

UML

Introduction aux diagrammes

Emmanuel Pichon

2013

V1.1

Introduction aux diagrammes UML

- ◎ Pour bien comprendre UML, il est nécessaire de définir un cadre général de présentation
 - Les concepts « objet »
 - Les activités de fabrication des logiciels en amont du code (sans définir une méthode particulière)

- ◎ Ce cadre général de présentation a pour but
 - De vous indiquer le contexte d'usage de chaque diagramme
 - Pour vous aider à choisir le bon diagramme en fonction de vos besoins de modélisation

Le concept d'objet

- ◉ Un objet est un élément dynamique (*run-time*)
 - Il n'existe que pendant le fonctionnement du système
 - Il est créé (instancié), utilisé et détruit par d'autres objets
 - Il collabore avec d'autres objets en échangeant des messages

"Objects respond to messages that are generated by objects executing communication actions" (OMG)

Le concept de classe

- ◉ Une classe est un élément statique du système
 - *Elle constitue le code du système (build)*
 - Elle définit la structure (attributs et relations) et le comportement (opérations) d'un ensemble d'objets
 - Elle permet de créer des objets pendant le fonctionnement du système (instanciation)

Principe d'encapsulation

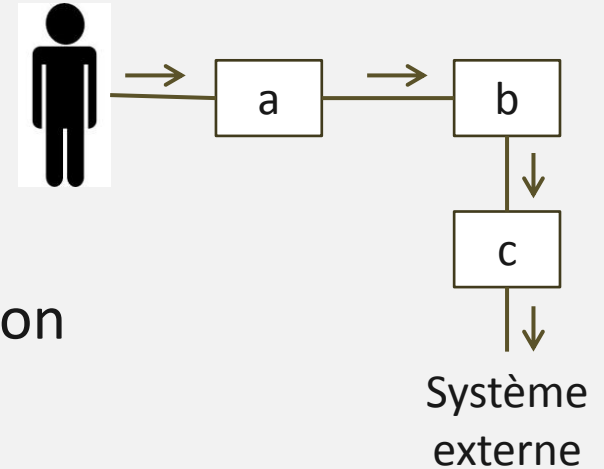
- ◎ **LE** principe « objet » à connaître
 - Un objet est une boîte noire qui offre des services
 - Distinction entre comportement et implémentation d'un objet
 - Un objet est responsable du son bon fonctionnement
- ◎ Objectif
 - Utiliser les services d'un objet sans connaître son implémentation
 - Pouvoir changer son implémentation sans impacter les services offerts aux autres objets
- ◎ Principe applicable à tous niveaux : système, sous-système, classe, ...

Distinction dynamique / statique dans UML

aspects dynamiques

- Diagrammes décrivant le comportement du système (*behavior*)

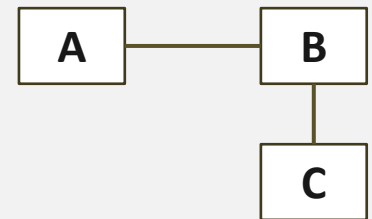
- Exemple : diagramme de communication



aspects statiques

- Diagrammes décrivant l'implémentation du système (*structure*)

