

Activités - Conception

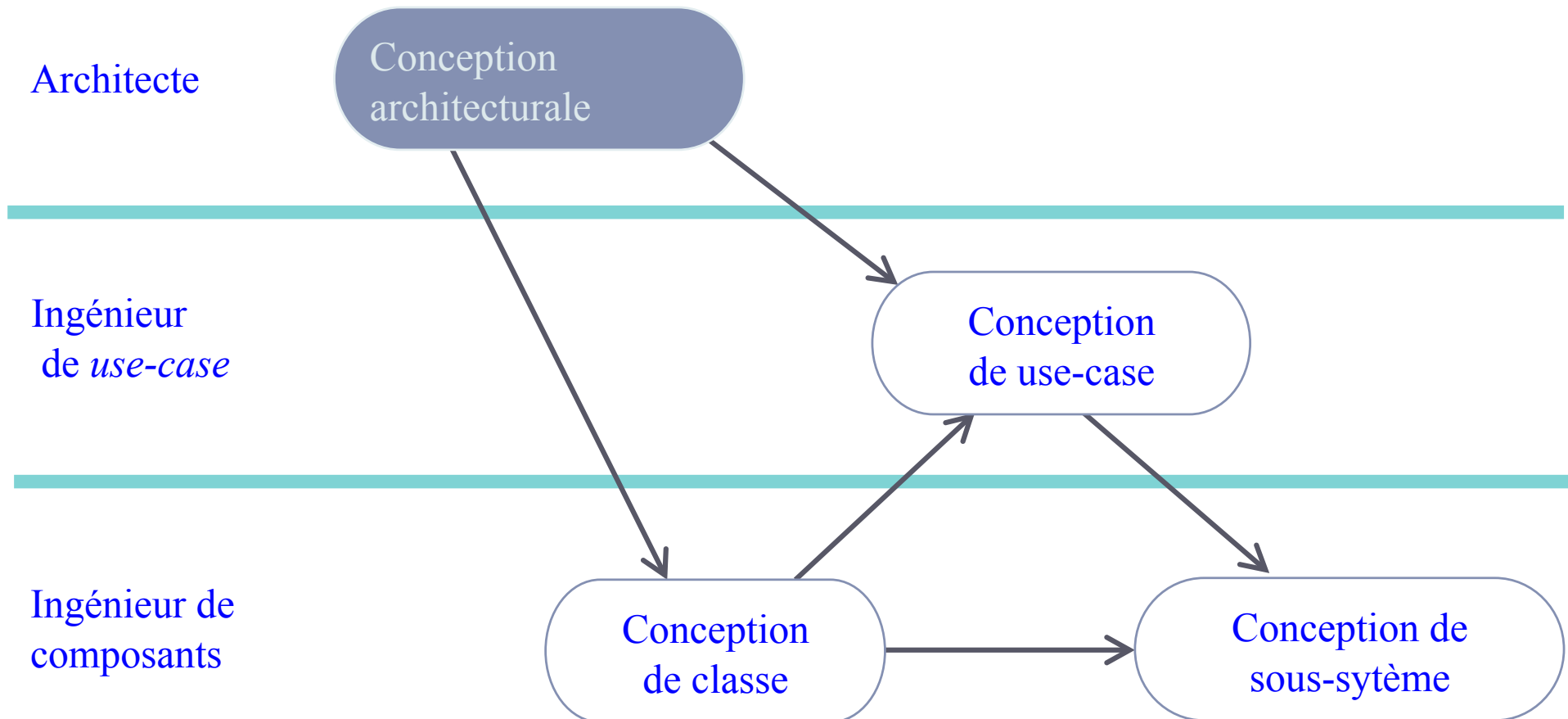
Activités de conception

- ▶ La conception regroupe les activités suivantes :
 - ▶ conception architecturale,
 - ▶ conception de classe,
 - ▶ conception de use-case,
 - ▶ conception de sous-système.

