Université Nice Sophia Antipolis

Conception Orientée Objets

Présentation

Frédéric Mallet

http://deptinfo.unice.fr/~fmallet/

Université Nice Sophia Antipolis

Organisation du cours

- Volume Horaire et EDT
  - 18h CM
    - Frederic Mallet
  - 18h TD
    - Frederic Mallet
    - Owen Rouille
- Evaluation
  - Evaluation théorique, 1h30 : 50%
  - Evaluation pratique (en TD) : 50%

Université Nice Sophia Antipolis

Plan du cours

- Introduction à UML2
- Les « diagrammes »
  - Les cas d'utilisation
  - Les classes et leurs instances
  - Les machines à états (et transitions)
  - Les activités
  - Les interactions
  - Le mécanisme de profilage
- Eclipse Modeling Framework (EMF)
- *Le langage de contraintes OCL*

Université Nice Sophia Antipolis

Les objets

- Objectifs
  - Montrer les forces de COO
  - Décrire l'histoire de POO
  - Commenter l'utilisation actuelle de la POO

Université Nice Sophia Antipolis

La technologie Orientée-Objet

- Guide la conception par
  - Un ensemble de concepts
    - abstraction, modularité, encapsulation, polymorphisme
  - Des langages et des outils qui supportent ces concepts
    - Classification vs. prototype
    - Héritage (multiple)
    - Typage : fort/faible, explicite/inféré
- Ses forces (supposées)
  - Reflète plus finement les objets du monde réel
    - Du code (plus) facile à maintenir
    - Plus stable : un changement s'applique à un sous-système facile à identifier et **isoler** du reste du système

Université Nice Sophia Antipolis

Système de gestion d'un lycée

Objets

- Personnes
  - Etudiant, enseignant, principal, secrétaire
- Diplôme
  - Année, matière, parcours
- Notes
  - Coefficients

Fonctions

- Calculer la moyenne
- Calculer les taux d'encadrement
- Calculer le nombre de redoublants
- Calculer le taux de réussite au baccalauréat

Université Nice Sophia Antipolis

Objectifs des technologies à objets

- Utiliser le langage du domaine
  - Modèle et vocabulaire métier
- Construire des modèles faciles à:
  - Étendre, modifier, valider, vérifier
- Faciliter l'implantation
  - Génération facilitée vers les langages à objets
- Nécessite une méthode et des outils
  - Rational Unified Process, Agile, … (cf. semestre 2)
  - UML est seulement un langage

Université Nice Sophia Antipolis

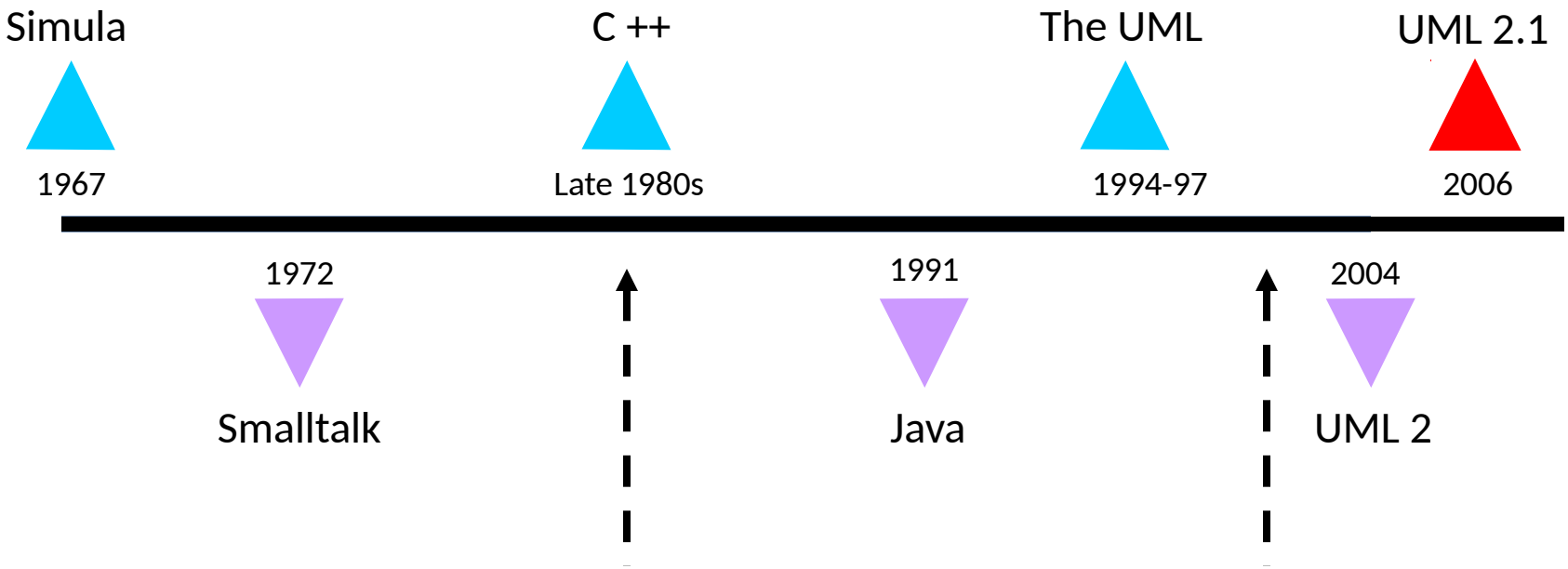
Les forces des technologies à objets

- Difficiles à imposer dans l'industrie
  - Certains pensaient que c'était une mode
  - D'autres que c'est une étape vers autre chose : programmation par composants, par aspects
- Les objets sont partout (*Turing Award* en 2000)
  - **Un seul paradigme** : de l'analyse système à l'implantation
    - En pratique : raffinements semi-automatiques parfois difficiles
  - Les objets représentent le **monde réel** : objets ou phénomènes
  - **Stable et adaptatif** :
    - Petits changements localisés et dé-corrélés du reste

L'histoire des technologies à objets

Étapes majeures

Booch : G. Booch
OMT : J. Rumbaugh
Objectory : I. Jacobson



Premier environnement commercial Smalltalk, suivi de C++

Turing Award pour O-J. Dahl et K. Nygaard

Université Nice Sophia Antipolis

COO vs. conception structurée

- La COO
  - Seulement une couche supplémentaire
  - Garde le meilleur de la conception structurée
  - **Mélange** les aspects statiques et dynamiques
  - Réutilise une **classe** plutôt qu'un ensemble de routines
    - Systématisation de la conception modulaire
  - Rend primitif des mécanismes puissants
    - Encapsulation # fichiers en-tête
    - Polymorphisme # pointeur sur *void*
    - Liaison dynamique # pointeur sur fonction

Université Nice Sophia Antipolis

Des vieilles idées resservies

Procédural	Objets (UML)	Matériel
Décomposition fonctionnelle	Unified Process	Platform-based design
Masquer l'information	Encapsulation	Interface
Module	Instance	Entité
Appel de procédure	Message	# Canal
Procédure	Opération	*Process*
Appel dynamique	Liaison dynamique	# Table vectorisation
Déclaration dynamique	Instanciation	Instanciation
Conception structurée	Conception OO	Conception par composant
Type	Classe	Cellule

Université Nice Sophia Antipolis

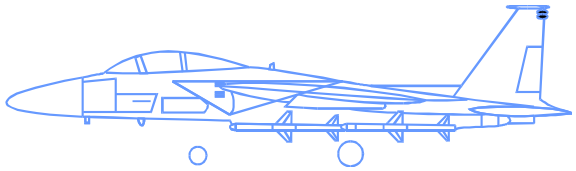
UML2 – Modélisation visuelle et MDA

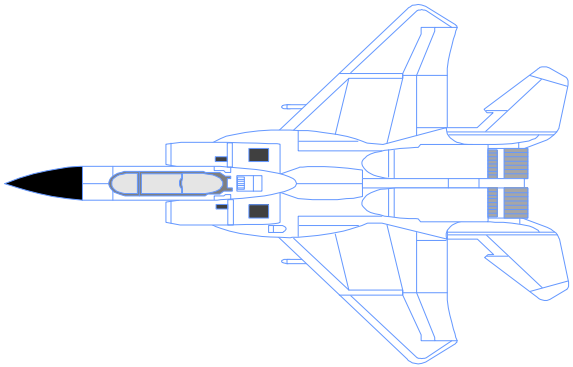
- Objectifs
  - Décrire l'intérêt de la modélisation visuelle
  - Énoncer les quatre principes de base
  - Expliquer ce que représente UML
  - Présenter les procédés les mieux adaptés à UML

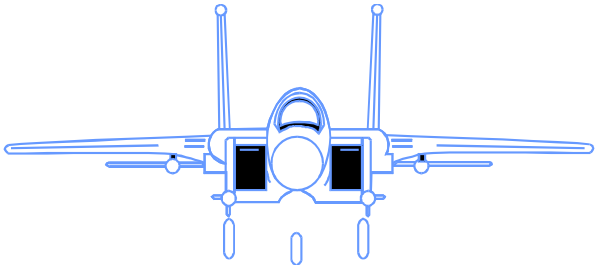
Université Nice Sophia Antipolis

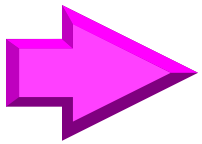
Qu'est-ce qu'un modèle ?

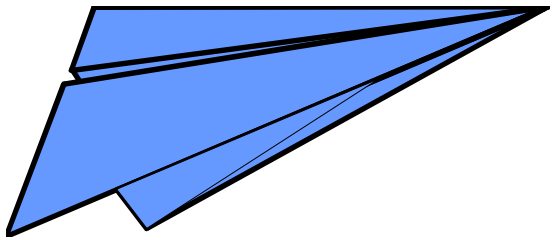
- Un modèle est UNE simplification de la réalité











Université Nice Sophia Antipolis

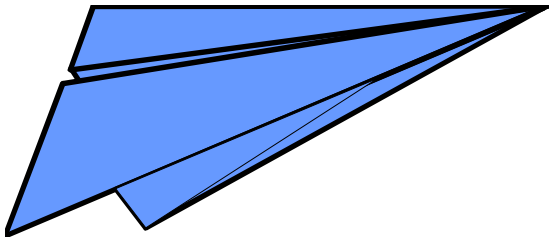
Pourquoi un modèle ?

- Quatre objectifs à la modélisation
  - Aider à **visualiser** le système
  - **Spécifier** la structure et le comportement
  - **Servir de plan** pour la construction effective
  - Permettre de **documenter** les choix
- Quatre avantages
  - Abstraction : diviser pour régner
  - Compréhension : mises au point avec le client
  - L'énergie déployées pour modéliser révèle les difficultés
  - Les erreurs sur les modèles **coûtent bien moins cher**

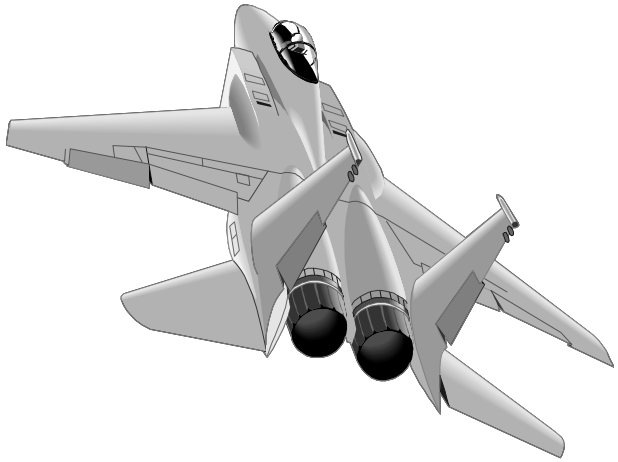
Université Nice Sophia Antipolis

L'importance des modèles

moins important — Plus important



Avion papier

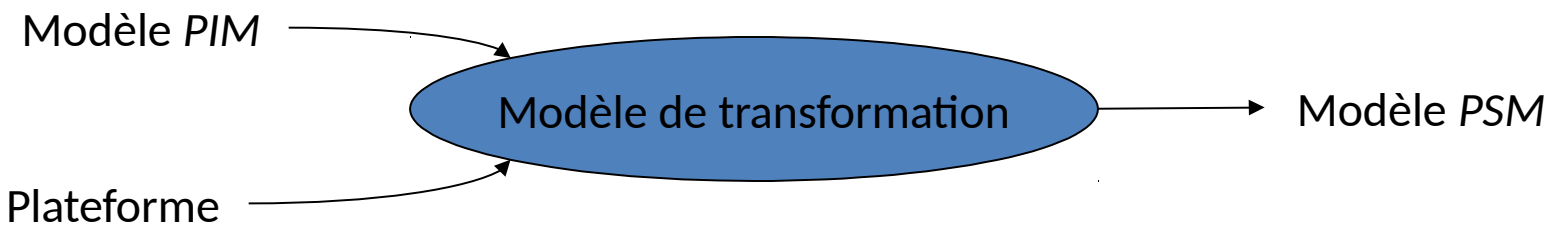


Avion militaire

Le développement logiciel AUSSI nécessite des modèles bien pensés !

