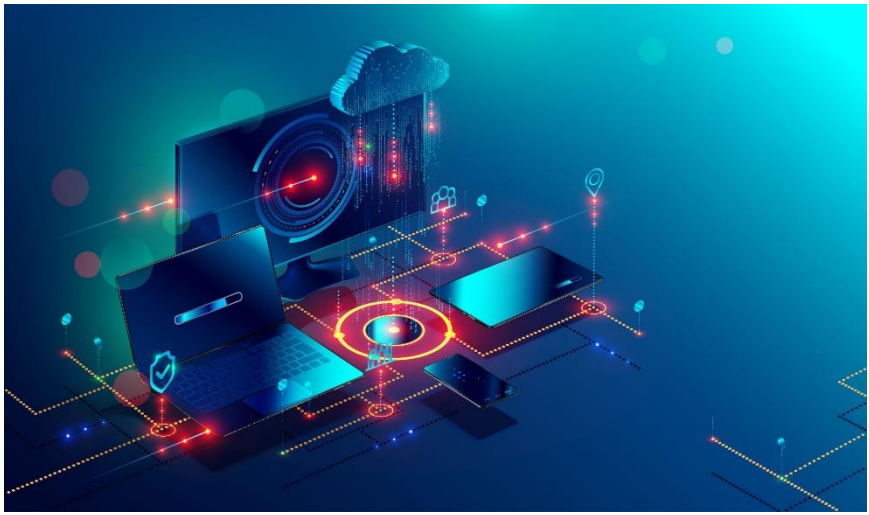
Analyse Orientée Objet

I. Introduction

II. Approche objet et système d'information.

III. Principes Objet

IV. UML

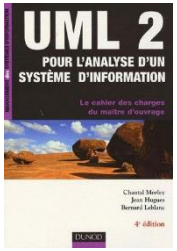


Analyse Orientée Objet

- Objectifs

- Organisation du cours

- Supports



- Evaluation

Analyse Orientée Objet

➢ Objectifs:

- *Introduire la modélisation orientée objet.*
- *Acquérir d'une manière pédagogique et rigoureuse les bases du concept objet.*
- *Introduire le langage UML.*
- *Présenter des éléments méthodologiques d'utilisation des différents types de diagrammes dans un processus de développement.*

Analyse Orientée Objet

- **Organisation du cours:**

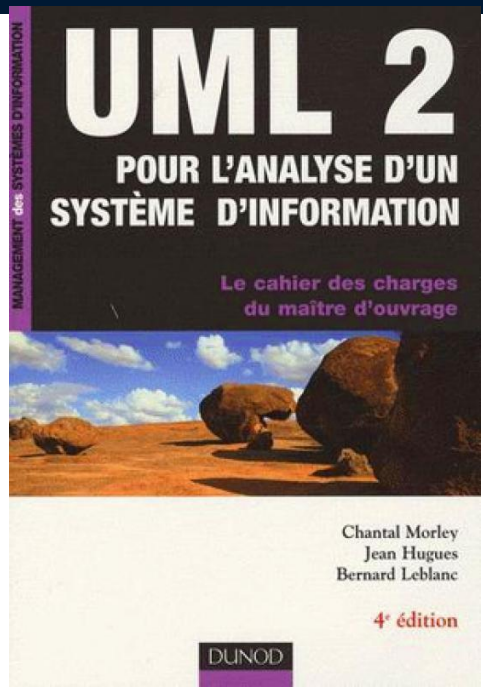
Bloc 2 toutes les options:

Théorie *: 15 h. 2h par semaine*

Laboratoire*: 30h. 2h par semaine en un quadrimestre.*

Analyse Orientée Objet

➢ Supports:



- *UML 2 Pour l'analyse d'un système d'information.*
  Chantal Morley
  Jean Hugues
  Bernard Leblanc

- *Documents dans mon centre de ressources*
  souad.serrhini@hepl.be

Analyse Orientée Objet

➢ Evaluation

- ➢ ***Théorie et Laboratoire:***
  Evaluation continue
  - *Interrogations écrites/orales*
  - *Devoirs/préparations*
  - *Participation au cours*

  → 100%

Examen en septembre.

