

# Langage de modélisation objet unifié

Langage UML

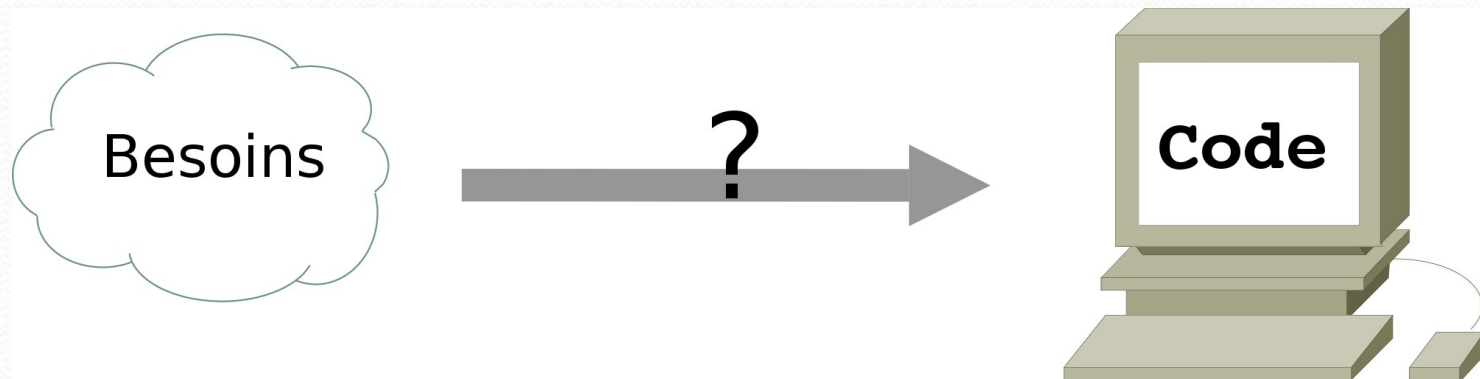


Pr R. EL OUAHBI

# Plan

- Problématique
- Modélisation
- Approche classique
- Approche Objet
- L'apport de l'approche OO
- Historique d'UML
- Langage UML
- Diagramme UML
- Modélisation par UML

# Problématique



Quelle méthode pour passer de l'expression des besoins au code de l'application ?



# Modèles et modélisation

**Modéliser** : comprendre et représenter

**Un modèle est une abstraction de la réalité**

Abstraction : ensemble des caractéristiques essentielles d'une entité, retenues par un observateur

**Un modèle est une vue subjective mais pertinente de la réalité**

Un modèle ne représente pas une **réalité absolue** mais reflète des aspects importants de la réalité, il en donne donc une vue juste et pertinente

# Exemple de modèles

## **Modèle météorologique :**

à partir de données (nuage, vents, pression atmosphérique...), permet de prévoir les conditions climatiques pour les jours à venir

## **Modèle économique :**

à partir d'hypothèses macro-économiques (évolution du chômage, taux de croissance...), permet de simuler l'évolution de cours boursiers

## **Modèle démographique :**

définit la composition d'un panel d'une population et son comportement, dans le but d'augmenter l'impact de démarches commerciales, etc...