Model Driven Architecture (MDA)

- Développement orienté modèles
  - Spécifier un modèle indépendant de la plateforme sur laquelle il sera déployé
  - Spécifier la ou les plateformes
  - Choisir une plateforme adaptée
  - Transformer le modèle de spécification en un modèle spécifique pour la plateforme



Université Nice Sophia Antipolis

MDA *viewpoints (1/2)*

- **C**omputation **I**ndependent **M**odel (**CIM**)
  - Modèle métier : pas la structure du système
  - Présente le système et son environnement
    - Vocabulaire utilisé
    - Doit pouvoir être tracé vers PIM et PSM
  - Ex: décrit un serveur web, un accès internet, mais cela ne signifie pas qu'il y aura une classe « WebServer » ou « Internet »
- **P**latform-**I**ndependent **M**odel (**PIM**)
- **P**latform-**S**pecific **M**odel (**PSM**)

Université Nice Sophia Antipolis

MDA *viewpoints (2/2)*

- **C**omputation **I**ndependent **M**odel (**CIM**)
- **P**latform-**I**ndependent **M**odel (**PIM**)
  - Modélise le système indépendamment d'une cible
  - Utilisation d'une machine virtuelle neutre **(analyse)**
  - La plateforme se résume à ses services (communication, ordonnancement, nommage)
  - Ex: Ne sait pas si on utilise Corba, J2EE, ou .net
- **P**latform-**S**pecific **M**odel (**PSM**)
  - Comment le système utilise les spécificités d'une plateforme **(conception)**
  - Conversion des noms
  - Devrait être généré automatiquement

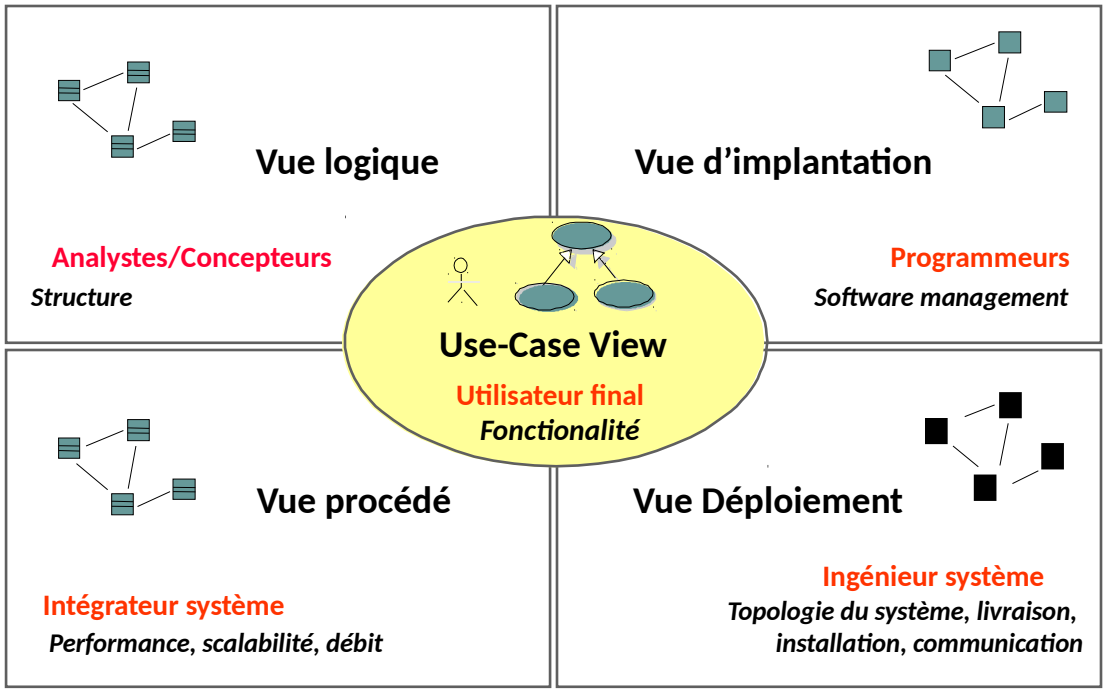
Université Nice Sophia Antipolis

Les quatre principes

- Un modèle influence énormément la façon d'aborder le problème et la solution
  - Vue du concepteur BD # vue du programmeur OO
- Chaque modèle peut être exprimer à différents niveaux de précision
  - Les meilleurs modèles permettent de choisir le niveau de détail en fonction de qui regarde et pourquoi il le regarde
- Les meilleurs modèles sont liés à la réalité
- Un seul modèle n'est jamais suffisant
  - Tous les systèmes gagnent à être décrits avec plusieurs petits modèles relativement indépendant => comment assurer la cohérence entre les modèles

Principe 4 : un seul modèle ne suffit pas !

- Créer plusieurs modèles indépendants mais avec des points communs



Université Nice Sophia Antipolis

Unified Modeling Language

- Langage visuel unifié
  - Tout le monde doit parler le même langage
- Langage pour spécifier
  - Executable-UML
  - Supposé précis et non ambigu
- Des liens vers +s langages de prog.
  - Java, C++, VB
  - RDMS ou OODMS
  - Génération de code et *reverse engineering*.

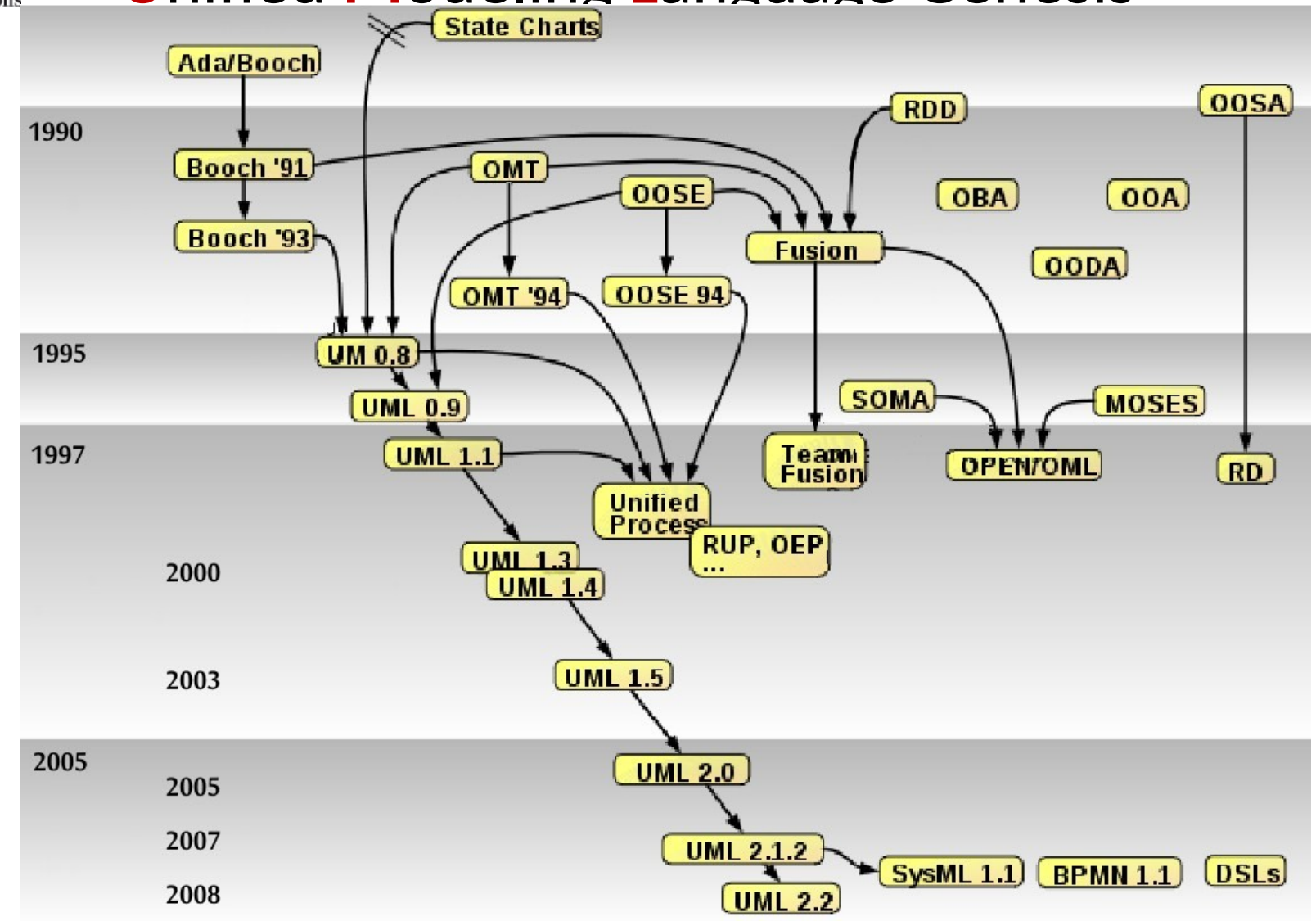
Université Nice Sophia Antipolis

Unified Modeling Language Genesis

- In 1994
  - Object-orientation was becoming popular
  - Too many methods/languages to describe similar concepts (>5000)
    - Metamodels were very similar
    - Graphical notations were completely different
  - The Industry was asking for a standard notation
- Rational Software Corporation starts a process
  - Booch method (Grady Booch) and OMT (Jim Rumbaugh)
  - Followed by OOSE (Ivar Jacobson from Objectory)
    - Use cases

Université Nice Sophia Antipolis

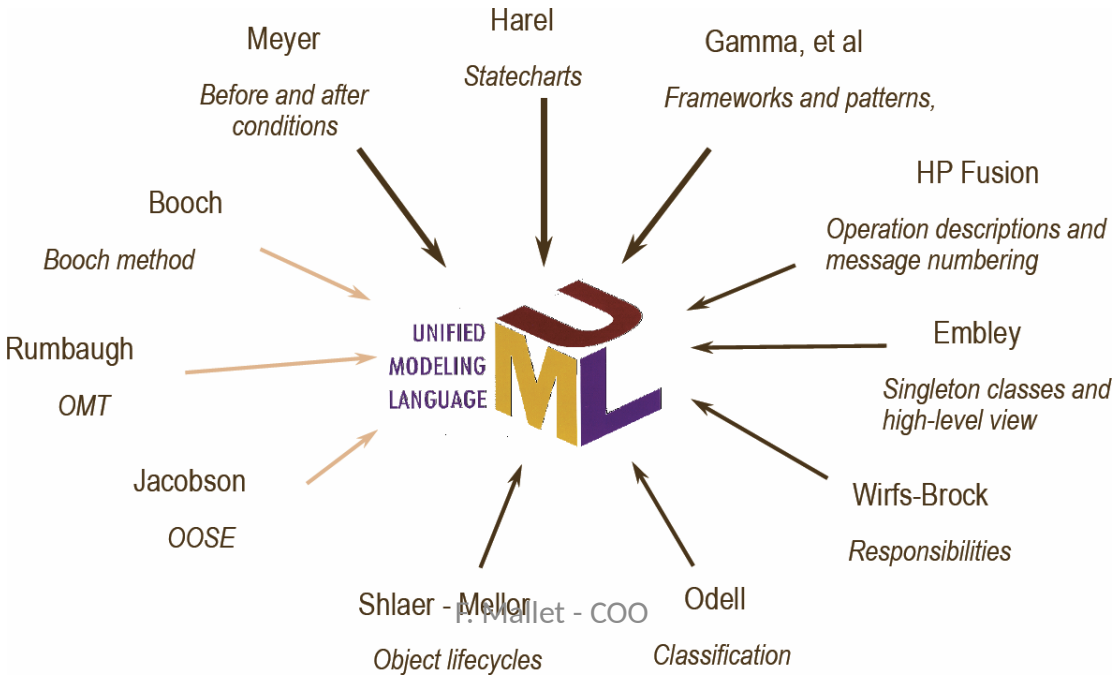
Unified Modeling Language Genesis



http://en.wikipedia.org/wiki/File:OO-historie.jpg

Unified Modeling Language (UML)

- Specified by the **OMG**
  - Object Management Group
    - OMG™ is an international, open membership, not-for-profit computer industry consortium since 1989
    - Most famous specifications: CORBA, UML, MDA, MOF, IDL
    - http://www.omg.org



Université Nice Sophia Antipolis

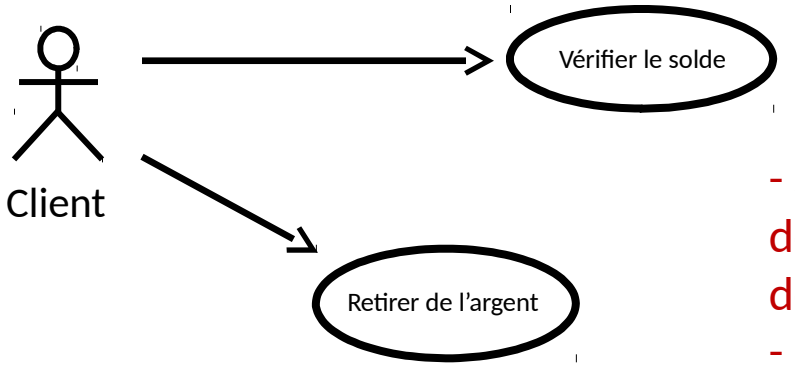
UML et RUP

- Un langage n'est pas suffisant, il faut aussi une méthode
- Les méthodes (*process*) qui fonctionnent le mieux avec UML sont :
  - Orienté par les *Use-case* ;
  - Centré sur l'architecture ;
  - Itératif et incrémental.
    - Les utilisateurs réagissent au fur et à mesure.
- *Rational Unified Process*

