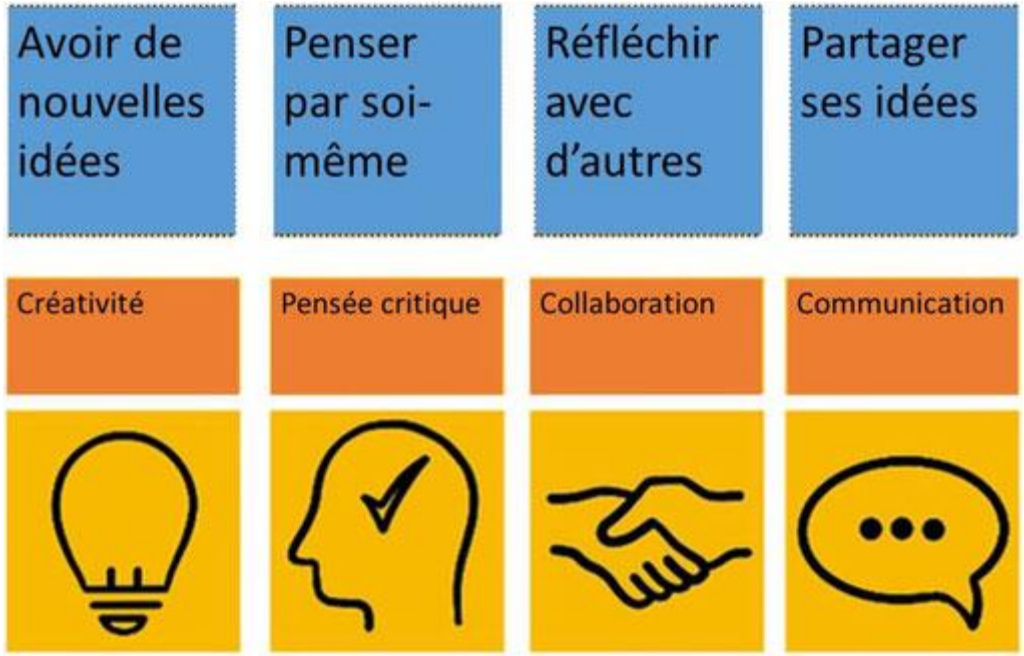
