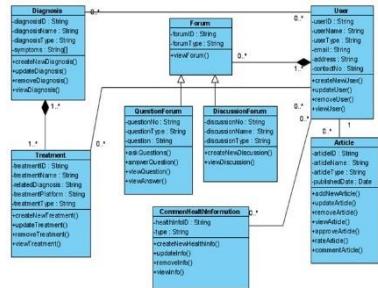


Génie Logiciel

Chapitre 3 : Processus Unifié



CHIKHI Nacim Fateh



Processus de développement logiciel

- ❖ Un processus définit une séquence d'étapes, en partie ordonnées, qui concourent à l'obtention d'un système logiciel ou à l'évolution d'un système existant.
- ❖ L'objet d'un processus de développement logiciel est de produire des logiciels de qualité qui répondent aux besoins de leurs utilisateurs dans des temps et des coûts prévisibles.
- ❖ Après le développement d'une notation unifiée de modélisation orientée objet (UML), est venu le besoin d'un processus de développement logiciel unifié.

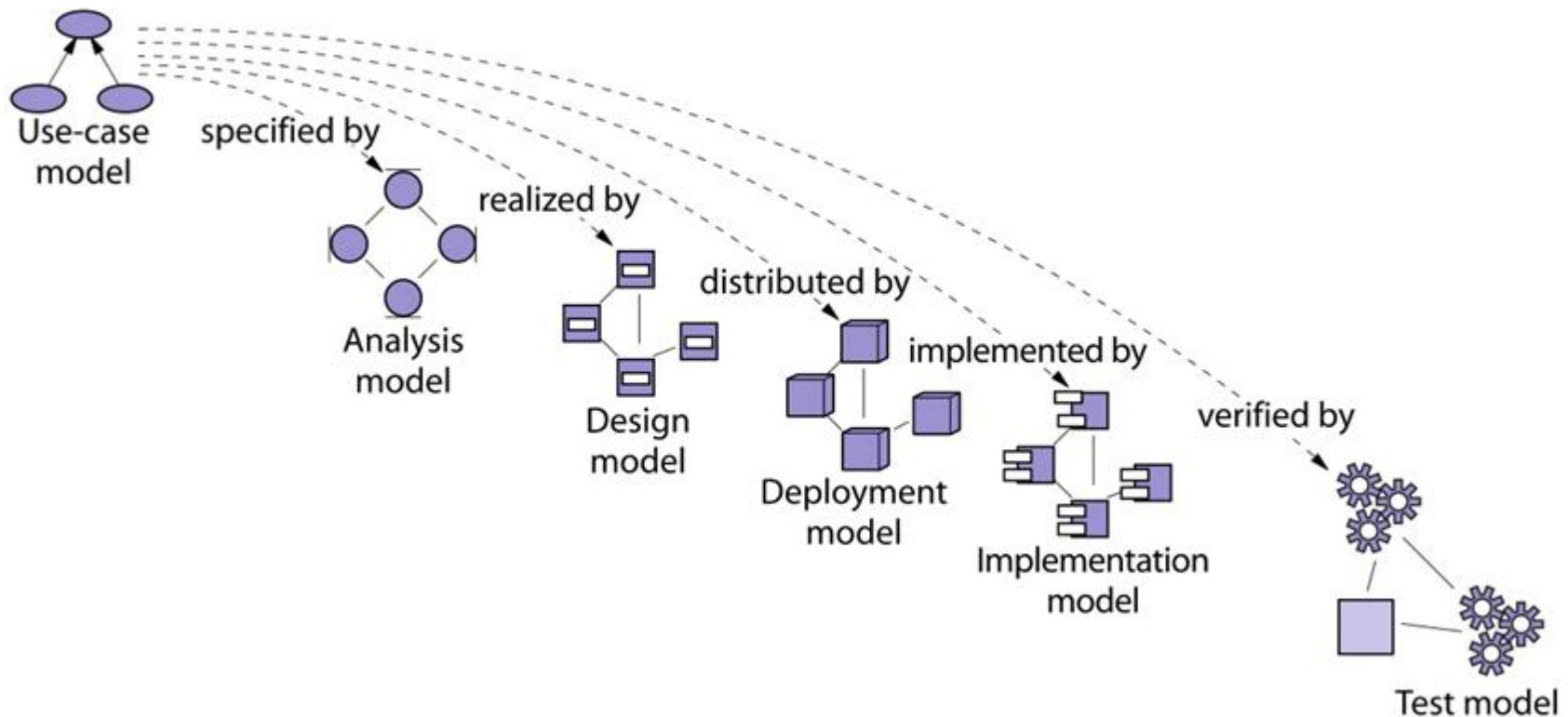
Qu'est-ce-que UP ?

- ❖ Unified Process ou Processus Unifié est une famille de méthodes de développement de logiciels orientés objets.
- ❖ UP est apparu au milieu des années 90 suite à la fusion entre la méthode de Rational Software et la méthode de Objectory AB.
- ❖ Contrairement à UML qui est standardisé par l'OMG, UP ne l'est pas.
- ❖ UP est un processus générique. Plusieurs variantes existent :
 - ❖ RUP : Rational Unified Process
 - ❖ EUP : Enterprise Unified Process
 - ❖ AUP : Agile Unified Process
 - ❖ 2TUP : Two Tracks Unified Process

Caractéristiques de UP

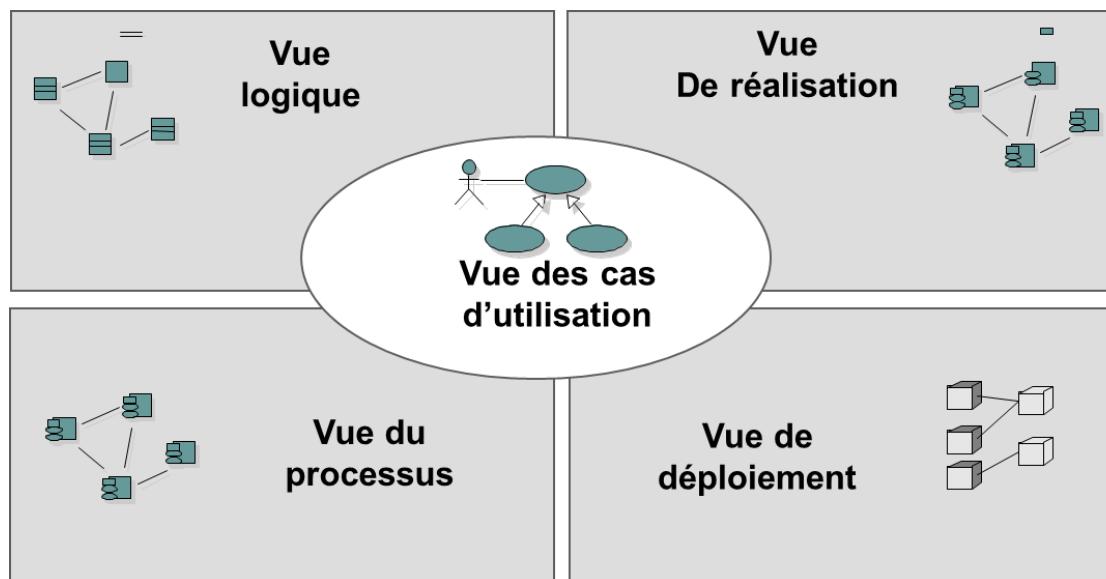
- ❖ UP est itératif et incrémental : Le projet est découpé en itérations ou étapes de courte durée qui permettent de mieux suivre l'avancement global. A la fin de chaque itération une partie exécutable du système final est produite de façon incrémentale. L'approche itérative permet aussi de mieux gérer la complexité du développement logiciel.
- ❖ UP est guidé par les cas d'utilisation : Le but principal d'un système d'information est de répondre aux besoins du client. Le processus de développement sera donc orienté utilisateur car la spécification et la conception sont construites à partir des modes d'utilisation attendus par les acteurs du système.

Caractéristiques de UP (suite)



Caractéristiques de UP (suite)

- ❖ UP suit une démarche centrée sur l'architecture : Tout système complexe doit être décomposé en parties modulaires afin d'en faciliter la maintenance et l'évolution. Cette architecture (fonctionnelle, logique, matérielle, etc.) doit être modélisée en UML, et pas seulement documentée en texte.

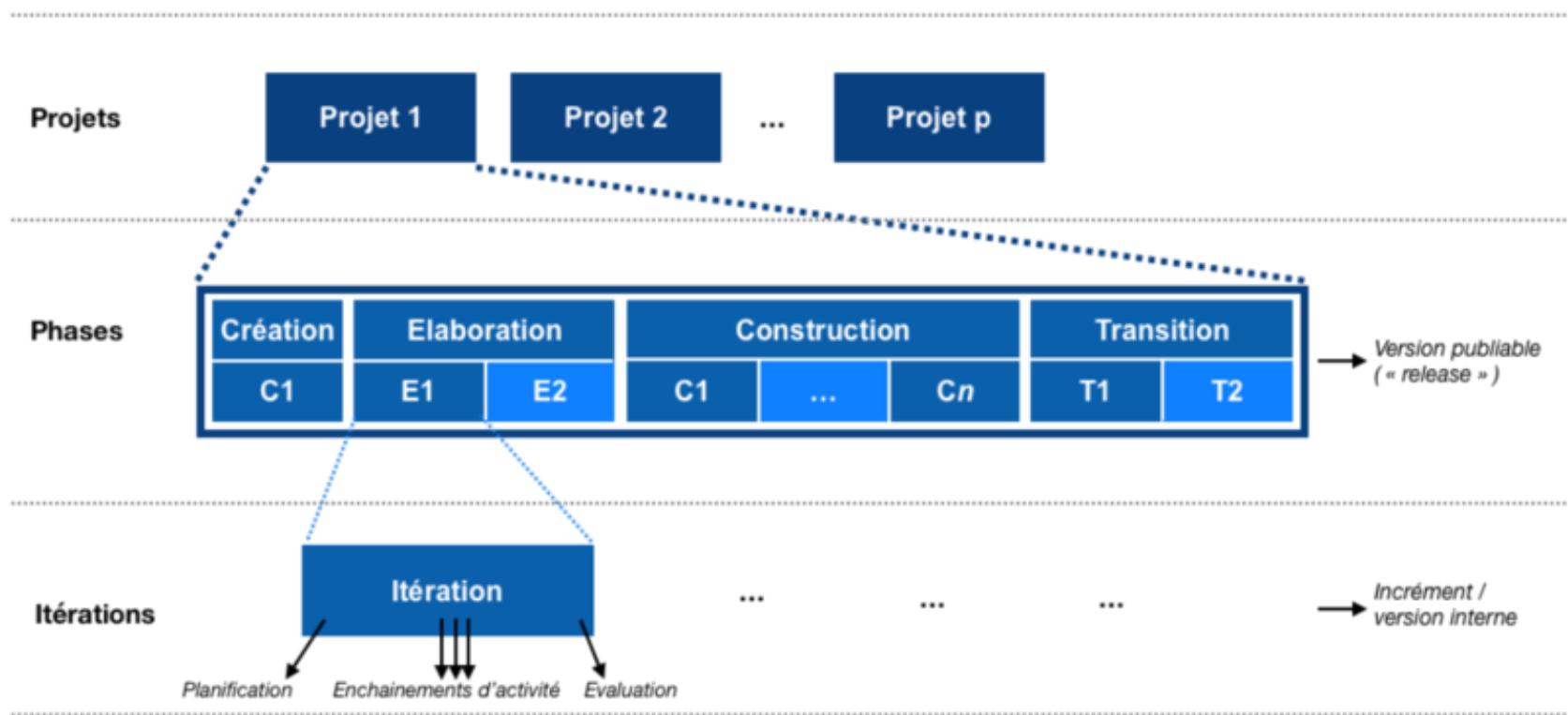


Caractéristiques de UP (suite)

- ❖ UP est piloté par les risques : Les risques majeurs du projet doivent être identifiés au plus tôt mais surtout levés le plus rapidement. Les mesures à prendre dans ce cadre déterminent l'ordre des itérations.

- ❖ Exemples de risques : L'incapacité de l'architecture technique à répondre aux contraintes opérationnelles, ou encore l'inadéquation du développement aux besoins des utilisateurs.

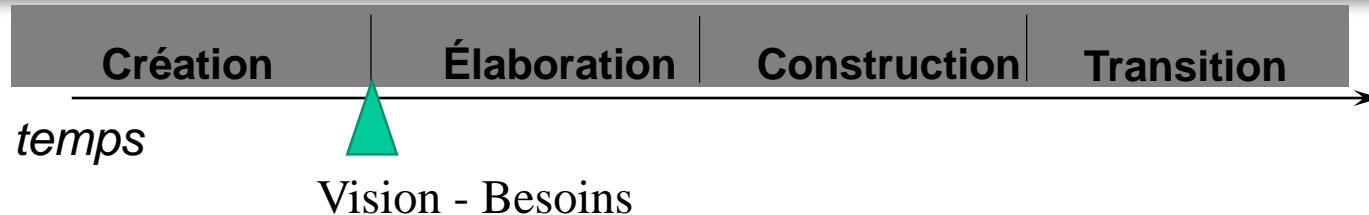
UP : le cycle de vie



UP : les phases

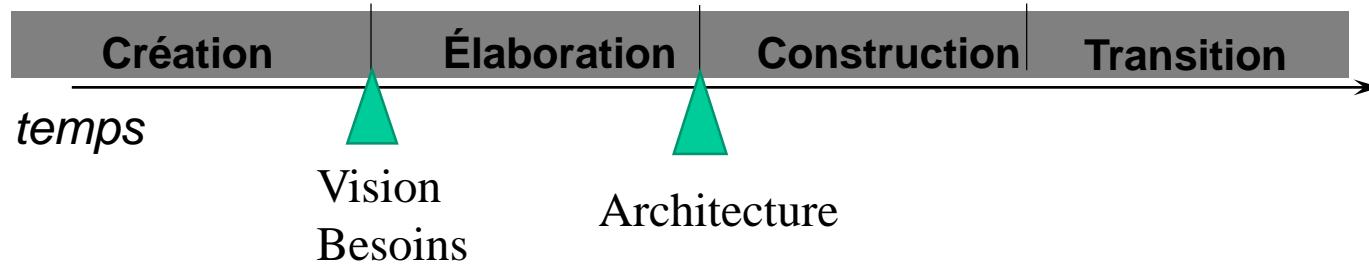
- ❖ La phase de **création** vise à définir le produit et les objectifs du projet.
- ❖ La phase d'**élaboration** vise à clarifier les exigences, à définir l'architecture du produit et à en valider la faisabilité.
- ❖ La phase de **construction** vise à construire et à mettre en œuvre le produit et les livrables associés
- ❖ La phase de **transition** vise à livrer, diffuser ou déployer le produit de sorte qu'il soit prêt à être utilisé.