Enchaînement des activités de conception



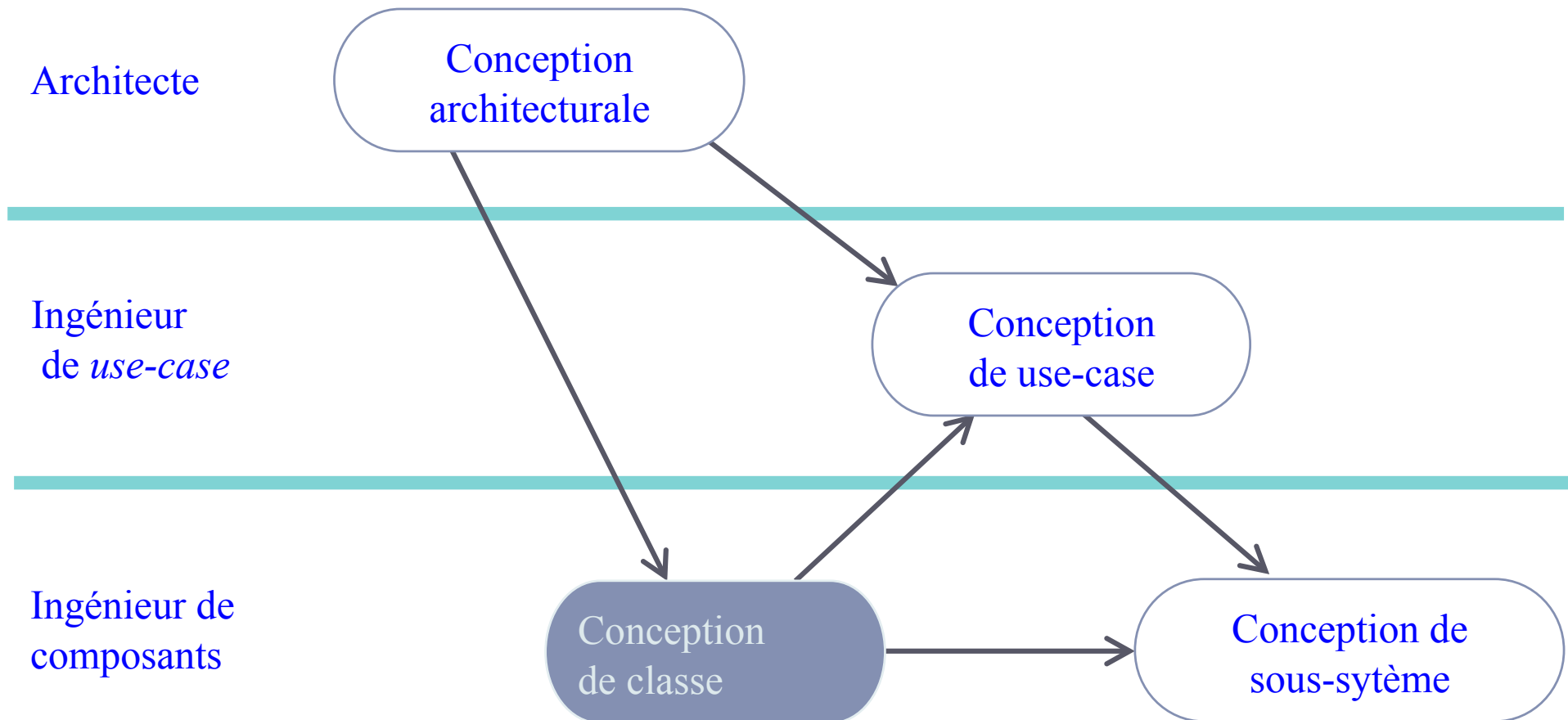
Enchaînement des activités de conception



Conception architecturale

- ▶ L'architecte assume la responsabilité de cette activité.
- ▶ À partir notamment du modèle des use-cases, du modèle d'analyse et de la première description de l'architecture issue de l'analyse, il doit en particulier produire :
 - ▶ une ébauche des sous-systèmes et de leur interface
 - ▶ une ébauche des classes et mécanismes de conception,
 - ▶ une ébauche du modèle de déploiement (nœuds et configurations réseau).