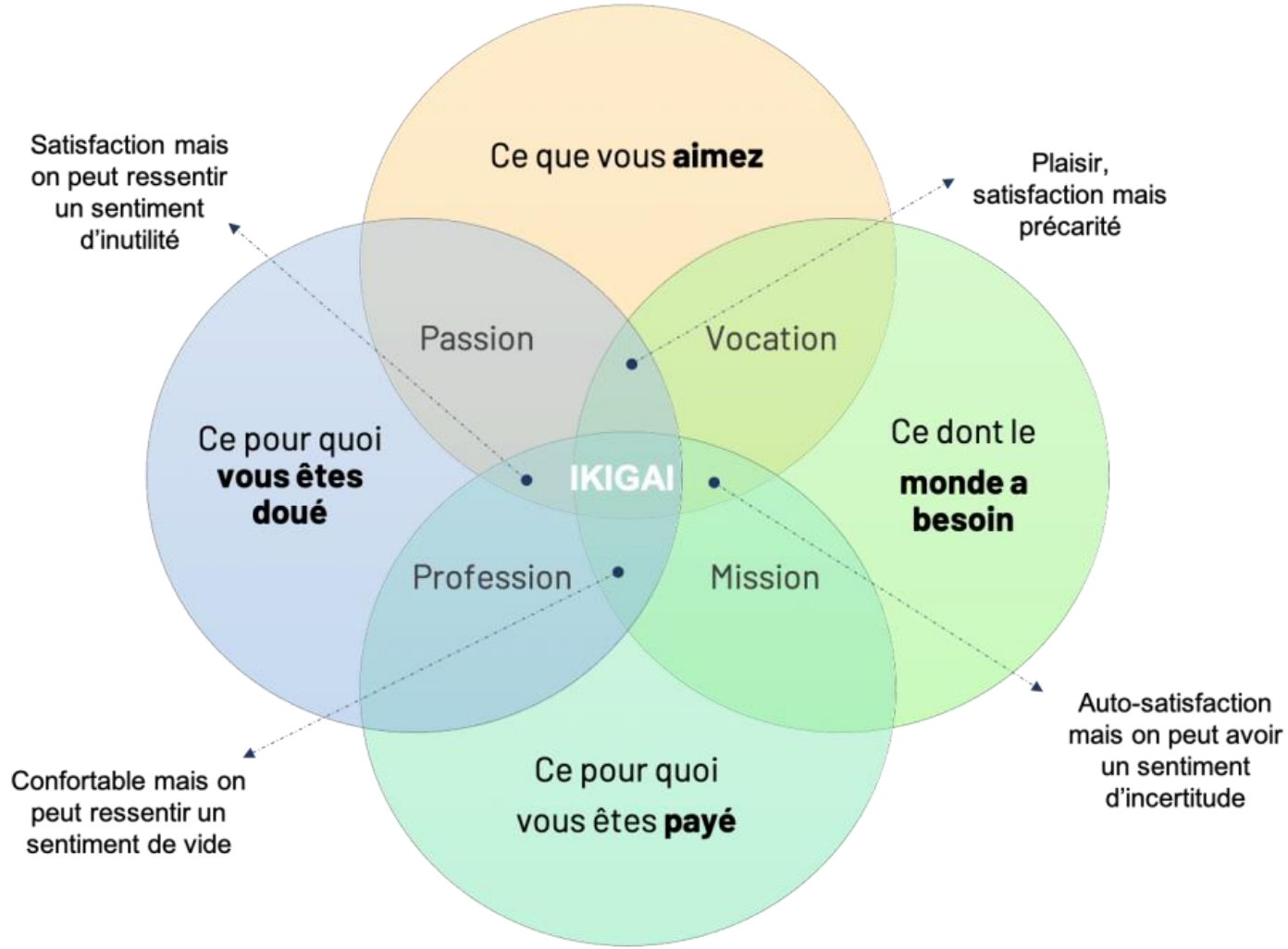
# Introduction au langage UML

