Génie Logiciel

Chapitre 2
Modélisation avec UML

Niveau: 3ième année Licence informatique

Année: 2025/2026

Plan

- Introduction
- UML: Langage de modélisation
- Les diagrammes UML
- Diagrammes structurels (statiques)
- Diagrammes comportementaux (dynamiques)
- Aller plus loin avec UML

Introduction

Modélisation

- Qu'est qu'un modèle ?
  - ✓ Une simplification de la réalité

- Pourquoi modéliser ?
  - ✓ Les modèles permettent de mieux comprendre le système que l'on développe

- Nous construisons des modèles pour les systèmes complexes parce que nous ne sommes pas en mesure d'appréhender de tels systèmes dans leur intégralité

systèmes complexes: climat, cerveau, économie, réseaux sociaux, écosystèmes, organismes vivants, systèmes informatiques à grande échelle, etc

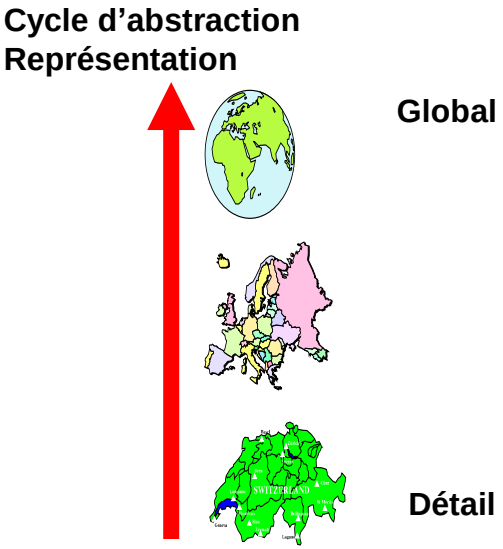
Modélisation (2)

- Le choix des modèles à créer a une forte influence sur la manière d'aborder un problème et sur la nature de sa solution
  - ✓ Un modèle
    - met l'accent sur certains aspects
    - en ignore ou simplifie d'autres

- Tous les modèles peuvent avoir différents niveaux de précision

- Les meilleurs modèles ne perdent pas le sens de la réalité

- Parce qu'aucun modèle n'est suffisant à lui seul,
  - ✓ il est préférable de décomposer un système important en un ensemble de petits modèles presque indépendants

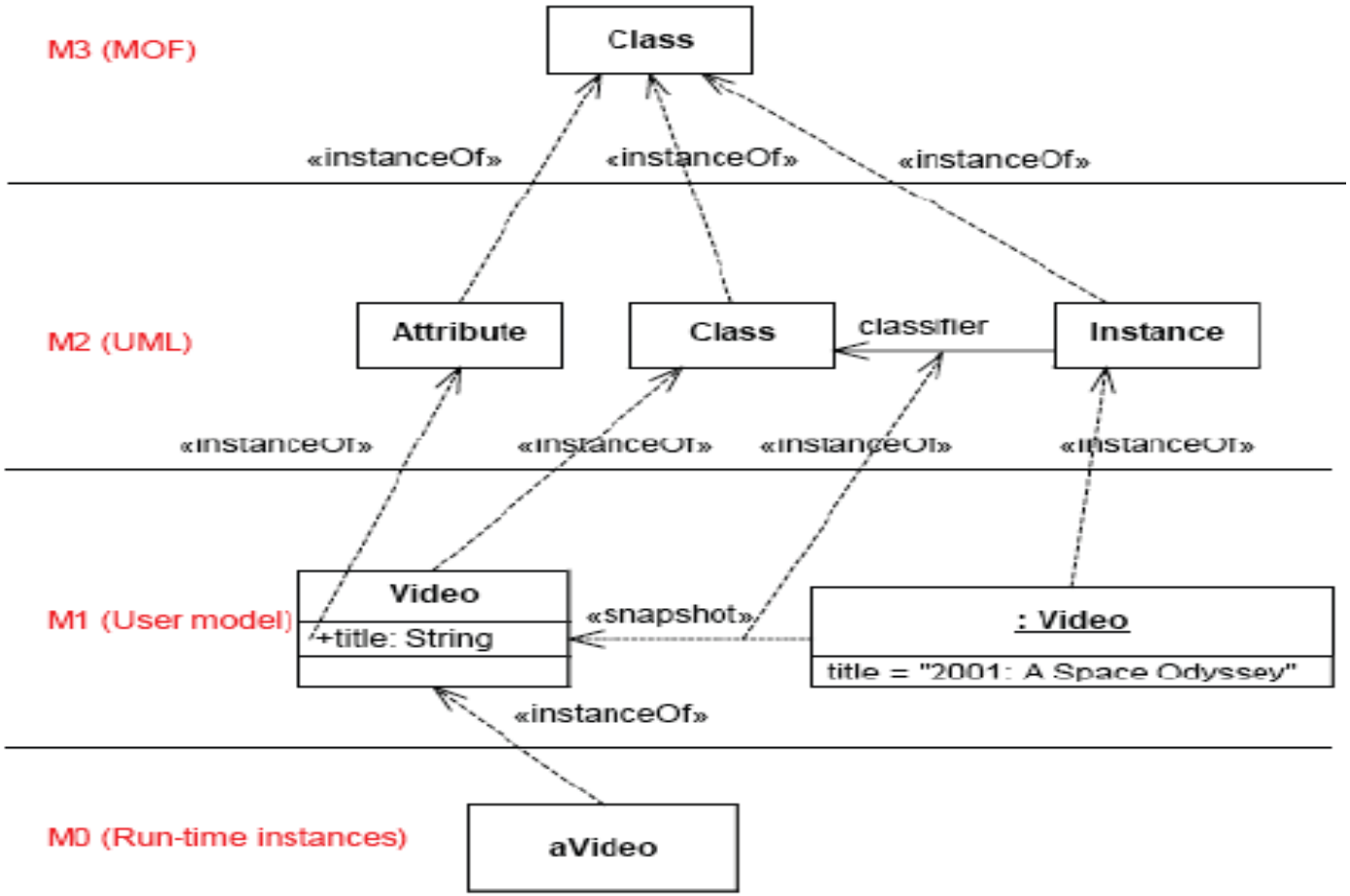


Niveaux de modélisation

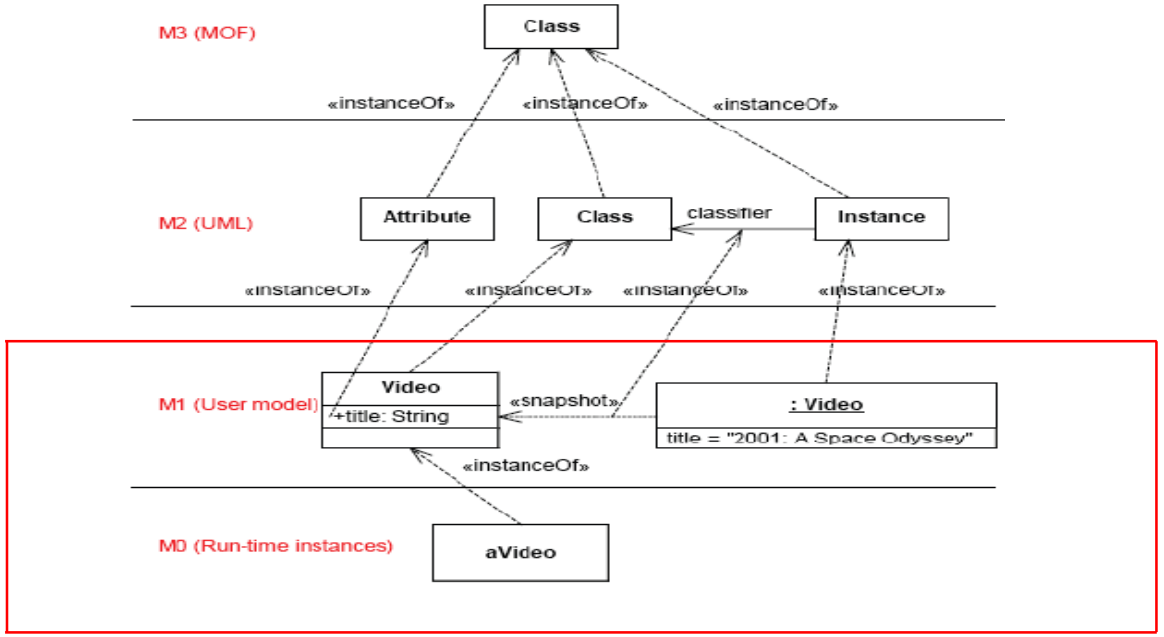
- L'OMG (Object Management Group) définit 4 niveaux de modélisation
  - ✓ M0 : système réel, système modélisé
  - ✓ M1 : modèle du système réel défini dans un certain langage
  - ✓ M2 : méta-modèle définissant ce langage
  - ✓ M3 : méta-méta-modèle définissant le méta-modèle