- Exemple : diagramme de classes

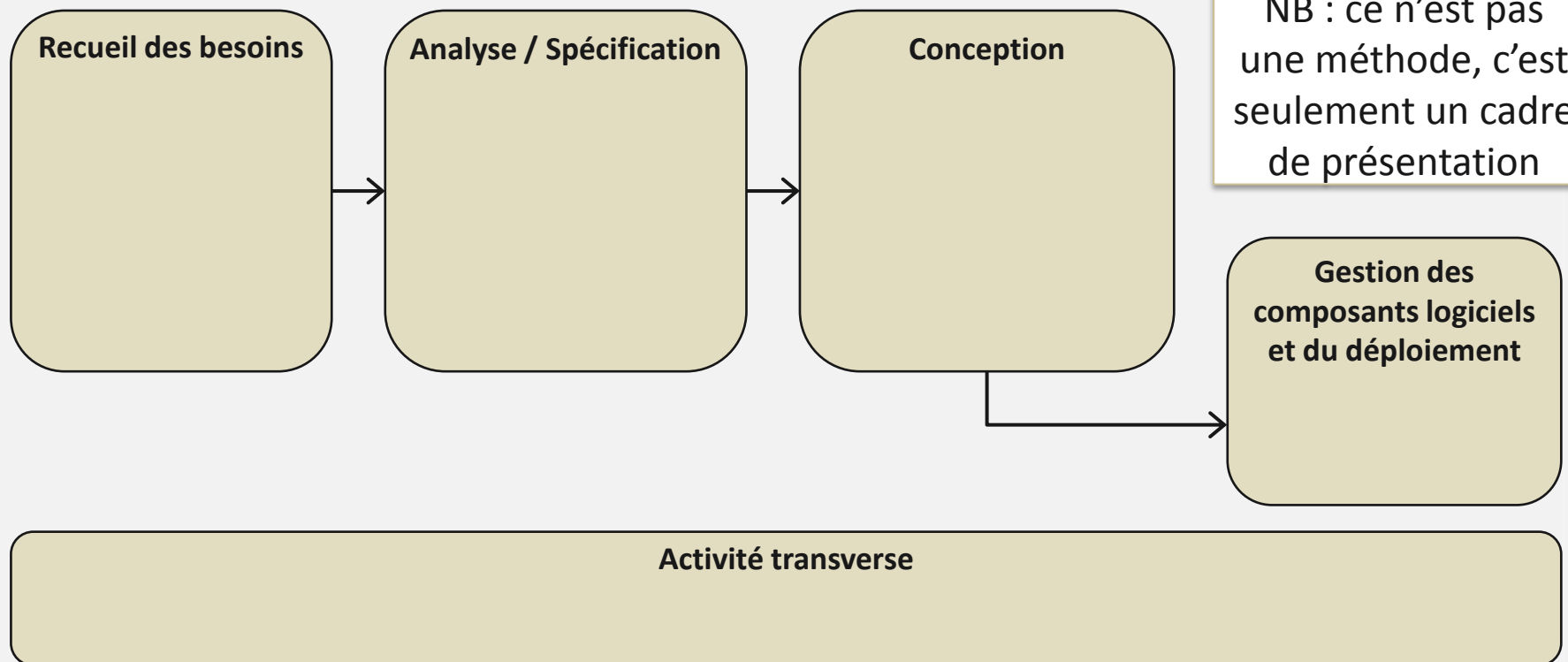


Cycle de vie d'objet

- ⊙ Tout objet suit un cycle de vie
 - Enchaînement d'états et de transitions entre états
 - A tout moment, un objet est dans un état
 - L'état d'un objet conditionne le fonctionnement de l'objet
- ⊙ Cycle de vie identique pour tous les objets d'une même classe
- ⊙ En UML, le diagrammes d'états est dédié à ce sujet
 - Applicable à tous niveaux : système, sous-système, classe, ...