# Le modèle

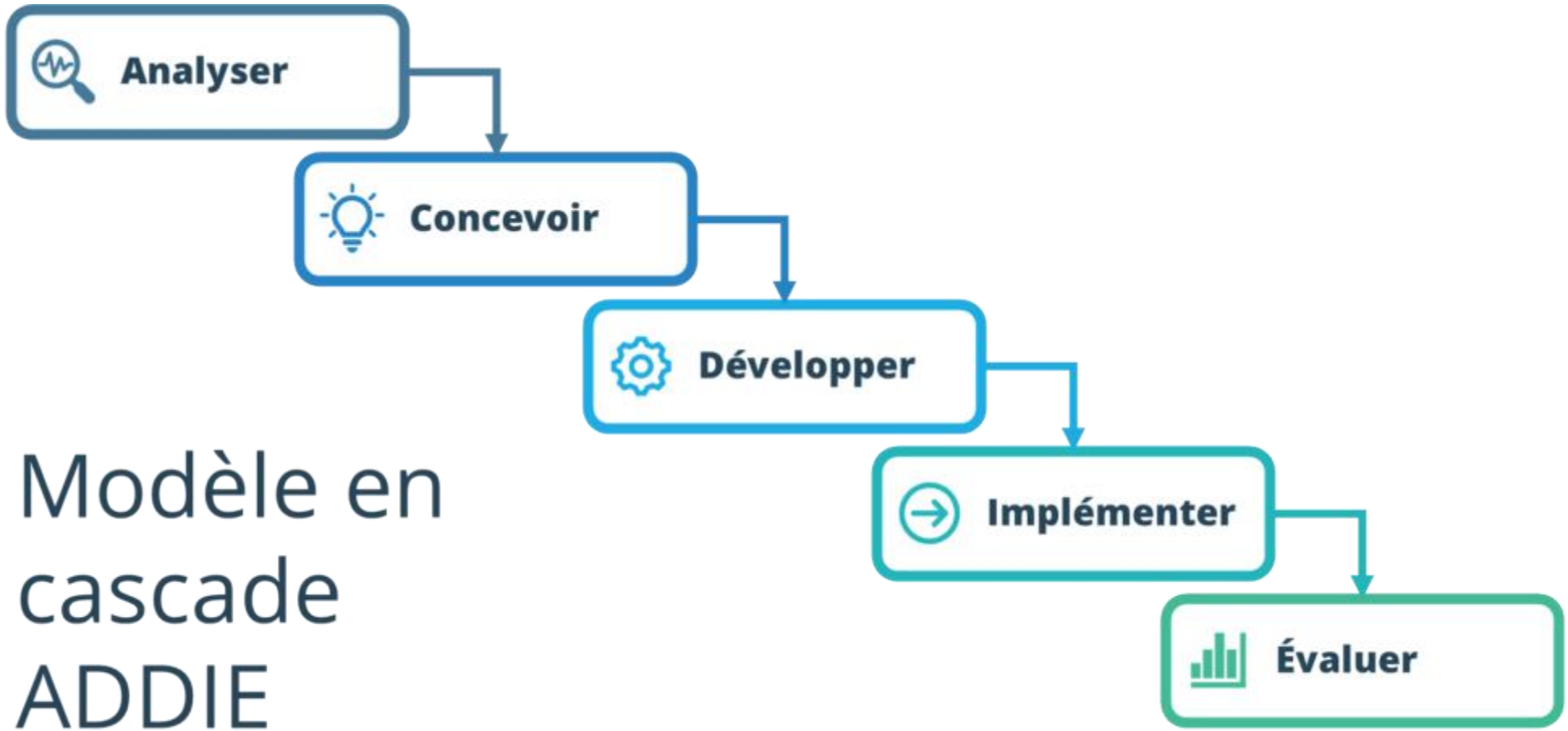
- Comme toutes les étapes du cycle de vie du logiciel, la modélisation est une phase importante au développement du logiciel. C'est l'étape ou on définit l'architecture du système logiciel à réaliser. Cette architecture aide à communiquer les besoins du client d'une partie et symbolise un modèle de solution ou ce qu'on appelle un **modèle**.
- Un modèle est une simplification de la réalité qui permet de mieux comprendre le système à développer.
- Un modèle permet de communiquer une solution acceptable du système informatique. Cette communication est essentielle pour aboutir à une compréhension commune aux différentes parties prenantes, et précise d'un problème donné.
- Un modèle est une abstraction d'objets de la réalité. C'est donc une simplification du monde réel. La problématique consiste alors à trouver le bon niveau d'abstraction et à exprimer les concepts du monde réel dans un langage assez abstrait mais aussi précis qu'un langage de programmation pour que ce langage soit interprétable par un programme informatique.

| Avoir de nouvelles idées | Penser par soi-même | Réfléchir avec d'autres | Partager ses idées |
|---|---|---|---|
| Créativité | Pensée critique | Collaboration | Communication |





# Modèle en cascade ADDIE



# La modélisation

- La modélisation est le processus de création d'un modèle représentatif du système réel en utilisant des méthodes et des langages spécifiques à l'exemple de l'UML, Réseaux de pétri, Merise, etc.
- Les principes de la modélisation comprennent:
  1. Abstraction: réduction d'un système complexe en termes simples et pertinents pour la résolution du problème.
  2. Simplification: suppression des détails inutiles pour se concentrer sur les aspects les plus importants du système.
  3. Généralisation: généralisation des caractéristiques communes à plusieurs objets pour les modéliser sous une forme plus générale.
  4. Formalisation: représentation du système sous forme de formules mathématiques, de diagrammes, de graphiques, etc.
  5. Validation: vérification de la pertinence et de la fiabilité du modèle en le comparant aux données du monde réel.
- Ces principes aident à représenter de manière cohérente et efficace les systèmes complexes et à en étudier le comportement.