- Le niveau M3 est le MOF
  - ✓ Meta-Object Facility
  - ✓ Dernier niveau, il est méta-circulaire : il peut se définir lui-même
  - ✓ Pour l'OMG, il est le méta-méta-modèle unique servant de base à la définition de tous les méta-modèles
  - ✓ Définit comment construire des méta-modèles

Niveaux de modélisation (2)



Niveaux de modélisation (3)



- Seules les deux couches user model et run time instance sont destinées à l'utilisateur
- Les niveaux M2 et M3 sont surtout destinés aux
  - ✓ Concepteurs de langages
  - ✓ Outils de modélisation
  - ✓ Frameworks (UML, EMF, etc)

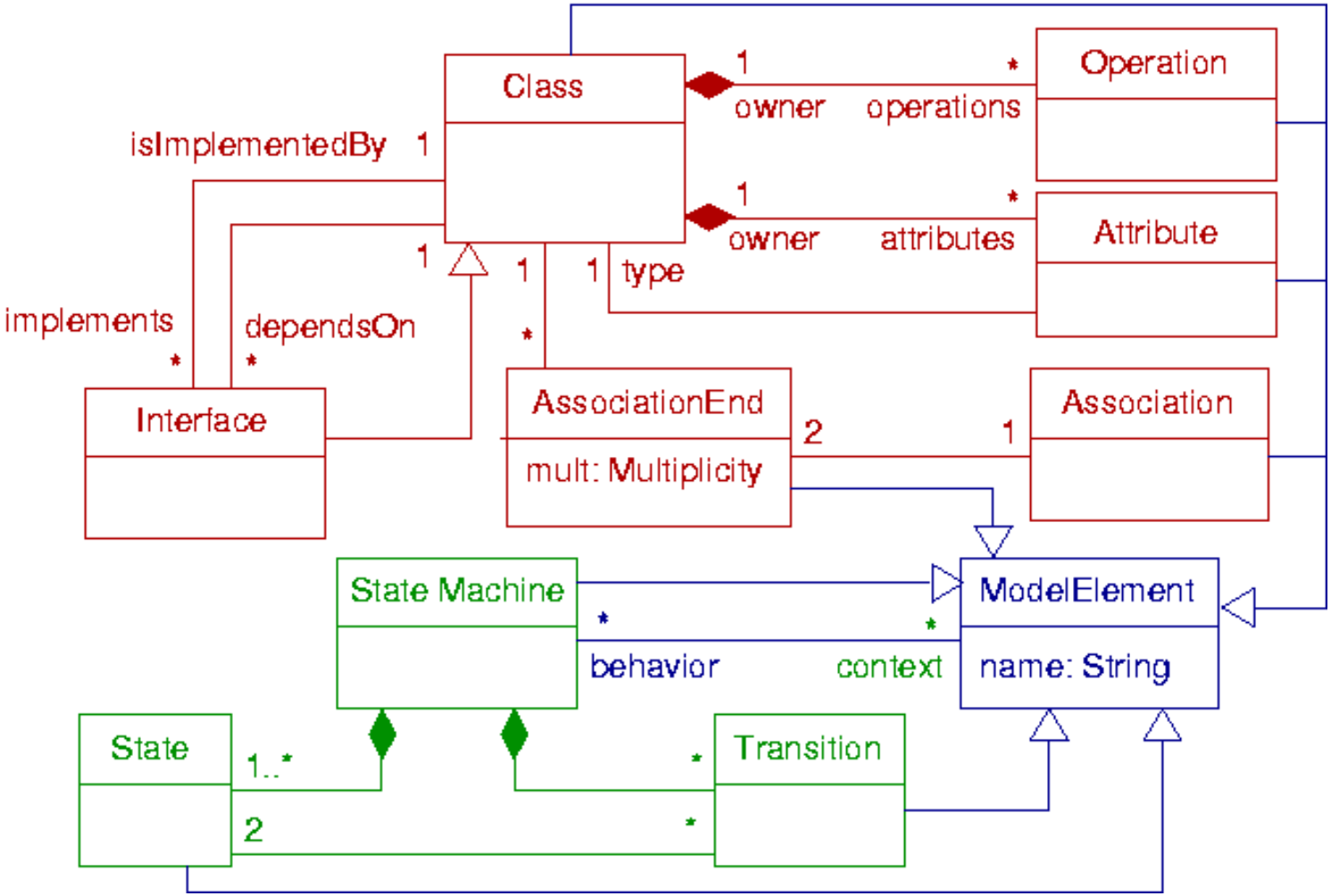
UML: Langage de modélisation

UML

- Unified Modeling Language
- Un langage de modélisation unifié
- Résultat de la fusion de plusieurs langages de modélisation graphique orientés objet: OMT, Booch et OOSE
- Rapidement devenu le standard incontournable de la modélisation orienté objet
- Un standard de l'OMG depuis 1997
- UML selon l'OMG :
  - ✓ Langage visuel dédié à la spécification, la construction et la documentation des artefacts d'un système logiciel

- L'OMG définit le méta-modèle d'UML

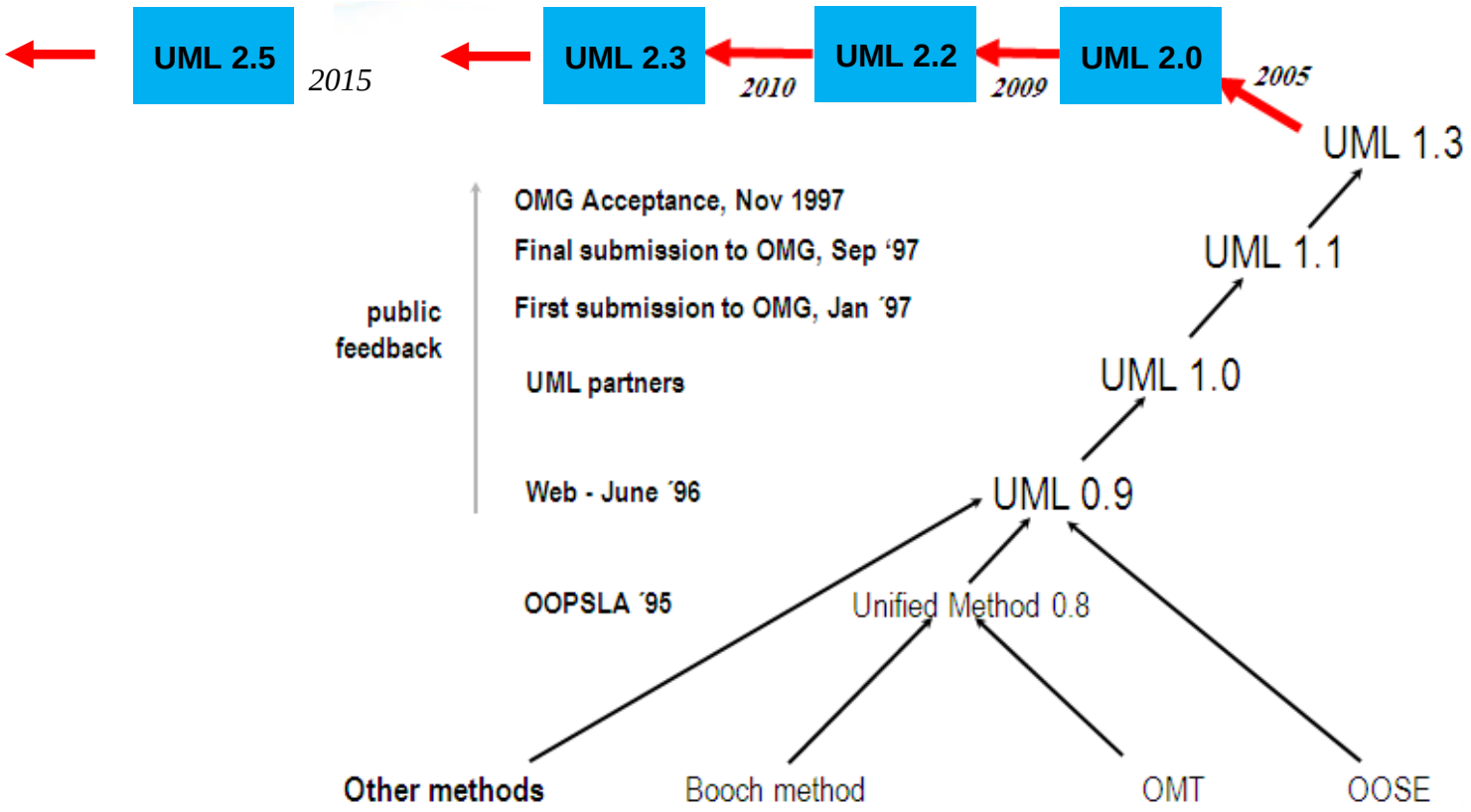
Méta-modèle UML - simplifié



OMG

- Object Management Group: www.omg.org
- Un organisme international à but non lucratif
- Fondé en 1989 pour standardiser et promouvoir le modèle objet
- Plus de 700 membres
- Fondateurs : American Airlines, Canon, Data General, Gold Hill Hewlett-Packard, Philips, Prime, HP, Sun, Soft-switch Unisys, 3Com
- Normes: UML, XMI, MOF, OCL, ...