Université Nice Sophia Antipolis

Modèle orienté par les *use-case*

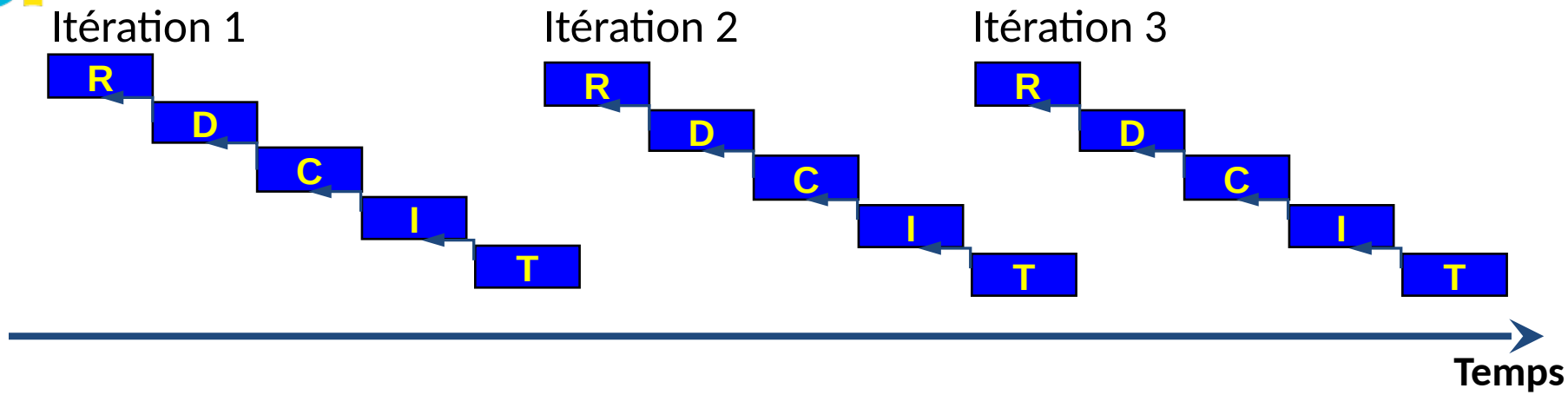
- Les *use-case* sont la base
  - Ils doivent être précis et concis
  - Ils sont compréhensibles par la majorité
  - Ils permettent de synchroniser les différents modèles
  - Ils décrivent l'ensemble des fonctions du système et les acteurs concernés



- SysML les reprend et définit des diagrammes de *Requirements*
- RUP recommande une description tabulaire

Université Nice Sophia Antipolis

RUP : procédé itératif



- Les premières itérations s'intéressent aux aspects les plus risqués
  - Requirements, Design, Coding & test unit, Implementation, Test
- Chaque itération produit un exécutable
- Chaque itération contient des tests d'intégration

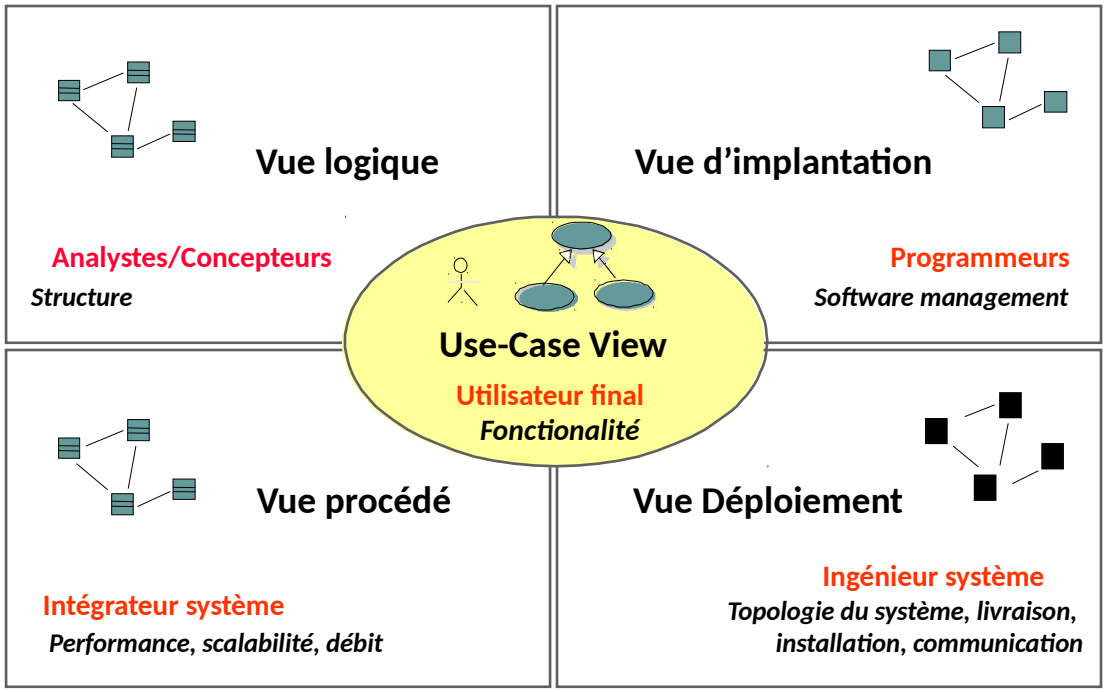
Université Nice Sophia Antipolis

Our process

- Describe Requirements with Use cases or SysML
  - Describe the behavior of requirements with state machines and activities
  - Describe the abstract features of the platform (Non Functional Properties)
- Build possible scenarios for each Use cases
  - Using interactions (sequence or collaboration diagrams)
  - This requires to identify
    - the classes and their instances
    - the methods and their parameters
  - In parallel, build a class diagram
- Describe the state of classes
  - With state machines or activities
- Make bundles
  - Component and deployment diagrams

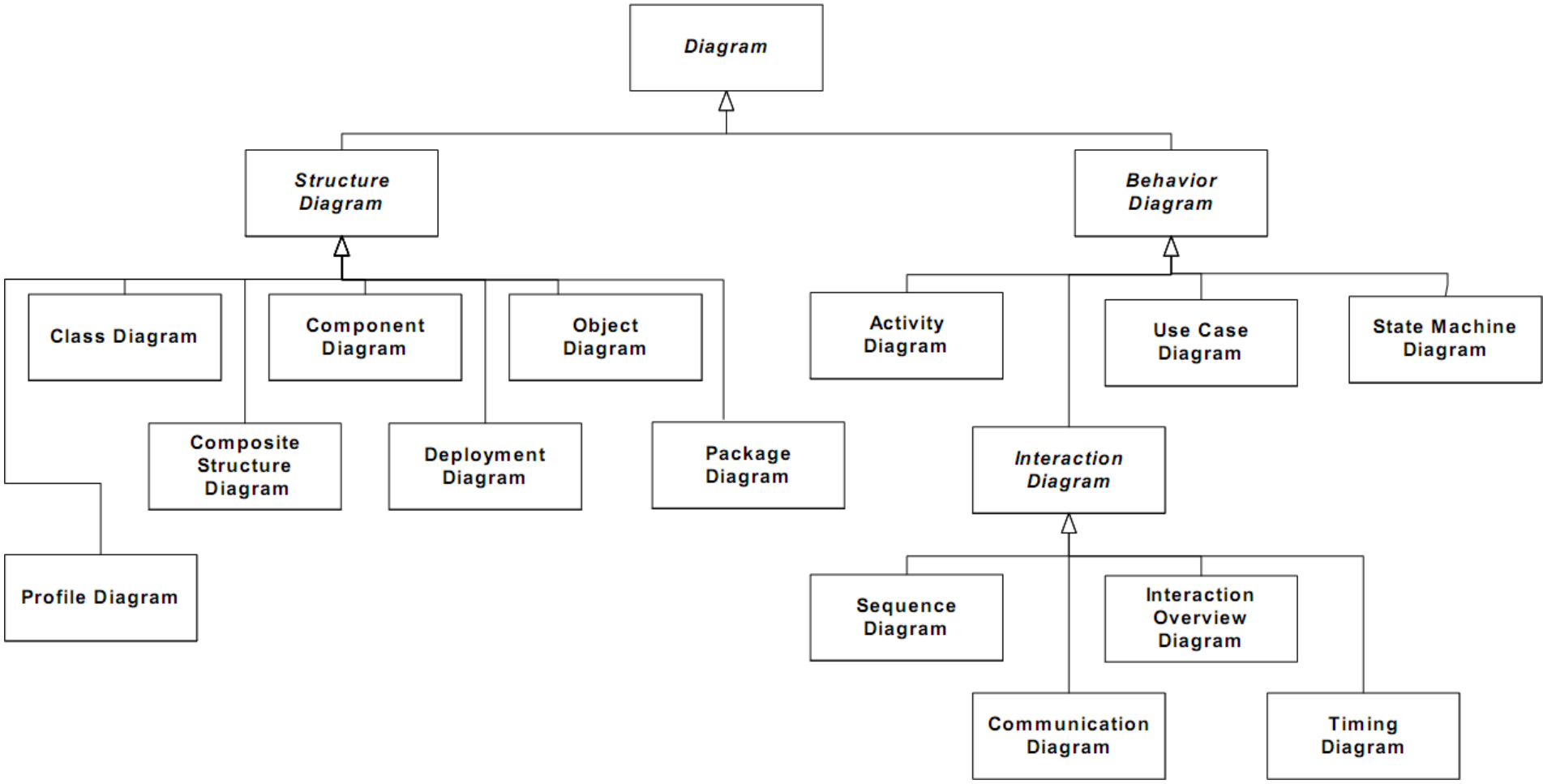
Principe 4 : un seul modèle ne suffit pas !

- Créer plusieurs modèles indépendants mais avec des points communs



Université Nice Sophia Antipolis

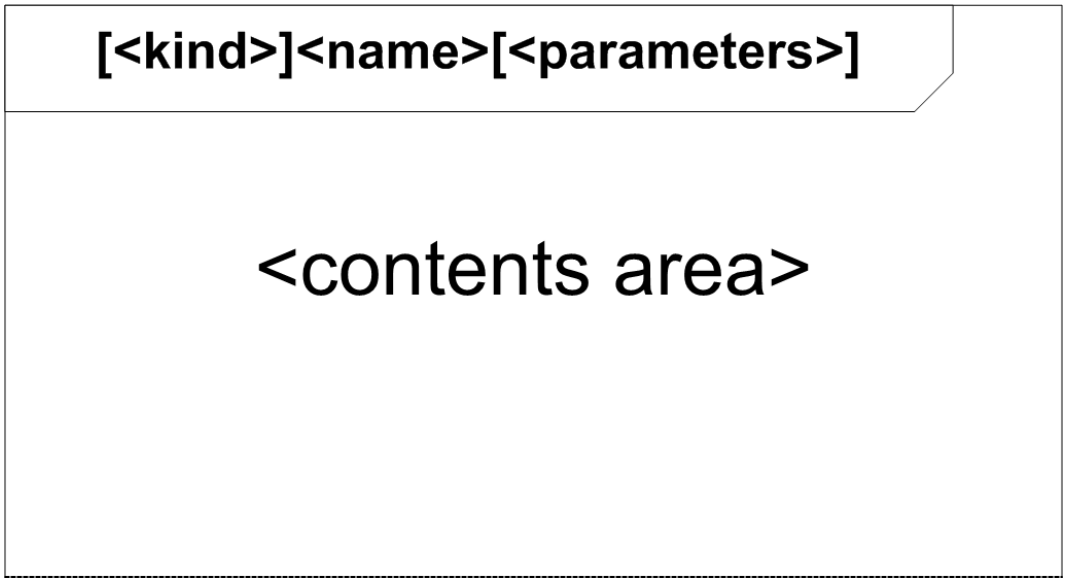
14 diagrams



Université Nice Sophia Antipolis

Diagram and frames

- Diagrams should be within frames
  - Heading should give a name, kind and parameters if any
    - kind ∈ { activity, class, component, deployment, interaction, package, state machine, use case }
    - Short form { act, class, cmp, dep, sd, pkg, stm, uc }



Université Nice Sophia Antipolis

UML2 – Qu'est-ce qu'un objet

- Objectifs
  - Encapsulation, abstraction, modularité, hiérarchie
  - Structure d'une classe
  - Relations entre une classe et un objet
  - Polymorphisme et généralisation
  - Les interfaces

Université Nice Sophia Antipolis

Plan

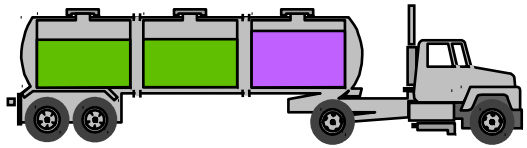
- Qu'est-ce qu'un objet ?
- Quatre concepts au centre de la COO
- Qu'est-ce qu'une classe ?
- Généralisation et polymorphisme
- Organisation des éléments modèles

Université Nice Sophia Antipolis

Qu'est-ce qu'un objet ?

- Un objet représente une entité physique, conceptuelle ou logicielle du monde réel.