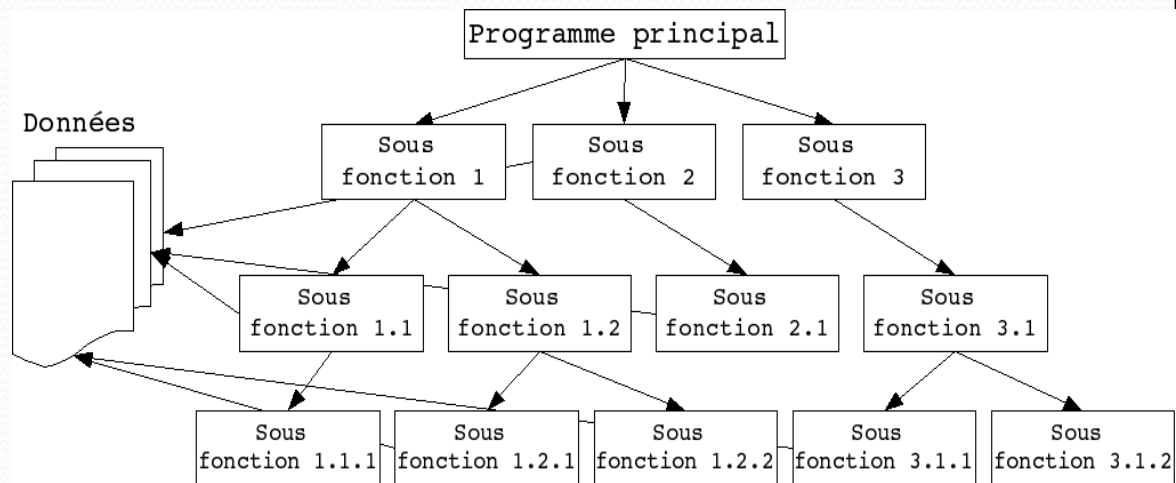
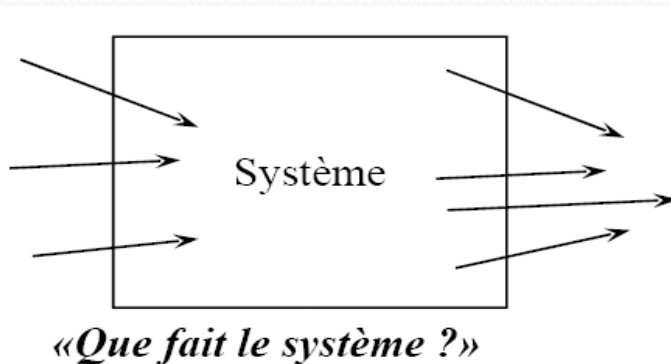


# Caractéristiques des modèles

- Le caractère abstrait d'un modèle doit notamment permettre :
  - de faciliter la compréhension du système étudié
    - Un modèle réduit la complexité du système étudié.
  - de simuler le système étudié
    - Un modèle représente le système étudié et reproduit ses comportements
- Un modèle réduit (décompose) la réalité, dans le but de disposer d'éléments de travail exploitables par des moyens mathématiques ou informatiques

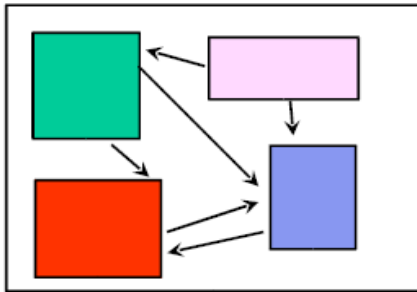
# Méthodes fonctionnelles

- Accent mis sur ce que fait le système (ses fonctions)
- Identification des fonctions du système puis décomposition en sous-fonctions, récursivement, jusqu'à obtention de fonctions élémentaires, implémentables directement
- La fonction détermine la structure

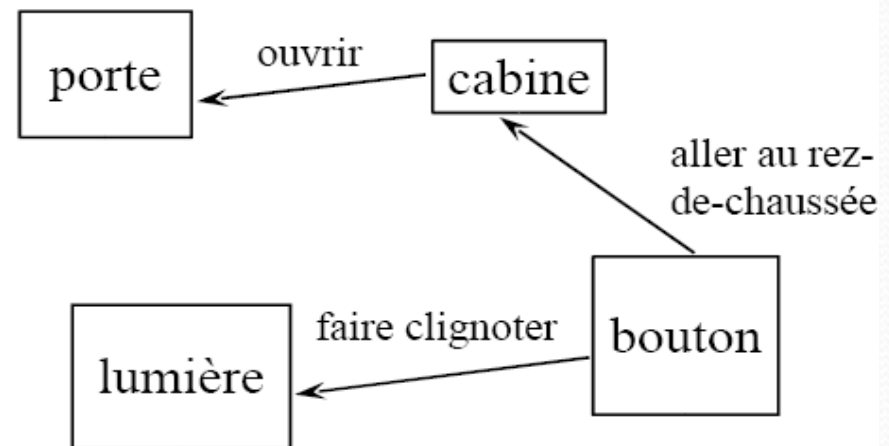


# Méthodes Orientée Objet

- Accent mis sur ce qu'est le système (ses composants)
- Identification des composants du système : les objets- fonction = collaboration entre objets
- Les fonctions sont indépendantes de la structure
- Un objet intègre à la fois des données et des opérations