# La modélisation

**Pourquoi** ?

- Le modèle du système dans les premières phases du cycle de vie de logiciel est nécessairement une simplification du système réel. Le processus de modélisation vise à mieux cerner les limites du système à réaliser. Ensuite, les modèles sont de plus en plus utilisés pour aboutir au code.
- Modéliser un système avant sa réalisation permet de mieux comprendre le fonctionnement du système. Elle permet de maîtriser la complexité d'un système et d'assurer sa cohérence.

**Comment ?**

- La modélisation se fait à l'aide d'une méthode et outils. Les méthodes d'analyse et de conception fournissent une méthodologie et des notations standards qui aident à concevoir des logiciels de qualité.

# La modélisation

**La modélisation orientée objet**

L'approche orientée objet considère le logiciel comme une collection d'objets dissociés, identifiés et possédant des propriétés. Une propriété est soit un attribut (i.e. une donnée caractérisant l'état de l'objet), soit une fonction.

La fonctionnalité du logiciel émerge alors de l'interaction entre les différents objets qui le constituent. L'une des particularités de cette approche est qu'elle rapproche les données et leurs traitements associés au sein d'un unique objet.

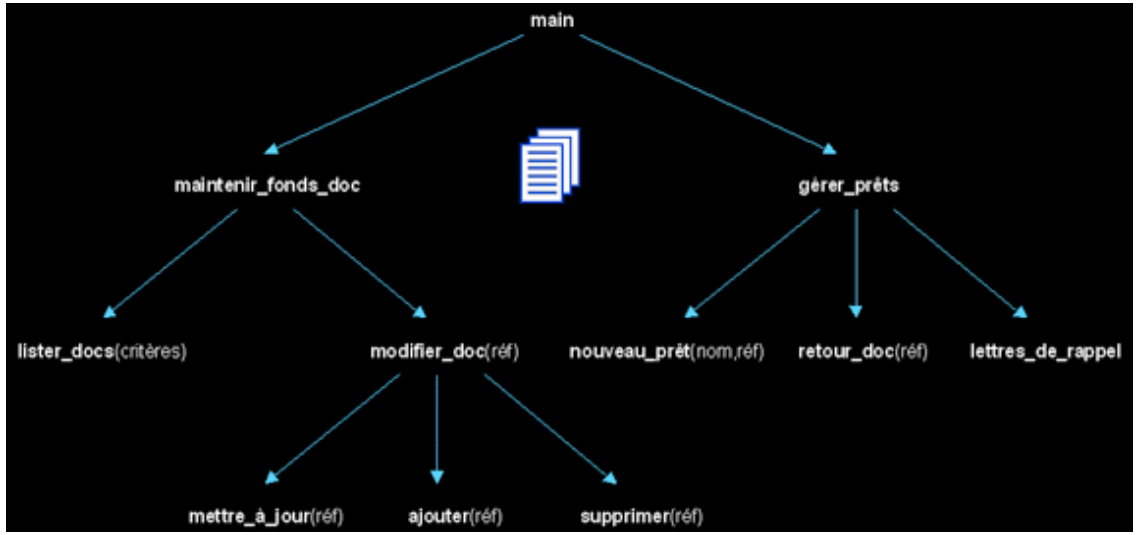
# La modélisation

Un objet est caractérisé par :