Phase de création



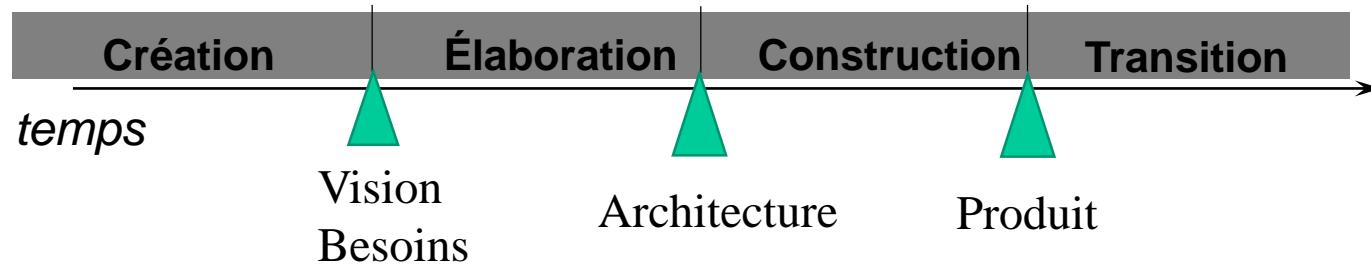
- Objectifs
 - Définir les limites du système
 - Identifier les usages
 - cas d'utilisation
 - interfaces utilisateurs
 - Esquisser une architecture initiale
 - Identifier les risques principaux
- Résultats
 - **Vision** : glossaire, liste des acteurs, liste des cas d'utilisation classée, **10 %** des cas d'utilisation représentatifs documentés, contraintes non fonctionnelles, liste des risques principaux, premier diagramme de classes, interfaces utilisateurs
 - **Feu vert du commanditaire**

Phase d'élaboration



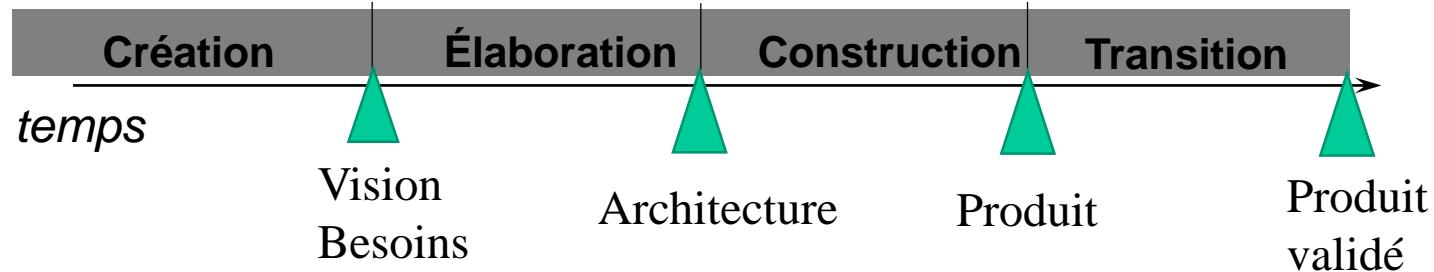
- Objectifs
 - Définir l'architecture de référence
 - Préciser les cas d'utilisation (**80 %** des besoins fonctionnels)
 - Planifier le projet
 - Préciser les risques du projet (besoins, technologiques, compétences, politiques)
- Résultats
 - **Architecture de référence** : concerne les cas d'utilisation centraux avec l'ensemble de leurs modèles d'analyse
 - **Un planning du projet** intégrant le calendrier, les coûts, le personnel
 - **Une évaluation des risques** et des éléments de protection

Phase de construction



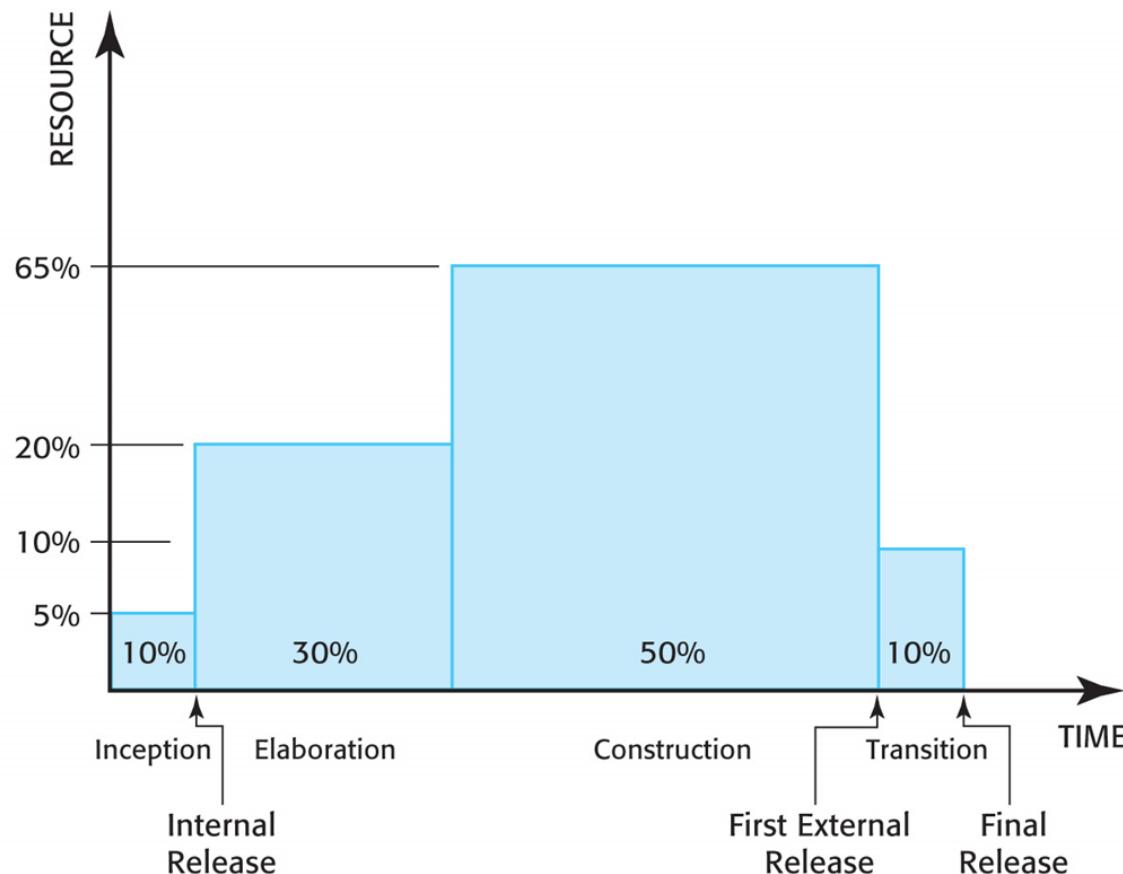
- Objectifs
 - Finalisation de la description et de la réalisation des cas d'utilisation
 - Finalisation de l'analyse, de la conception, de l'implantation et des tests
 - Validation par rapport aux risques
 - Validation par rapport à l'architecture de référence
- Résultats
 - **Produit version Béta**, logiciel testé en interne, manuels

Phase de transition



- Objectifs
 - Correction des derniers bugs liés au béta test
 - Validation par rapport aux besoins du clients
 - Logiciel
 - Manuels
- Résultats
 - **Produit validé** par les utilisateurs