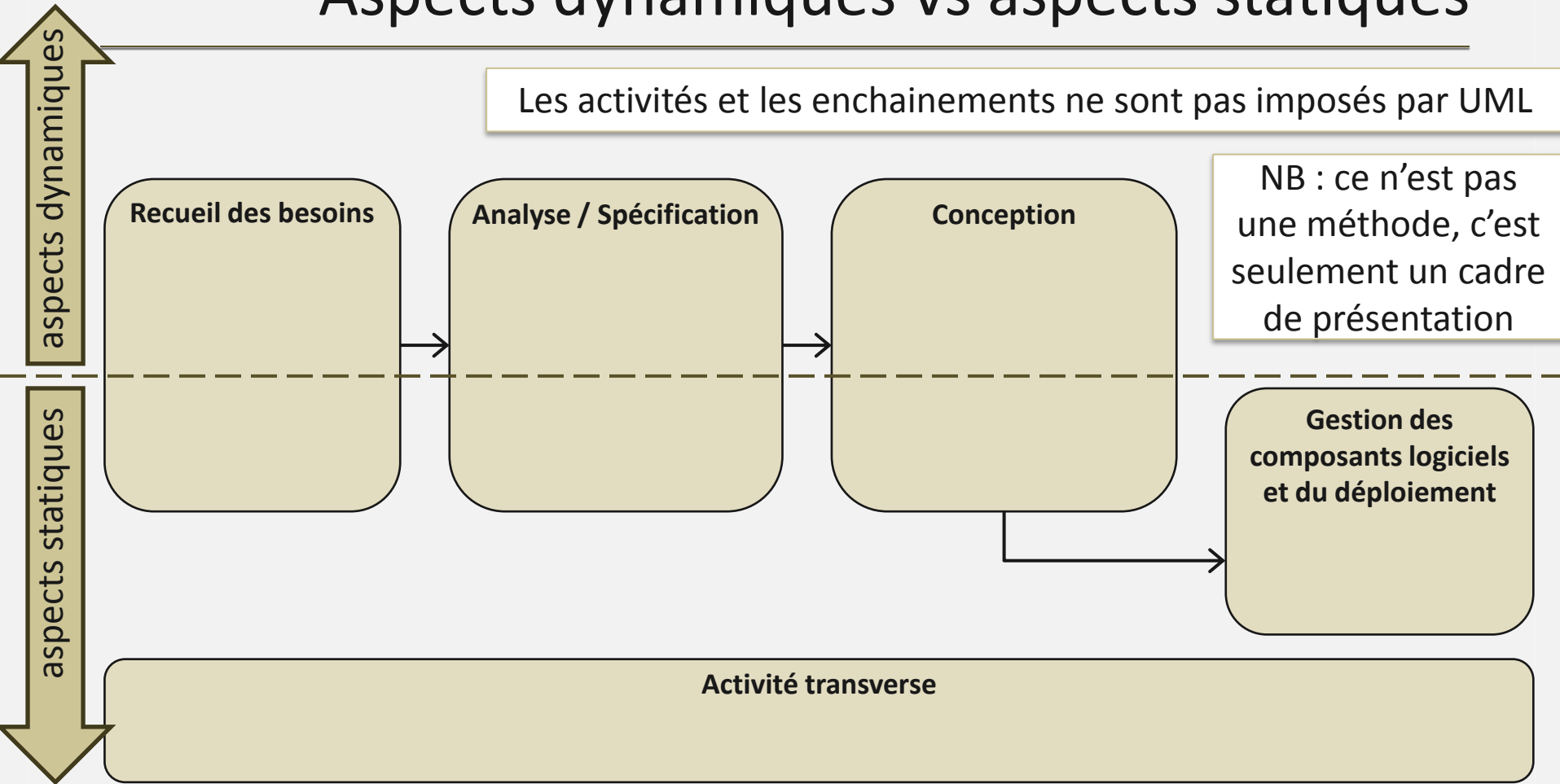
Activités concernées par UML

Les activités et les enchainements ne sont pas imposés par UML



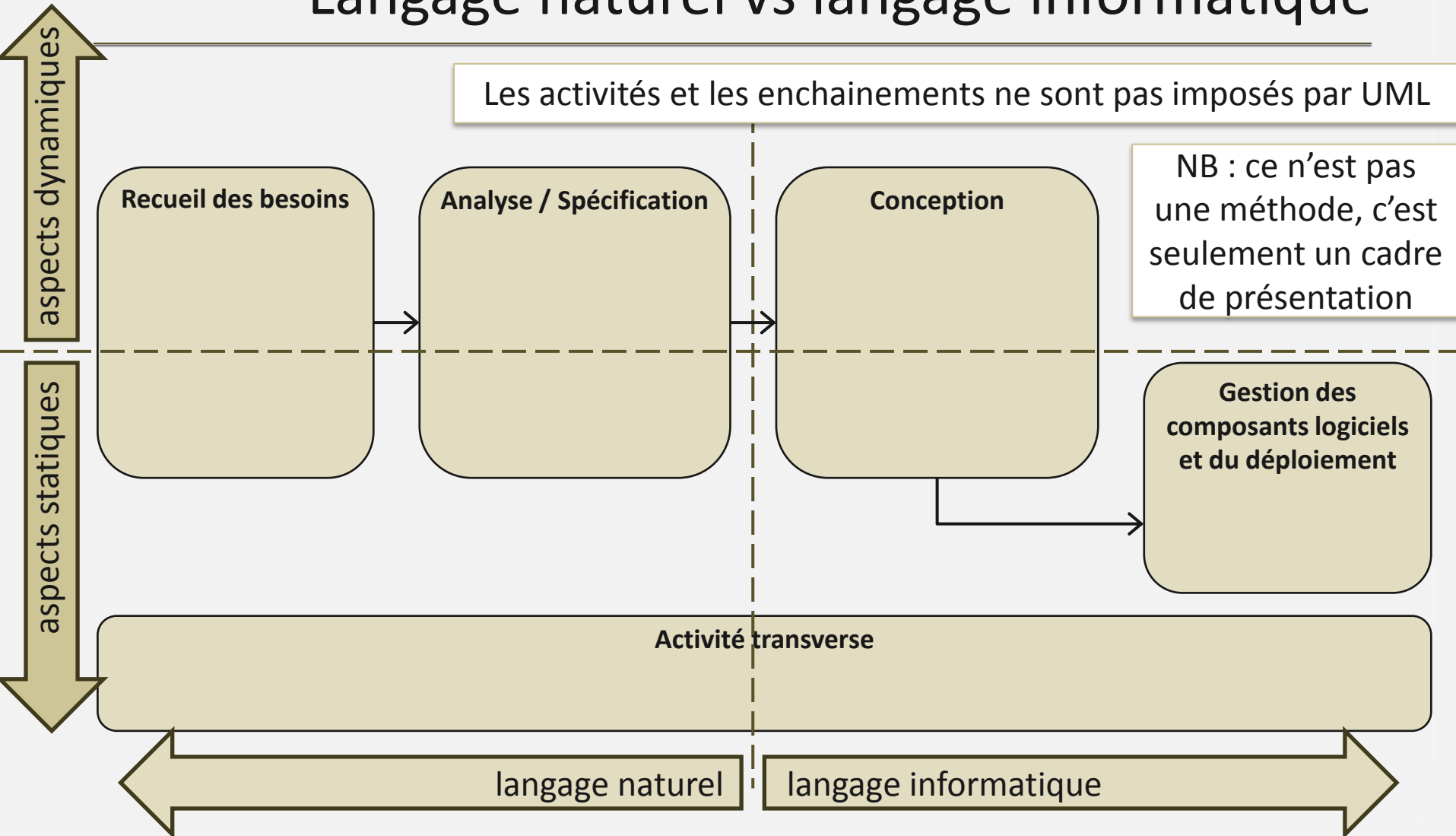
Activités concernées par UML

Aspects dynamiques vs aspects statiques

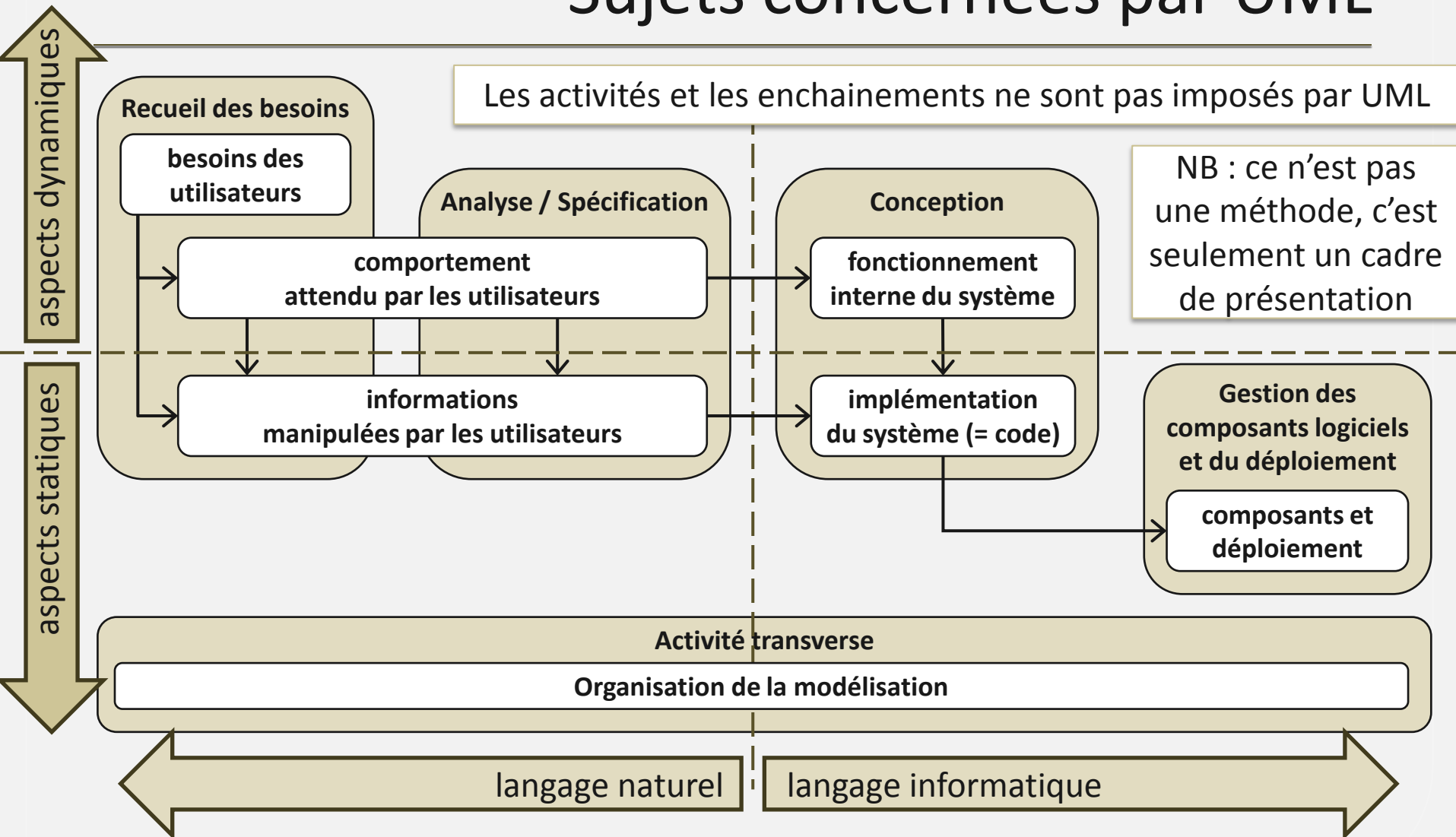


Activités concernées par UML

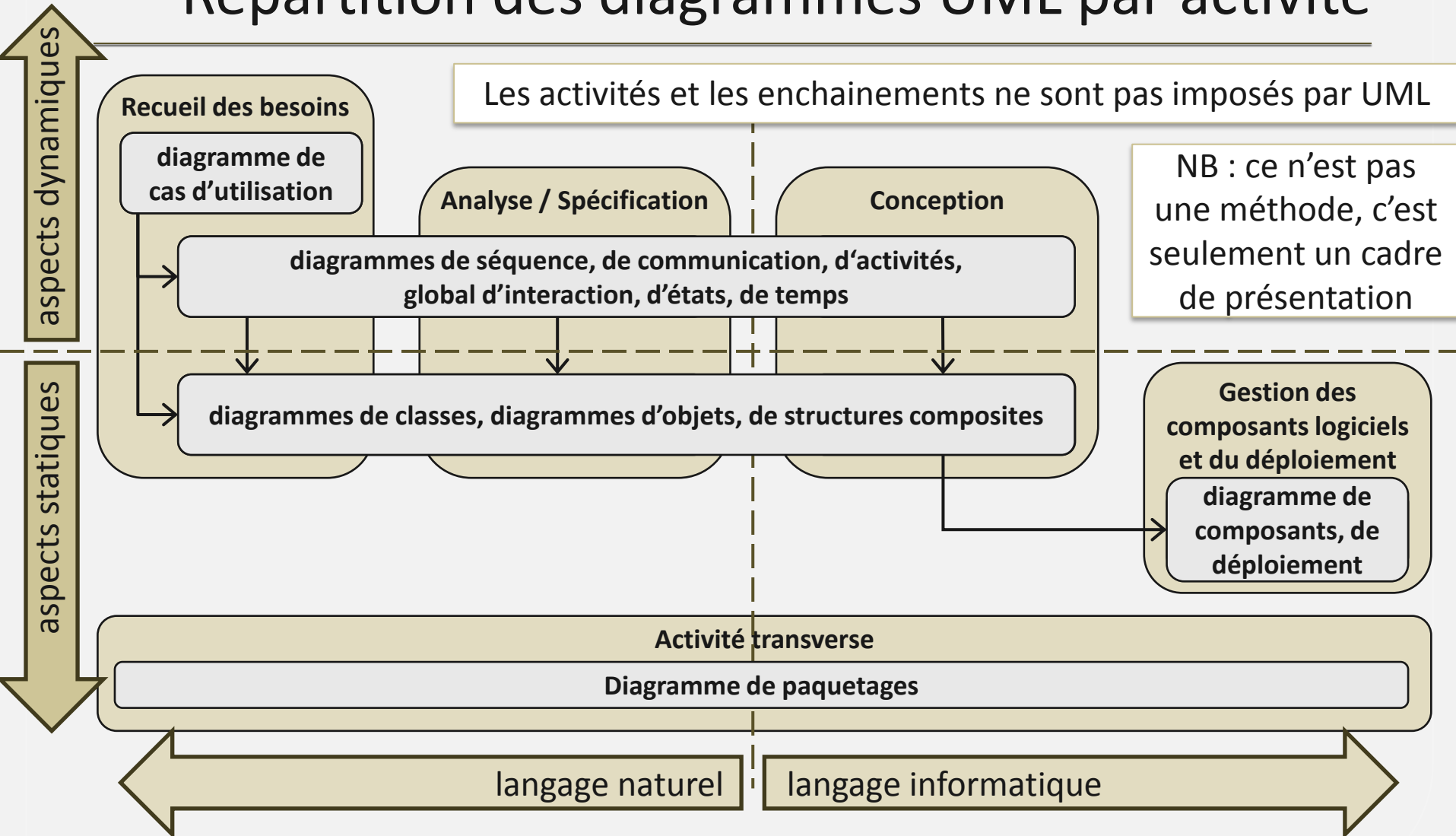
Langage naturel vs langage informatique



Sujets concernés par UML



Répartition des diagrammes UML par activité



Répartition des diagrammes UML par sujet

SUJETS

DIAGRAMMES UML

↑ aspects dynamiques	● Besoins des utilisateurs	● Cas d'utilisation	↑ Diagrammes d'interaction
	● Comportement attendu par l'utilisateur	● Séquence	
	● Fonctionnement interne du système	● Communication	
		● Global d'interaction	