Evolution d'UML



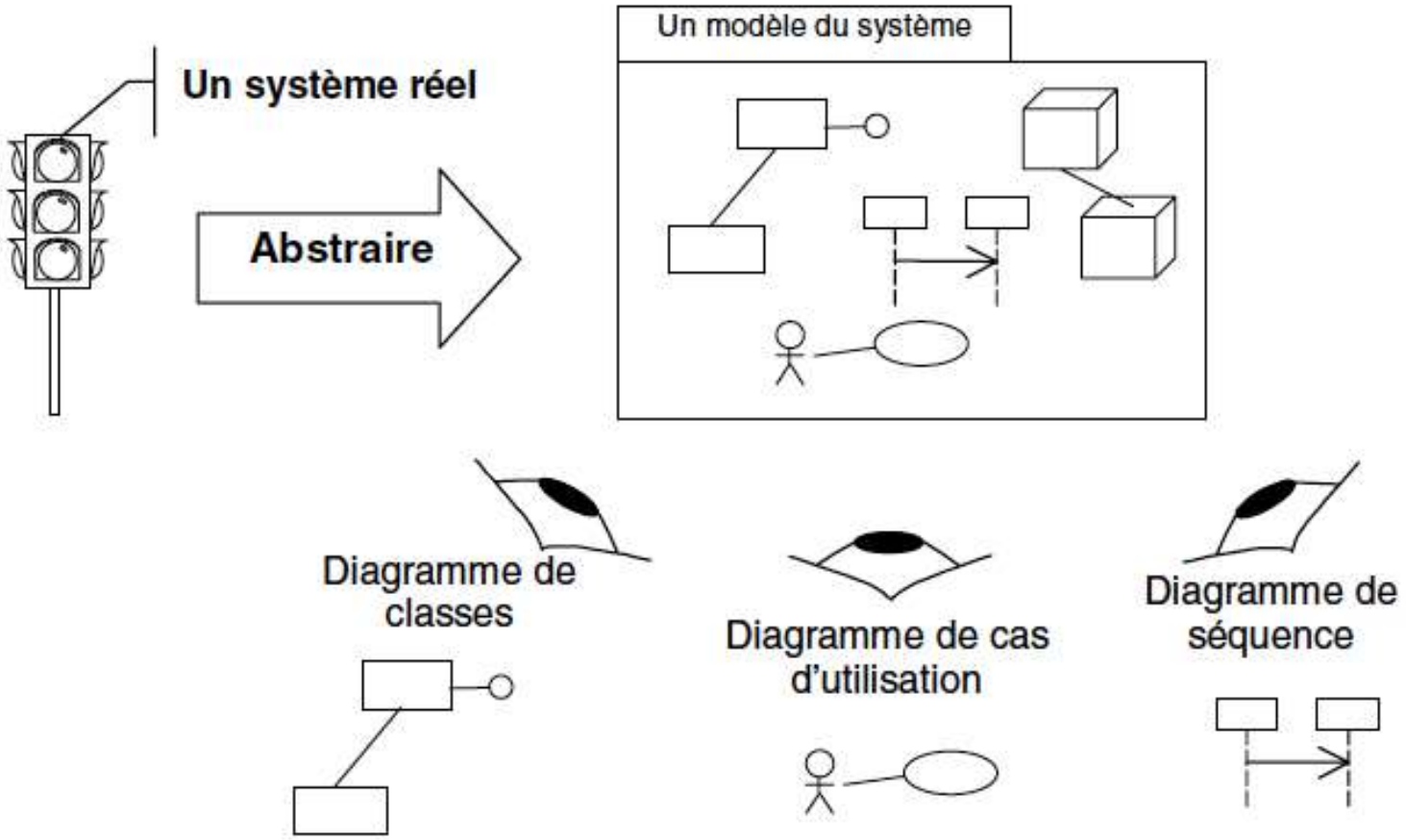
- Les travaux d'amélioration d'UML 2 se poursuivent …
  - ✓ 2017, UML 2.5.1
  - ✓ 2023, UML 2.5.2 (corrections mineures)

Langage UML

- UML est un langage et pas une méthode
- Permet de modéliser toutes les étapes du développement d'une application de l'analyse au déploiement
- S'articule autour de plusieurs types de diagrammes
- Les diagrammes décrivent les modèles

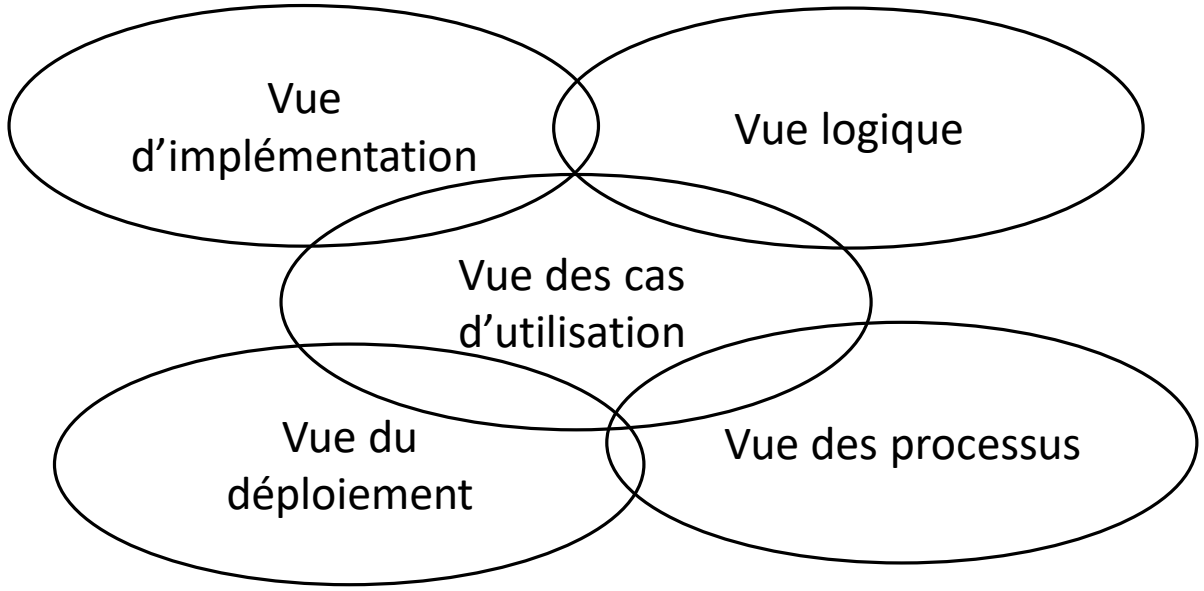
- Chaque diagramme donne un point de vue différent sur le système

Diagrammes UML - Points de vue sur le système



Notion de vue

- UML est fondé sur la notion de vue
  - ✓ traduit le fait qu'il y a plusieurs manières de regarder un système
  - ✓ UML suit le modèle « 4+1 vues »



- Ces vues se concrétisent en 14 types de diagrammes

Notion de vue (2)

- Vue logique
  - ✓ s'attache aux aspects statiques du système en termes d'objets et de classes, enrichis de descriptions dynamiques en termes d'activités, de séquences et de communications.
- Vue des processus
  - ✓ s'intéresse aux flots d'exécution et à leur synchronisation, en termes de tâches, threads et processus, avec leurs états et leurs interactions.
- Vue de la réalisation
  - ✓ décrit l'organisation des composants logiciels.
- Vue du déploiement
  - ✓ spécifie les ressources matérielles et l'implantation des composants logiciels sur ces ressources.
- Vue des cas d'utilisation
  - ✓ donne une vision d'ensemble des finalités du système en termes d'acteurs, de fonctionnalités et de scénarios d'utilisation.