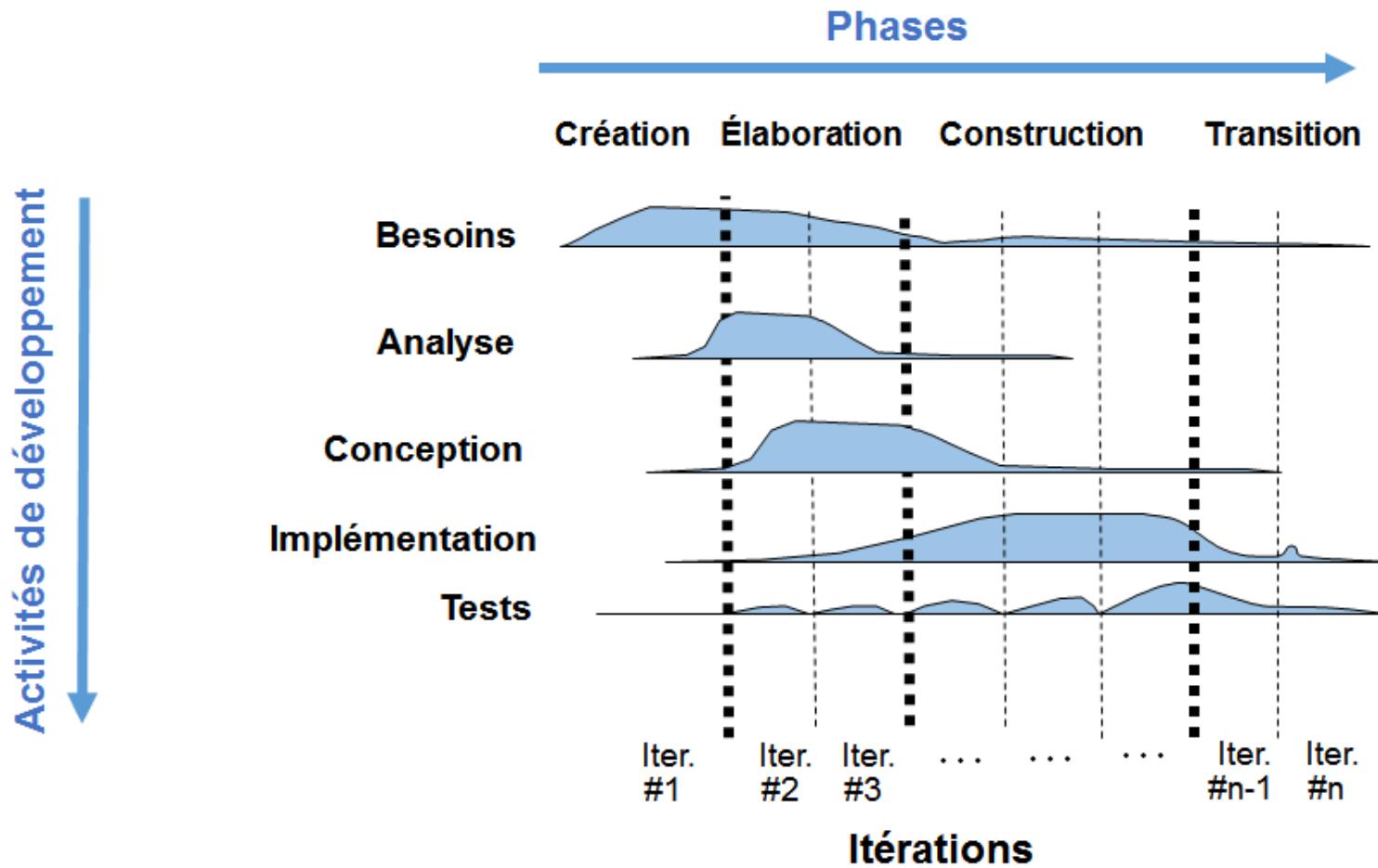
Distribution des ressources par phase



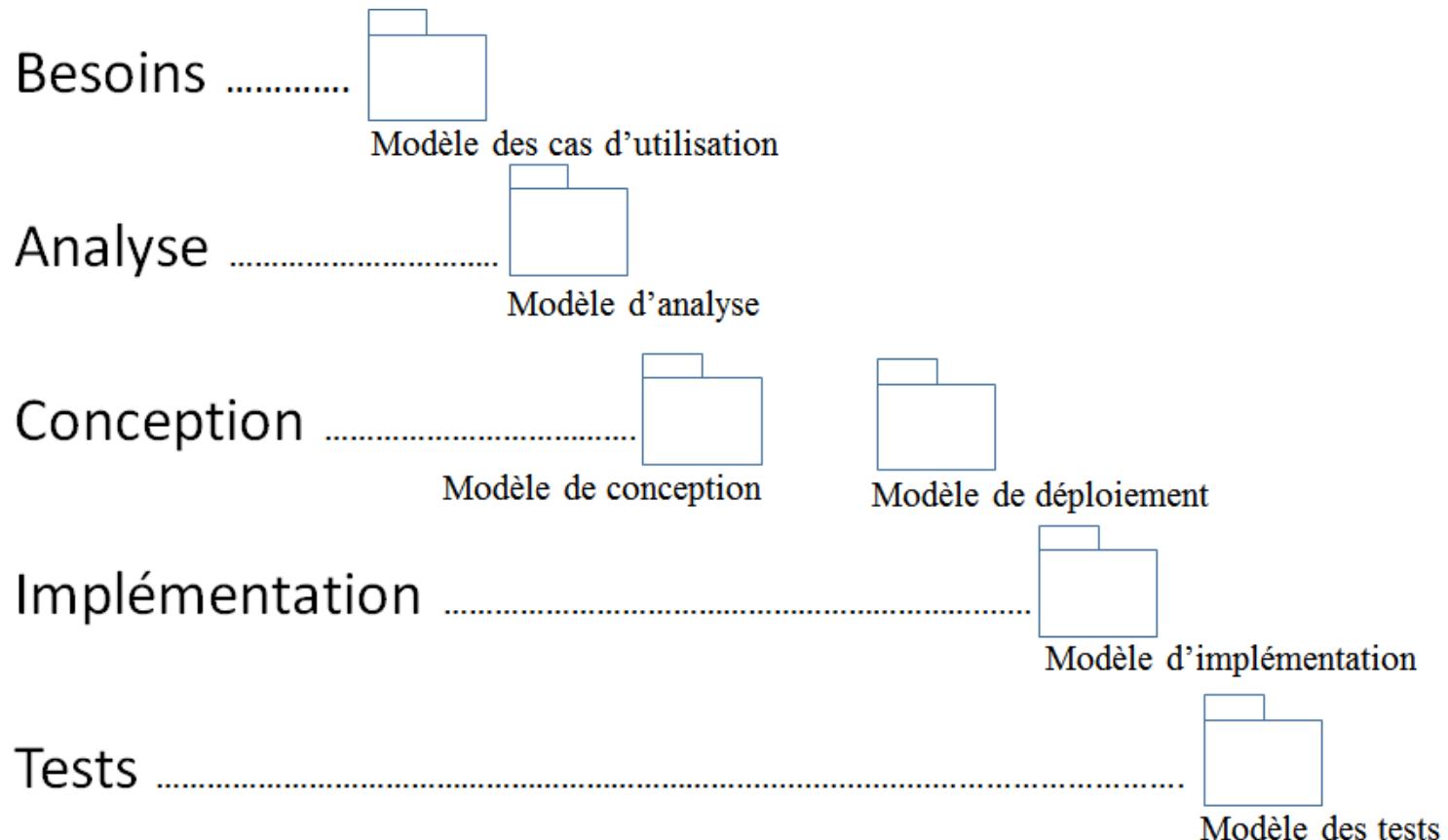
Progression du développement des cas d'utilisation dans les phases

	Modèle du métier complet	Cas d'utilisation identifiés	Masse des cas d'utilisation décrits	Masse des cas d'utilisation analysés	Masse des cas d'utilisation conçus, implémentés et testés
Phase de création	50 à 70 %	50 %	10 %	5 %	Un faible pourcentage pour un prototype de mise à l'épreuve du concept
Phase d'élaboration	Près de 100 %	Au moins 80 %	40 à 80 %	20 à 40 %	Moins de 10 %
Phase de construction	100 %	100 %	100 %	100 % si maintenus	100 %
Phase de transition					