		● Temps	
↓ aspects statiques		● Activités	
		● Etats	
	● Informations manipulées par l'utilisateur	● Classes	
	● Implémentation du système	● Objets	
	● Composants	● Structures composites	
	● Déploiement	● Composants	
	● Transverse	● Déploiement	
		● Paquetages	

UML est utilisable à plusieurs niveaux de détail

- ◎ Esquisse
 - Modélisation approximative destinée à échanger des idées
- ◎ Analyse
 - Modélisation partielle pour illustrer les points à analyser
- ◎ Cadrage
 - Modélisation complète du périmètre du système
- ◎ Spécification
 - Modélisation complète des exigences fonctionnelles
- ◎ Conception, plusieurs niveaux possibles dont
 - Architecture système
 - Modélisation Indépendante de la Plate-forme (*PIM*)
 - Modélisation Spécifique à une Plate-forme (*PSM*)

Conseils de modélisation

- ◎ Rester simple !
 - C'est la principale qualité d'UML
- ◎ En fonction de votre contexte projet
 - Choisir le *bon* diagramme en fonction du sujet modélisé
 - Choisir le *bon* niveau de détail
- ◎ Eviter de faire apparaître trop de détails dans les diagrammes
 - Les *bons* diagrammes sont synthétiques
 - Le détail se trouve dans la documentation associée à chaque élément de modélisation

Conseils de modélisation

- ◉ Limiter la taille des diagrammes (à titre indicatif)
 - Un diagramme reste lisible jusqu'à environ 12 éléments
 - Exception : le diagramme de séquence reste lisible jusqu'à une vingtaine de messages
- ◉ Au-delà, distinguer plusieurs niveaux de détail
 - Un diagramme pour les éléments principaux
 - Un diagramme pour les éléments secondaires attachés à chaque élément principal
 - Avec deux niveaux, on peut modéliser jusqu'à 140 éléments avec 13 diagrammes lisibles

Les outils UML permettent de ...

- ⊙ Réaliser des diagrammes UML (souvent une partie d'UML)
- ⊙ Organiser un modèle UML (*packages*)
- ⊙ Assurer la cohérence d'un modèle UML
 - Un élément peut apparaître sur plusieurs diagrammes
 - Mise à jour automatique des diagrammes suite à la modification d'un élément (renommage, déplacement, ...)
 - Recherches et analyses d'impacts
- ⊙ Travailler en groupe avec une modélisation commune
- ⊙ Produire de la documentation (Word, HTML)
- ⊙ Produire du code logiciel (et se synchroniser)
- ⊙ Effectuer une rétro-ingénierie (*reverse engineering*)