  - Entité physique

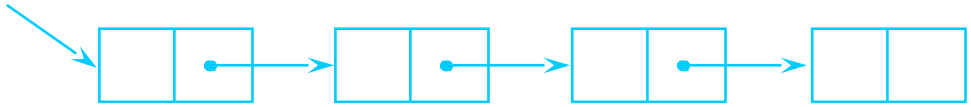


Camion

  - Entité conceptuelle

Procédé chimique

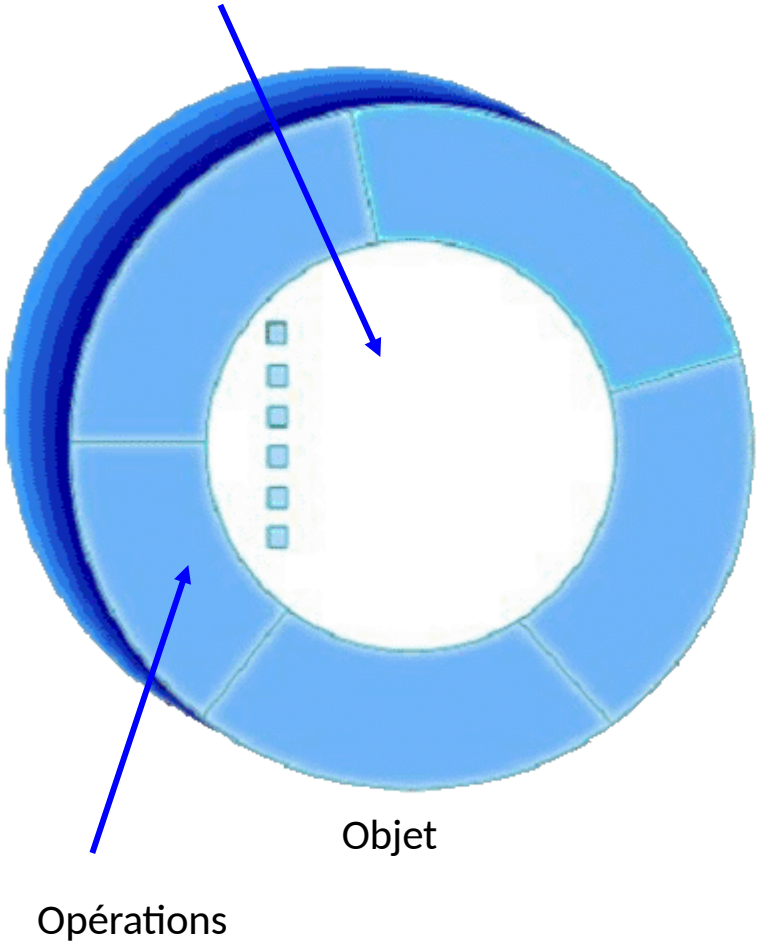
  - Entité logicielle



Liste chaînée

Qu'est-ce qu'un objet ?

- Un objet a une frontière bien définie, une identité : **état** et ***comportement***.
  - L'état est représenté par des slots et des références
  - Le comportement est représenté par les opérations et les machines à états

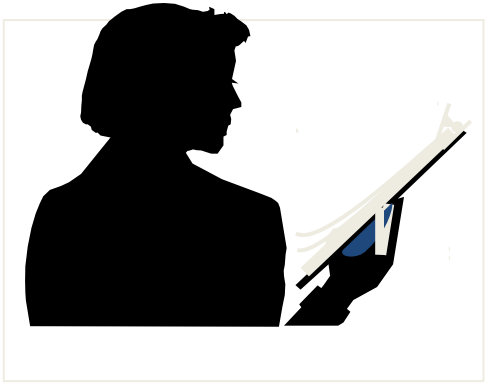


InstanceSpecification

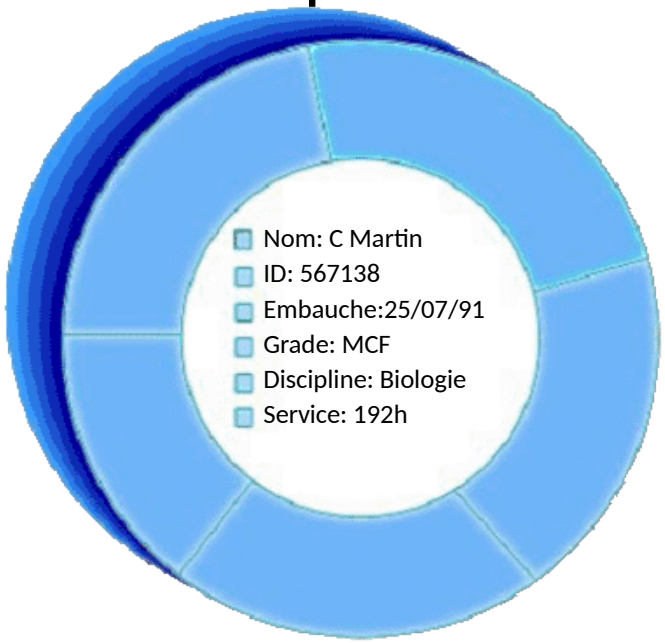
Université Nice Sophia Antipolis

L'état d'un objet

- L'état est une condition ou situation pendant la vie d'un objet qui satisfait une condition, effectue une activité ou attend pour un événement.
- L'état d'un objet peut changer dans le temps.



Nom: C Martin
ID: 567138
Embauche: 25/07/1991
Grade: Maitre de conférence
Discipline: Biologie
Service dû : 192h

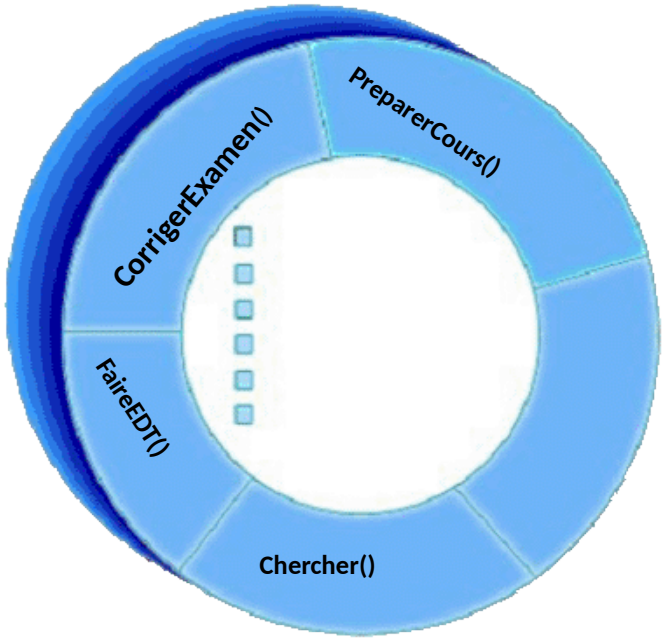


Professeur Martin

Un objet a un comportement

- Le comportement détermine comment l'objet agit ou réagit
- Le comportement visible d'un objet est son interface (ensemble d'opérations).

Comportement du professeur Martin
- Corriger les examens
- Préparer un nouveau cours
- Chercher
- Faire l'emploi du temps



Professeur Martin

Université Nice Sophia Antipolis

Chaque objet a une identité

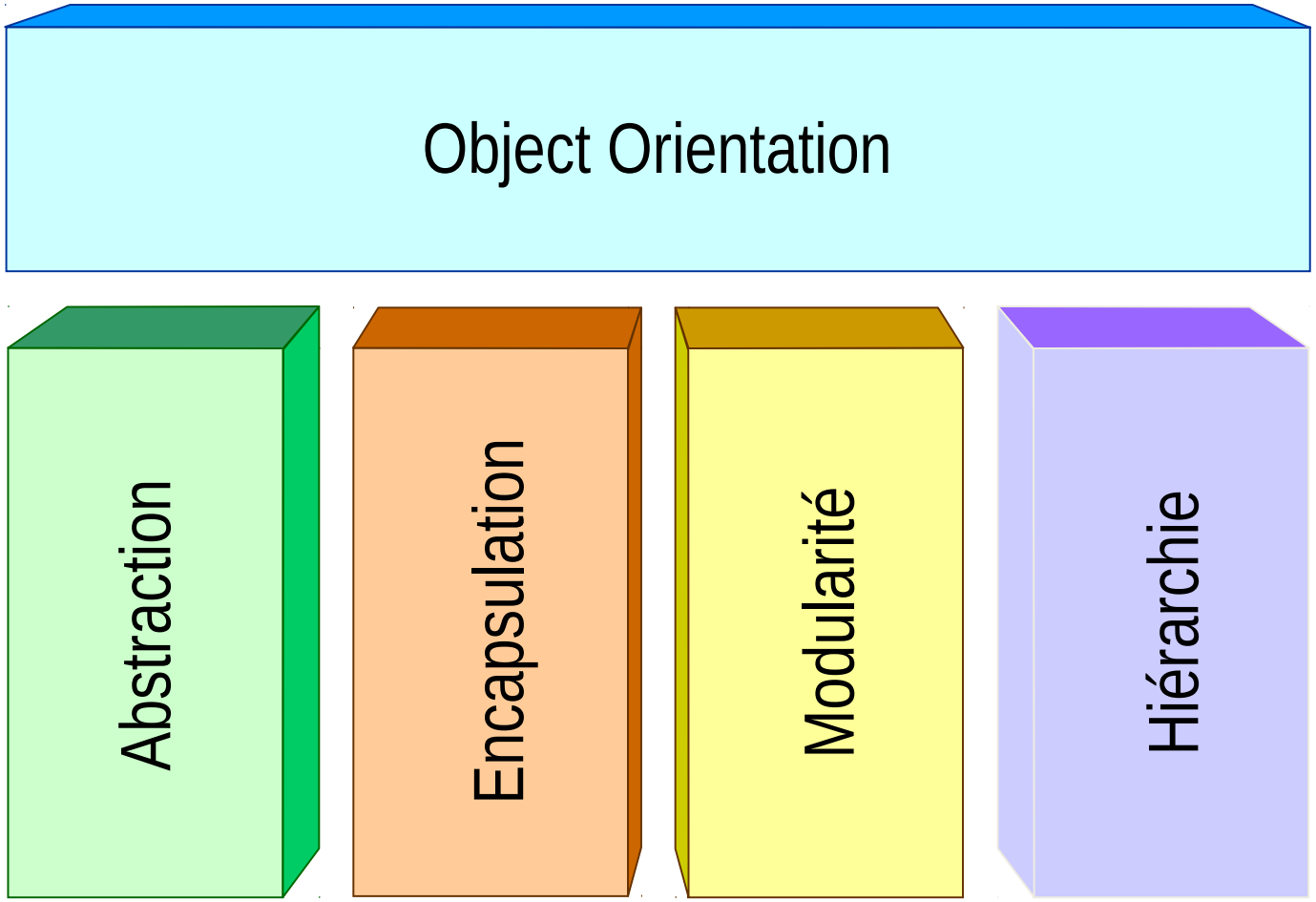
- L'identité est unique même si l'état est le même que celui d'un autre objet

Professeur "C Martin"
enseigne la biologie

Professeur "C Martin"
enseigne la biologie

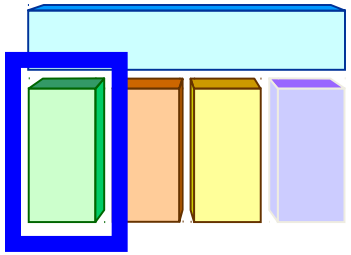
Université Nice Sophia Antipolis

Fondements de COO



Université Nice Sophia Antipolis

L'abstraction?

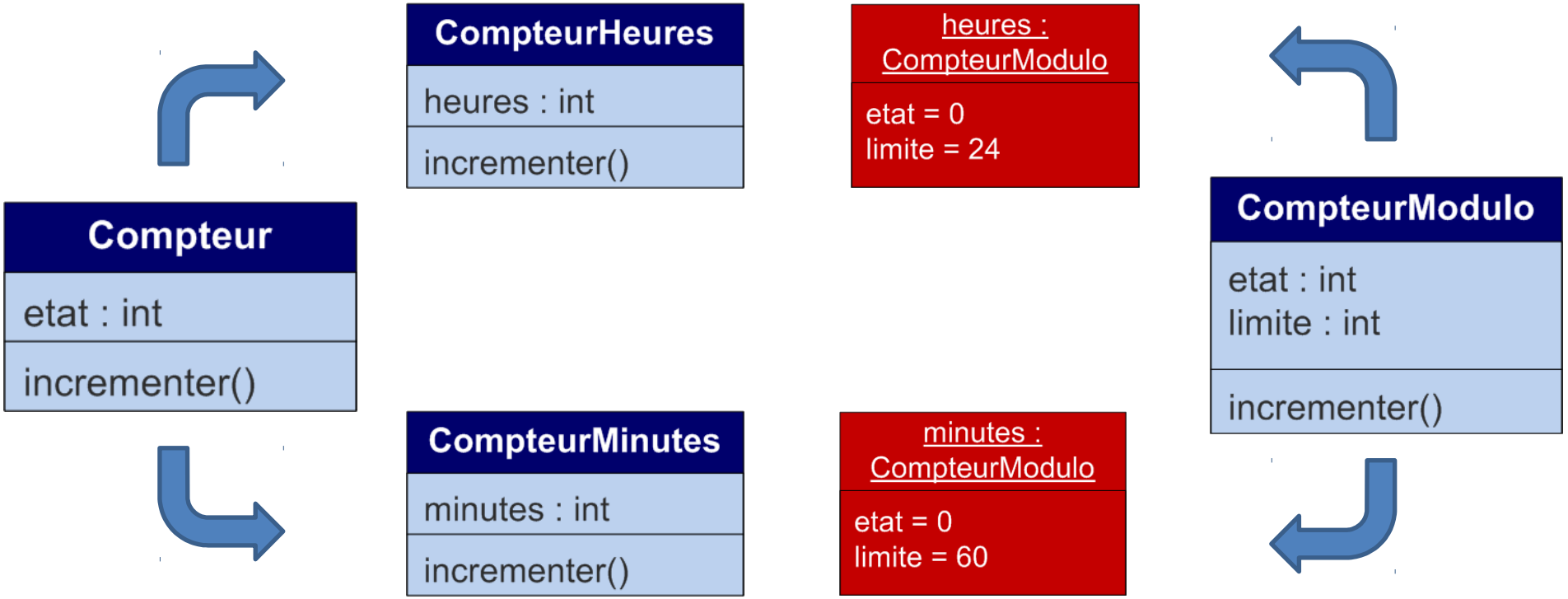


- Caractéristiques essentielles d'une entité qui la distingue des autres
  - Dépend de la perspective et du contexte
  - N'est pas une manifestation concrète mais dénote l'essentiel