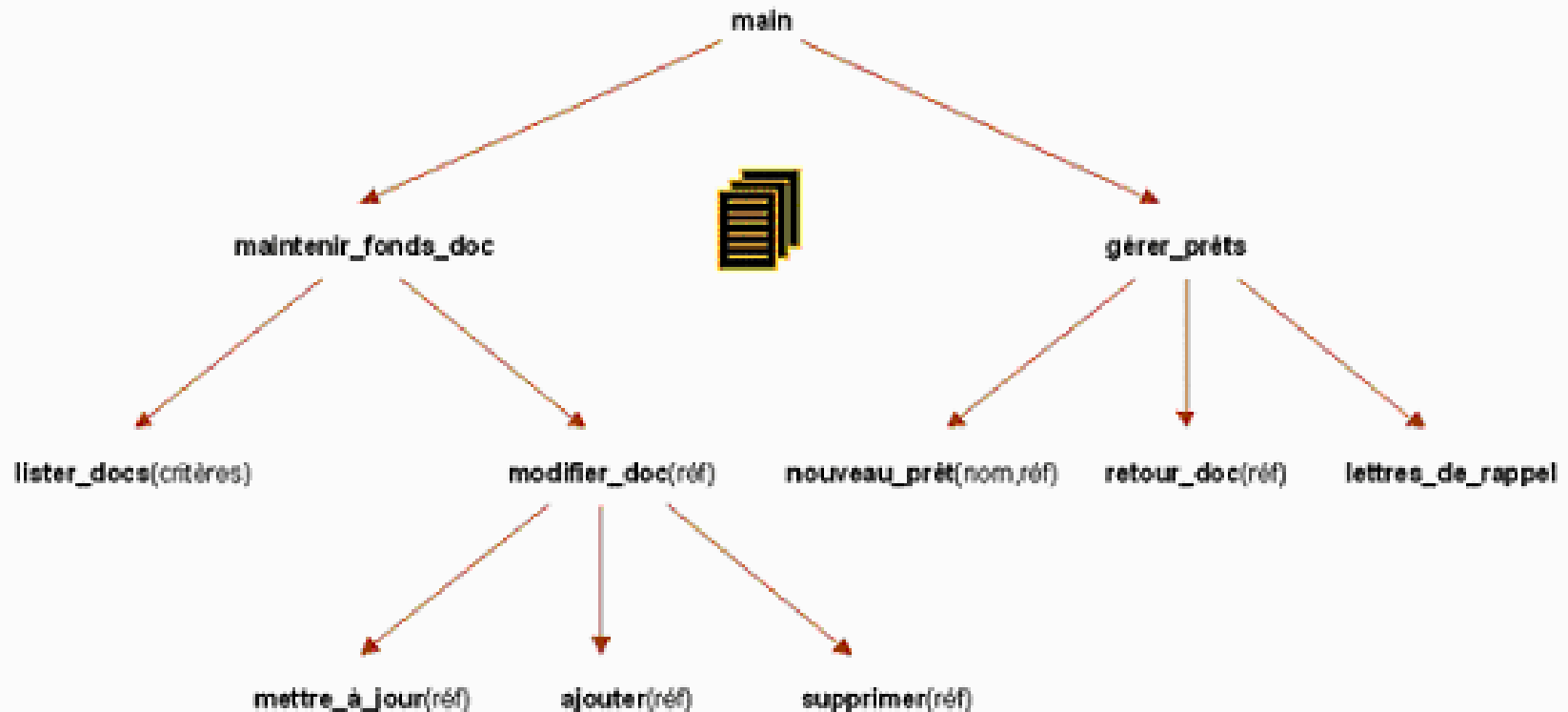
«De quoi se compose le système ?»