- **L'identité**: L'objet possède une identité, qui permet de le distinguer des autres objets, indépendamment de son état
- **Les attributs**:  Il s'agit des données caractérisant l'objet. Ce sont des variables stockant des informations sur l'état de l'objet.
- **Les méthodes**: Les méthodes d'un objet caractérisent son comportement, c'est-à-dire l'ensemble des actions (appelées opérations) que l'objet peut réaliser. Ces opérations permettent de faire réagir l'objet aux sollicitations extérieures (ou d'agir sur les autres objets). De plus, les opérations sont étroitement liées aux attributs, car leurs actions peuvent dépendre des valeurs des attributs, ou bien les modifier

# UML

- Lorsqu'il faut approcher un problème informatique, l'approche intuitive est de le découper en aspects fonctionnels




- En découpant le logiciel en fonctions, il devient difficile de faire une simple mise à jour. En effet, un changement effectué sur une des fonctions peut impliquer des changements majeurs dans une multitude d'autres fonctions.

# UML

**Pourquoi utiliser une approche orientée objet?**

- Un *objet*, en génie logiciel, constitue une entité autonome, qui regroupe un ensemble de propriétés cohérentes et de traitements associés.
- Un objet est une instance de classe
- Une classe est un type de données abstrait, caractérisé par des propriétés (attributs et méthodes) communes à des objets et permettant de créer des objets possédant ces propriétés

# UML

Une approche Objet permet d'obtenir

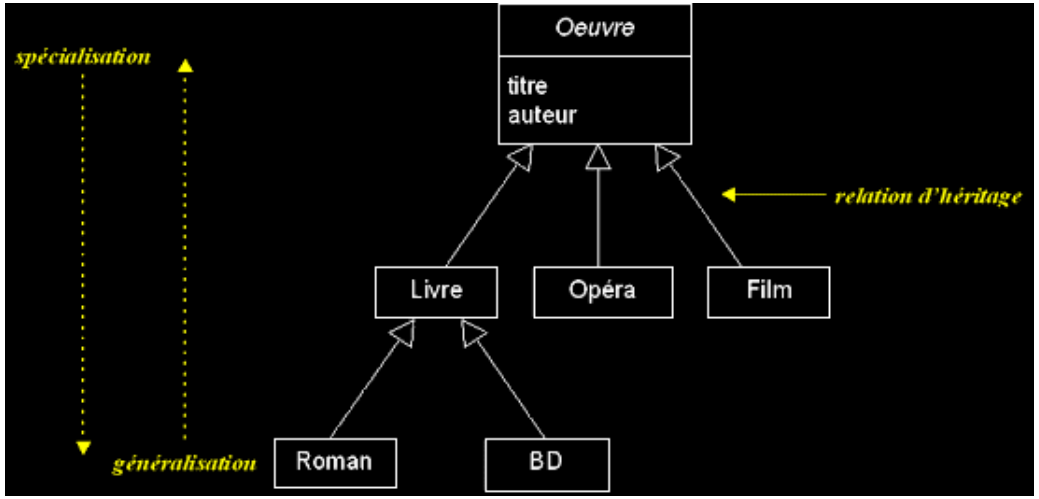
Encapsulation

- consiste à masquer les détails d'implémentation d'un objet, en définissant une interface
- l'interface est la vue externe d'un objet, elle définit les services accessibles (offerts) aux utilisateurs de l'objet
- l'encapsulation facilite l'évolution d'une application car elle stabilise l'utilisation des objets : on peut modifier l'implémentation des attributs d'un objet sans modifier son interface
- l'encapsulation garantit l'intégrité des données, car elle permet d'interdire l'accès direct aux attributs des objets (utilisation d'accesseurs)