Analyse Orientée Objet

I. Introduction

II. Approche objet et système d'information.

III. Les principes et concepts objets.

IV. UML.

Analyse Orientée Objet

I. Introduction

II. Approche objet et système d'information.

III. Les principes et concepts objets.

IV. UML.

I. Introduction

Les techniques de programmation n'ont cessé de progresser depuis l'époque de la programmation par cartes perforées à nos jours.

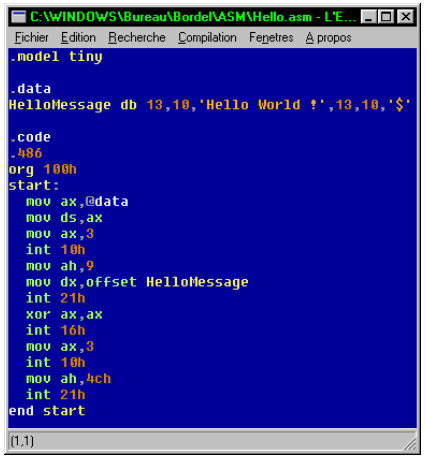
Pourquoi cette évolution?

Besoin de concevoir et de maintenir des applications toujours de plus en plus complexes.

I. Introduction

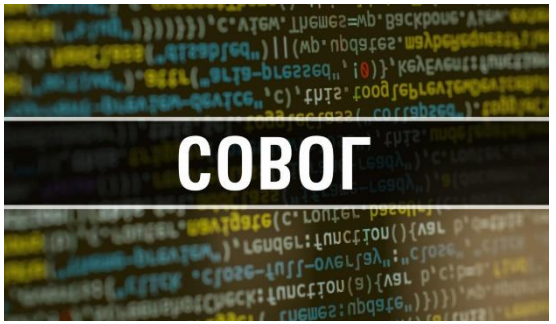
Un peu d'histoire:

- Programmation par cartes perforées switch câblage de 1800 à 1940.



- Assembleur en 1947 avec l'arrivée de l'ordinateur électronique né des besoins de la guerre.

- Des langages évolués: Fortran en 1956 et Cobol en 1959.



I. Introduction

Un peu d'histoire:
Les techniques de programmation étaient basées sur le branchement conditionnel et inconditionnel (goto).

Très difficile à développer et à maintenir.

I. Introduction

Un peu d'histoire:
Apparition de la programmation structurée qui a permis de développer et de maintenir des applications plus ambitieuses.

I. Introduction

Un peu d'histoire:

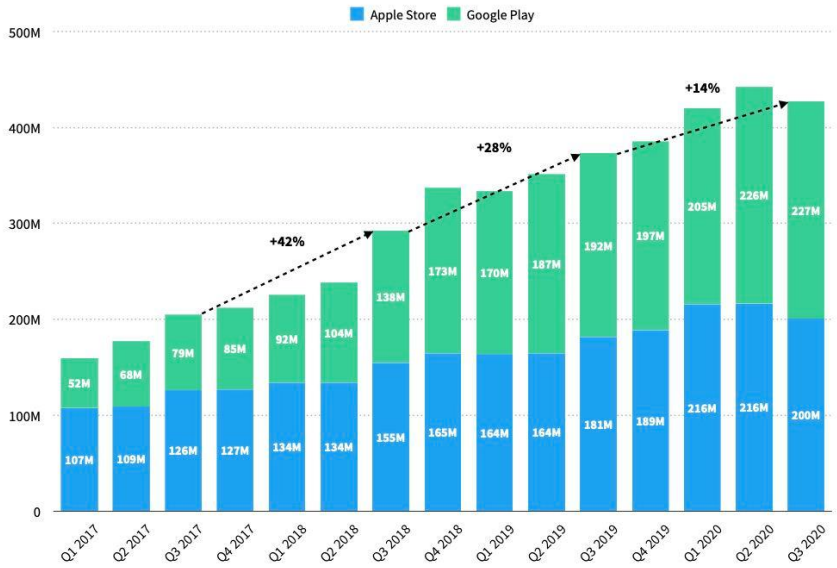
- Pascal en 1970.
- C en 1972.
- Modula et Ada en 1979.