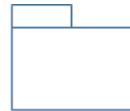
UP : les activités



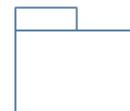
Modèles produits par les activités



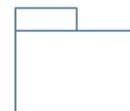
Correspondance Modèles – Diagrammes UML



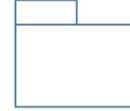
Modèle des cas d'utilisation



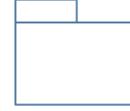
Modèle d'analyse



Modèle de conception



Modèle de déploiement



Modèle d'implémentation



Modèle des tests

Diagramme de
cas d'utilisation

Diagramme de
classes

Diagramme
d'objets

Diagramme de
composants

Diagrammes de
déploiement

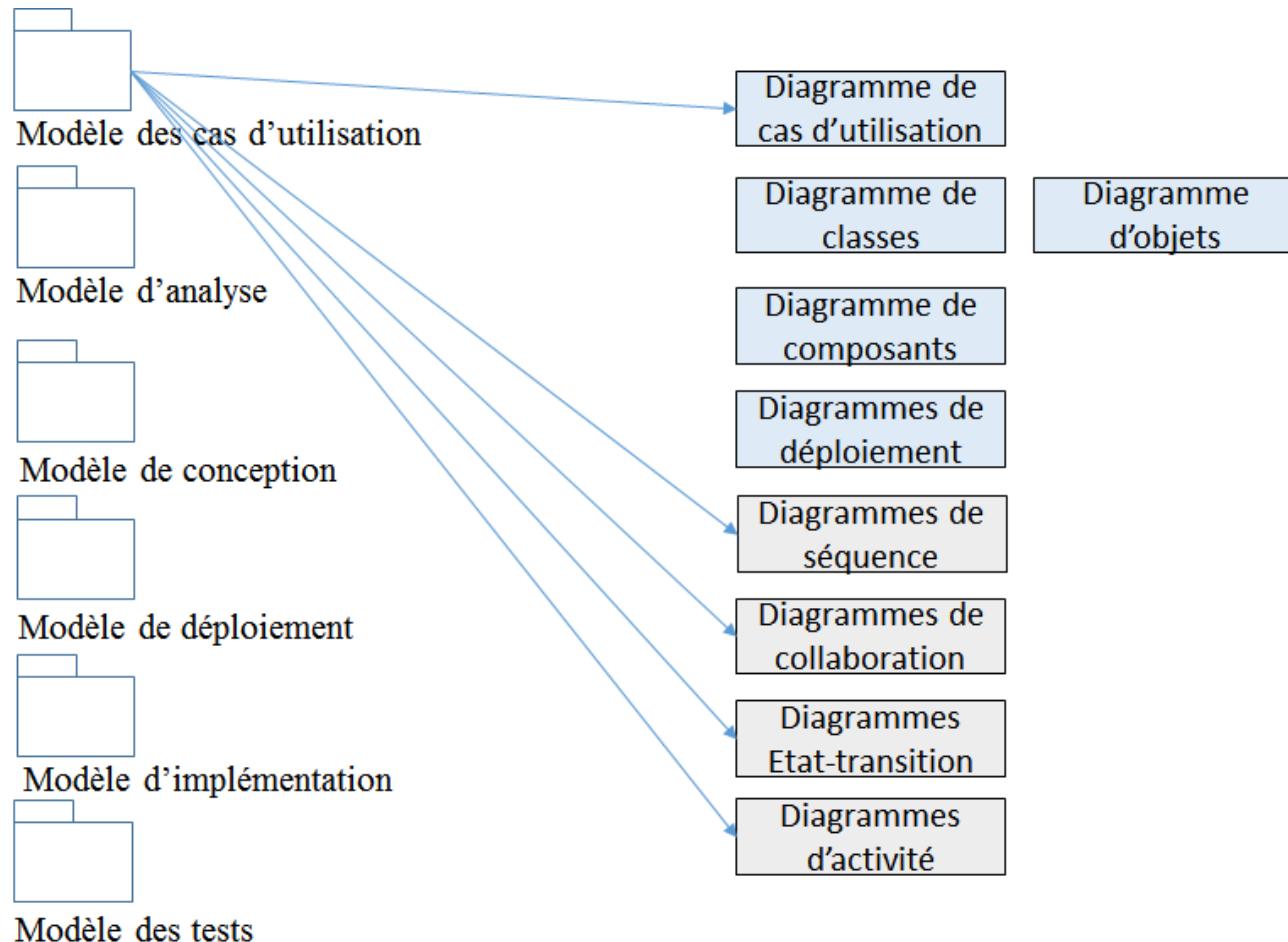
Diagrammes de
séquence

Diagrammes de
collaboration

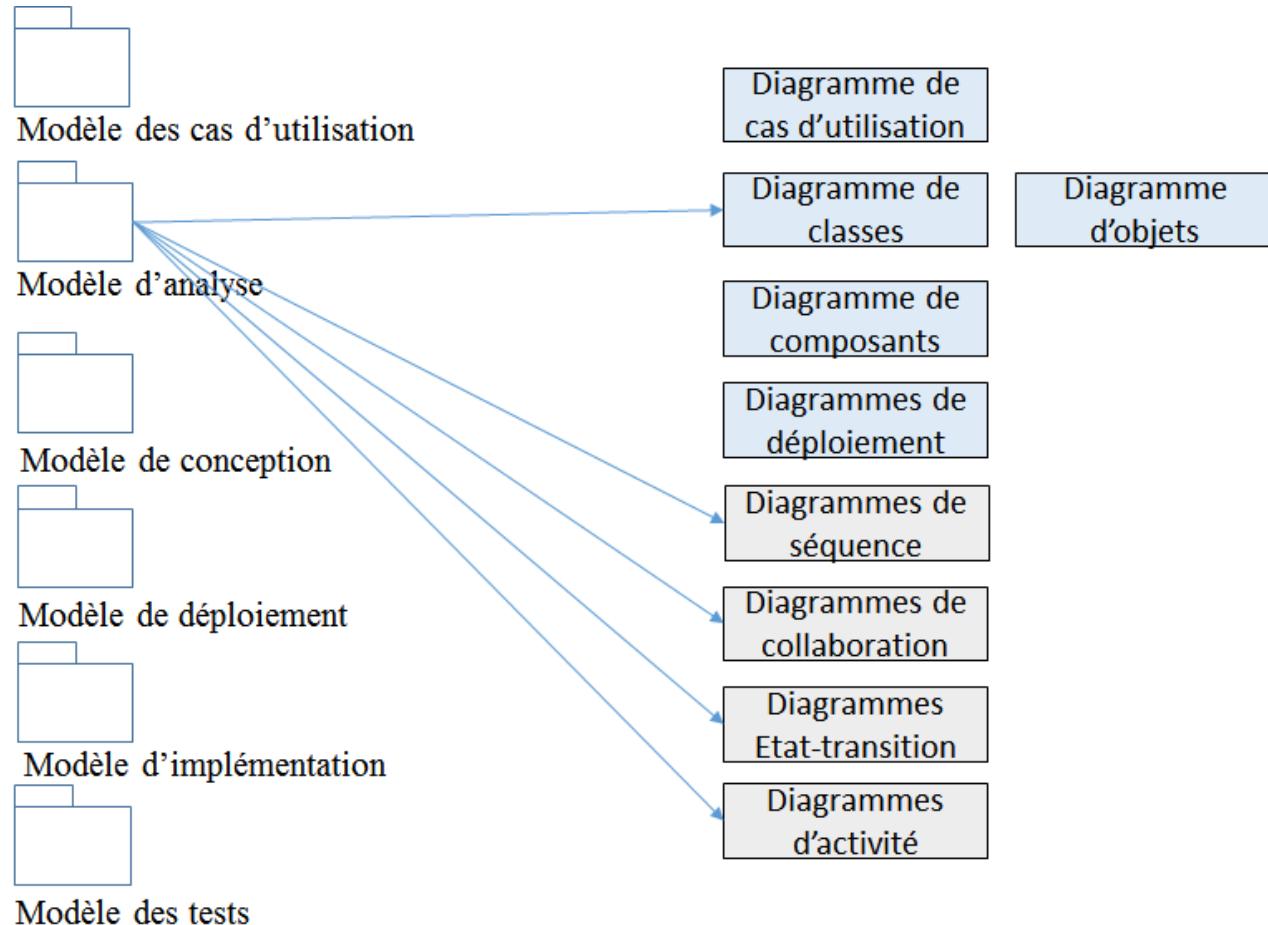
Diagrammes
Etat-transition

Diagrammes
d'activité

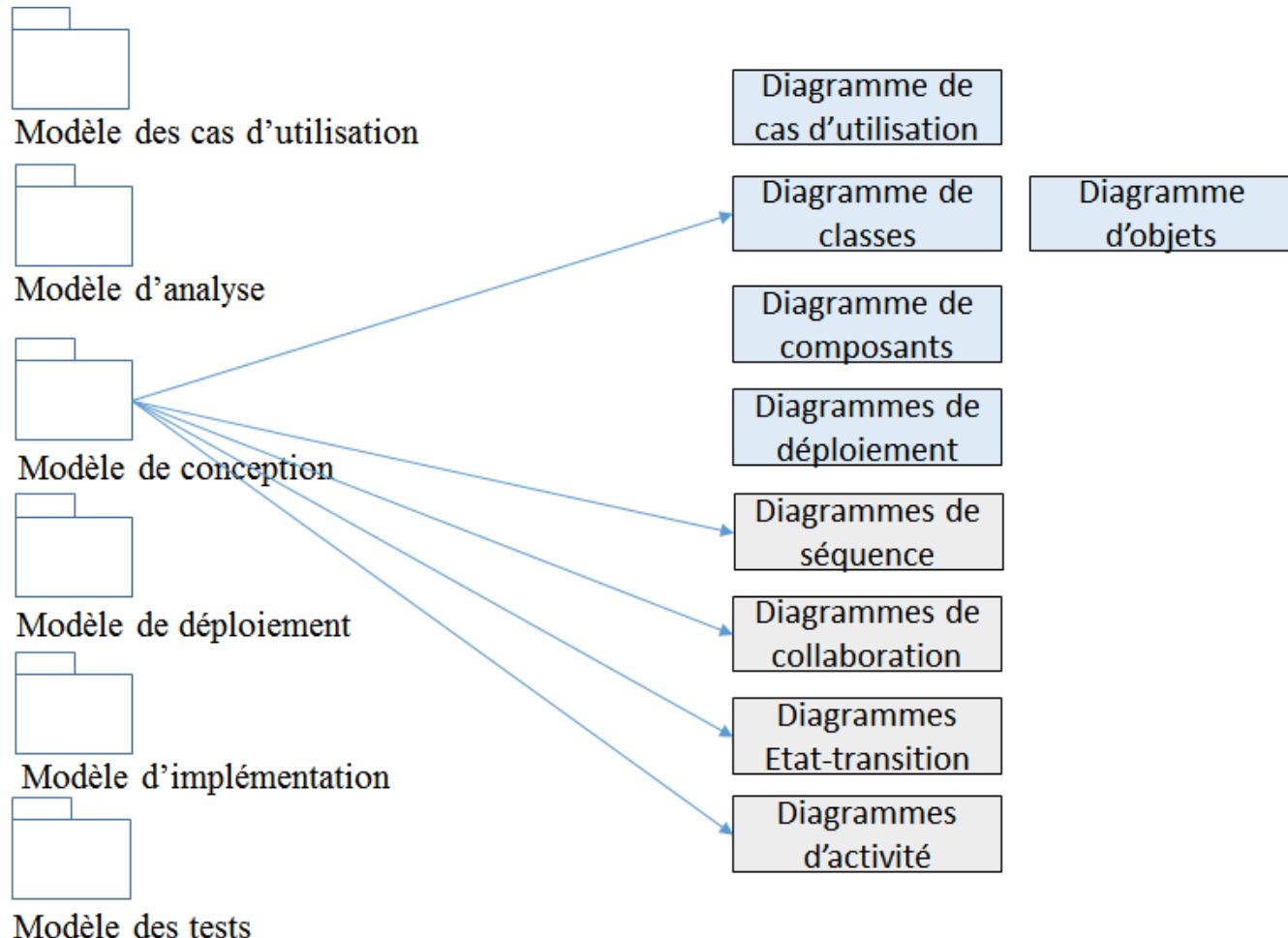
Correspondance Modèles – Diagrammes UML (suite)



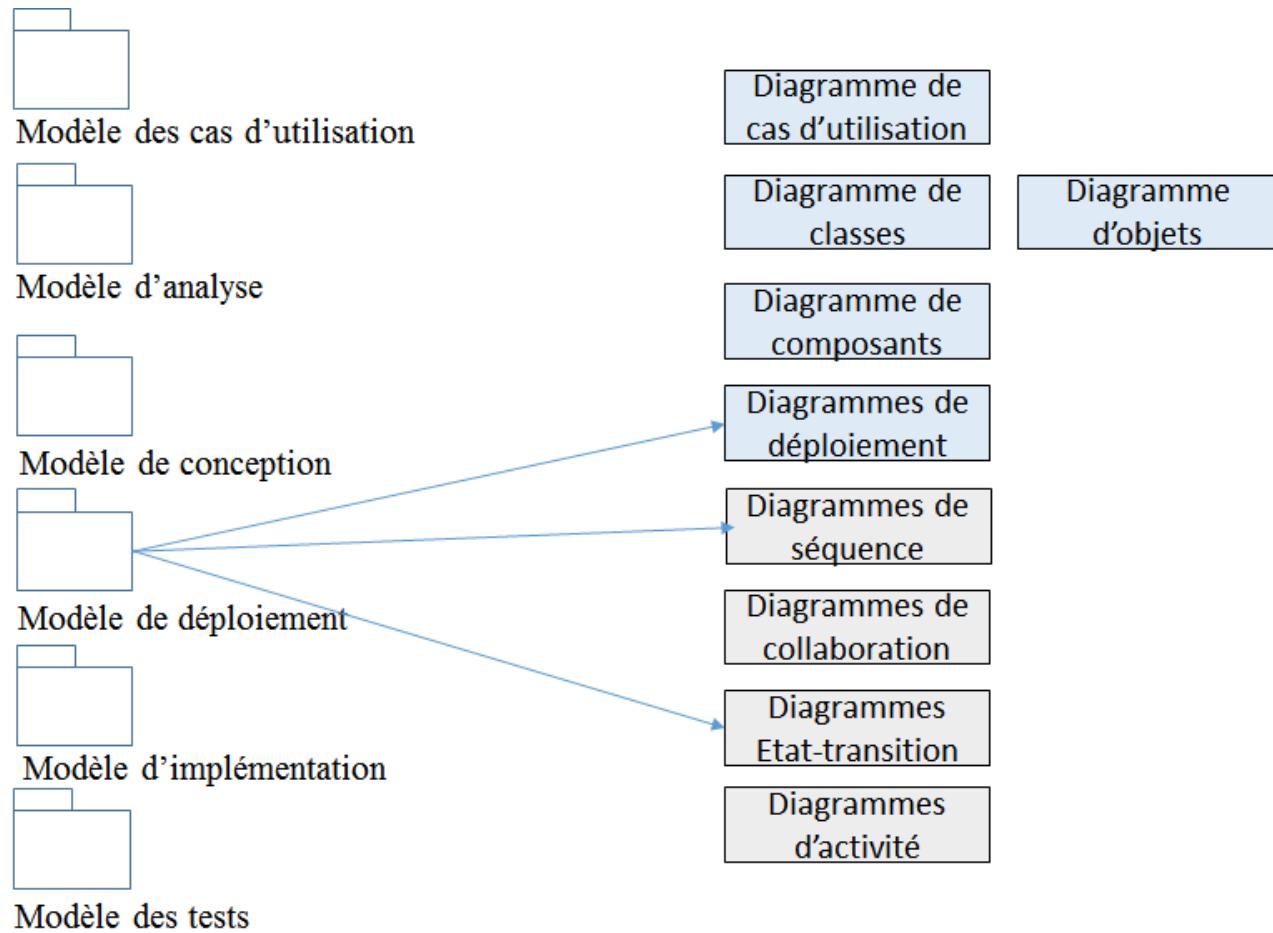
Correspondance Modèles – Diagrammes UML (suite)



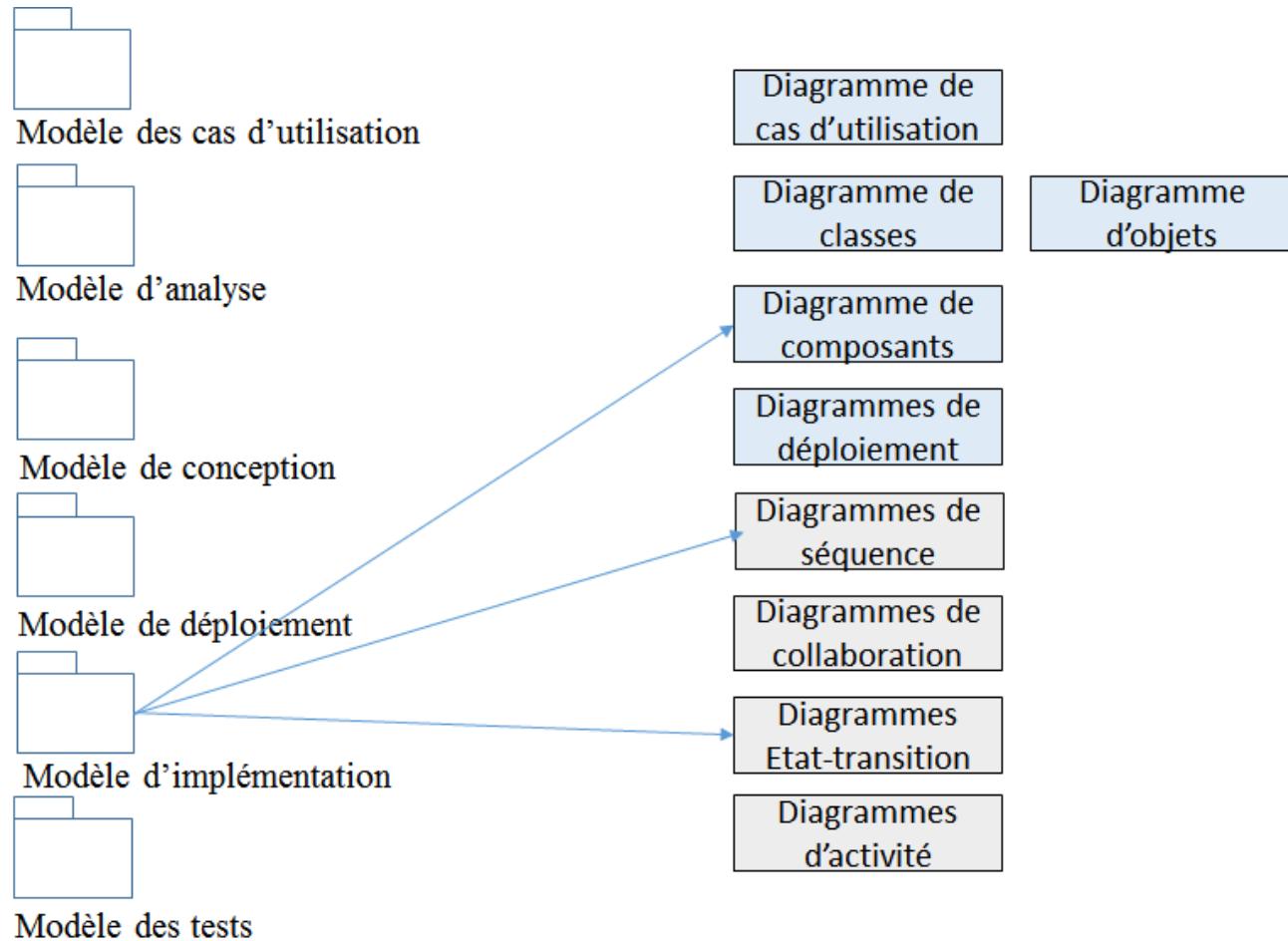
Correspondance Modèles – Diagrammes UML (suite)



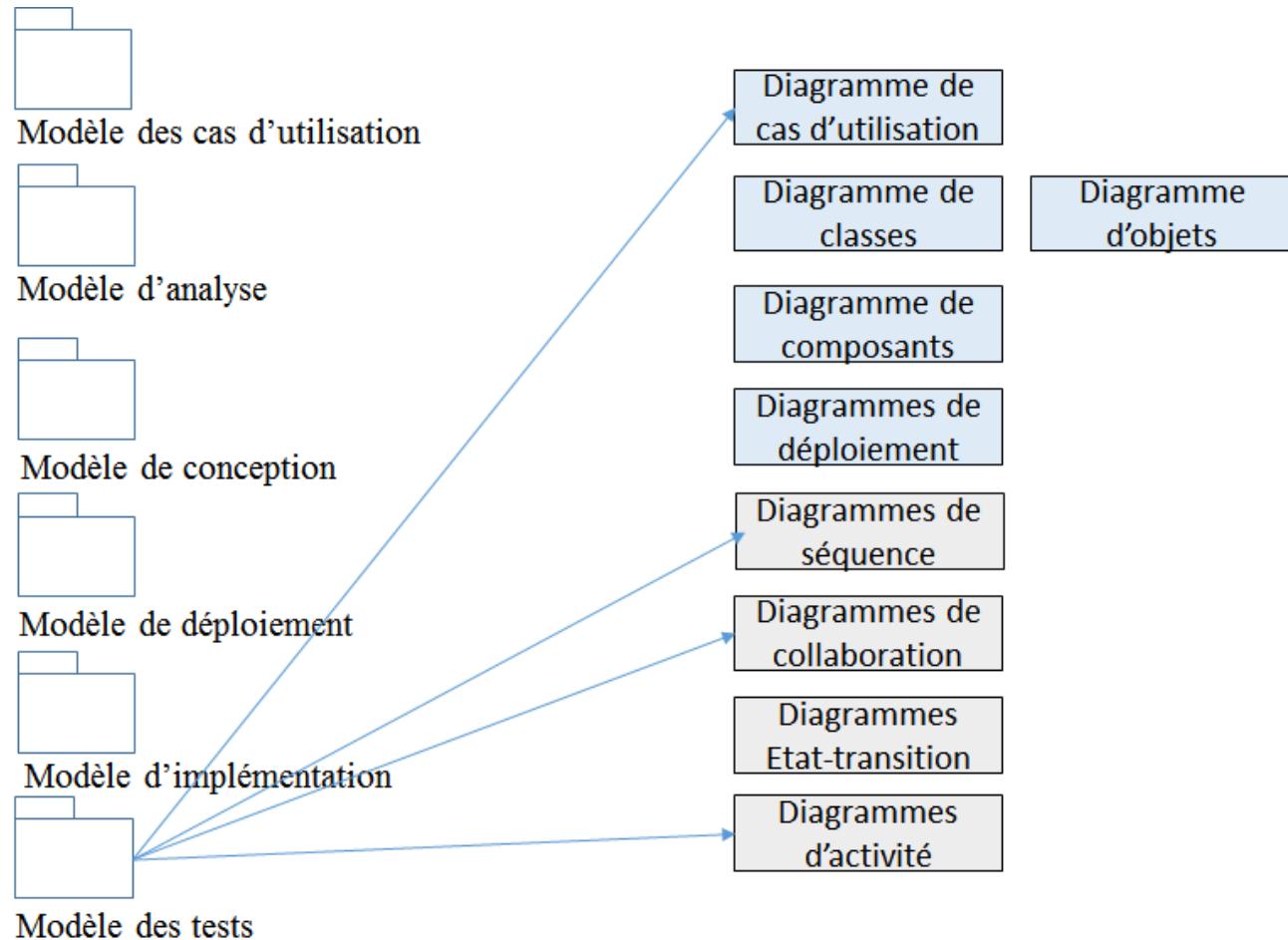
Correspondance Modèles – Diagrammes UML (suite)