Les diagrammes UML

Les diagrammes d'UML

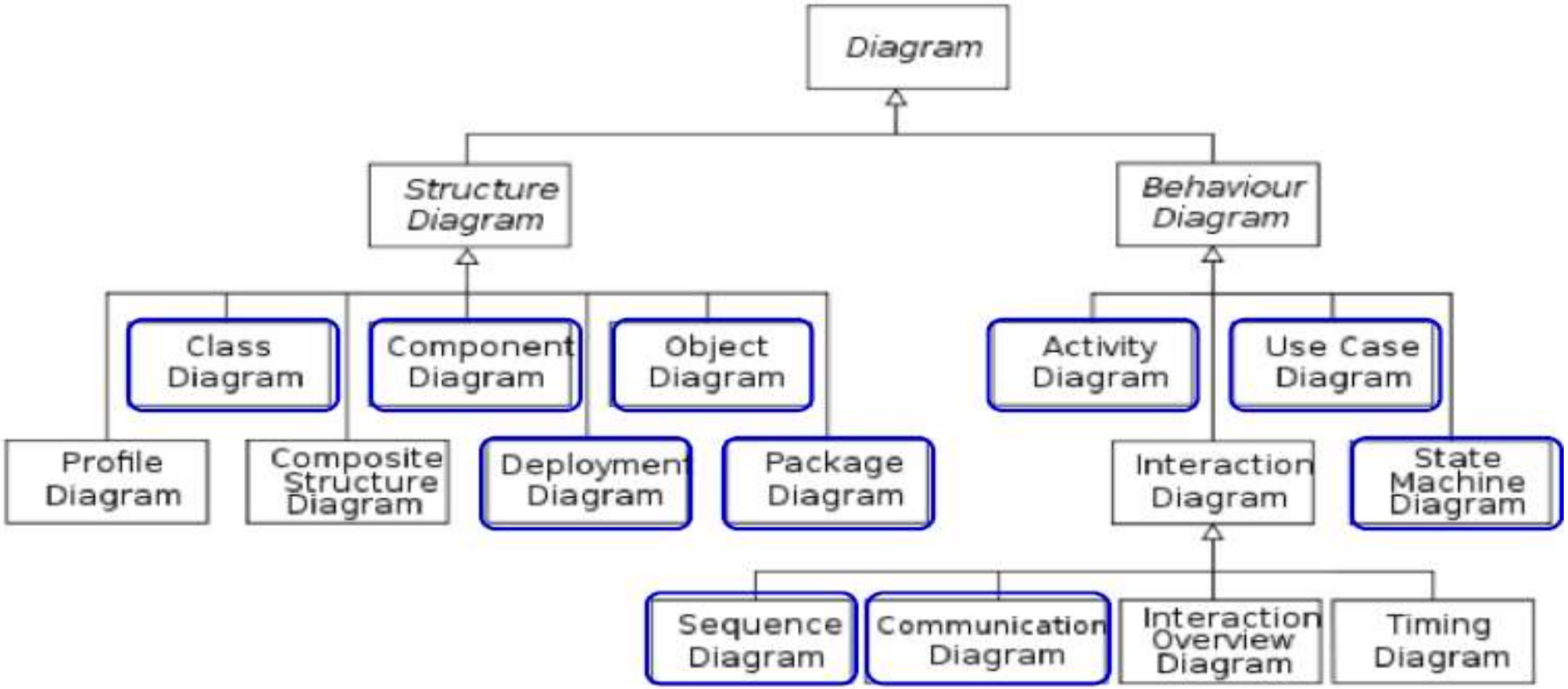
- 14 diagrammes depuis UML 2.3

- Classés en deux catégories:

1) Diagrammes de structure (statiques) :
   - ✓ Permettent de décrire la structure d'un système selon plusieurs points de vue (classes, composants, nœuds, objets, packages, ..)

2) Diagrammes de comportement (dynamiques) :
   - ✓ Permettent de décrire le comportement d'un système selon plusieurs points de vue (temporel, changement d'état, ..)

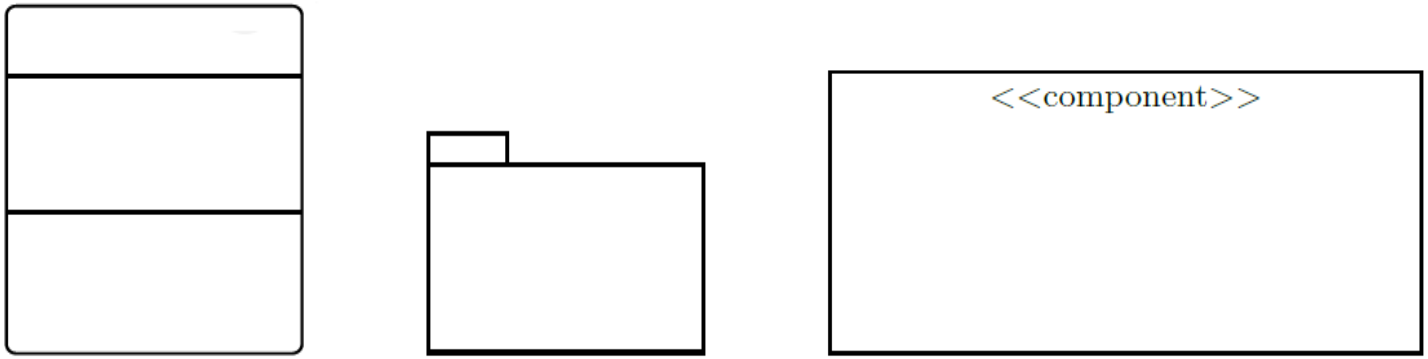
Les diagrammes d'UML (2)



- ✓ Et un langage pour exprimer des contraintes : OCL

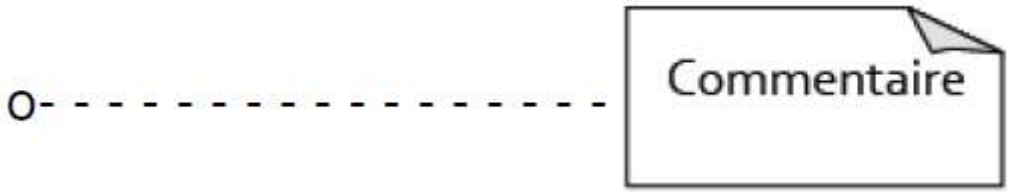
Notations communes

- Classeur : a une forme rectangulaire et permettant de représenter plusieurs éléments dans de différents diagrammes
- Package (paquetage) : un regroupement d'éléments de système ou de diagrammes
- Stéréotype : annotation entourée par <<nomAnnotation>> permettant d'ajouter une précision sur l'élément annoté

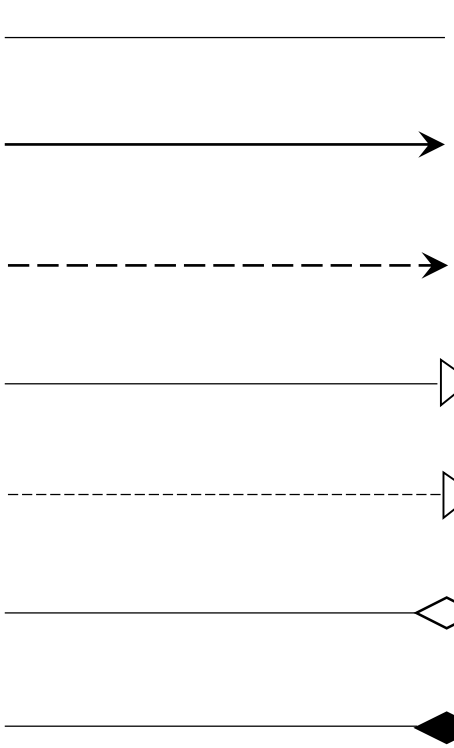


Notations communes (2)

- Valeurs marquées :
  - ✓ Ajout d'une propriété à un élément de modélisation
  - ✓ Notation : { nom1 = valeur1, ..., nomn = valeurn}
  - ✓ Valeurs marquées prédéfinies (ex. : derived : Bool)
  - ✓ Personnalisées (ex. : auteur : Chaîne)
- Commentaires :
  - ✓ Information en langue naturelle
  - ✓ Notation :



Les flèches en UML



Association bidirectionnelle

Association unidirectionnelle

Dépendance

Héritage

Implémentation

Agrégation

Composition

Diagrammes structurels (statiques)

Diagramme de classes

- Représente la description statique du système
- On intègre dans chaque classe
  - ✓ la partie dédiée aux données
  - ✓ et celle consacrée aux traitements
- C'est le diagramme pivot de l'ensemble de la modélisation d'un système

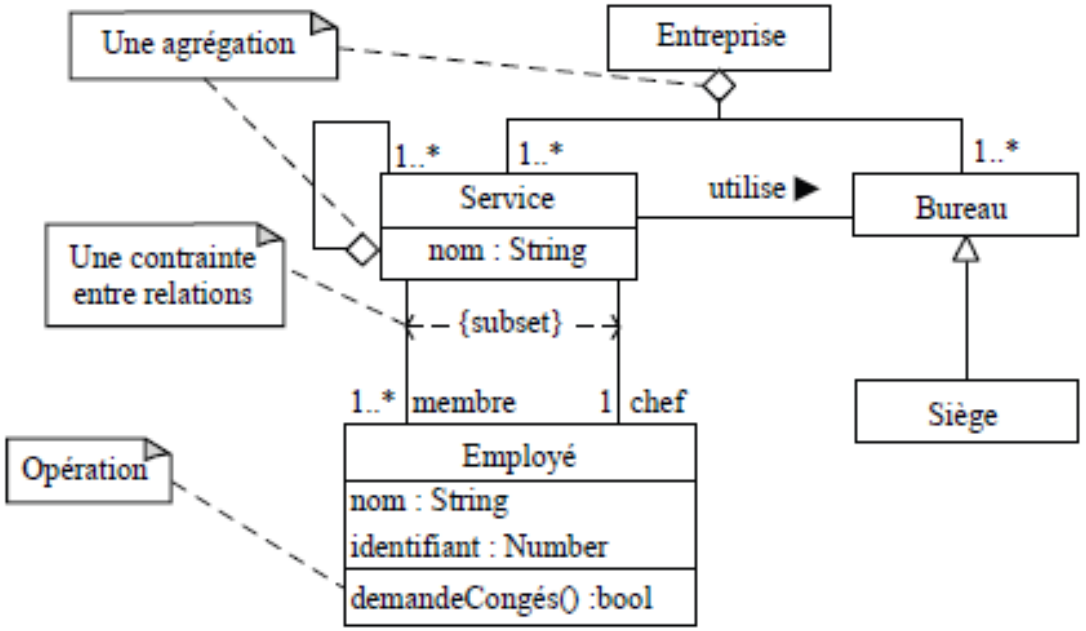


Diagramme d'objets

- Sert à illustrer des structures de classes compliquées en montrant des exemples d'instances
- Utilisé en analyse pour vérifier l'adéquation d'un diagramme de classes à différents cas possibles