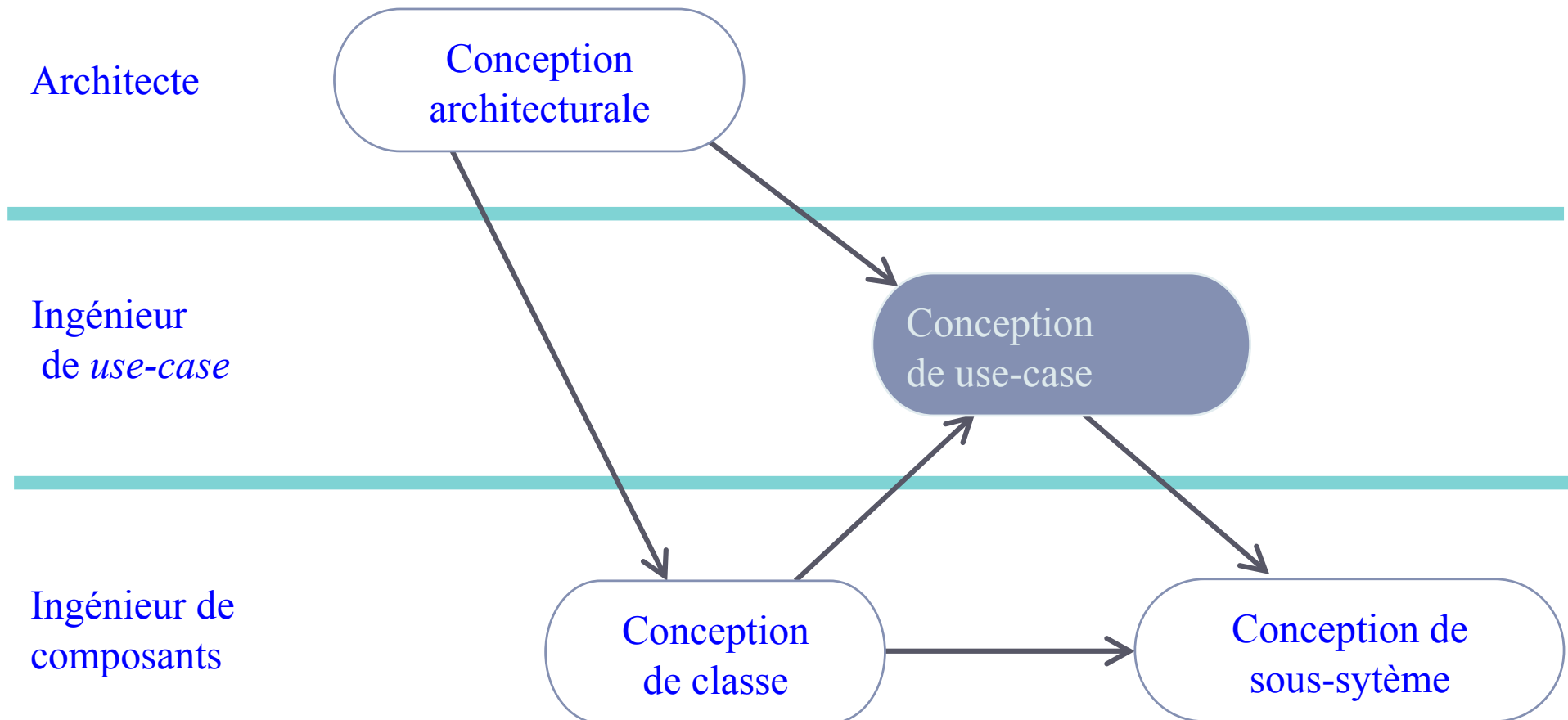
Enchaînement des activités de conception



Conception de classe

- ▶ Cette activité fait suite à la conception architecturale, elle est effectuée par les ingénieurs de composants.
- ▶ Cette activité consiste à :
 - ▶ décrire précisément les opérations et les attributs des classes de conception,
 - ▶ décrire les relations auxquelles elle prend part,
 - ▶ décrire ses états imposés (diagramme d'états),
 - ▶ expliciter ses méthodes (qui réalisent les opérations) et la réalisation correcte de ses interfaces,
 - ▶ exprimer les exigences sur son implémentation.