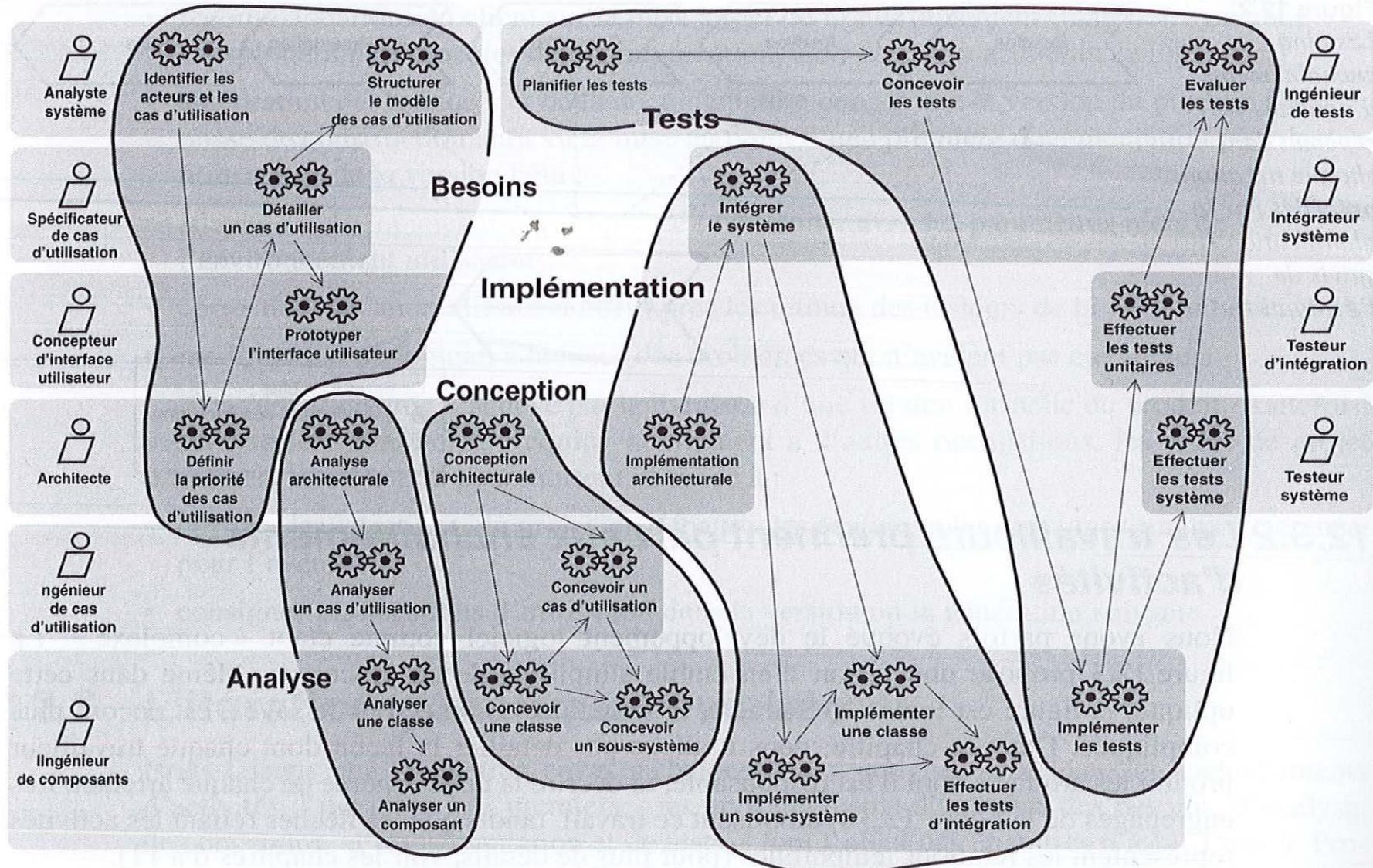
Correspondance Modèles – Diagrammes UML (suite)



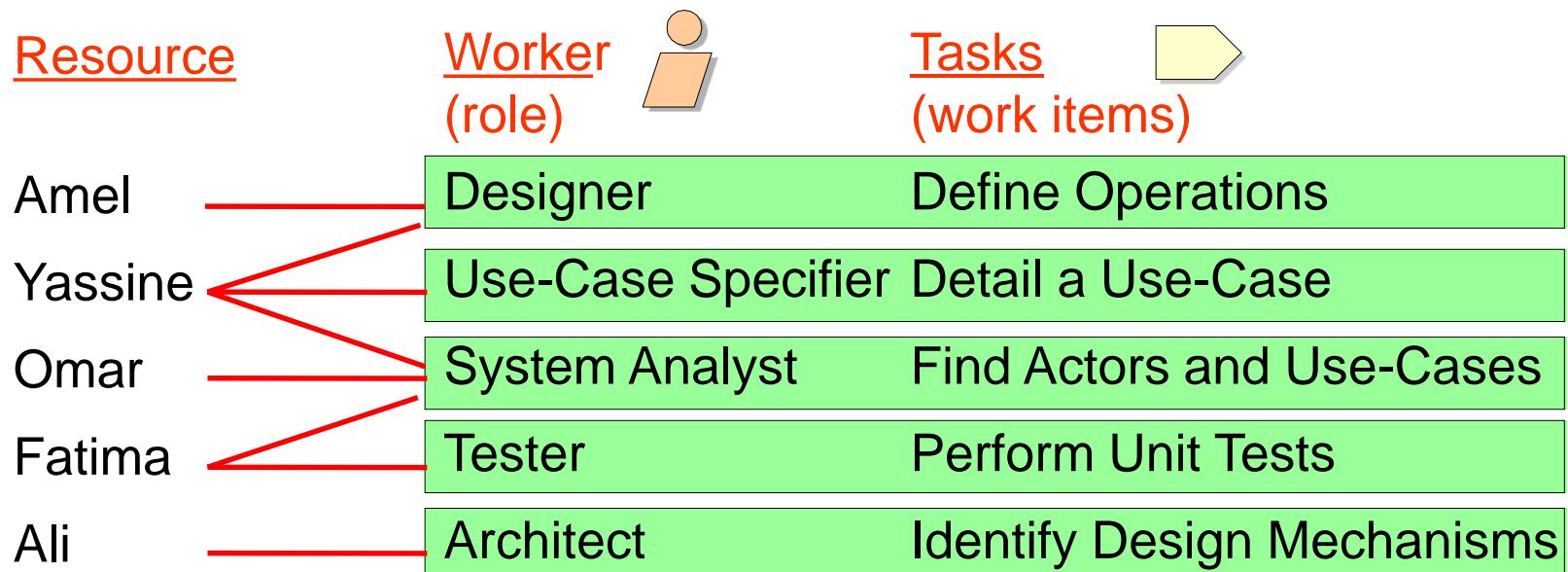
Correspondance Modèles – Diagrammes UML (suite)



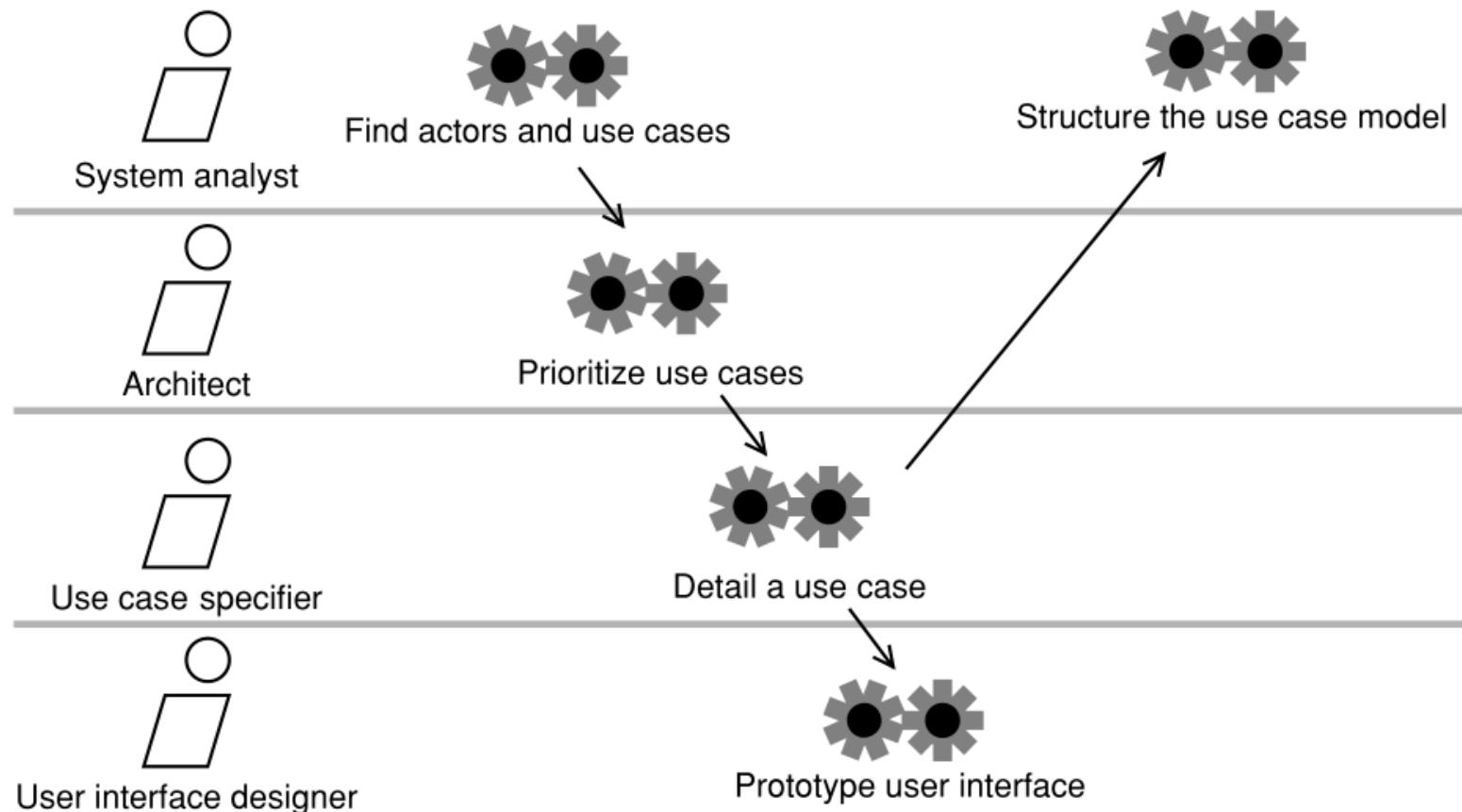
Les activités et les travailleurs



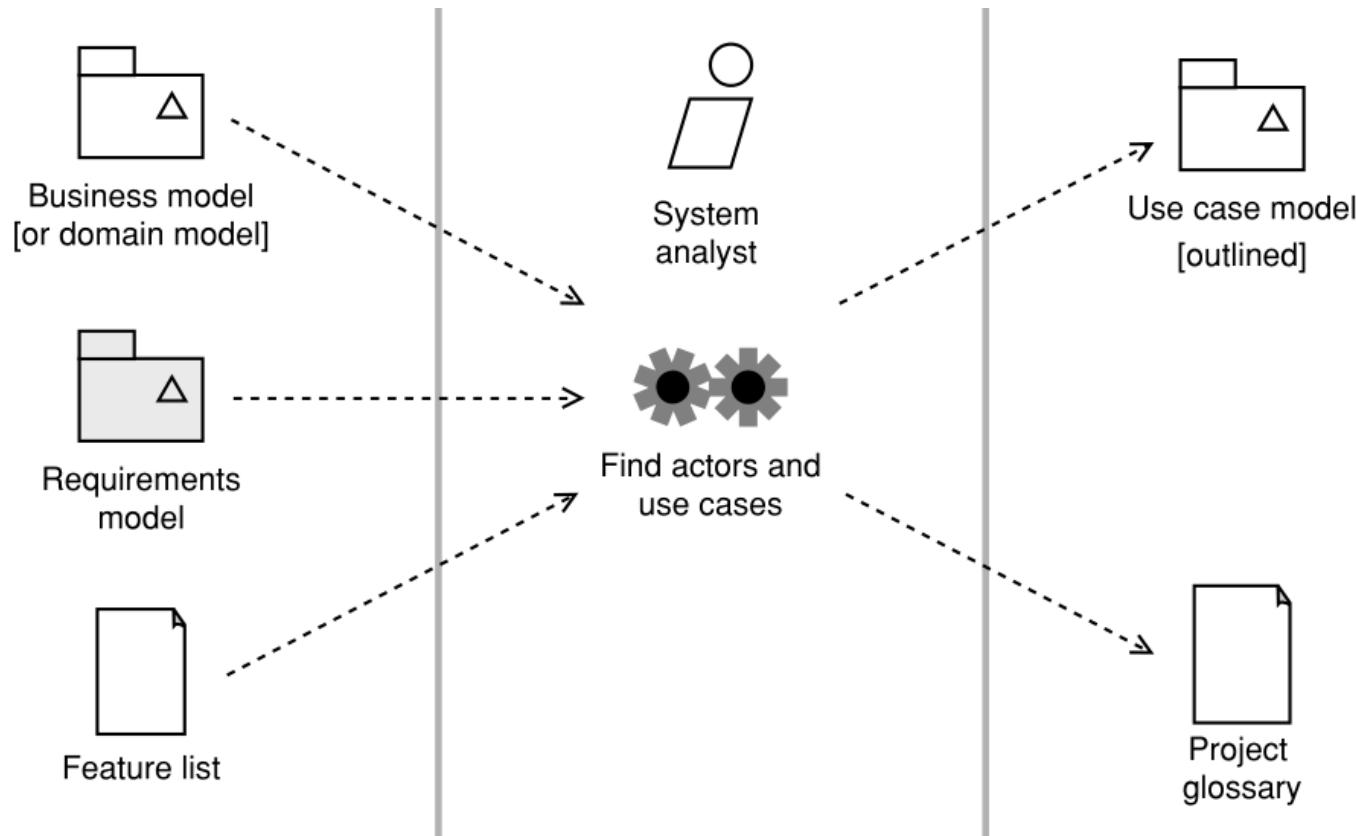
UP : les travailleurs (rôles)