# Approche fonctionnelle vs approche objet

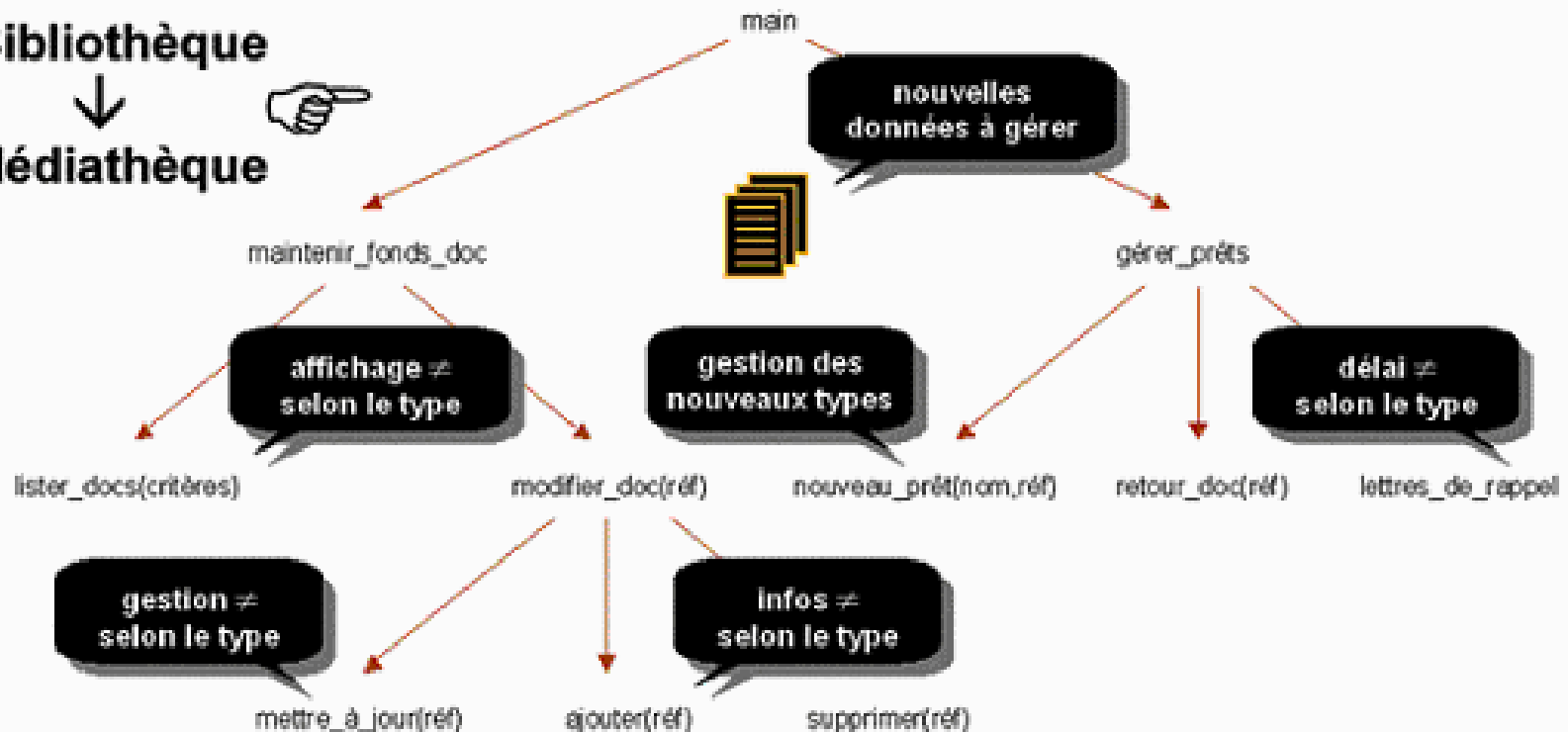
Exemple de découpe fonctionnelle d'un logiciel dédié à la gestion d'une bibliothèque :