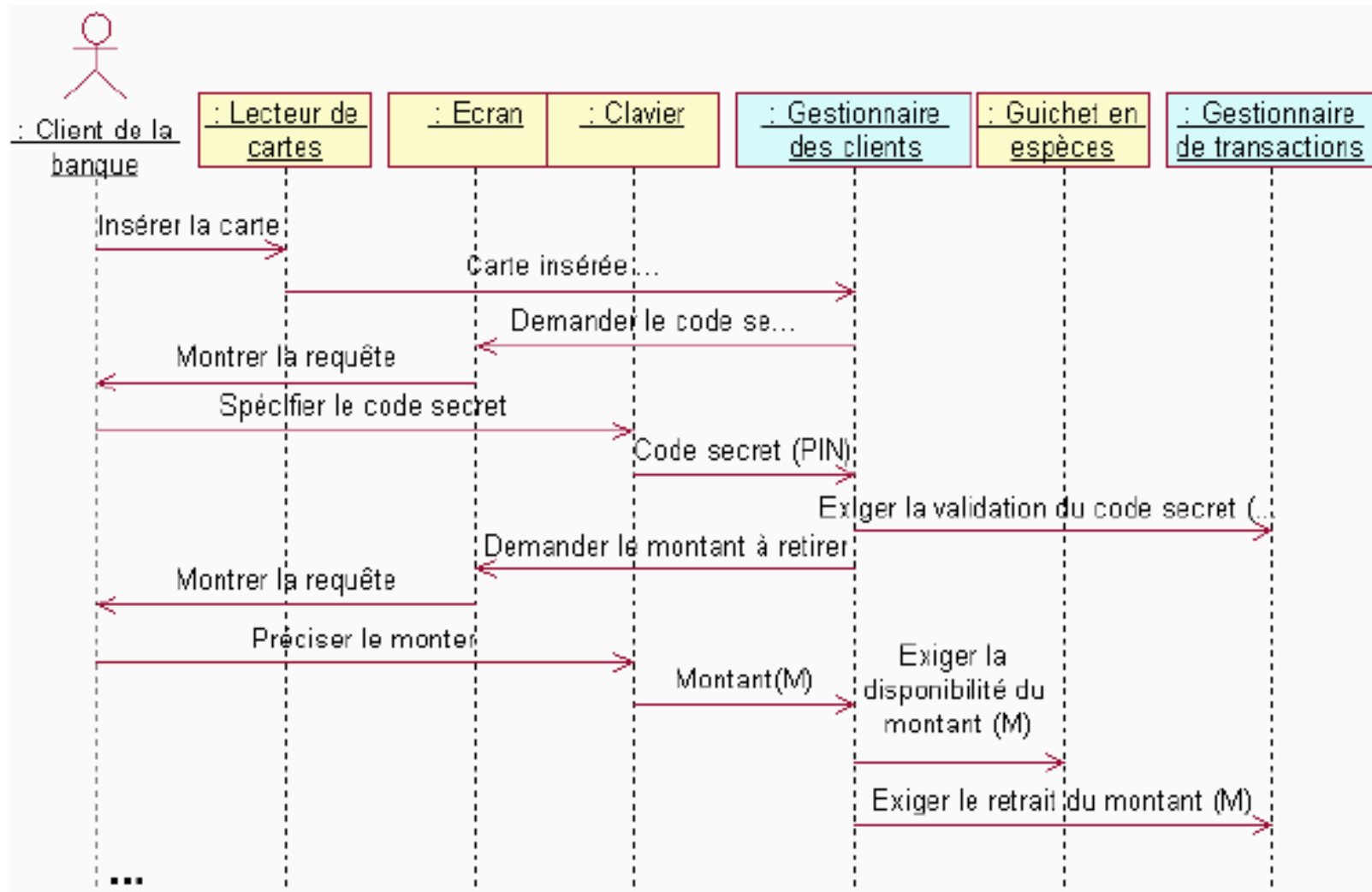
Enchaînement des activités de conception