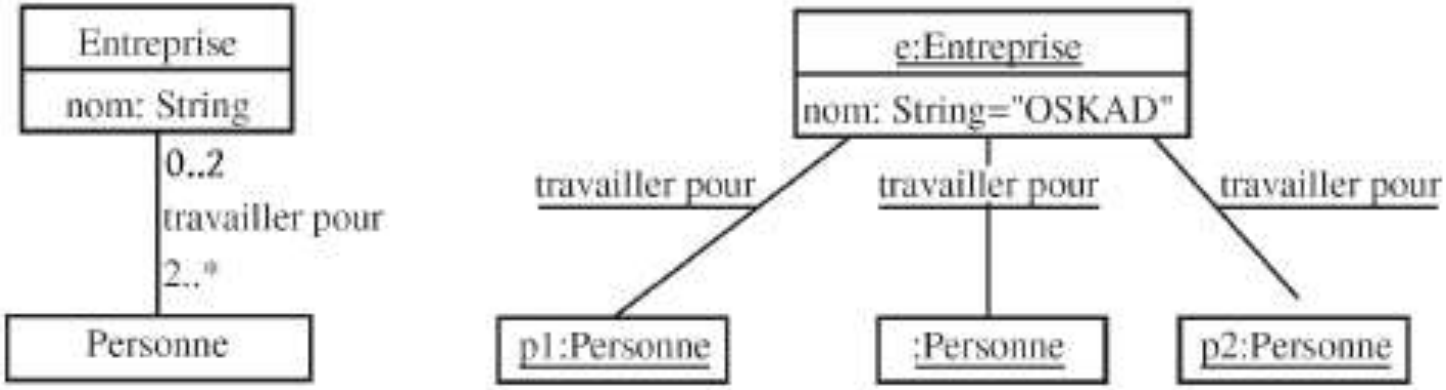


Diagramme de paquetages

- Montre les paquetages et, éventuellement, les relations entre eux

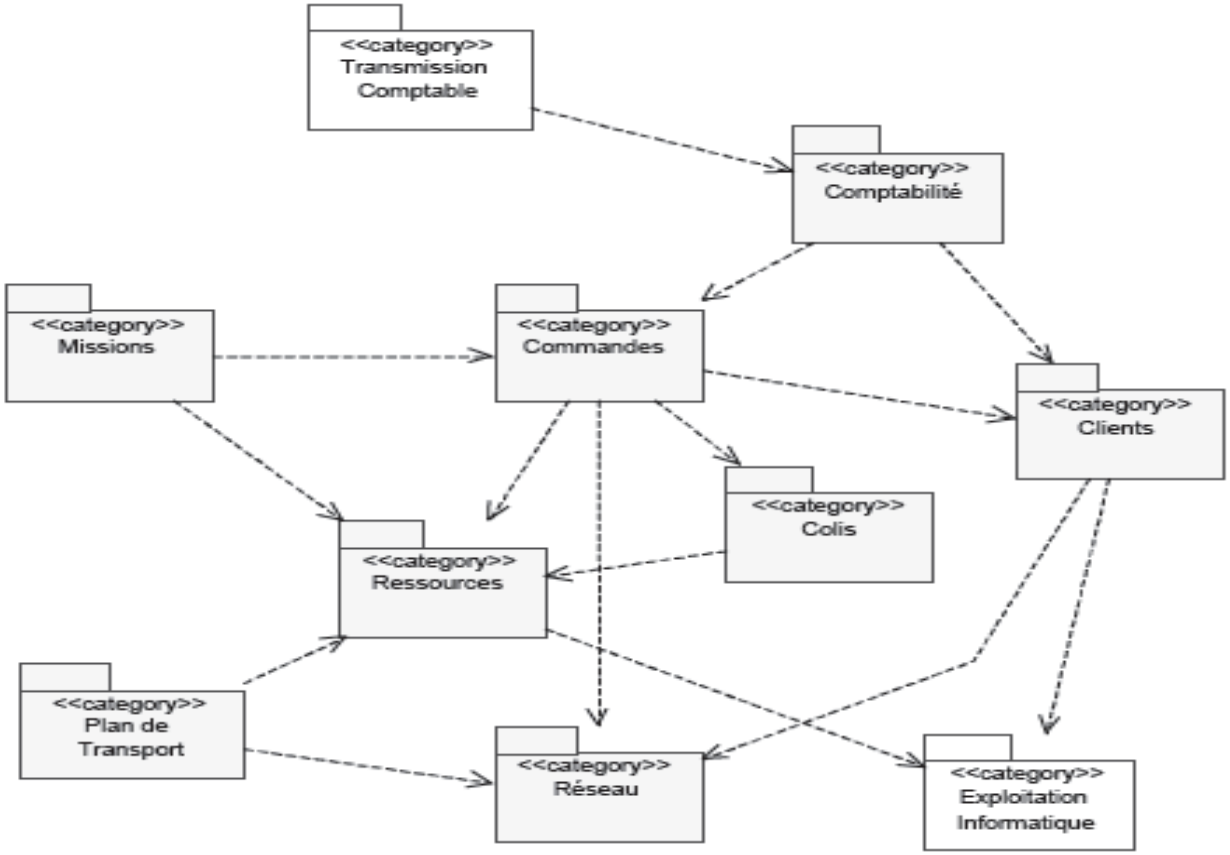


Diagramme de composants

- Représente les concepts connus de l'exploitant pour
  - ✓ installer et dépanner le système
- Il s'agit dans ce cas de déterminer la structure des composants d'exploitation
  - ✓ qui sont les librairies dynamiques, les instances de bases de données, les applications, les progiciels, les objets distribués, les exécutables, etc.

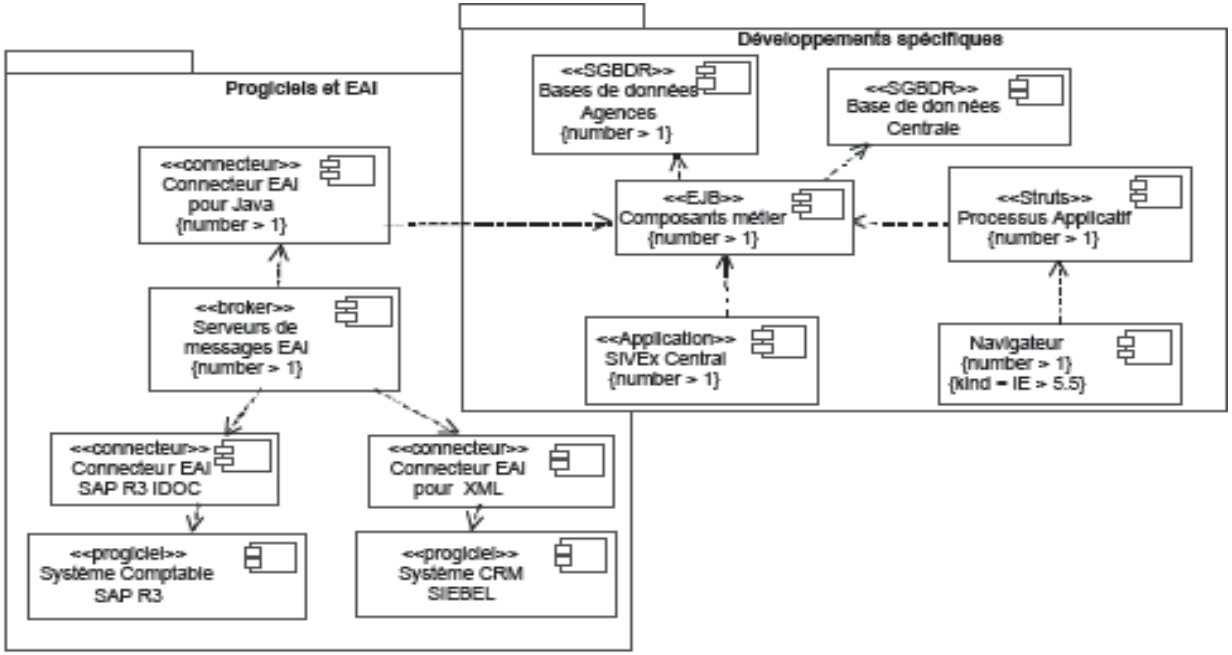


Diagramme de déploiements

- Correspond à la fois
  - ✓ à la structure du réseau informatique qui prend en charge le système logiciel,
  - ✓ et la façon dont les composants d'exploitation y sont installés.