# Approche fonctionnelle vs approche objet

Le passage d'une bibliothèque à une médiathèque implique le retouche du logiciel sa globalité

**Bibliothèque**  
↓  
**Médiathèque**

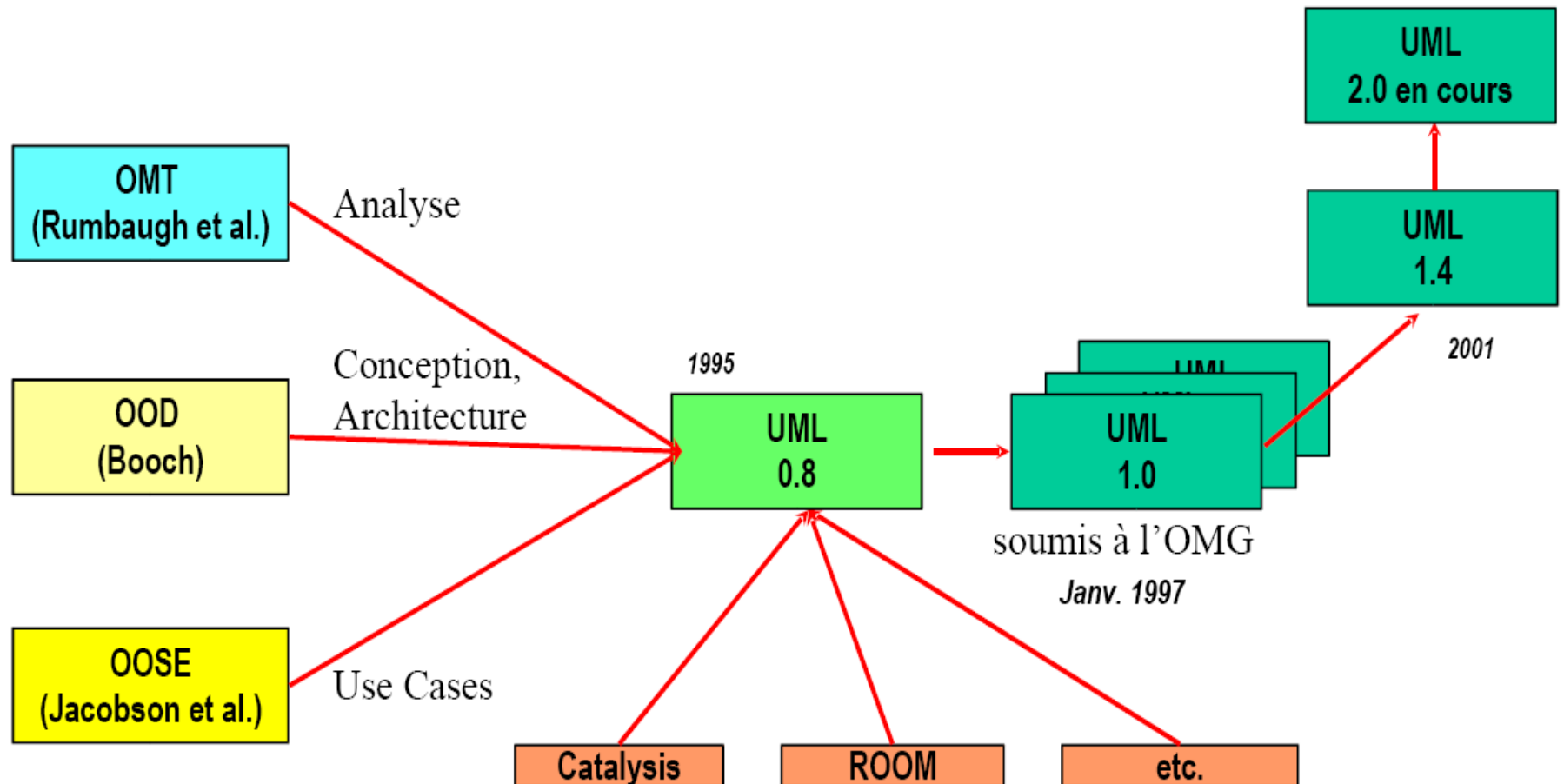


# Qu'apporte la modélisation objet

- Plus grande indépendance du modèle par rapport aux fonctionnalités demandées.
- Des fonctionnalités peuvent être rajoutées ou modifiées, le modèle objet ne change pas.
- Plus proche du monde réel.



# Historique d'UML



# Qu'est-ce que UML ?

- **UML** = **U**nified **M**odeling **L**anguage  
for Object-Oriented Development
- C'est le 1er standard international en  
conception de système d'information.
- Il provient de l'unification de différents modèles  
Orientés-Objet.

# UML : le standard

- UML est riche (il couvre toutes les phases d'un cycle de développement).
- UML est ouvert (il est indépendant du domaine d'application et des langages d'implémentation).
- les outils qui supportent UML se multiplient (GDPro, Umbrello, Objectteering, OpenTool, Rational Rose, Rhapsody, STP, Visio, Visual Modeler, WithClass...).



# UML : méthode ou modèle ?

Une méthode comporte :

- un **langage de modélisation** : une notation utilisée pour décrire les éléments de modélisation
- un **processus** décrivant les étapes et les tâches à effectuer pour mener à bien la conception.

Ce dernier point est absent de UML, donc :

- *UML n'est pas une méthode* mais un langage d'expression des éléments de la modélisation.

# Les 13 diagrammes UML

- **Diagrammes structurels ou diagrammes statiques (*UML Structure*)**
- **Diagrammes comportementaux ou diagrammes dynamiques (*UML Behavior*)**