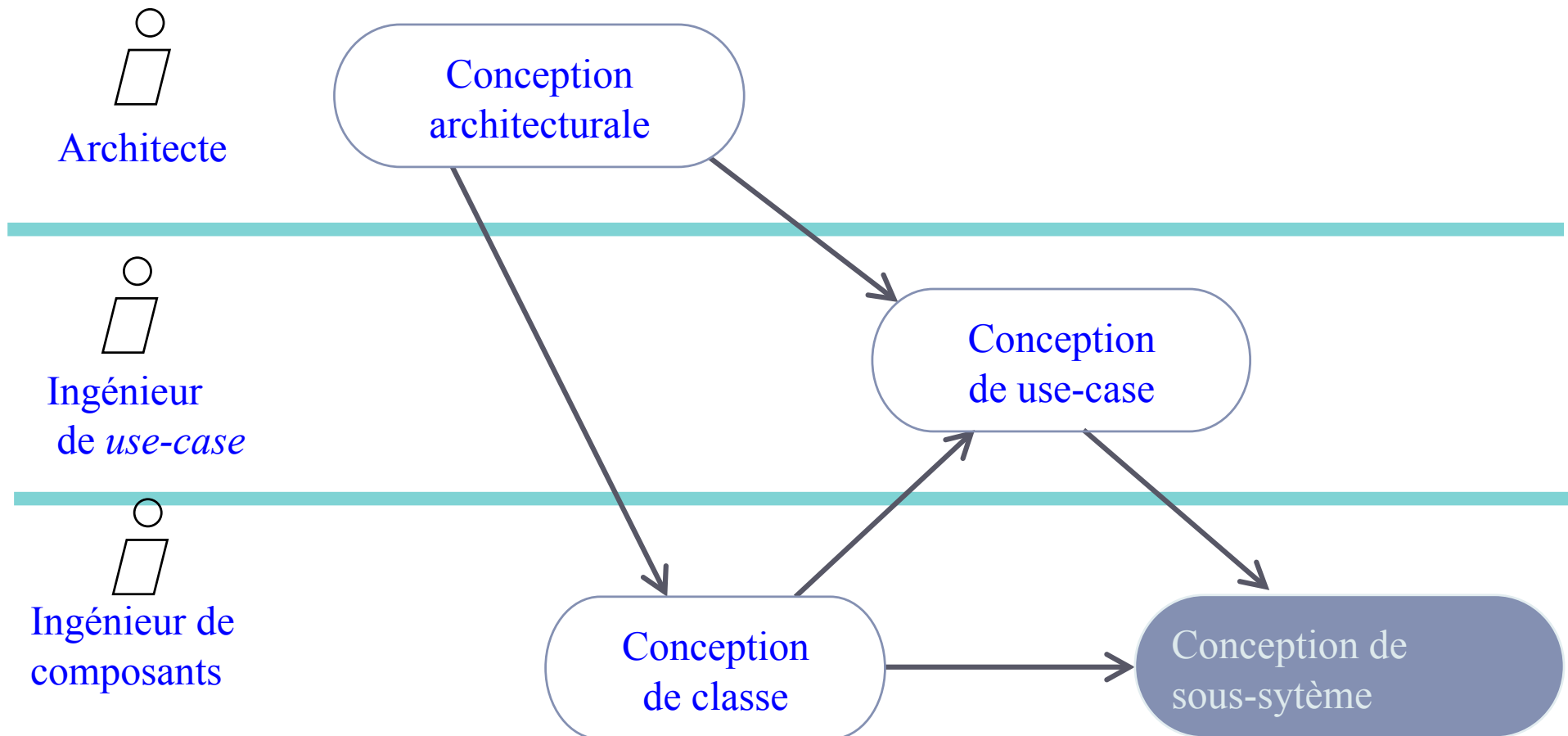
Conception de use-case

- ▶ Cette activité fait suite à la conception architecturale et à la conception des classes de conception, elle est effectuée par les ingénieurs de use-case.
- ▶ Cette activité consiste à :
 - ▶ reconnaître les classes de conception prenant part à la réalisation d'un use-case, de nouvelles classes de conception peuvent être éventuellement identifiées,
 - ▶ décrire les interactions entre classes en termes de messages,
 - ▶ définir les exigences sur les opérations et l'implémentation,
 - ▶ contrôler et enrichir la description des sous-systèmes et de leur interface.