Université Nice Sophia Antipolis

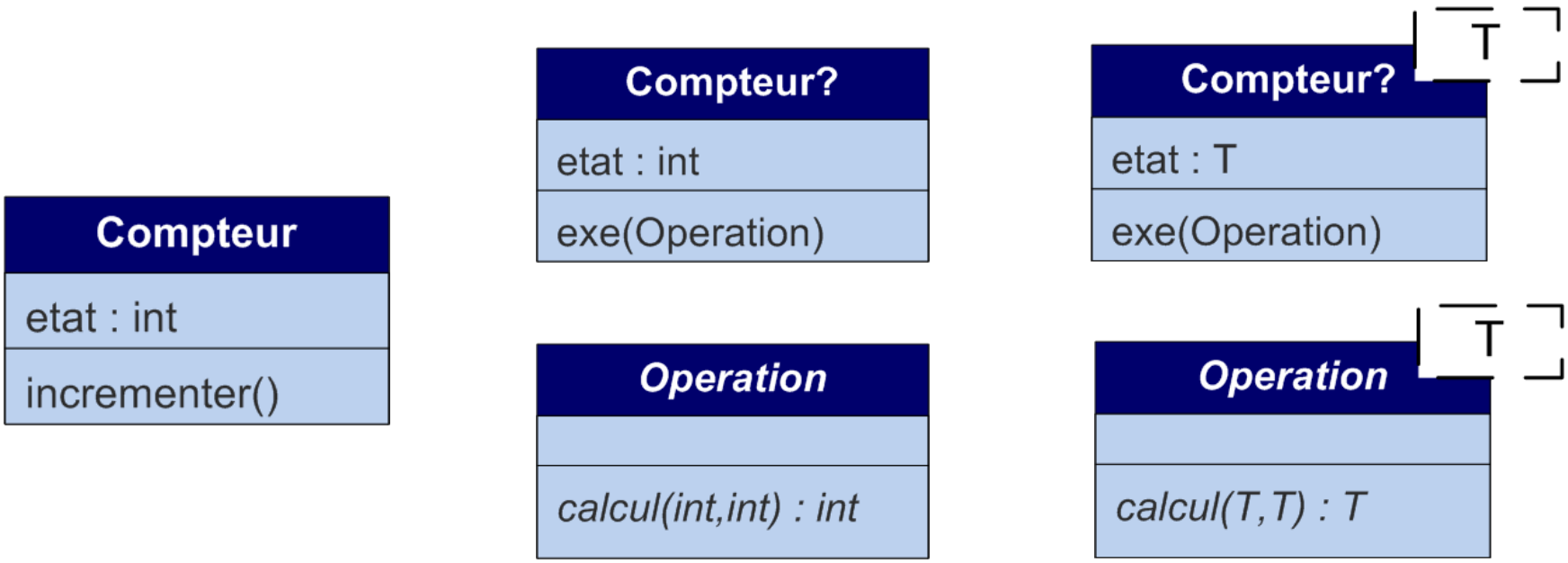
Abstraction et réutilisation

- Quelle est la limite à l'abstraction ?
  - Les classes deviennent des objets



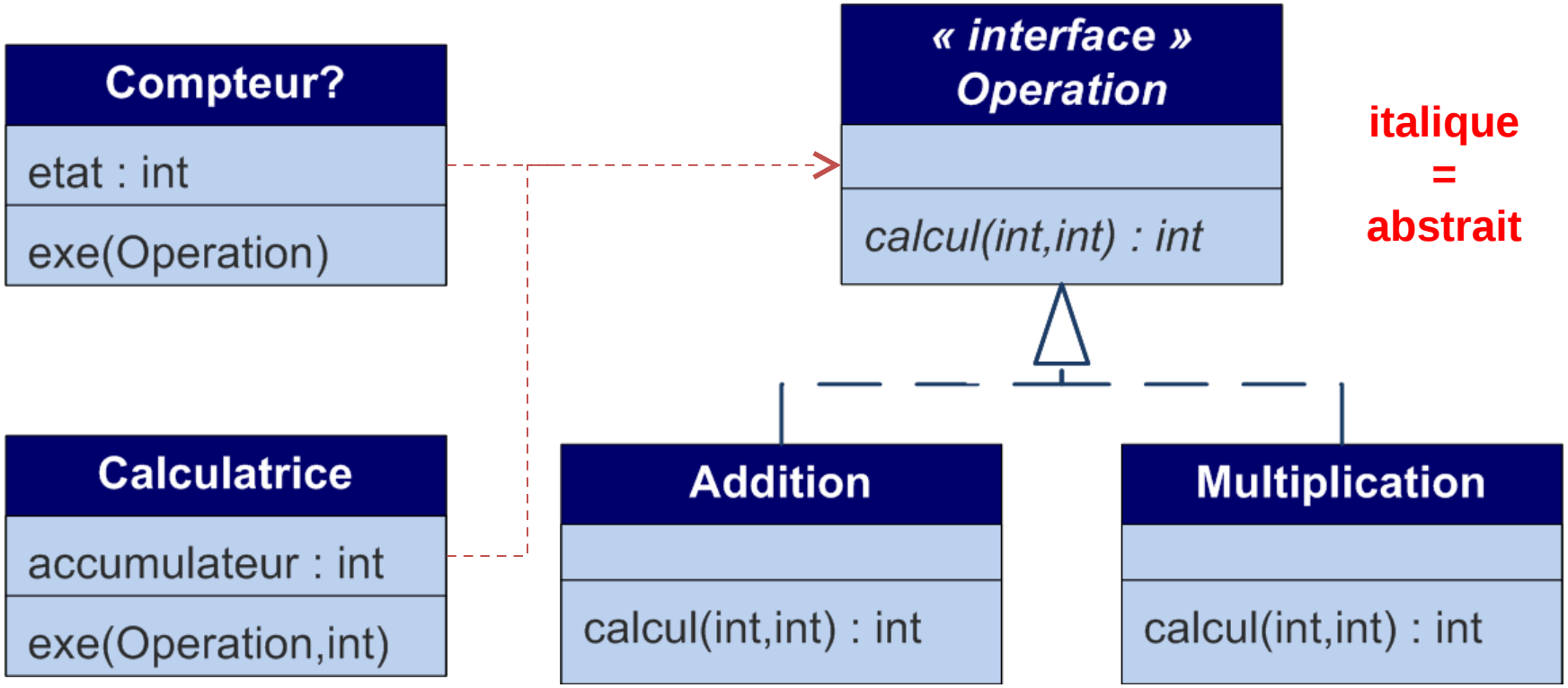
Abstraction et réutilisation

- Quelle est la limite à l'abstraction ?
  - Les méthodes deviennent des classes



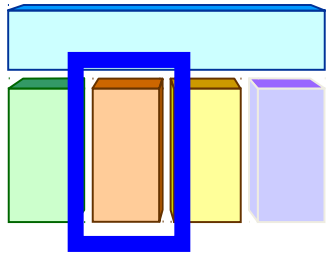
Université Nice Sophia Antipolis

Abstraction vs. Compréhension

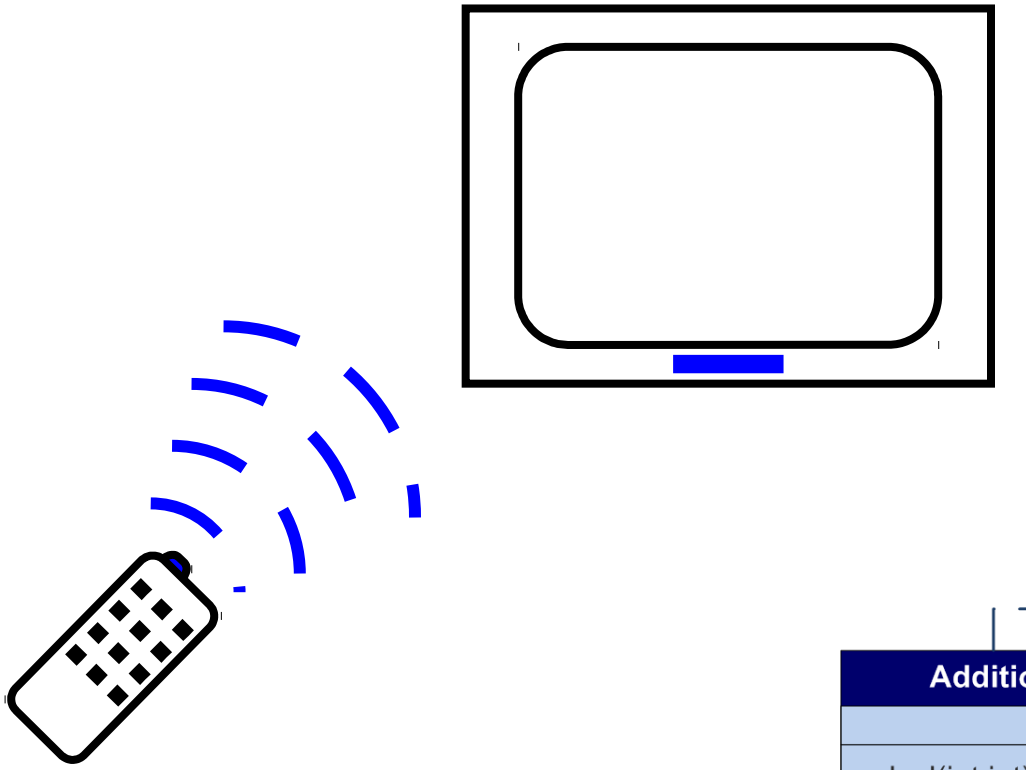


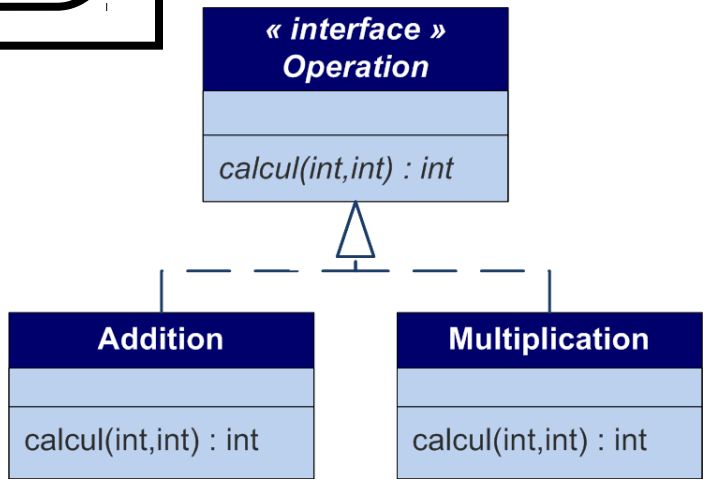
Université Nice Sophia Antipolis

L'encapsulation?



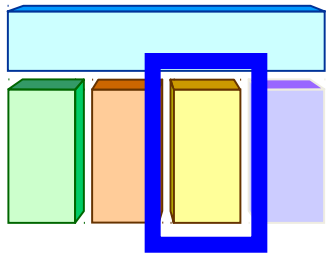
- Cache l'implantation aux clients
  - Les clients dépendent d'une interface



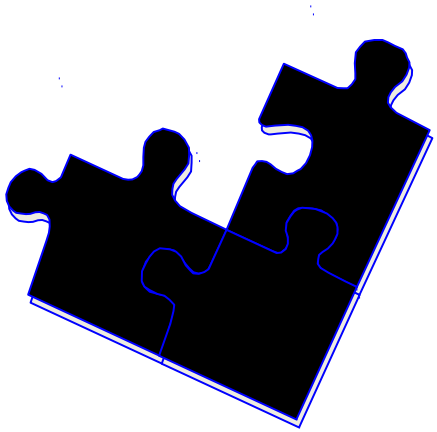


Université Nice Sophia Antipolis

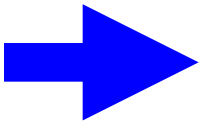
La modularité ?