# UML

## Héritage

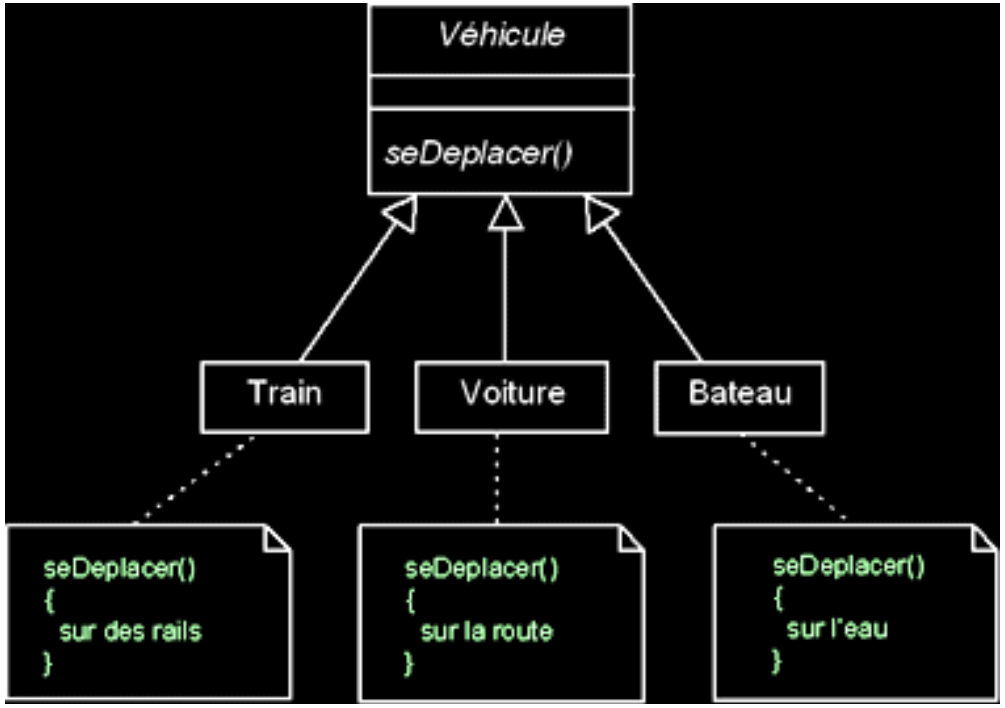
- l'héritage est un mécanisme de transmission des propriétés d'une classe (ses attributs et méthodes) vers une sous-classe
- une classe peut être spécialisée en d'autres classes, afin d'y ajouter des caractéristiques spécifiques ou d'en adapter certaines
- plusieurs classes peuvent être généralisées en une classe qui les factorise, afin de regrouper les caractéristiques communes d'un ensemble de classes
- la spécialisation et la généralisation permettent de construire des hiérarchies de classes. L'héritage peut être simple ou multiple
- l'héritage évite la duplication et encourage la réutilisation*