Conception de use-case



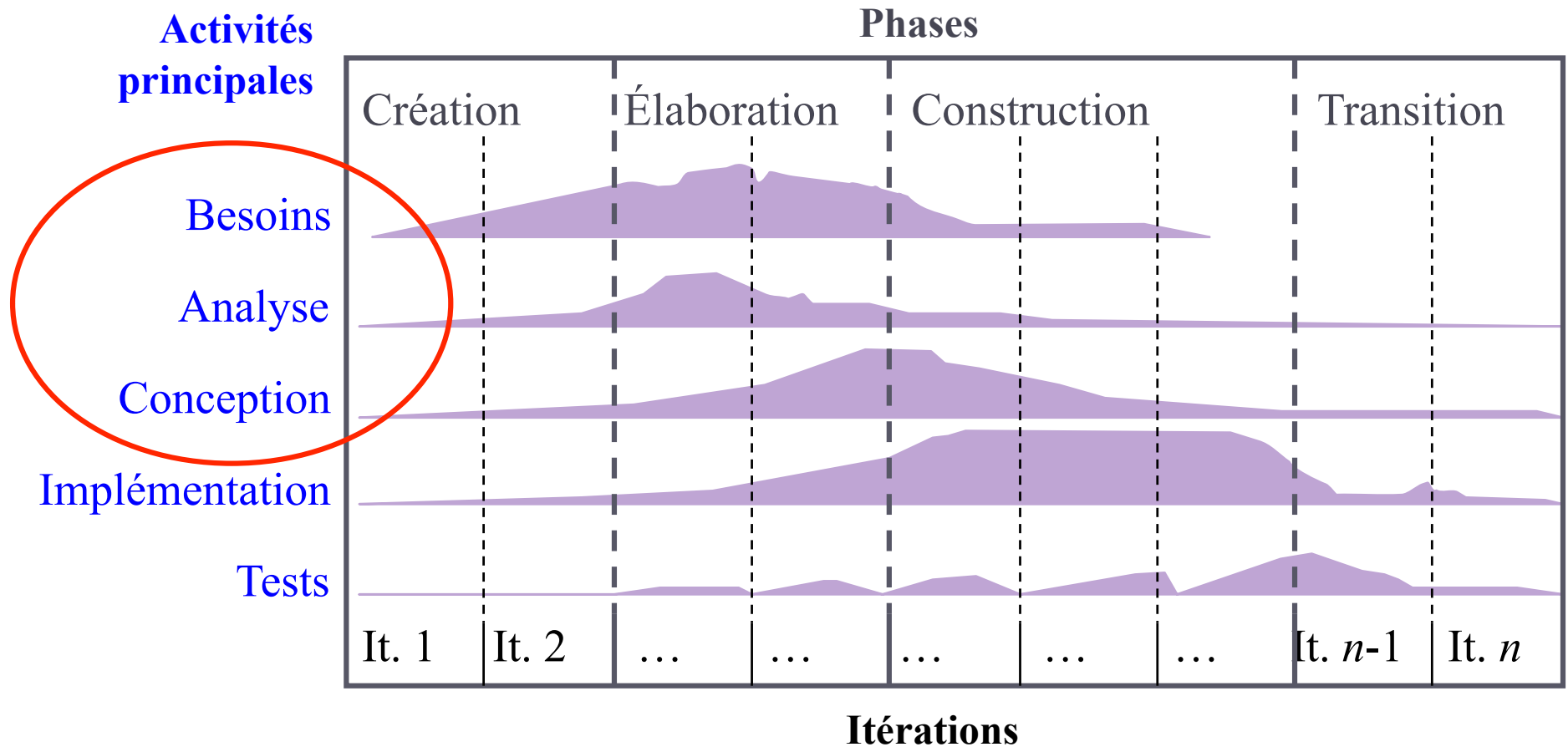
Enchaînement des activités de conception



Conception de sous-système

- ▶ Cette activité fait suite à la conception des classes et des use-cases, elle est effectuée par les ingénieurs de composants.
- ▶ Cette activité consiste à finaliser la description des sous-systèmes et de leur interface, c'est-à-dire à :
 - ▶ actualiser les dépendances entre sous-systèmes (et les minimiser),
 - ▶ actualiser leur interface,
 - ▶ actualiser le contenu des sous-systèmes.