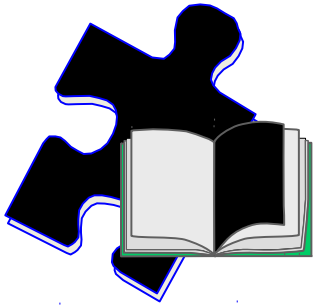
- Casser un système en petits modules



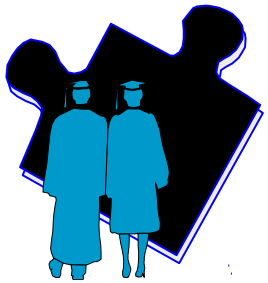
Système de gestion de l'université



Inscription administrative



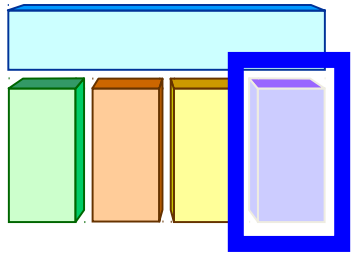
Inscription pédagogique

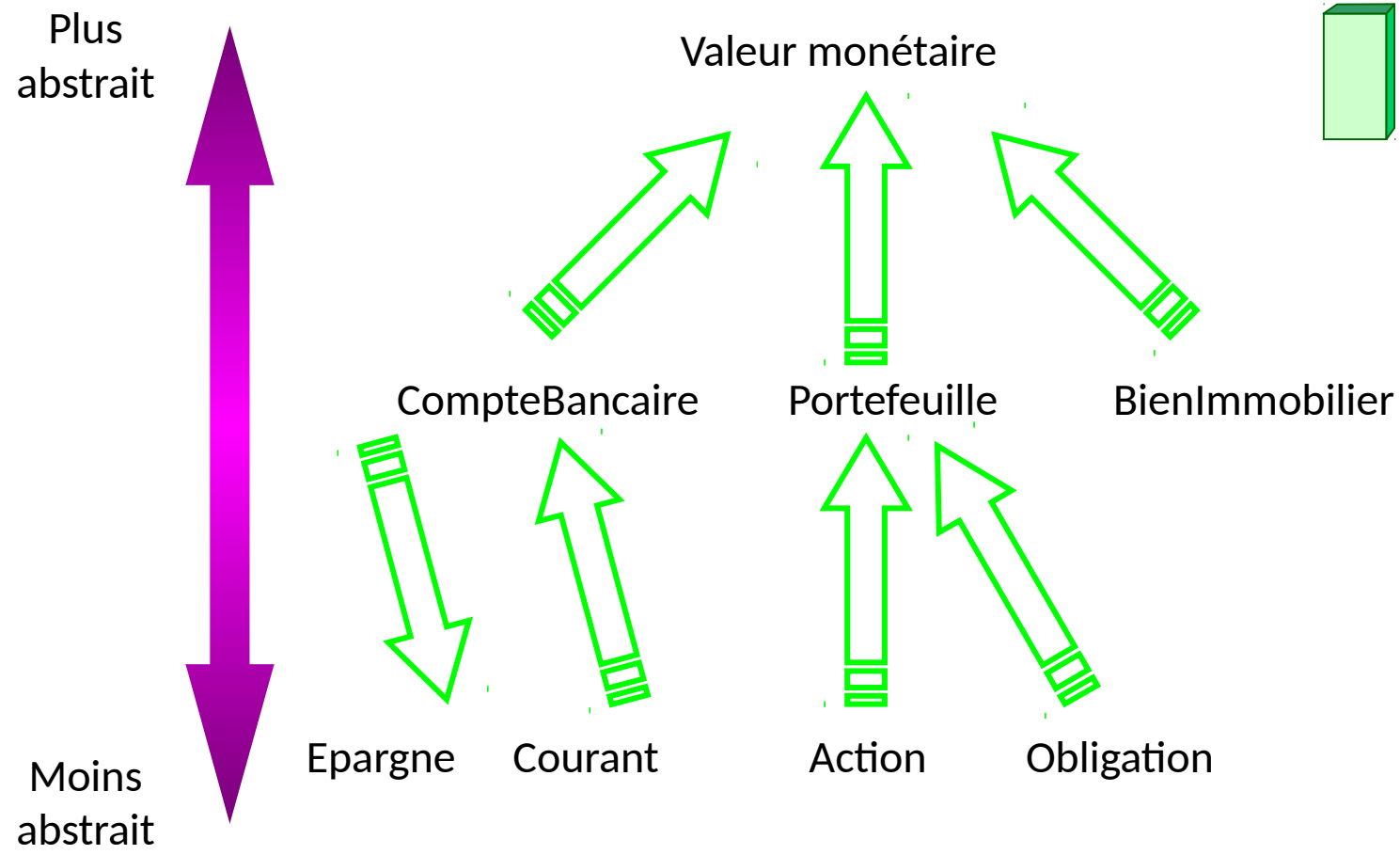


Gestion des parcours

Université Nice Sophia Antipolis

La hiérarchie?





Les éléments au même niveau hiérarchique devraient être au même niveau d'abstraction

Représentation des objets en UML

- Un objet (*InstanceSpecification*) est représenté par un rectangle.
- Le nom est souligné

Professeur C Martin

C Martin : Professeur

Objet nommé

: Professeur

Objet anonyme

Université Nice Sophia Antipolis

Qu'est-ce qu'une classe ?

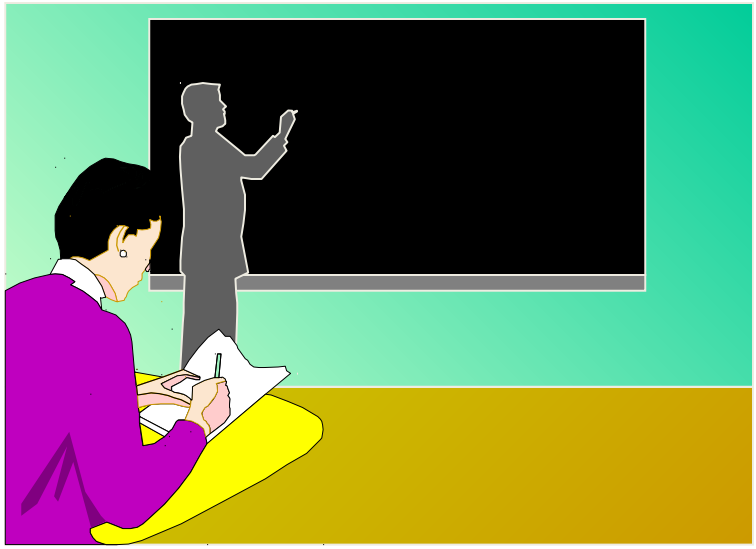
- Une classe décrit un ensemble d'objets qui partagent les mêmes *attributs*, *opérations*, *références*, et *sémantique*.
  - Un objet est l'instance d'une classe.
- Une classe est une abstraction car elle
  - Met en évidence certaines caractéristiques
  - Supprime d'autres caractéristiques

Université Nice Sophia Antipolis

La classe Cours

Classe
Cours

Attributs
Nom
Salle
Durée
Crédits
Semestre

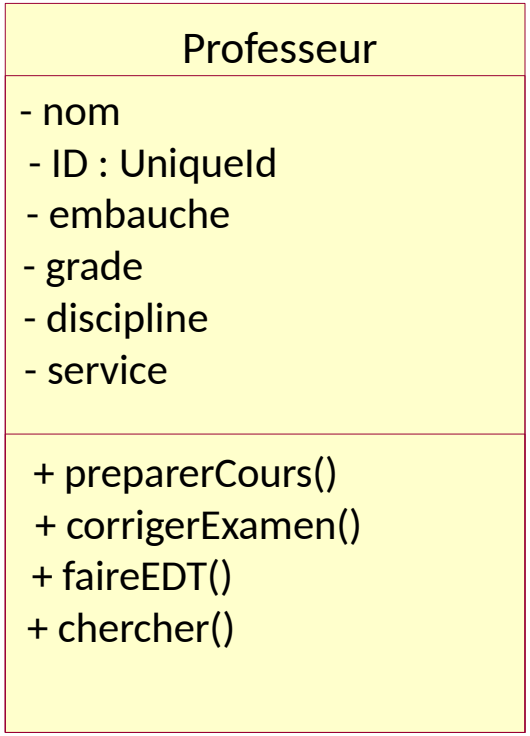


Comportement
Ajouter un étudiant
Enlever un étudiant

Université Nice Sophia Antipolis

Les classes en UML

- Une classe est représentée par un rectangle avec 3 compartiments
  - Le nom de la classe
  - La structure (les attributs)
  - Le comportement (opérations)



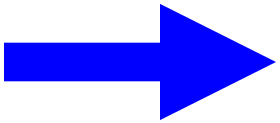
Relations entre classes et objets

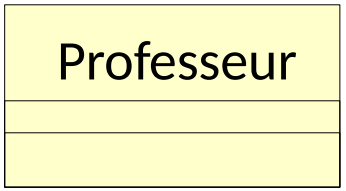
- La classe est une définition abstraite
  - Elle définit la structure et le comportement de chaque objet issue de cette classe
  - Sert de modèle pour la création d'instances
- Les classes ne sont pas des listes d'objets.

Professeur T

Professeur M

Professeur A

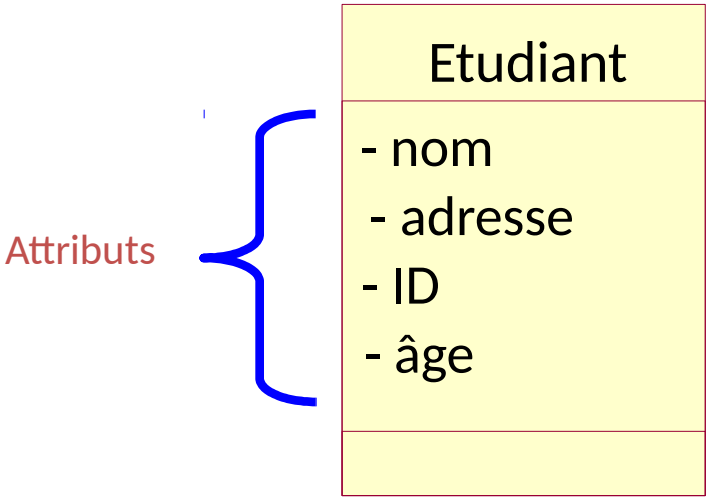




Université Nice Sophia Antipolis

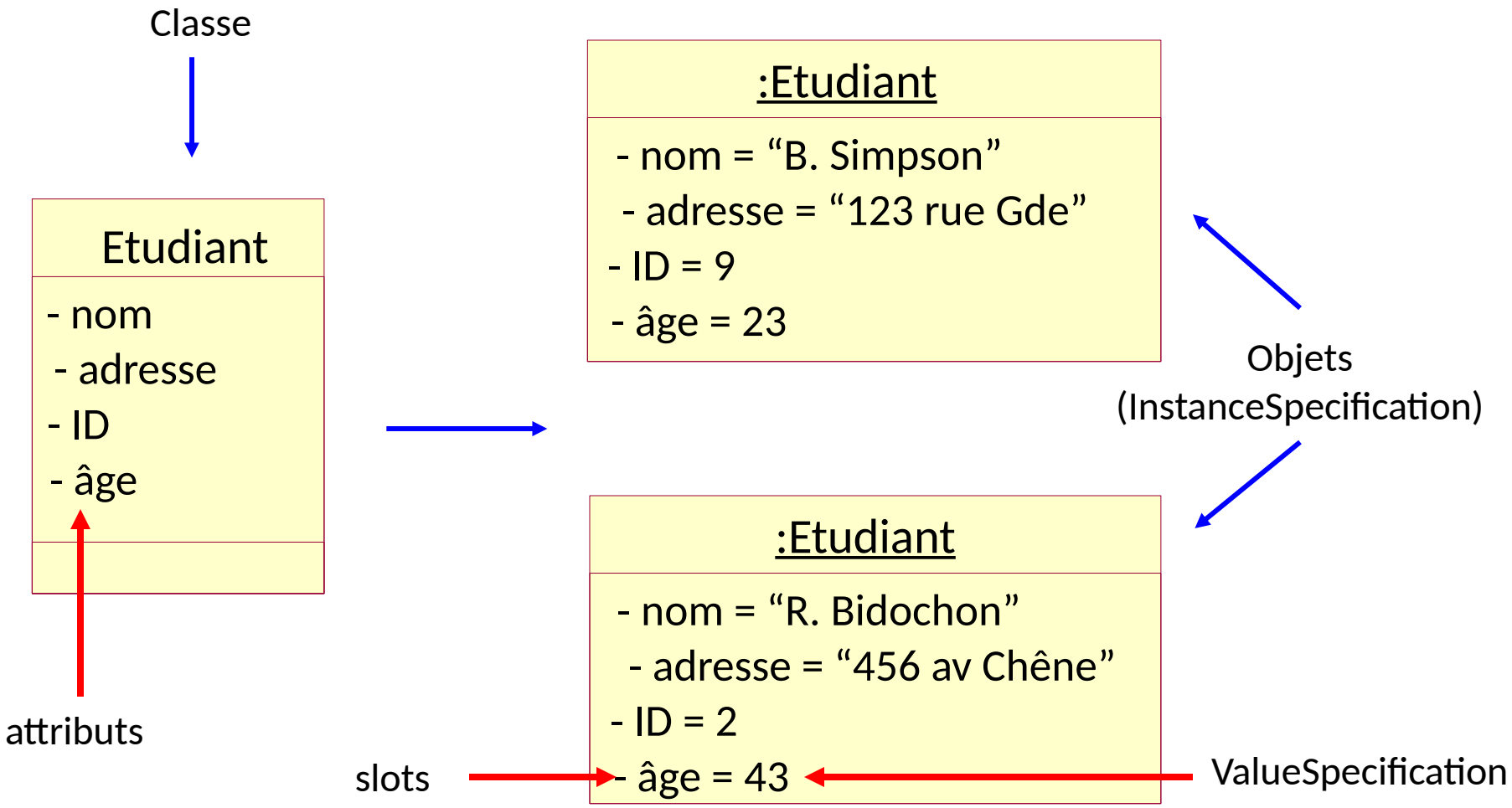
Qu'est-ce qu'un attribut?