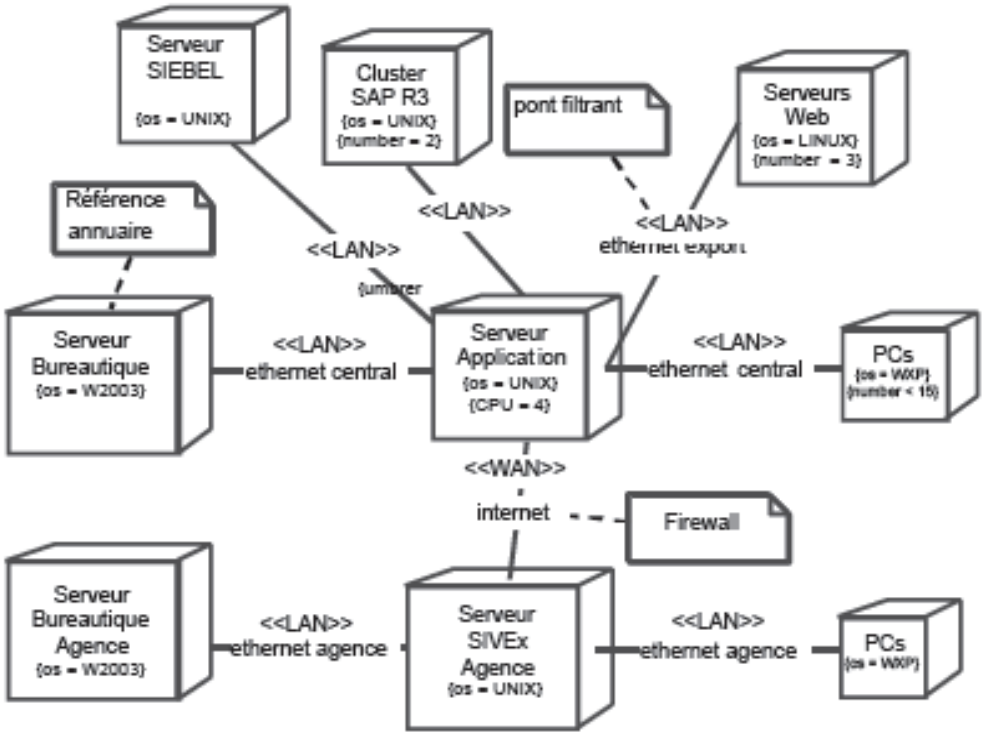


Diagramme de structures composites

- Décrit la composition d'un objet complexe lors de son exécution
- Ce diagramme est propre à UML 2
- Il introduit la notion de structure d'un objet complexe, tel qu'il se présente en phase de run-time

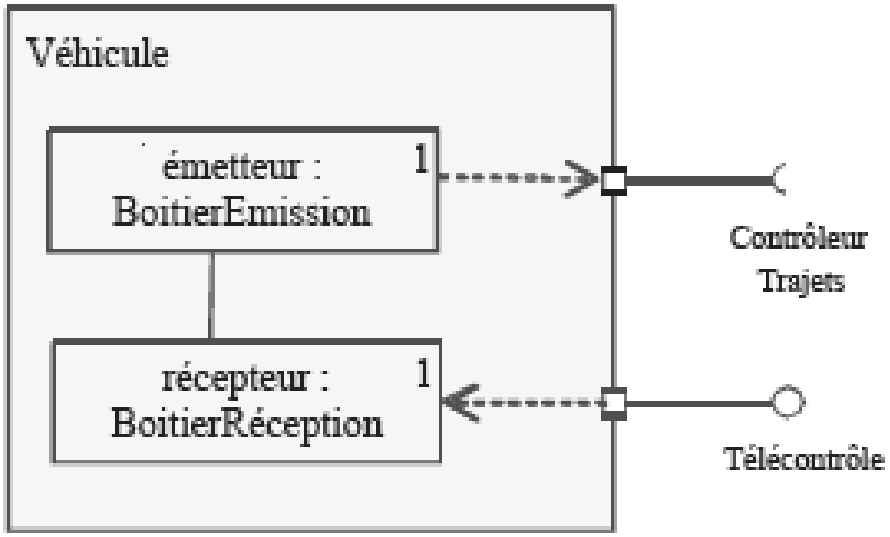
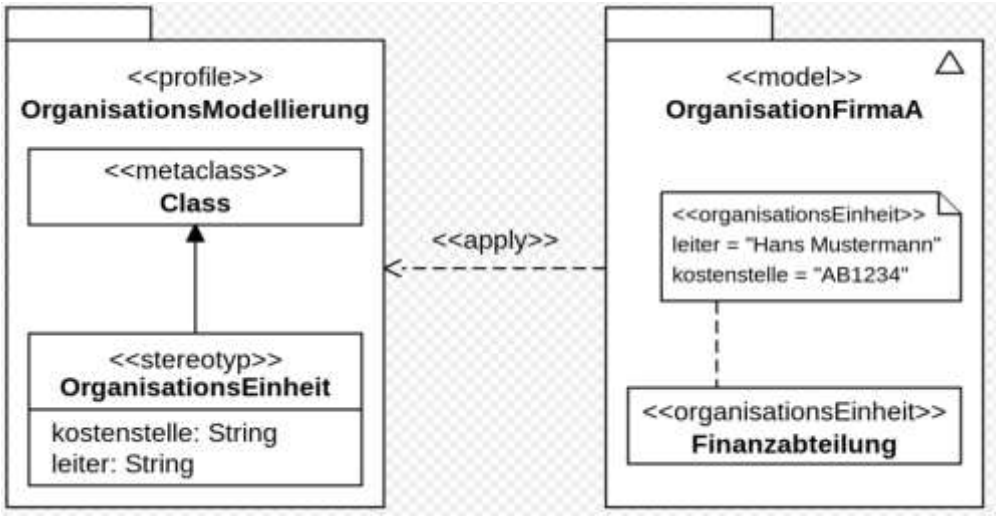


Diagramme de profils

- Permet de personnaliser UML afin de l'adapter à un domaine spécifique (santé, finance, aéronautique, IoT, etc.) sans modifier la norme UML elle-même.
- Il permet de créer des extensions d'UML via :
  - ✓ stéréotypes
  - ✓ étiquettes (tagged values)
  - ✓ contraintes
  - ✓ profils (ensemble d'extensions cohérentes)
- Apparu avec UML 2.2

Les profils UML sont un mécanisme officiel d'UML qui permet d'adapter UML à un domaine spécifique (métier, technique ou organisationnel) sans modifier le langage UML lui-même.



Diagrammes comportementaux (dynamiques)

Diagramme de cas d'utilisation (use case)

- Destiné à représenter les besoins des utilisateurs par rapport au système

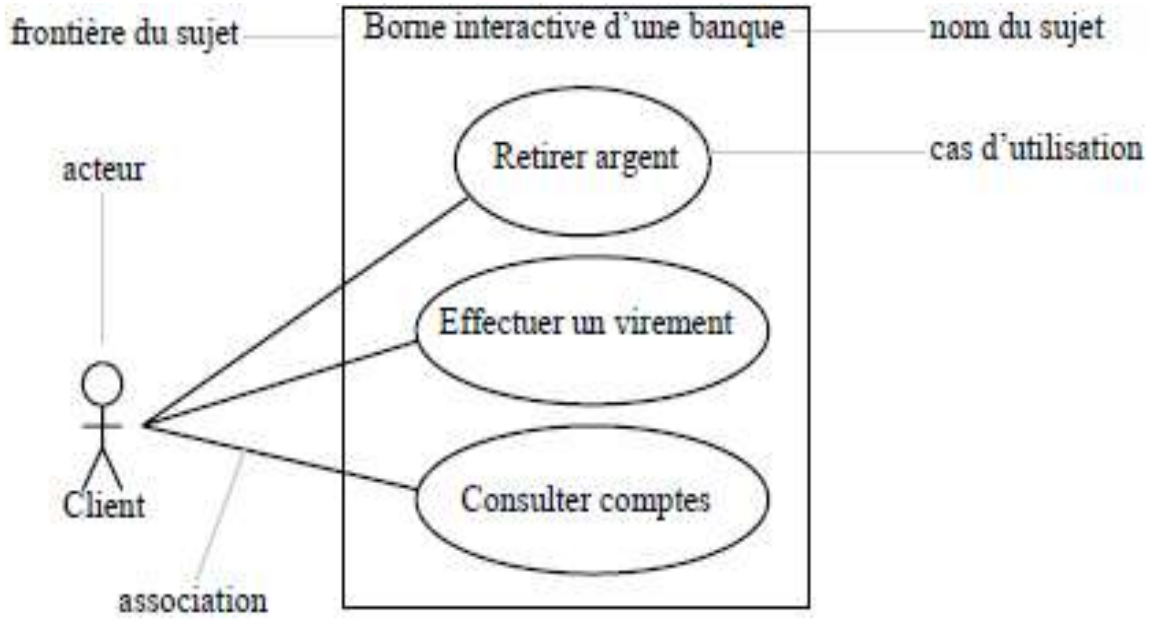


Diagramme de séquence

- Permet de décrire les scénarios de chaque cas d'utilisation en mettant l'accent sur la chronologie des opérations en interaction avec les objets