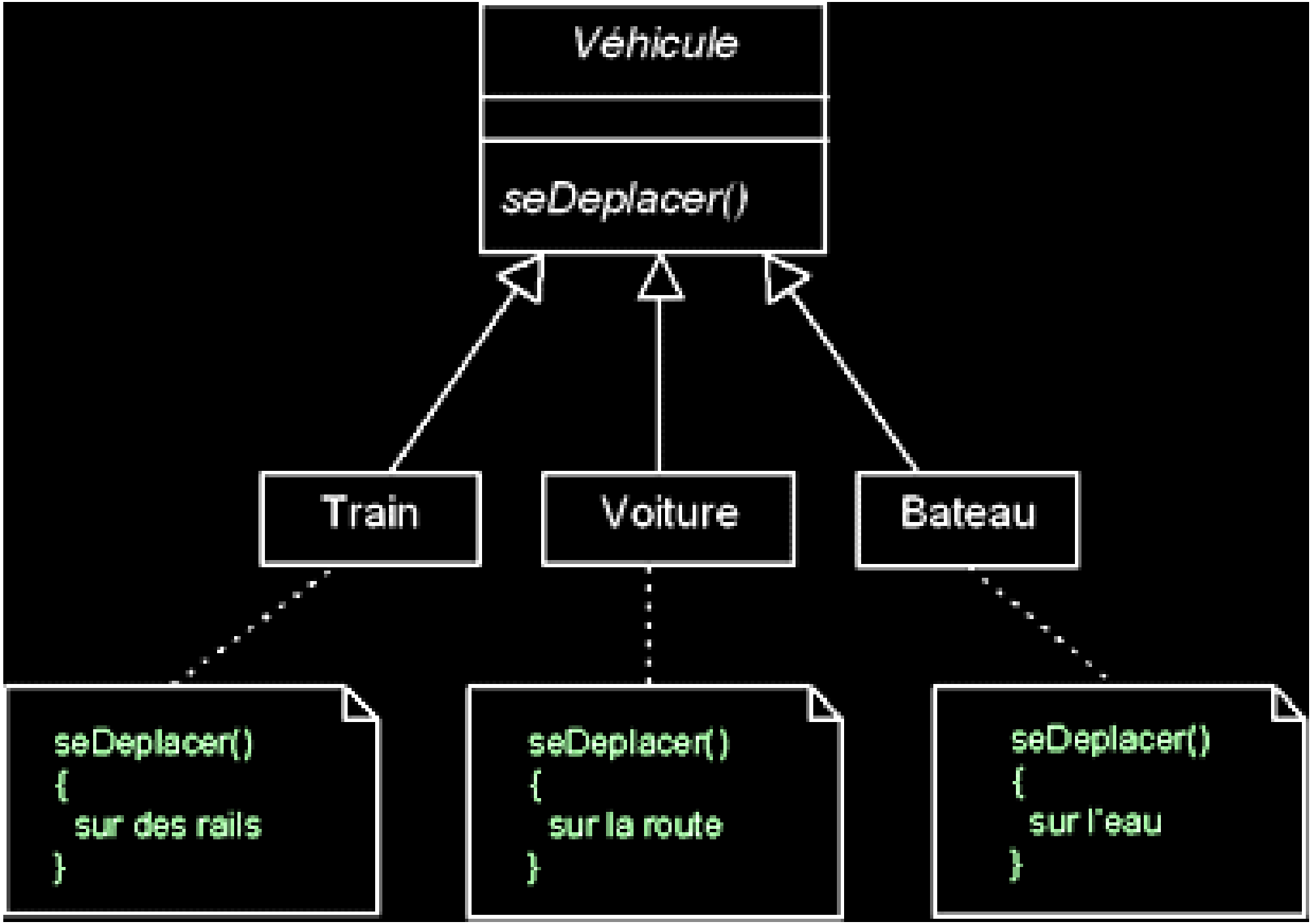


# UML

Polymorphisme

- le polymorphisme représente la faculté d'une méthode à pouvoir s'appliquer à des objets de classes différentes
- le polymorphisme augmente la généricité du code