- Un attribut est une propriété structurelle nommée dont le type décrit le domaine des valeurs que l'instance peut prendre.
  - Une classe peut avoir un nombre quelconque d'attributs y compris 0.



Université Nice Sophia Antipolis

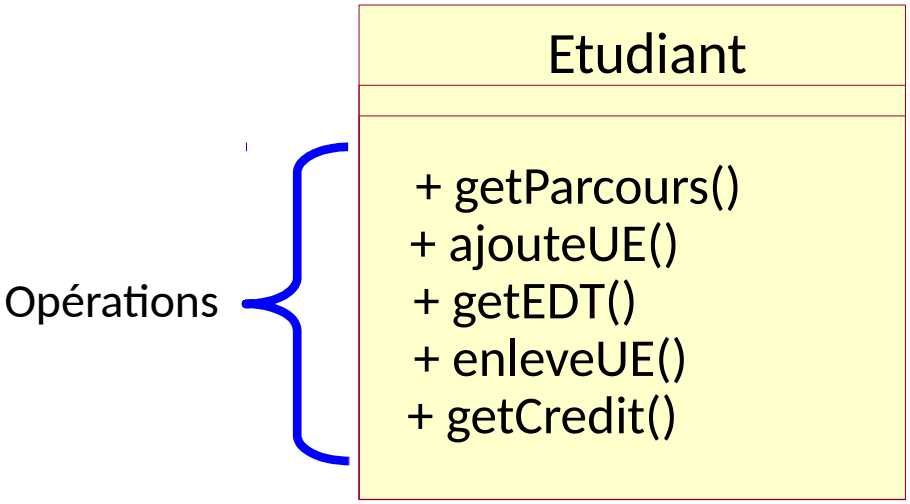
Attributs et *slots*



Université Nice Sophia Antipolis

Qu'est-ce qu'une opération?

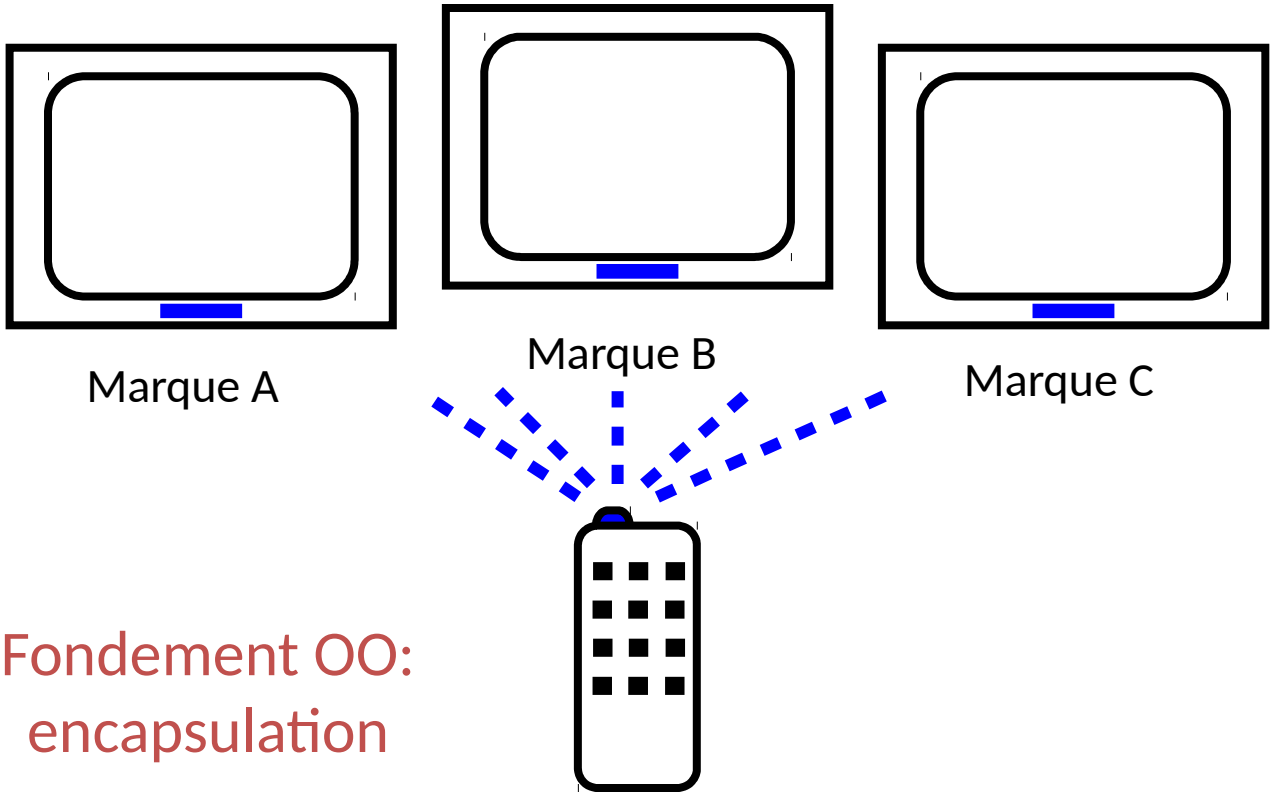
- Un service qui peut être invoqué par un objet pour effectuer un comportement. Une opération a une signature, qui définit les paramètres formels possibles
- Une classe peut avoir un nombre quelconque d'opérations



Université Nice Sophia Antipolis

Le polymorphisme ?

- Capacité à cacher une ou plusieurs implantations derrière une interface

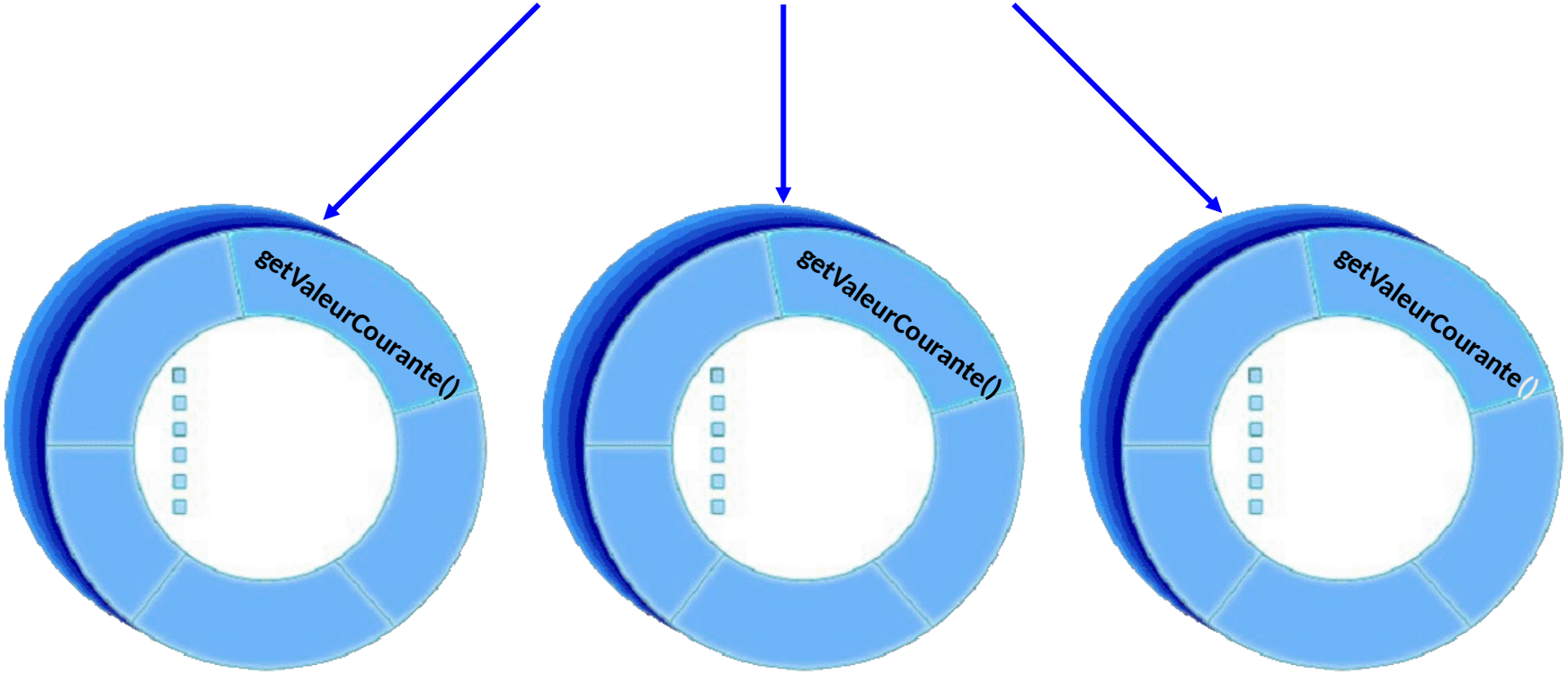


Fondement OO: encapsulation

Université Nice Sophia Antipolis

Exemple : polymorphisme

valeurMonetaire.getValeurCourante()



Obligation Action Fond mutualiste

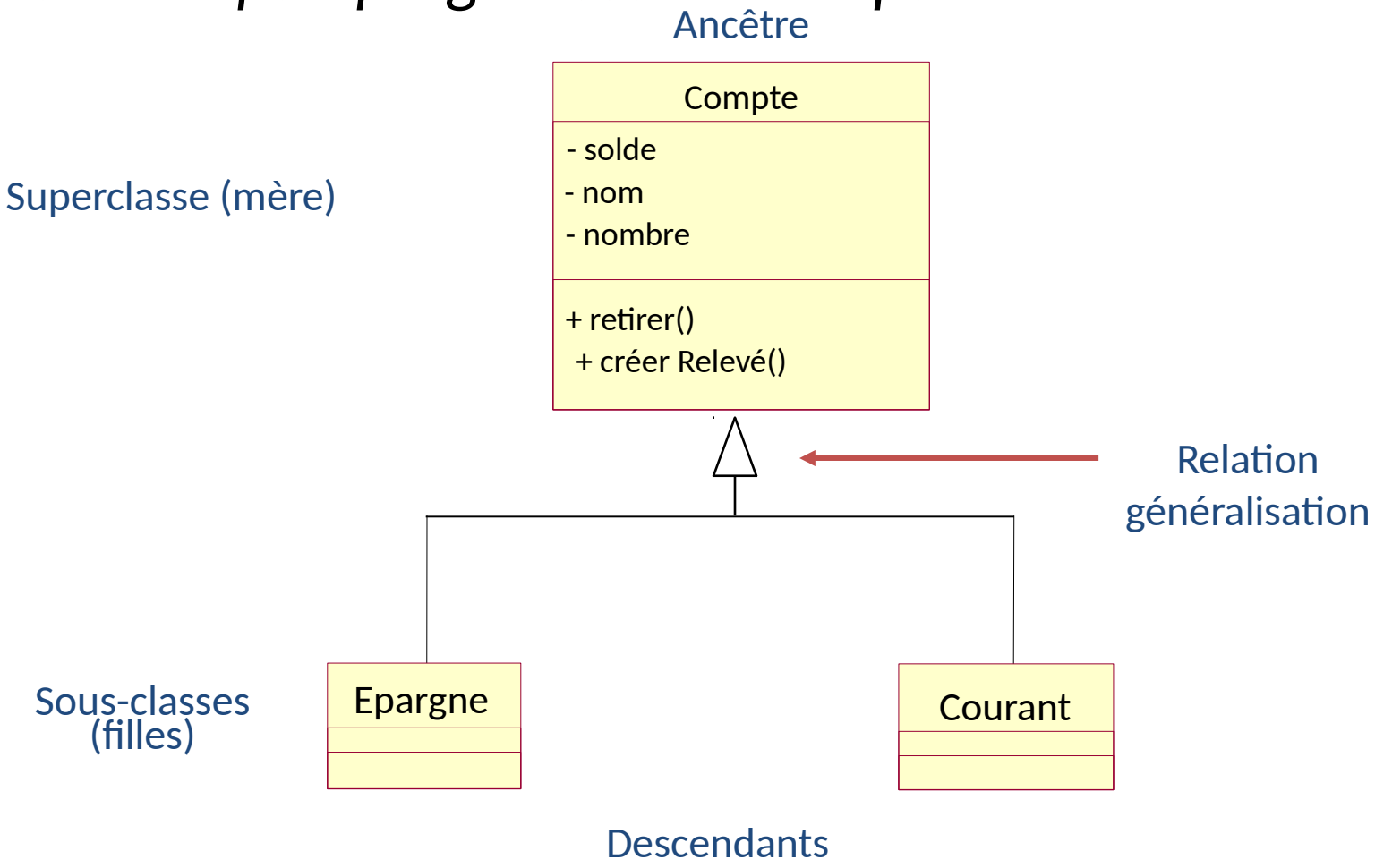
Université Nice Sophia Antipolis

La généralisation ?