Organisation en phases, itérations et activités



Répartition du travail en phases, itérations et activités principales

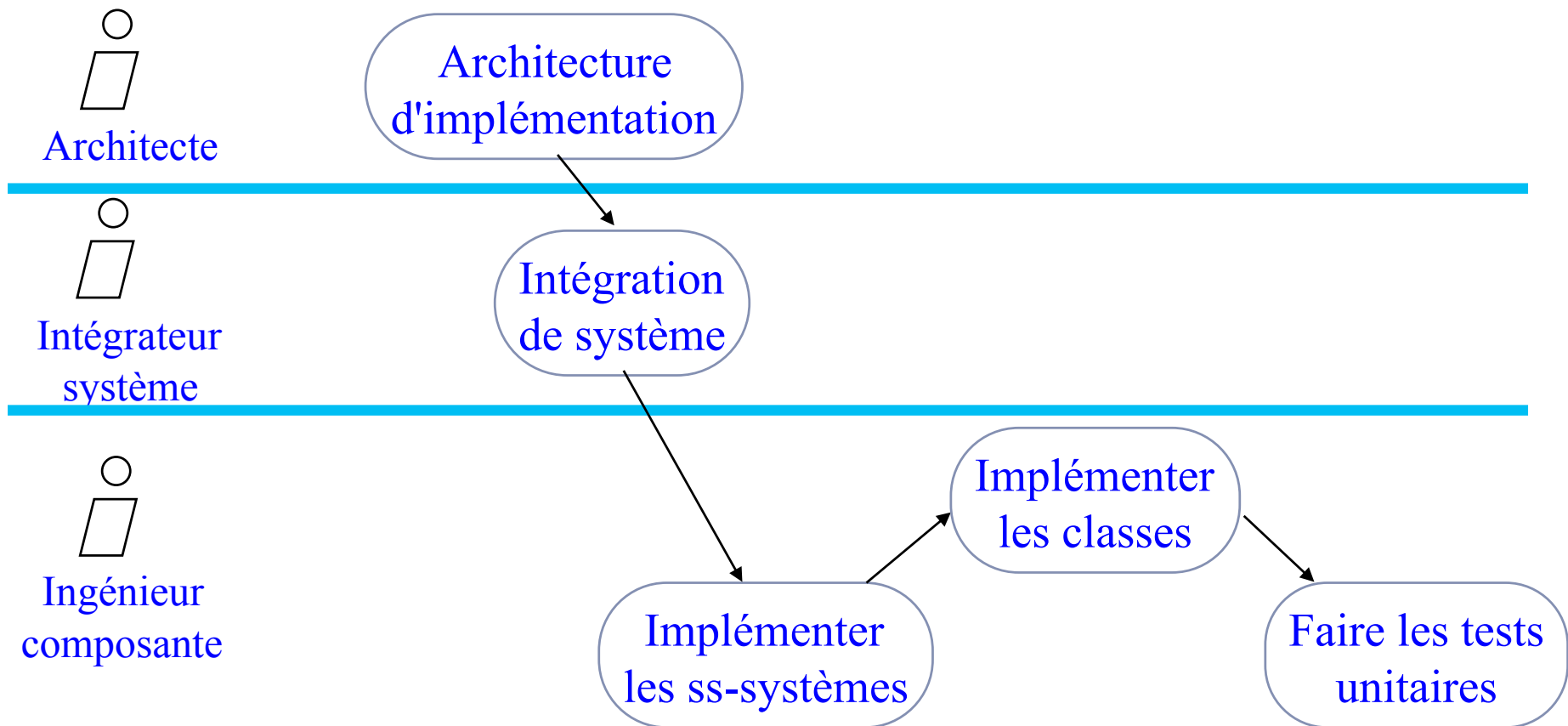
Activités - Implémentation

Implémentation

- ▶ Cette activité produit les artefacts suivant :
 - ▶ Architecture d'implémentation
 - ▶ Composantes organisées en sous-systèmes
 - ▶ Plan d'intégration
- ▶ Elle comporte les sous activités suivantes :
 - ▶ Architecture d'implémentation
 - ▶ Intégration de système
 - ▶ Implémenter les ss-systèmes
 - ▶ Implémenter les classes
 - ▶ Faire les tests unitaires