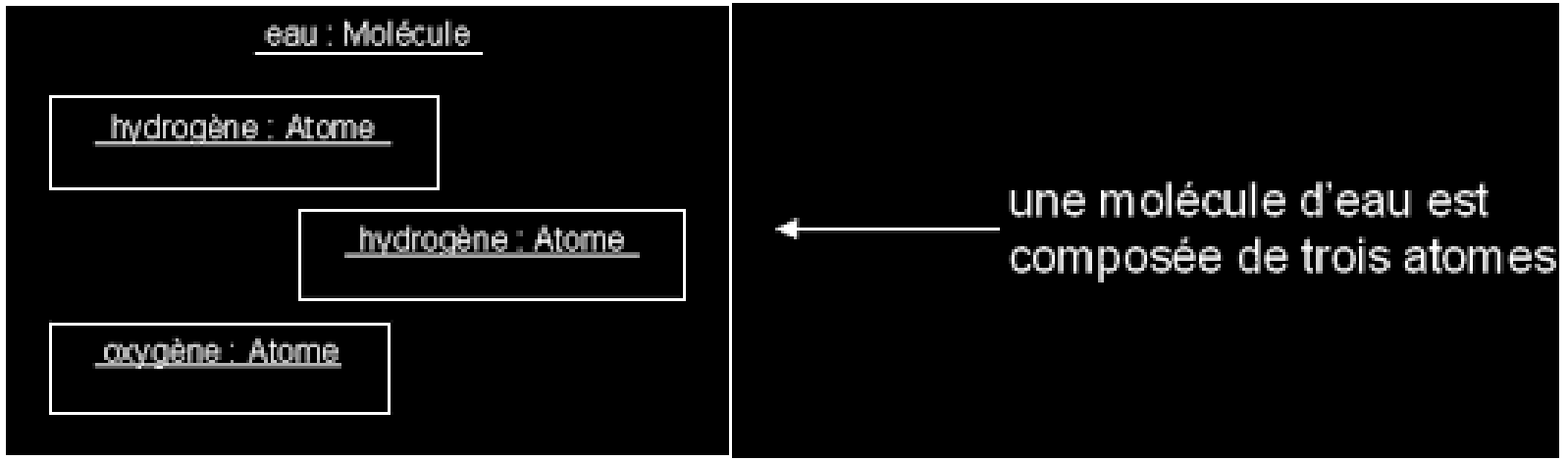



Véhicule
seDeplacer()
Train
Voiture
Bateau
seDeplacer()
{
sur des rails
}
seDeplacer()
{
sur la route
}
seDeplacer()
{
sur l'eau
}

# UML

Agrégation

- relation entre deux classes, spécifiant que les objets d'une classe sont des composants de l'autre classe. Une relation d'agrégation permet donc de définir des objets composés d'autres objets.
- l'agrégation permet d'assembler des objets de base, afin de construire des objets plus complexes



# UML

L'approche objet c'est:

- un ensemble de concepts stables, éprouvés et normalisés
- une solution destinée à faciliter l'évolution d'applications complexes

Les obstacles de l'approche objet:

- l'approche objet est moins intuitive que l'approche fonctionnelle : il est plus facile de penser en terme de fonctions que de penser en terme d'objets et d'interactions entre objets
- l'application des concepts objets nécessite une grande rigueur: il est plutôt difficile de décrire la structure objet d'un système de manière pertinente sans risque d'ambiguïtés

Il nous faut donc un "langage" pour exprimer de façon claire et précise les concepts objets du système que l'on veut modéliser. Ce "langage" est UML (Unified Modeling Language)

# UML

- **UML est un langage de modélisation orienté objet**, c’est-à-dire que toutes les entités modélisées sont des objets ou se rapportent à des objets, par exemple : un objet possède une structure de données (avec ce qui s’appelle des « attributs ») et des comportements (avec ce qui s’appelle des « opérations »).
- **UML est un langage et possède les attributs d’un langage**. Ainsi, étant graphique, UML permet de visualiser le système réalisé ; le modèle est divisé en vues sélectionnant les éléments pertinents puis en diagrammes de différents types. L’aspect graphique d' UML retient le premier l’attention de ses utilisateurs. Comme pour tout langage, les éléments du langage sont définis de manière précise, complète et sans ambiguïté.

# Modéliser avec UML

- La partie la plus importante dans l'élaboration d'un modèle avec UML est de définir et visualiser le modèle à l'aide de diagrammes.
- Un diagramme UML est une représentation graphique, qui s'intéresse à un aspect précis du modèle ; c'est une perspective du modèle, pas "le modèle"
- Chaque type de diagramme UML possède une structure (les types des éléments de modélisation qui le composent sont prédéfinis)
- Un type de diagramme UML véhicule une sémantique précise (un type de diagramme offre toujours la même vue d'un système)
- Combinés, les différents types de diagrammes UML offrent une vue complète des aspects statiques et dynamiques d'un système