- Une relation entre classes dans laquelle une classe **partage** la structure et le comportement de une ou plusieurs autres classes.
- Définit une hiérarchie d'abstractions selon laquelle une classe fille **spécialise** une ou plusieurs classes mères.
  - Héritage simple.
  - Héritage multiple.
- C'est une relation de type **"est-un".**

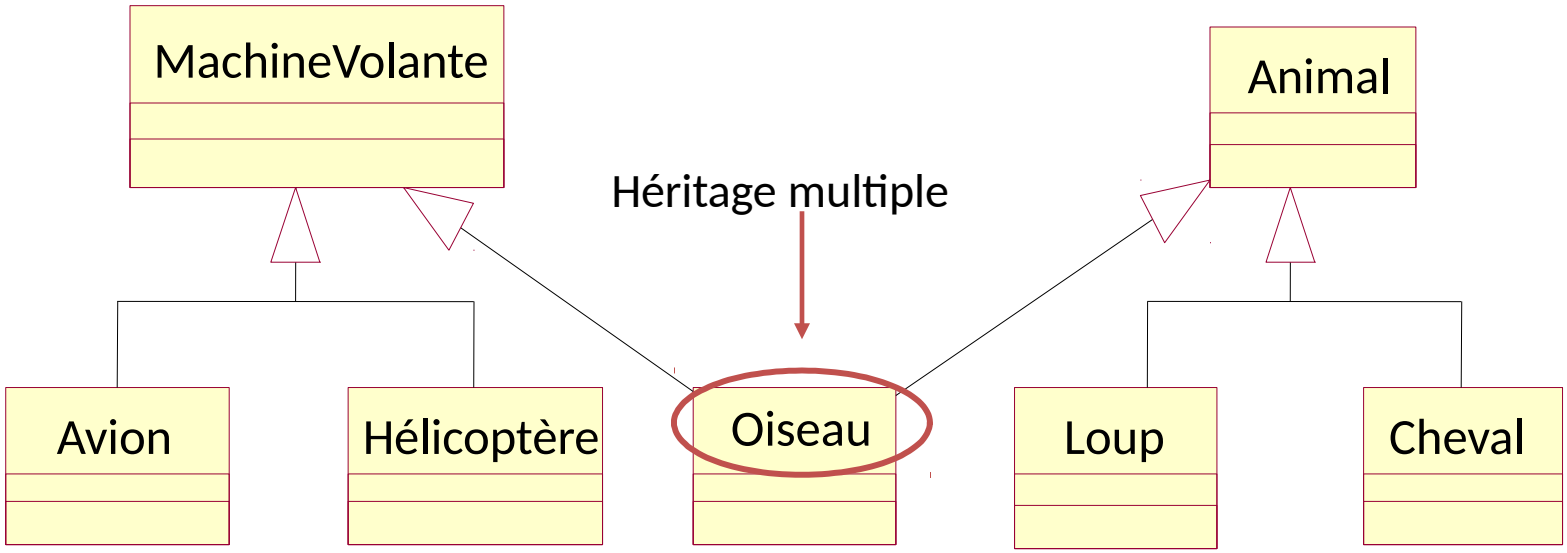
Exemple: Héritage simple

- Un *CompteEpargne* **est-un** *Compte*



Exemple: héritage multiple

- Une classe peut hériter de plusieurs

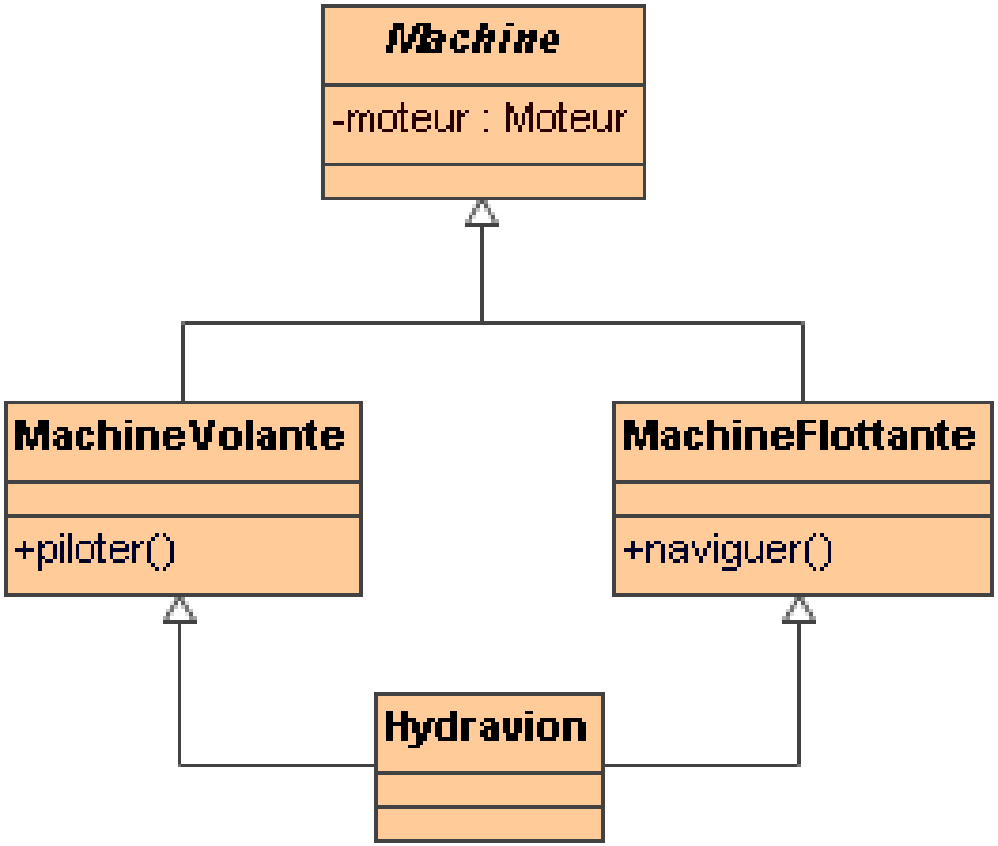


Utiliser l'héritage avec prudence et seulement si indispensable!

Université Nice Sophia Antipolis

Problème avec l'héritage multiple

- Combien de moteurs à l'hydravion ?



Université Nice Sophia Antipolis

De quoi hérite-t-on ?

- Une sous-classe hérite les attributs, les opérations et les références de ses parents.
- Une sous-classe peut:
  - Ajouter des attributs, des opérations, des références.
  - Redéfinir des opérations héritées.
- Les catégories communes sont montrées dans la classe mère la plus haute possible

L'héritage permet d'unifier les aspects communs entre classes.

L'héritage est un des mécanismes pour réaliser le polymorphisme!

Université Nice Sophia Antipolis

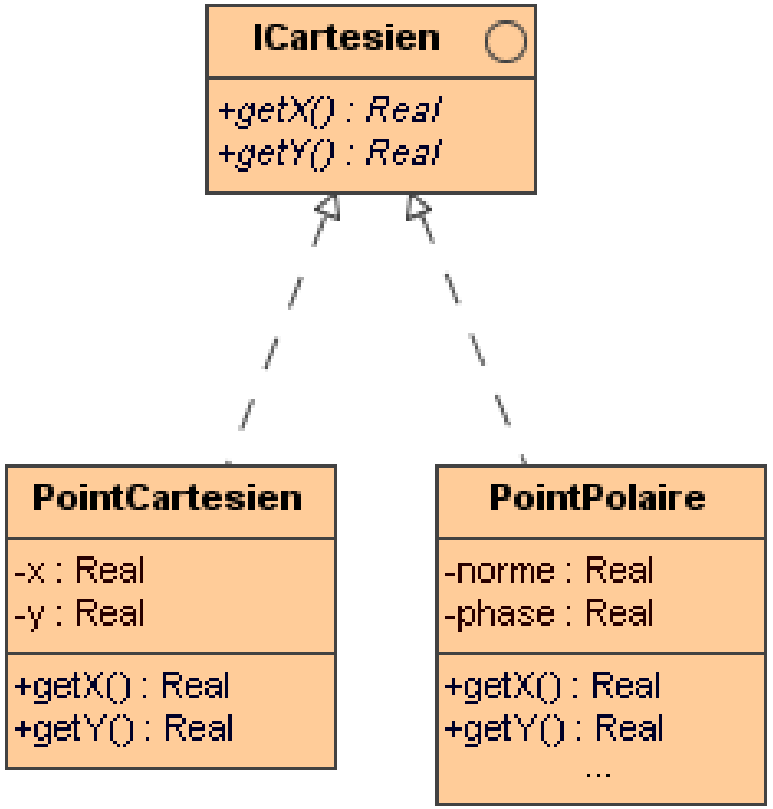
Les interfaces

- Les interfaces définissent un ensemble de caractéristiques et **obligations** cohérentes que doit remplir une classe donnée pour **offrir un service** particulier
  - Les classes peuvent réaliser une interface
  - Cela ne signifie pas nécessairement que les classes possèdent les attributs définis par l'interface !
  - L'utilisation d'interface garantie une modularité.

Université Nice Sophia Antipolis

Exemple : géométrie cartésienne

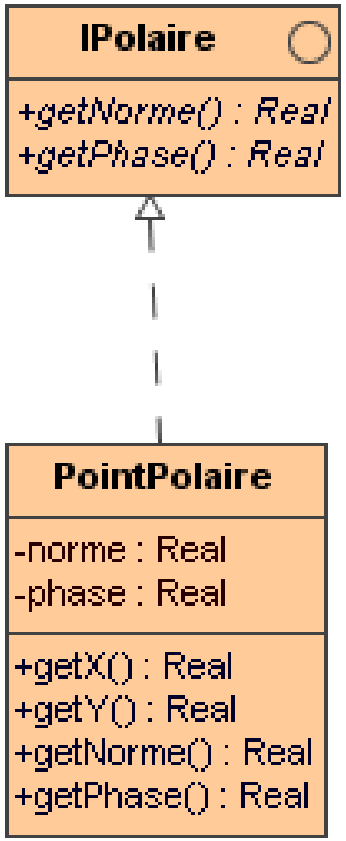
- De quoi a-t-on besoin en géométrie cartésienne ?
  - D'une abscisse et d'une ordonnée

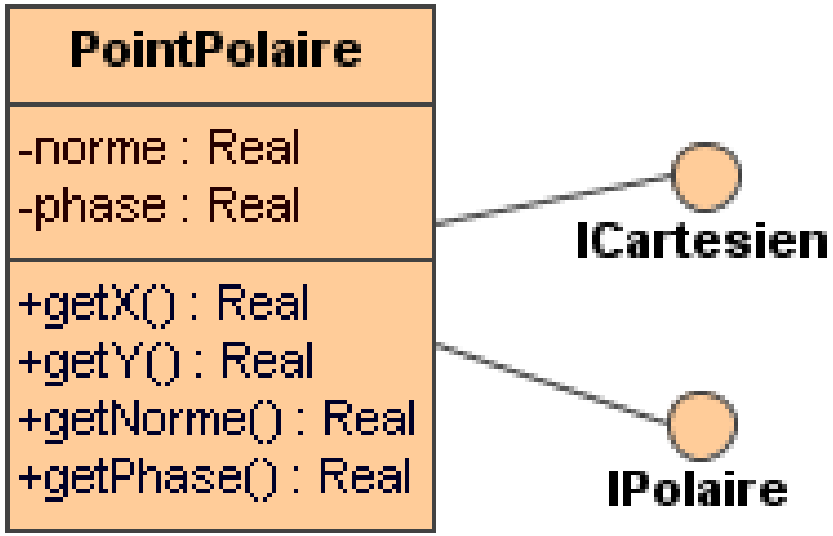


Université Nice Sophia Antipolis

Géométrie polaire

- En géométrie polaire ?
  - D'une norme et d'une phase

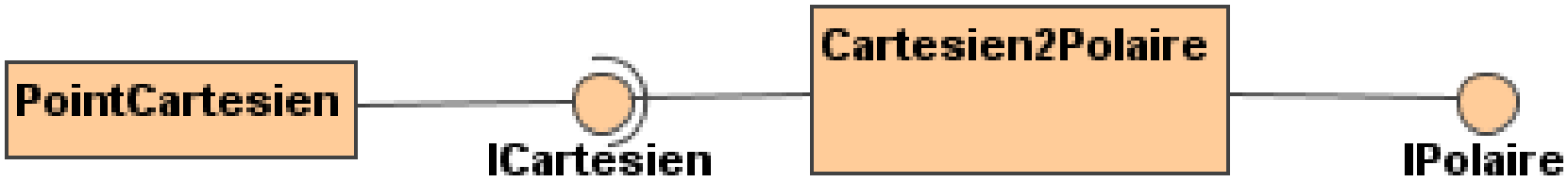




Université Nice Sophia Antipolis

De cartésien vers polaire

- Au moins 2 solutions
  - On pourrait modifier **`PointCartesien`**
  - On peut mettre un filtre de transformation



Université Nice Sophia Antipolis

Les paquetages?

- Un mécanisme pour grouper les éléments
- Un élément de modèle qui en contient d'autres
- Les paquetages peuvent contenir
  - Des classes, mais aussi des machines à états, des activités, …

Université Nice Sophia Antipolis

Exemple de paquetage

- Le paquetage, Gestion Université, contient un paquetage et cinq classes.