I. Introduction

Un peu d'histoire:
Le génie logiciel est venu placer la méthodologie en cœur du développement logiciel : Merise 1978.

La programmation structurée à également rencontré ses limites.

La taille des applications.



I. Introduction

Historique:

Apparition de la programmation Orientée Objet:

- Simula 67 en 1967.
- Smaltalk en 1976.
- C++ en 1982.
- Java en 1995.
- .....

I. Introduction

Historique:
Cette nouvelle technique de programmation a nécessité la conception de nouvelles méthodes de modélisation des logiciels.

Apparition de Génie logiciel

I. Introduction

Quelques définitions

Informatisation:

Le phénomène le plus important de notre époque.

L'informatique est au cœur de toutes les grandes entreprises.

Le système d'information d'une entreprise se compose de :

- 20 % de matériels.
- 80% de logiciels.

I. Introduction

Quelques définitions

Informatisation:

Aujourd'hui, 90% de nouvelles fonctionnalités des automobiles sont apportées par l'électronique et l'informatique embarquées.

I. Introduction

Quelques définitions

Logiciel:

Ensemble de programmes et éventuellement de documentation, relatifs au fonctionnement d'un ensemble de traitements de l'information.

Le développement de grands logiciels par de grandes équipes pose d'importants problèmes de conception et de coordination.

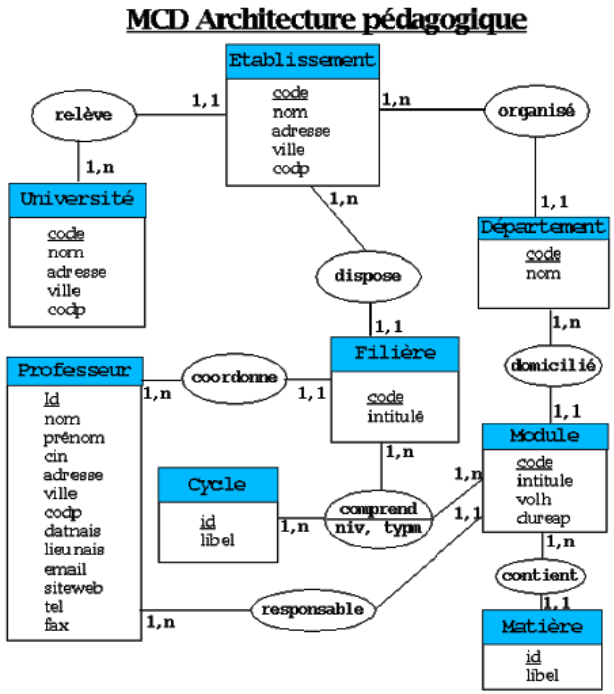
Naissance de MODÉLE

I. Introduction

Qu'est qu'un MODÈLE?

Un modèle est une représentation abstraite et simplifiée d'une entité du monde réel en vue de la décrire et de l'expliquer.

C'est la théorie orientée vers l'action qu'elle doit servir.