Implémentation

Le processus

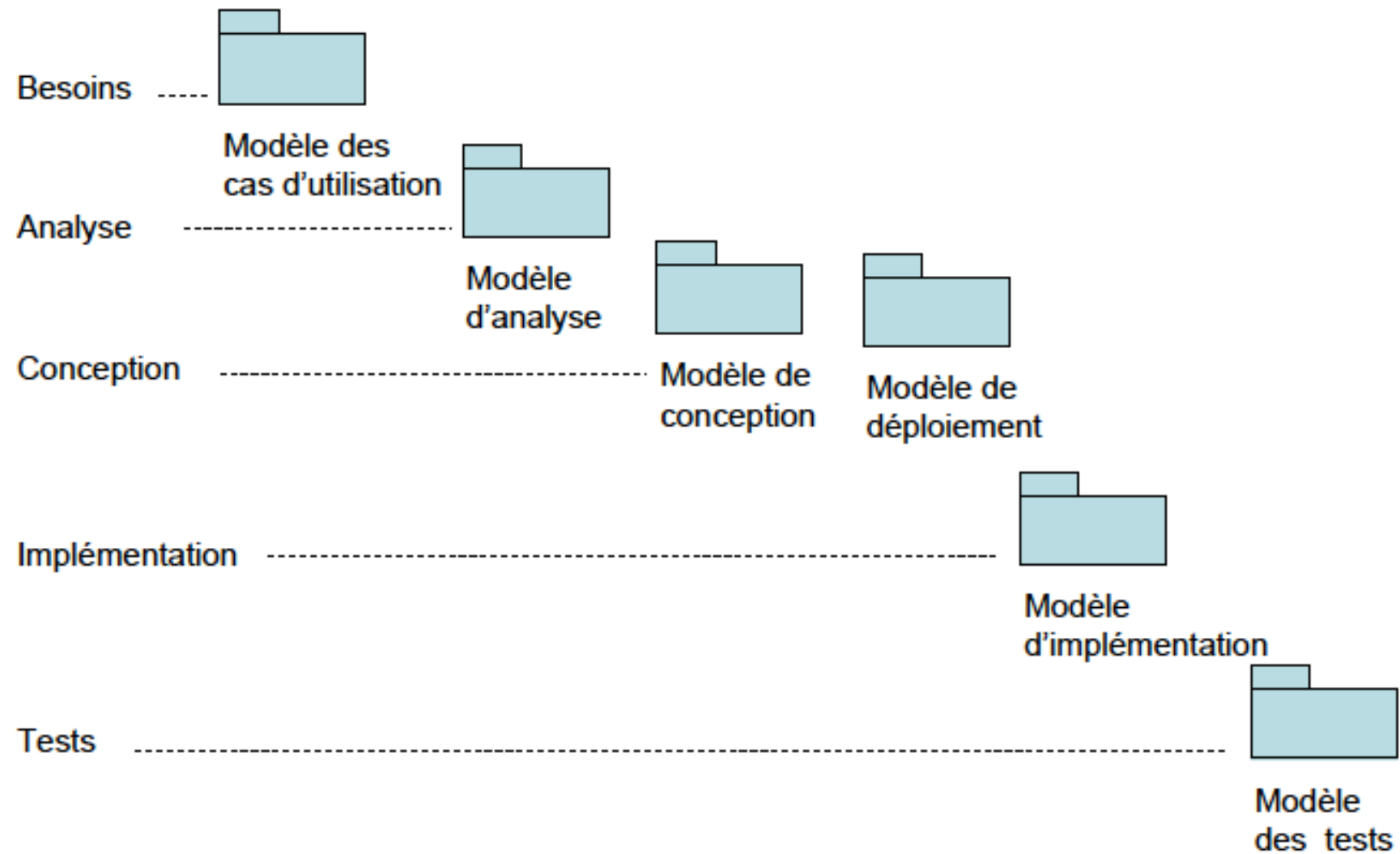


Activités - Test

Test

- ▶ Les produits de cette activité sont :
 - ▶ Plan de tests
 - ▶ Cas de test
 - ▶ Procédures de test
 - ▶ Composantes de test

UP au bilan ...



Résumé: UP (Processus unifié)

- ▶ Enraciné dans:
 - ▶ La méthode de Booch (bonne pour la conception et l'implémentation)
 - ▶ OMT (bonne pour l'analyse)
 - ▶ Objectory (bonne pour l'ingénierie des besoins et l'architecture du système)
- ▶ UP a rassemblé ces méthodes
- ▶ Aussi volumineux et complexe : temps d'apprentissage long, ou bien l'adapter au besoin