# Diagrammes structurels ou diagrammes statiques (*UML Structure*)

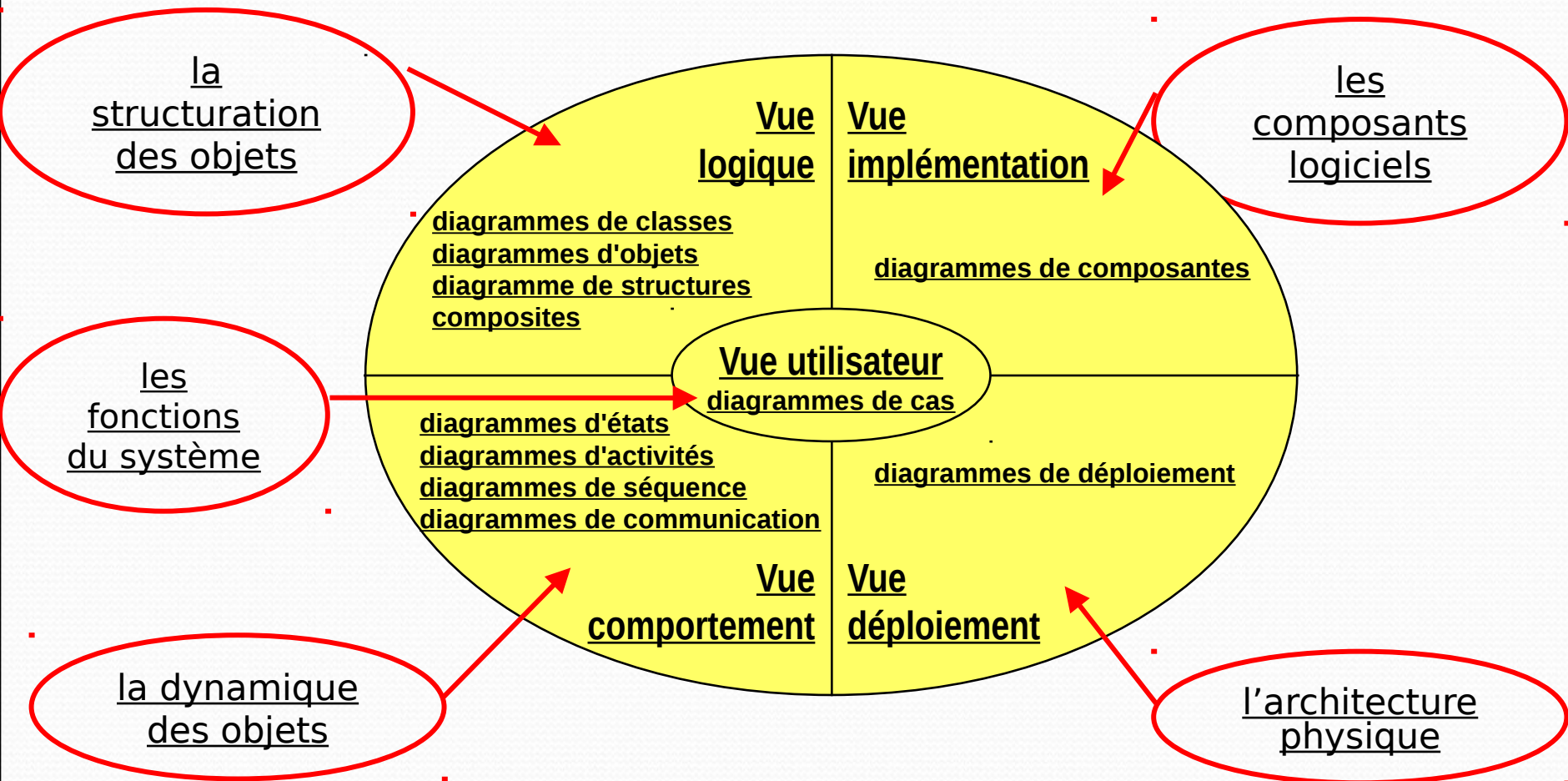
- Diagramme de classes (*Class diagram*)
- Diagramme d'objets (*Object diagram*)
- Diagramme de composants (*Component diagram*)
- Diagramme de déploiement (*Deployment diagram*)
- Diagramme de paquetages (*Package diagram*)
- Diagramme de structures composites (*Composite structure diagram*)



# Diagrammes comportementaux ou diagrammes dynamiques (UML Behavior)

- Diagramme de cas d'utilisation (*Use case diagram*)
- Diagramme d'activités (*Activity diagram*)
- Diagramme d'états-transitions (*State machine diagram*)
- **Diagrammes d'interaction (*Interaction diagram*)**
  - Diagramme de séquence (*Sequence diagram*)
  - Diagramme de communication (*Communication diagram*)
  - Diagramme global d'interaction (*Interaction overview diagram*)
  - Diagramme de temps (*Timing diagram*)

# Comment modéliser avec UML





# Vue utilisateur

- La vue utilisateur présente le système du point de vue du propriétaire et de l'utilisateur final.
- La vue présente les buts et objectifs du système.
- La vue présente les requis et fonctionnalités du système.



# Vue logique

- La vue logique présente les aspects statiques et structuraux du problème et de la solution.
- Décomposition orientée-objet
  - Décomposition en objets et classes.
  - Regroupement en paquetages.
  - Connexions par héritage, association, etc.
  - Accent sur l'abstraction, l'encapsulation, l'uniformité.
  - Réalisation des scénarios des cas d'utilisation.

# Vue comportement

- La vue comportement présente les aspects dynamiques du problème et de la solution.
- La vue comportement présente les interactions et collaborations entre les éléments du système.



# Vue implémentation

- La vue implémentation présente les aspects statiques, structuraux et dynamiques de la réalisation de la solution.
- La vue implémentation présente l'organisation du code de la solution.



# Vue déploiement

- La vue déploiement présente les aspects statiques et dynamiques de l'exécution de la solution.
- La vue déploiement présente les éléments physiques de la solution (processeurs, périphériques, ...)