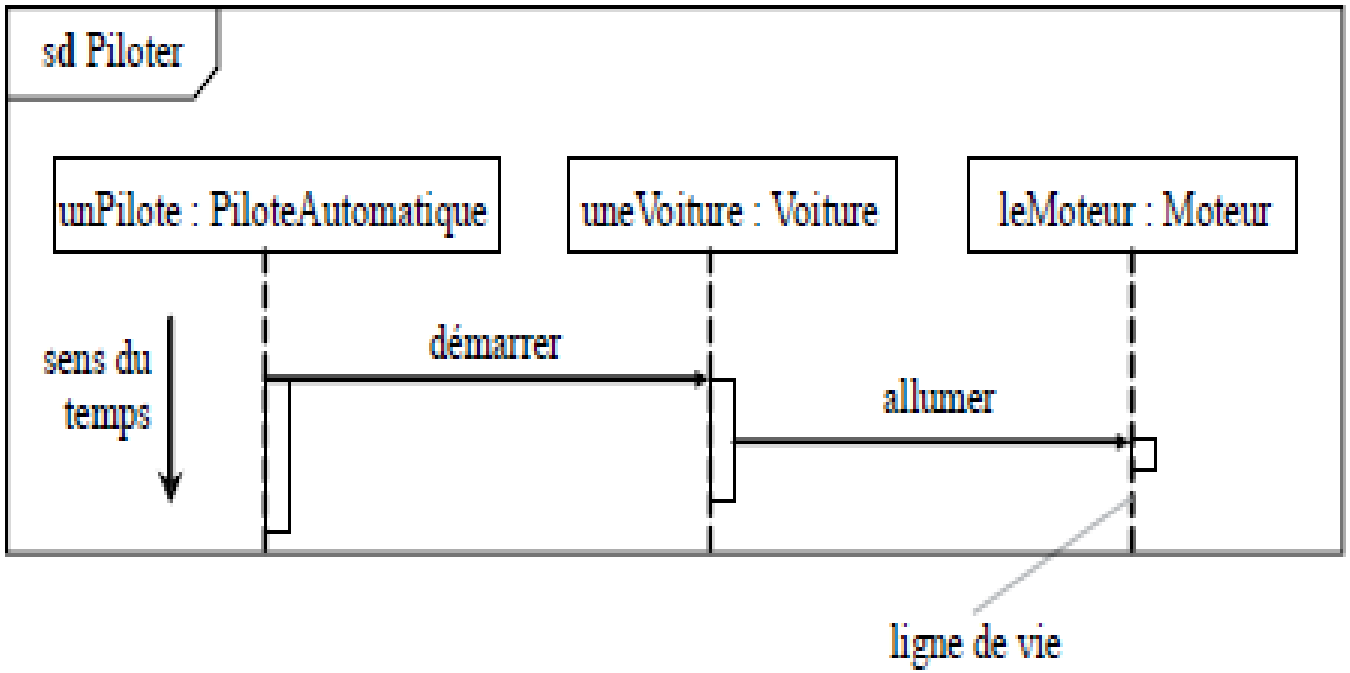
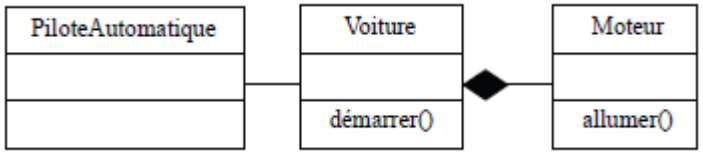


Diagramme de communication

- Graphe dont les nœuds sont des objets et les arcs (numérotés selon la chronologie) les échanges entre objets



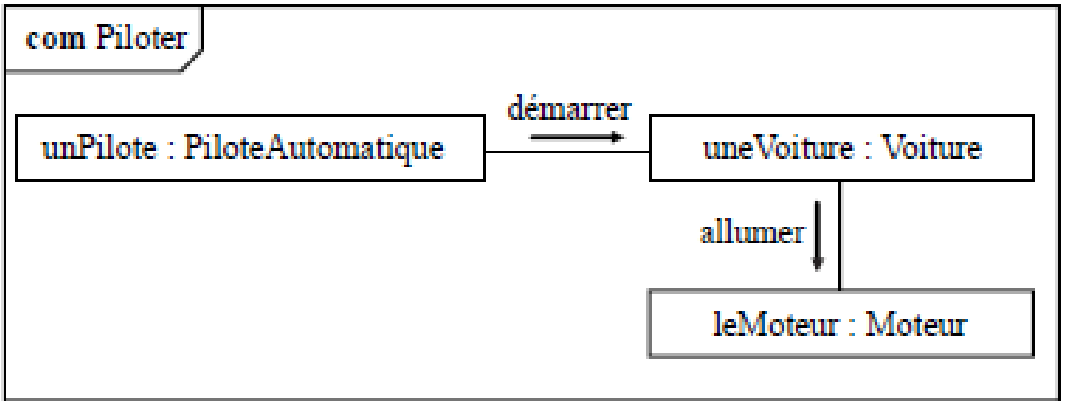


Diagramme d'états-transitions

- Permet de représenter le cycle de vie commun aux objets d'une même classe.
- Il complète la connaissance des classes en analyse et en conception

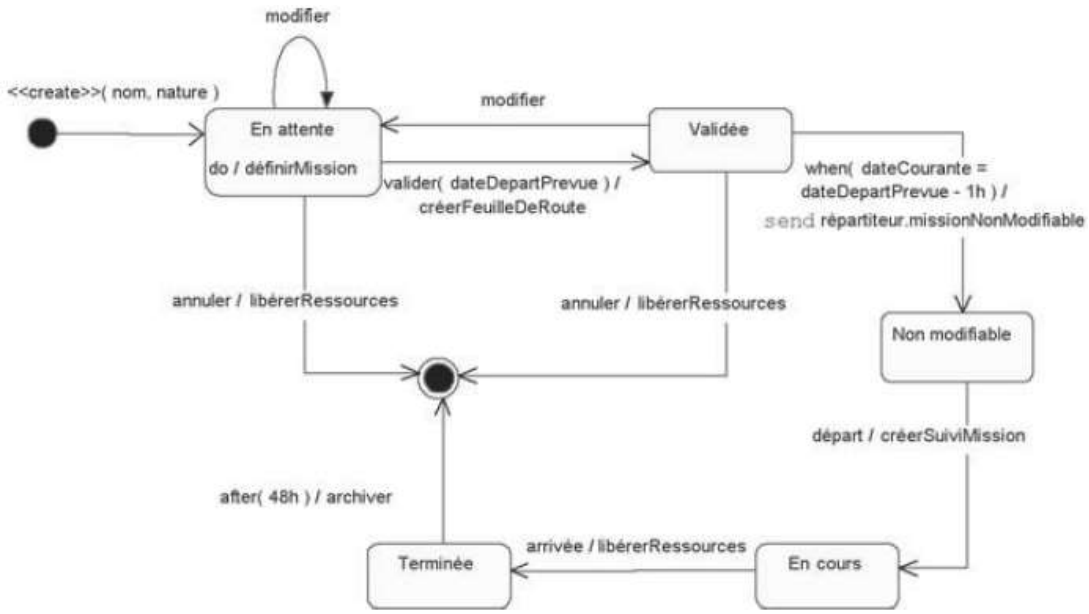


Diagramme d’activités

- Permet de décrire l'enchaînements des activités propre à une méthode ou à un cas d’utilisation