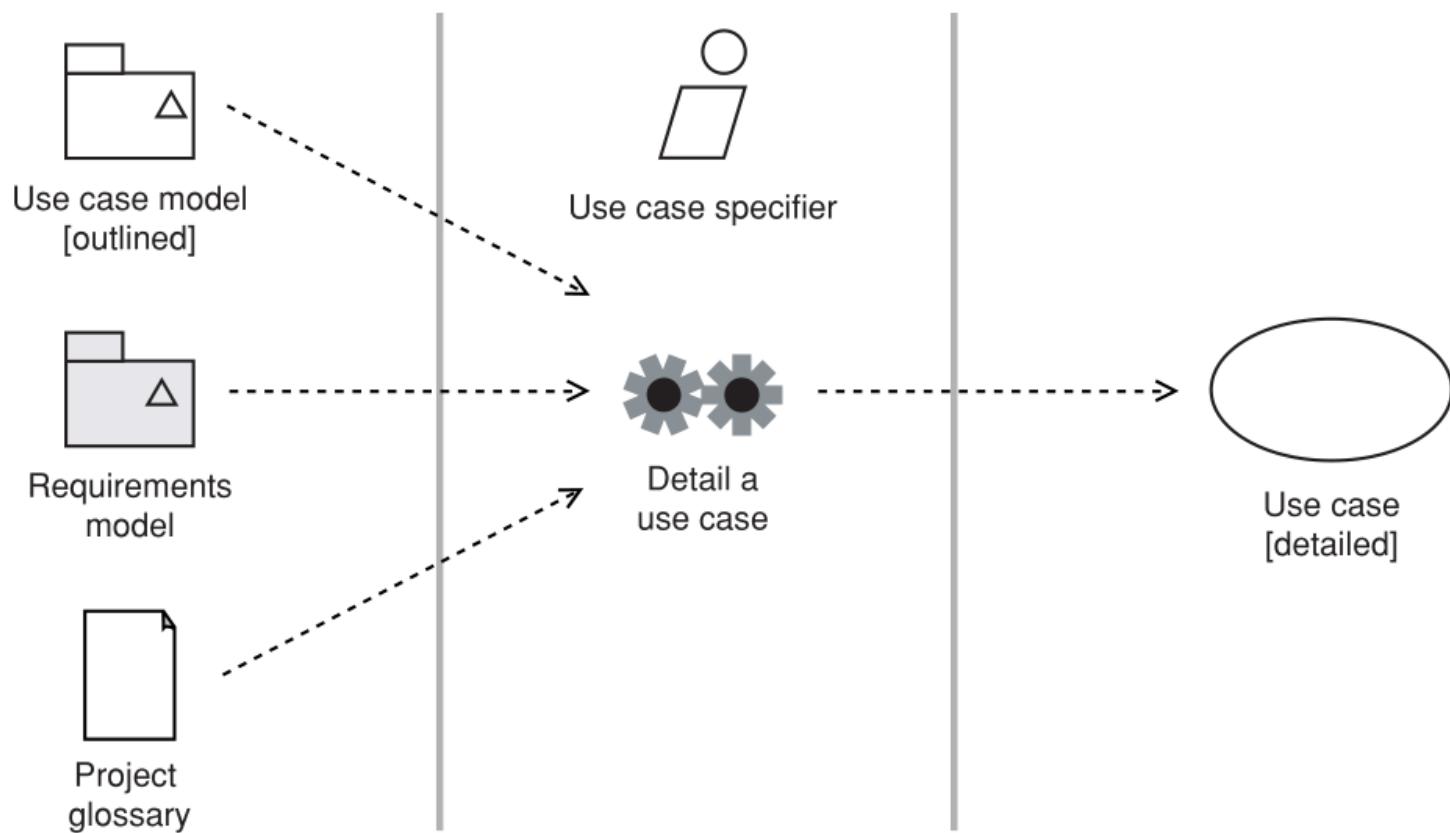
Workflow de l'activité Besoins



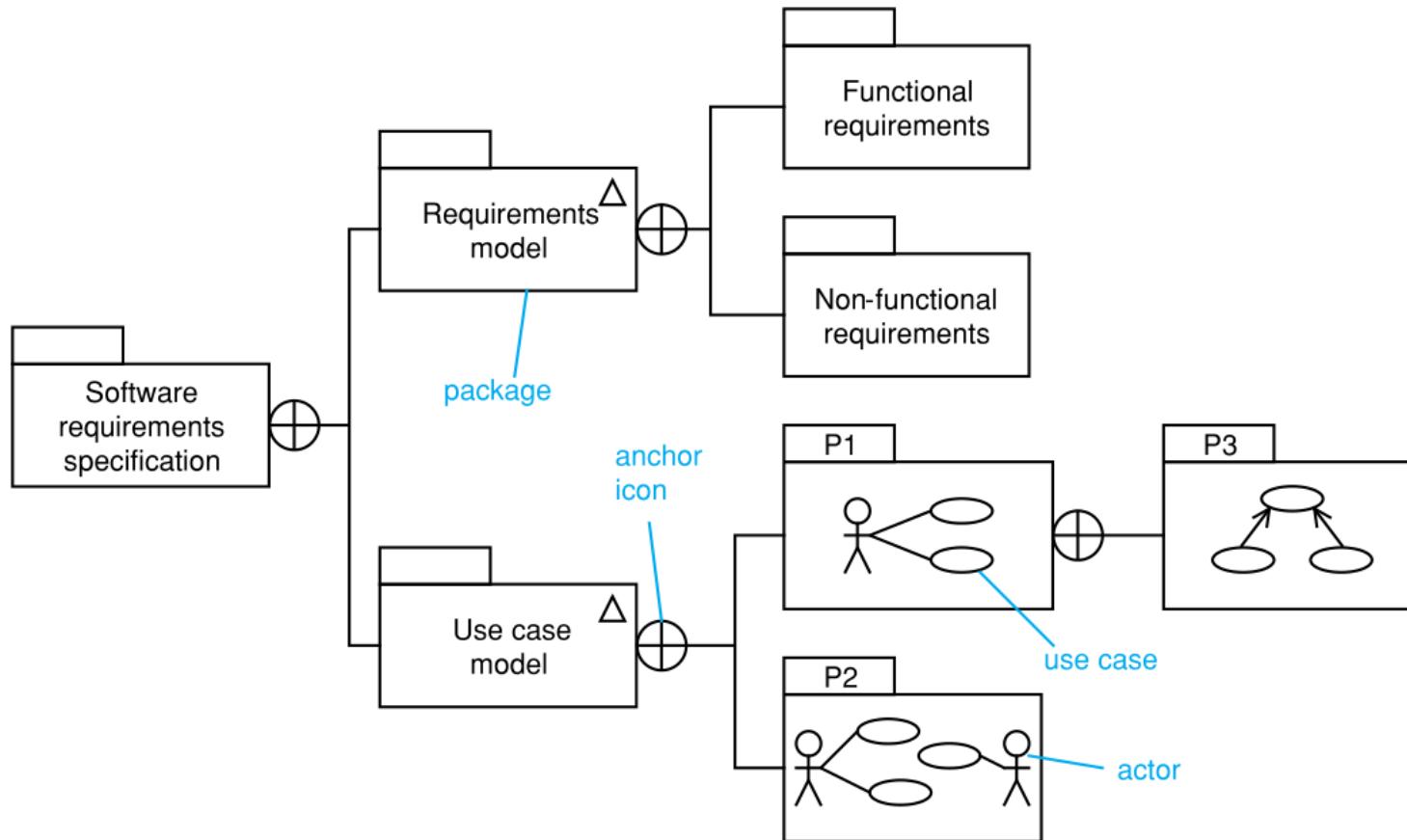
La tâche Identifier les acteurs et les cas d'utilisation