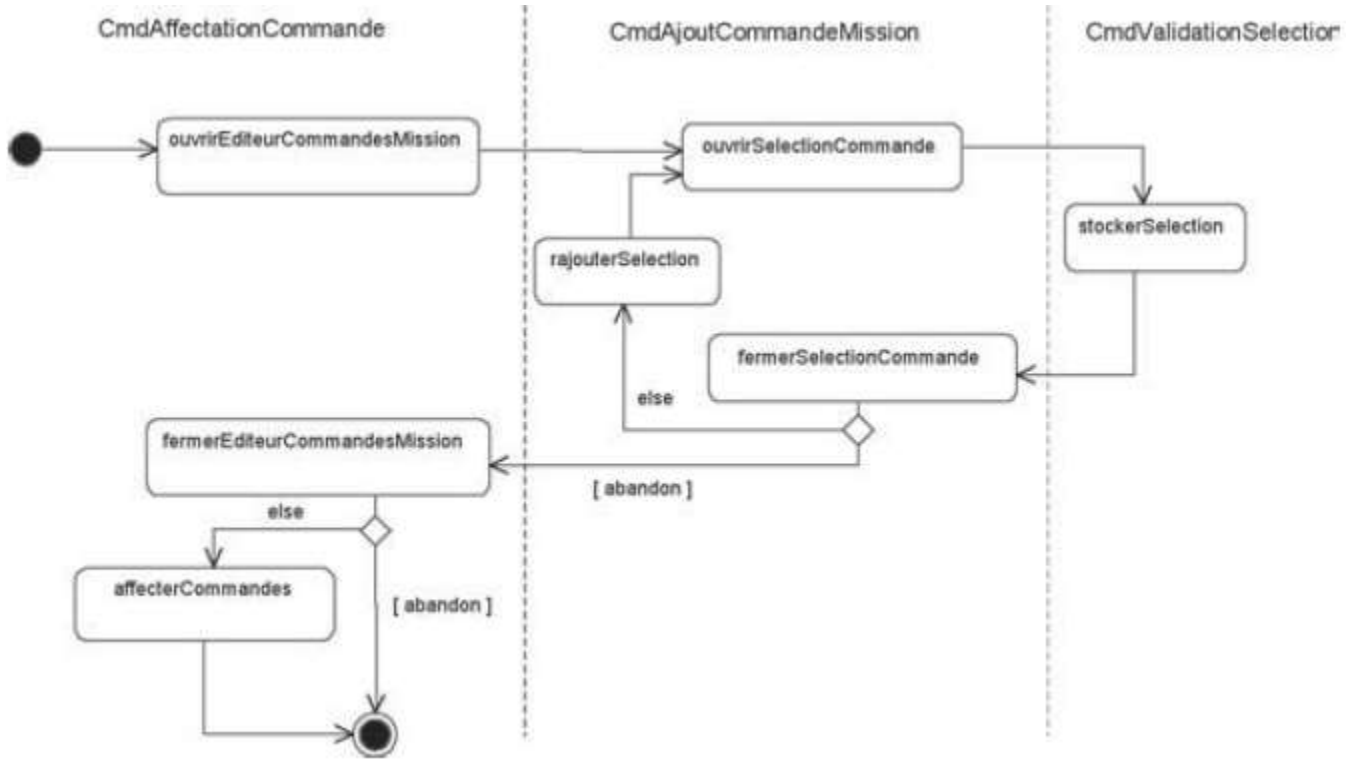


Diagramme global d'interaction

- Overview interaction
  - ✓ a été introduit par UML 2.0.
- Il propose d'associer les notations du diagramme de séquence avec celles du diagramme d'activité.
- À ce titre, il peut être aussi bien utilisé en phase d'analyse qu'en phase de conception pour la description d'une méthode complexe

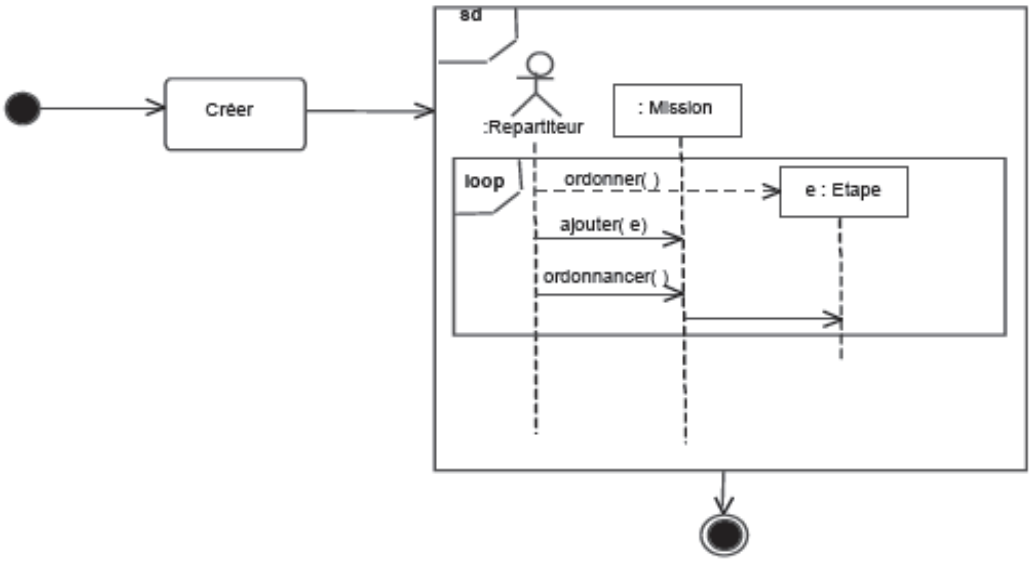
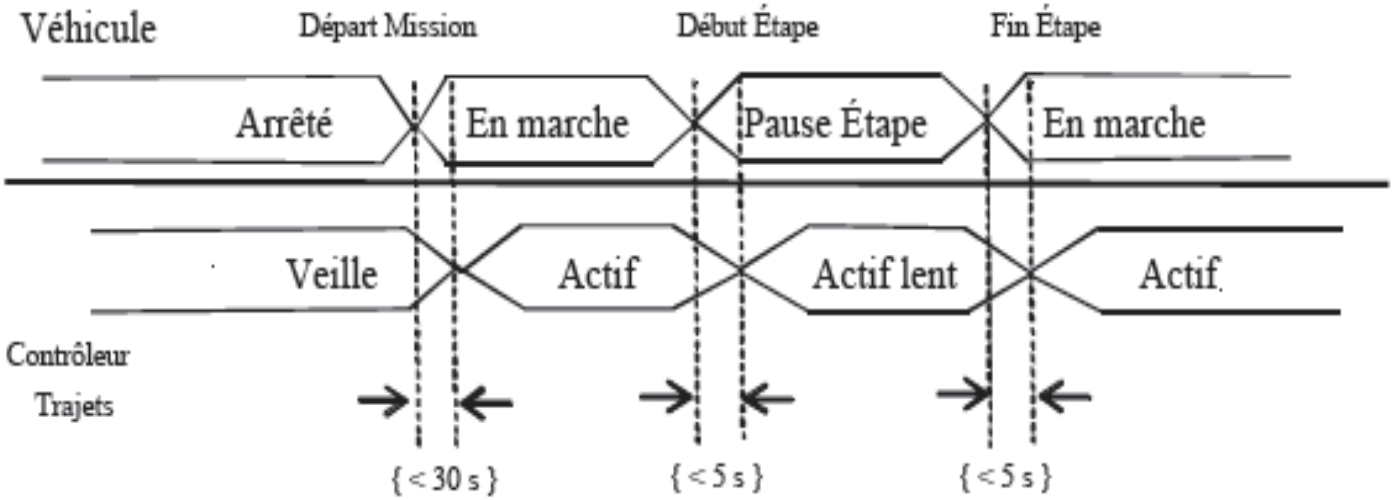


Diagramme de temps (timing)

- Timing diagram)
  - ✓ Un nouveau type de formalisme apporté par UML 2.0.
- Provient de techniques connues de l'ingénierie système et répond à des besoins de modélisation très spécifiques lorsque l'interaction entre plusieurs objets exige des contraintes temps-réel extrêmement précises et non équivoques.



Aller plus loin avec UML

Mécanismes d’extension UML

- Stereotype (Stéréotype)
  - ✓ Méta-classe spécialisée (ex: « real-time »)
  - ✓ Ajout de nouveaux stéréotypes à extension
- Tagged value (valeur étiquetée)
  - ✓ méta-attribut (ex: {abstract})
  - ✓ Ajout d’un nouveau méta-attribut à extension
- Constraint (Contrainte)
  - ✓ Règle de formation d’expression (ex: {ordered})
  - ✓ Nouvelles contraintes sur le méta-modèle à extension

Notion de profil UML

- Standardisation d'un méta-modèle étendu d'UML
- Adapté à un domaine métier ou middleware
- Un profil UML peut contenir
  - ✓ Les éléments sélectionnés dans le méta-modèle de référence
  - ✓ Des extensions utilisant les différent mécanismes d'extension
  - ✓ Descriptions sémantiques des extensions
  - ✓ Notations supplémentaires
  - ✓ Règles de validation, présentation, transformation

Quelques outils pour la modélisation

- ArgoUML (Open source)
- Eclipse: MoDisco
- Eclipse: Papyrus
- Enterprise Architect (payant)
- PlantUML
- StarUML
- BoUML
- Power Designer (payant)
- Visual Paradigm (payant)
- Enterprise Architect (payant)

Questions