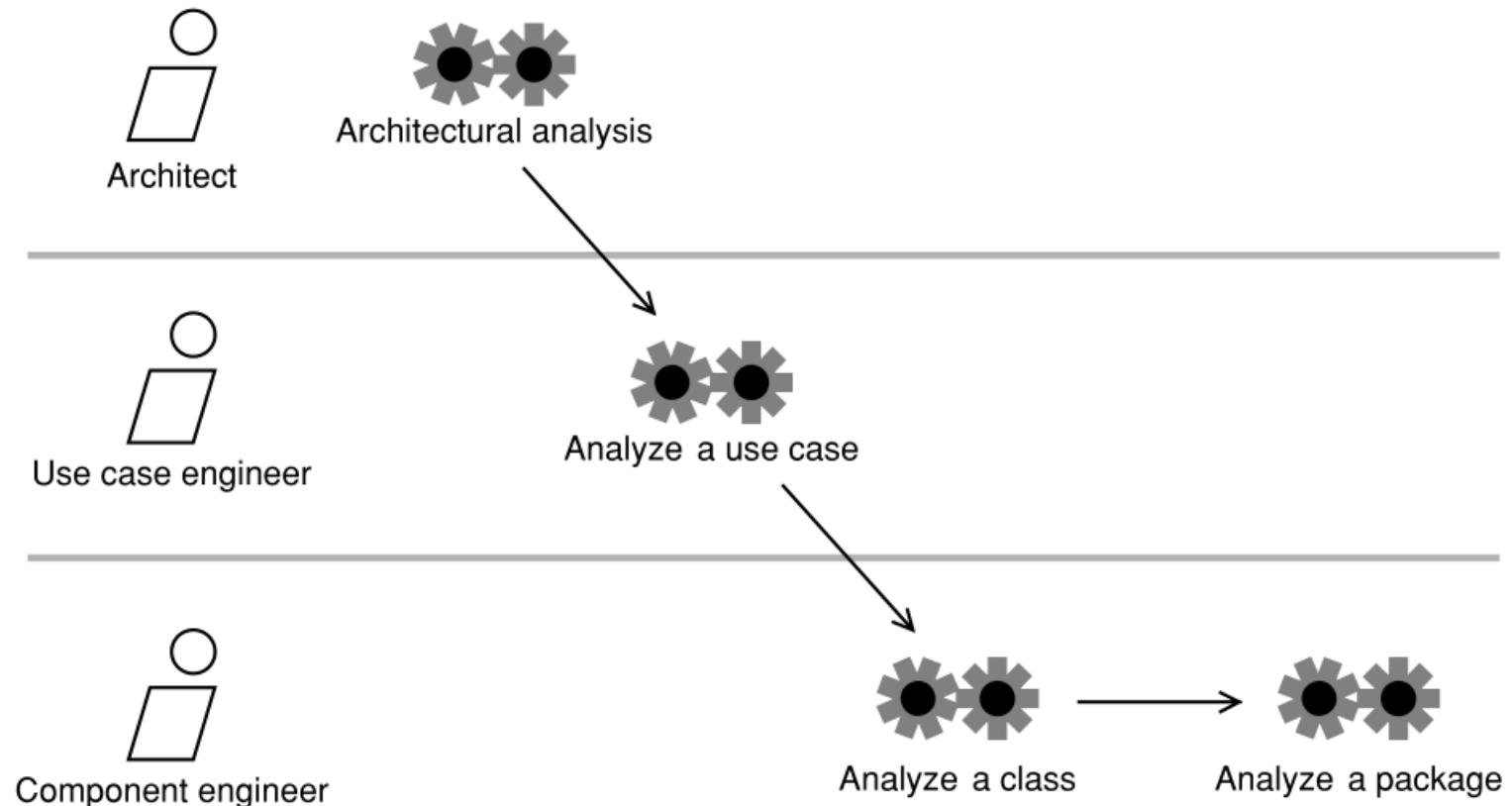
La tâche Détailier un cas d'utilisation



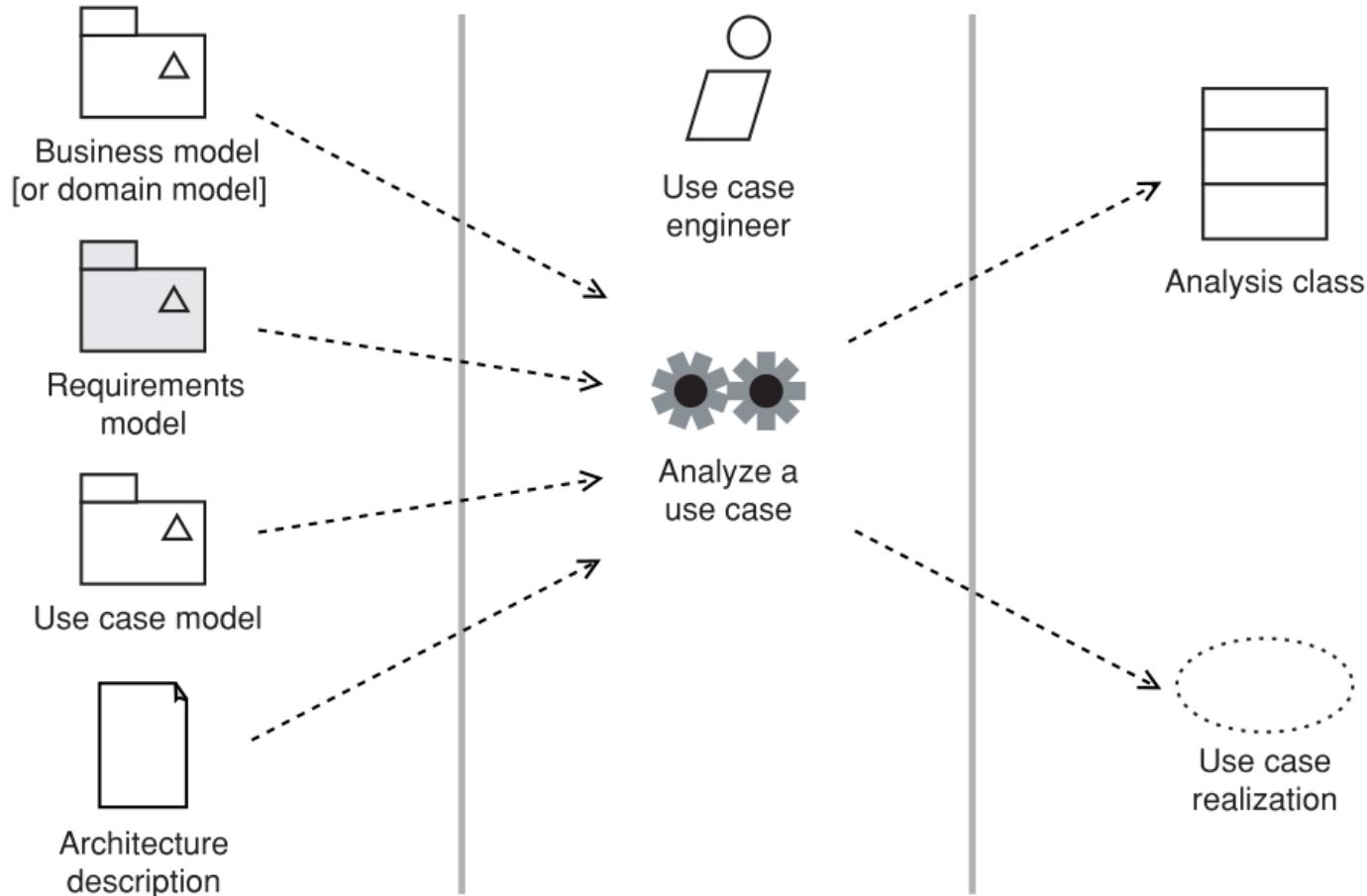
Modèles des besoins et modèle des cas d'utilisation



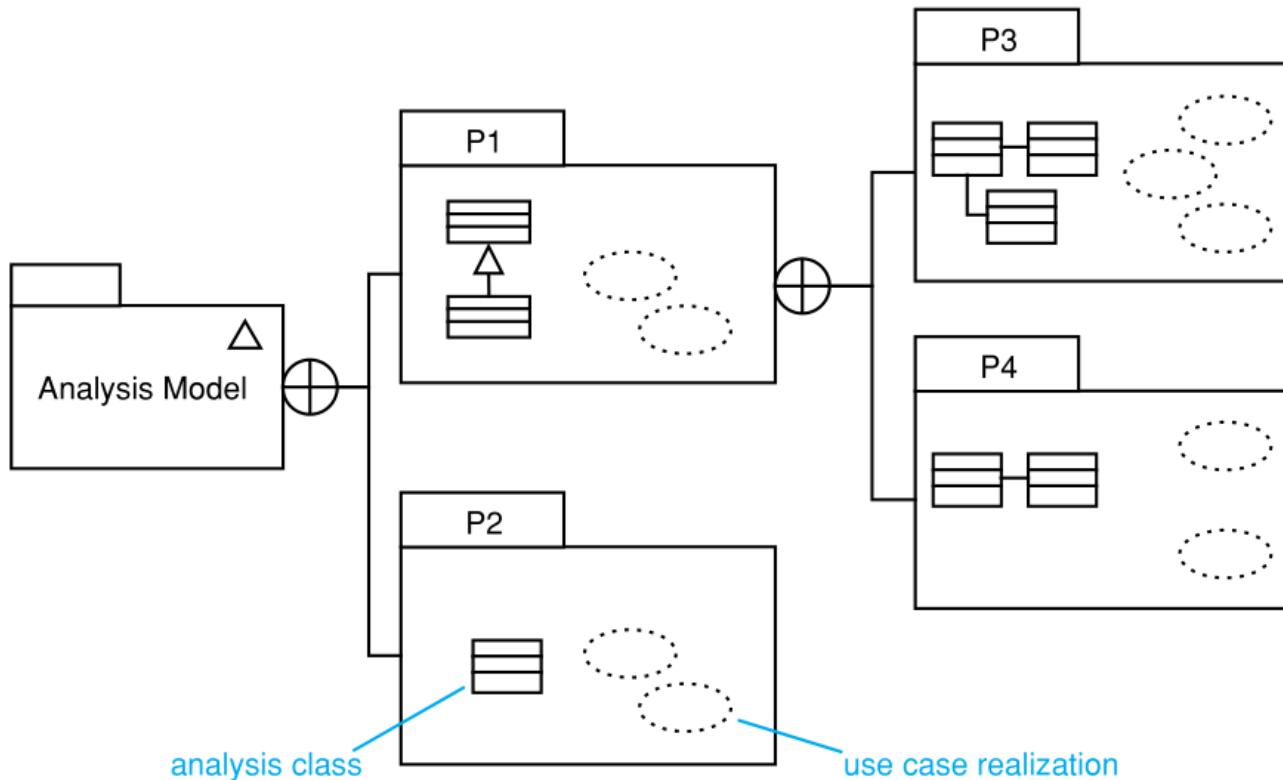
Workflow de l'activité Analyse



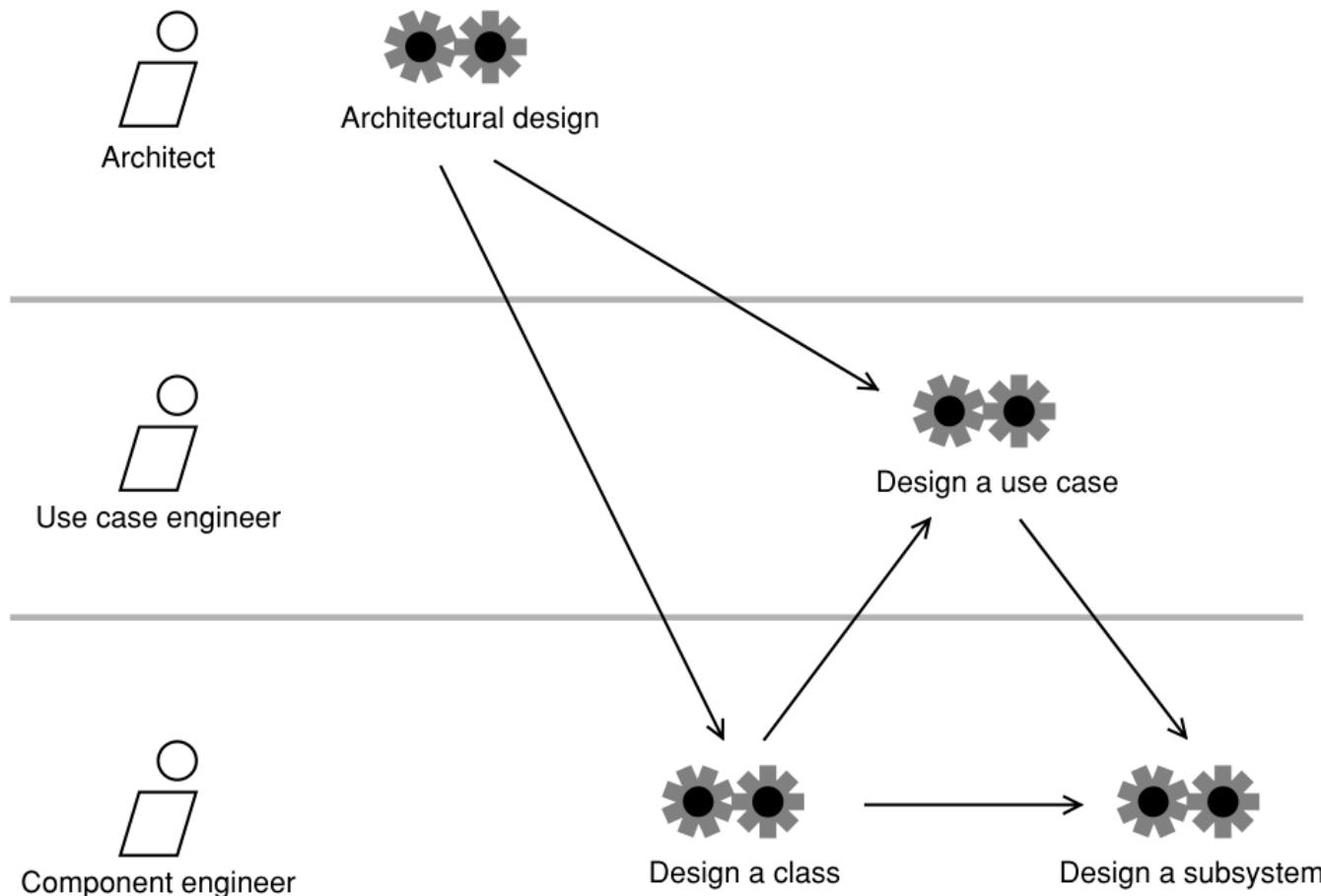
La tâche Analyser un cas d'utilisation



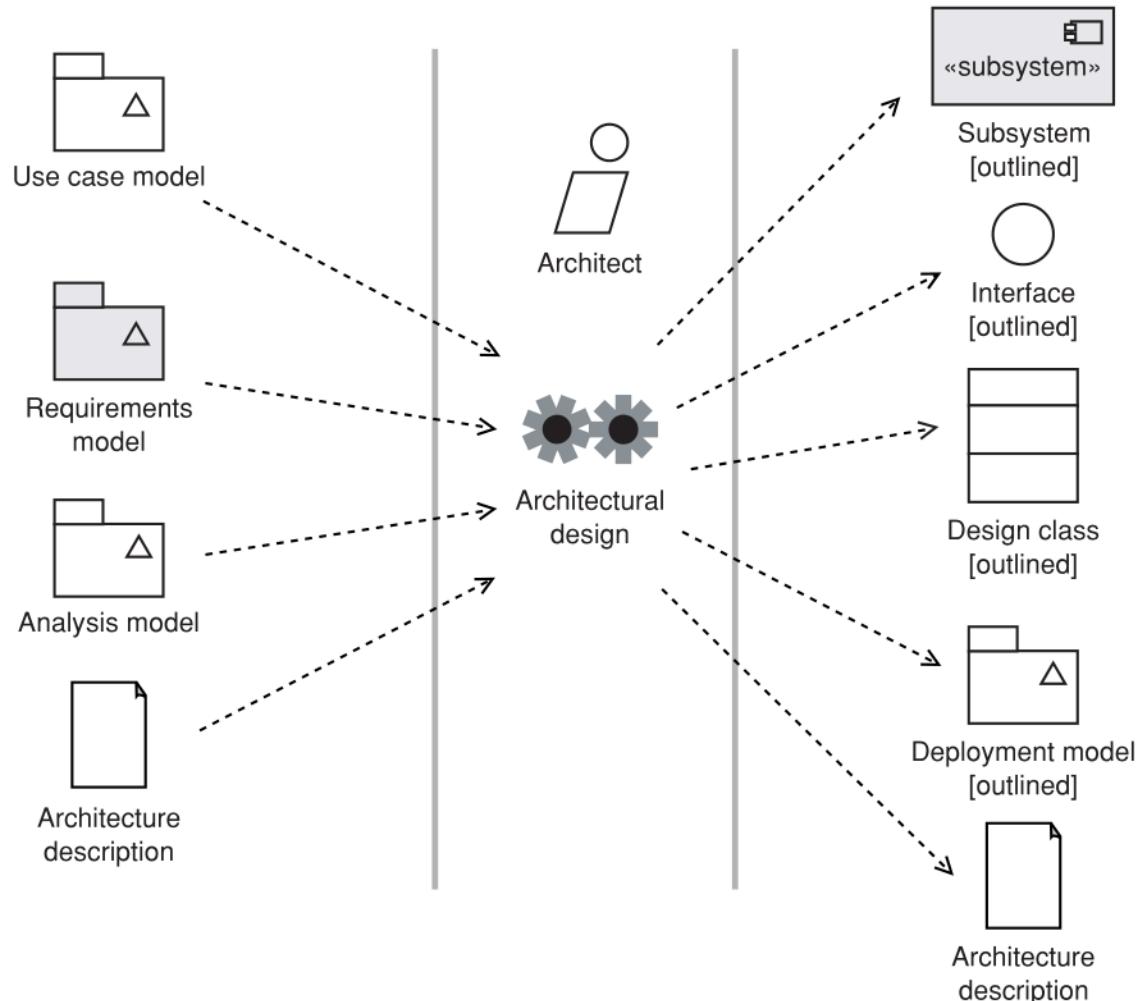
Le modèle d'analyse



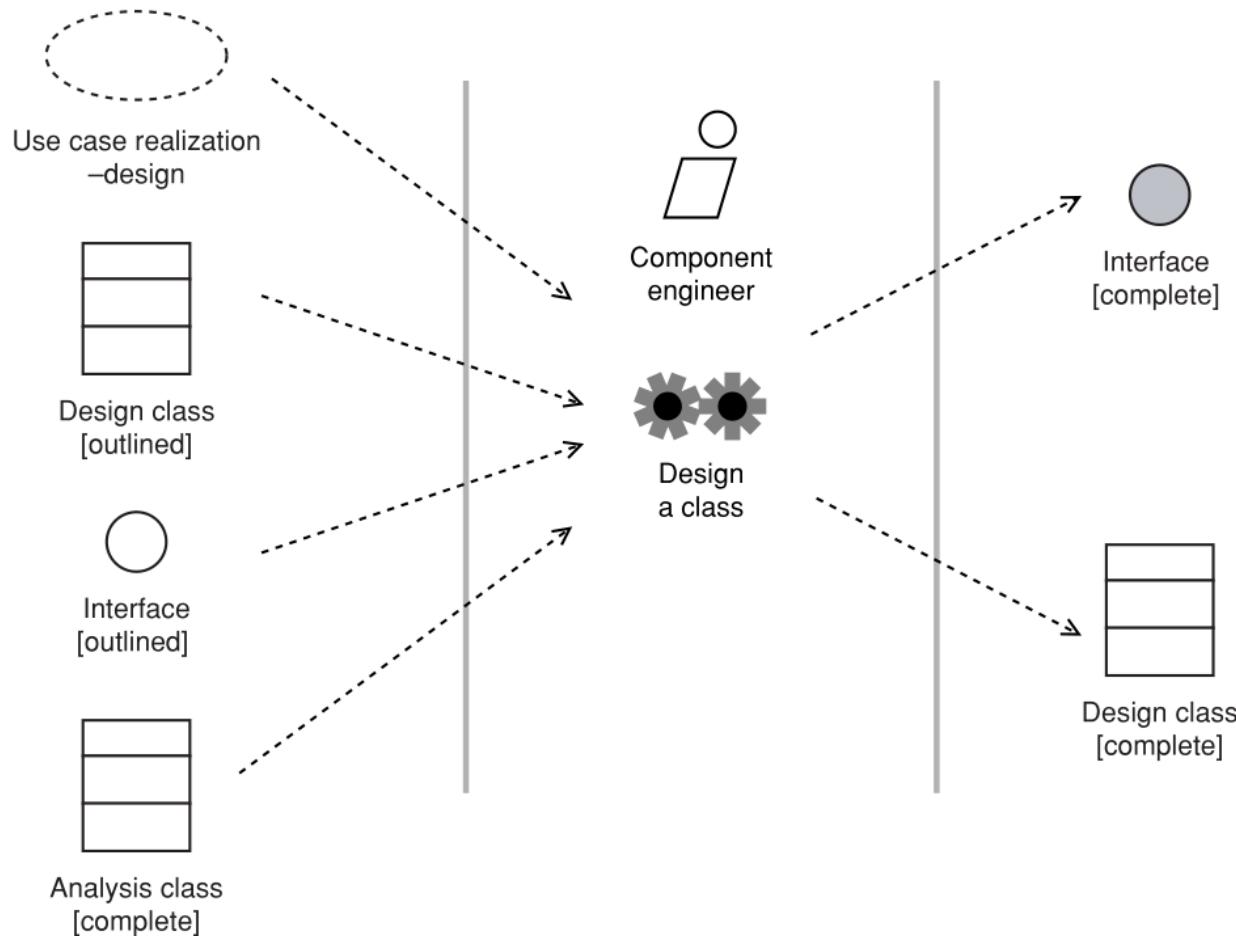
Workflow de l'activité Conception



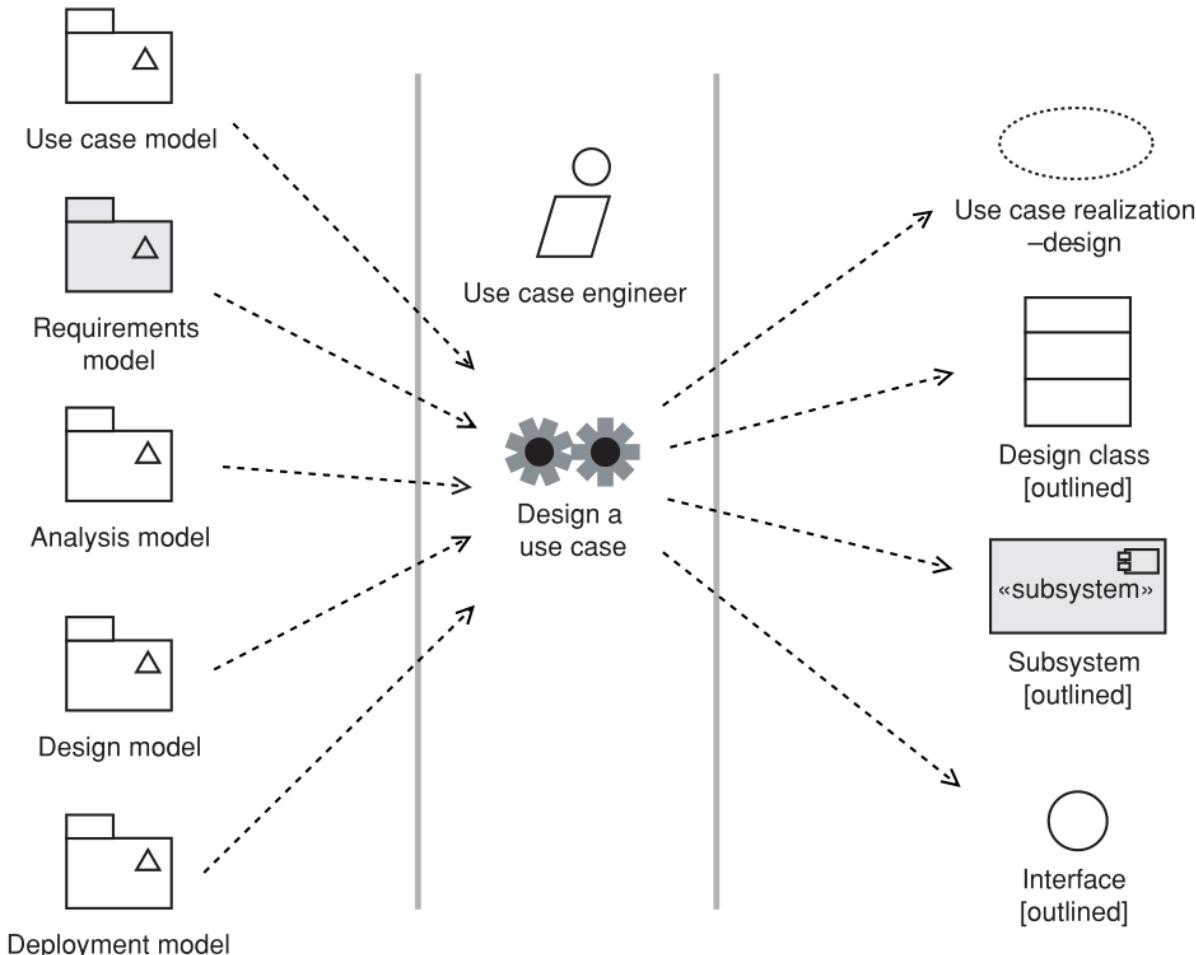
La tâche Conception architecturale



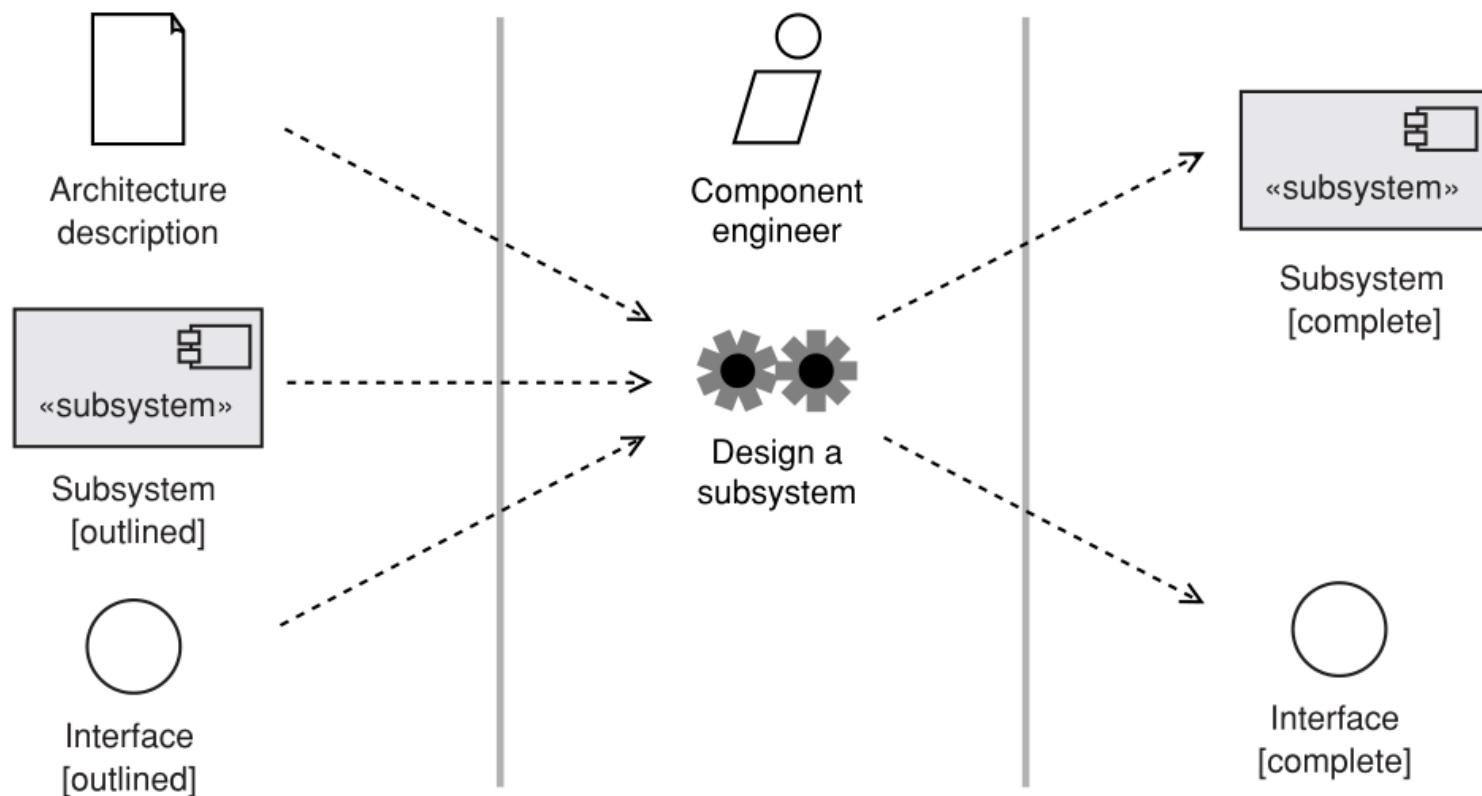
La tâche Conception d'une classe



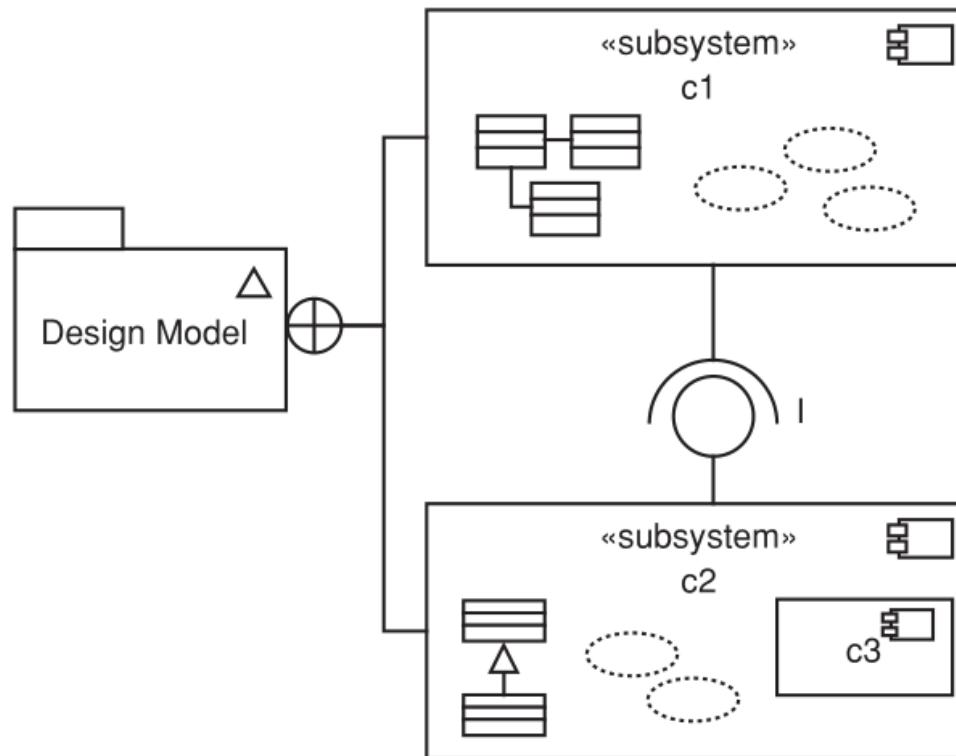
La tâche Conception d'un cas d'utilisation



La tâche Conception d'un sous-système



Le modèle de conception



Workflow de l'activité Implémentation