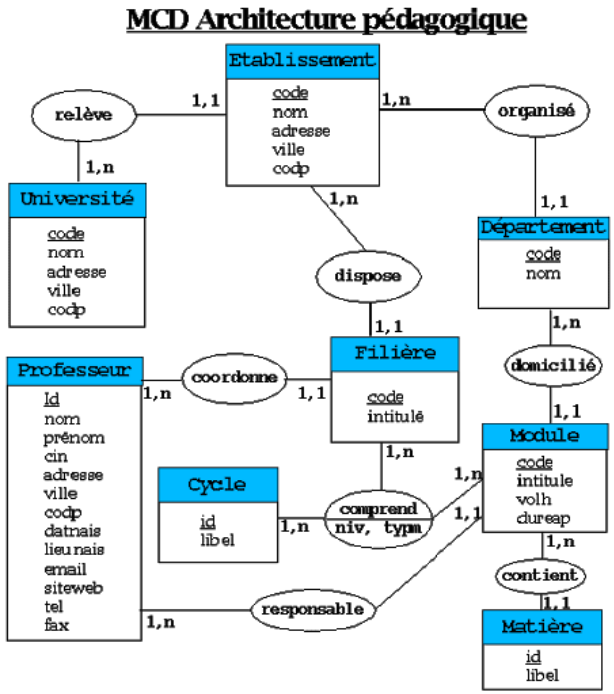
I. Introduction

Pourquoi MODÉLISER?

Modéliser un système avant sa réalisation permet de mieux comprendre son fonctionnement, de maîtriser sa complexité et d'assurer sa cohérence.

Langage commun, précis connu par tous les membres de l'équipe.

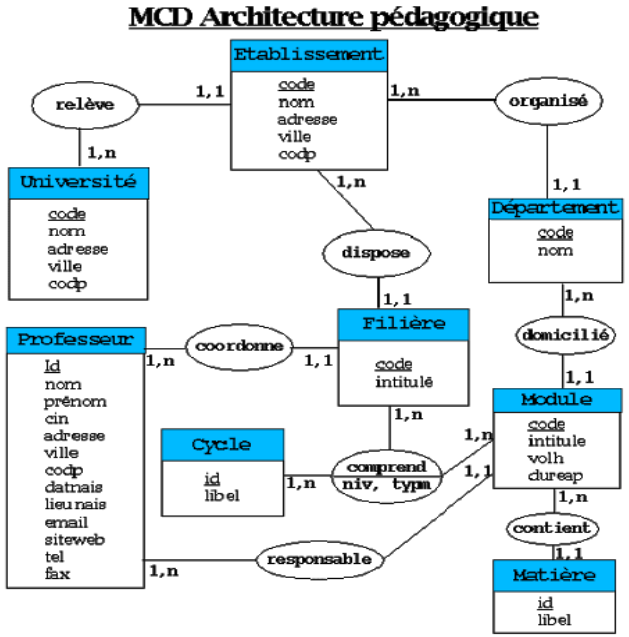
Vecteur privilégié pour COMMUNIQUER



I. Introduction

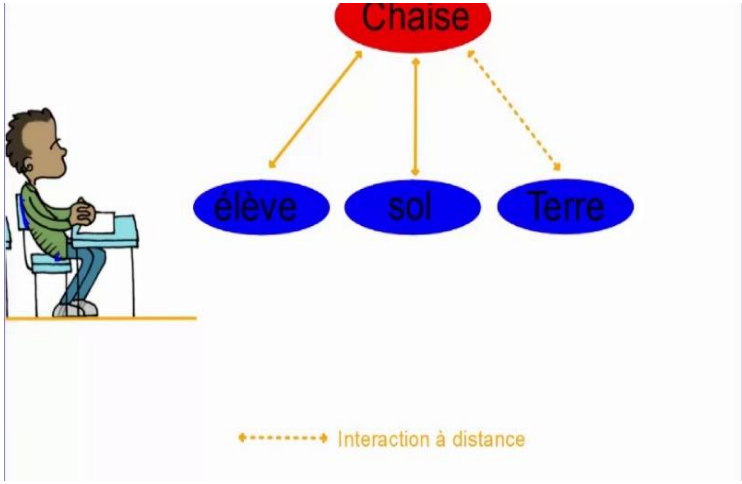
Approche classique

Outil pour représenter l'activité de l'entreprise indépendamment de son organisation.
Séparation **données/Traitements.**



Approche Objet

Considère le logiciel comme une collection d'objets dissociés, identifiés et possédant des caractéristiques structurelles et comportementales.



I. Introduction

La fonctionnalité du logiciel émerge alors de l'interaction entre les différents objets qui le constituent.

Analyse Orientée Objet

I. Introduction

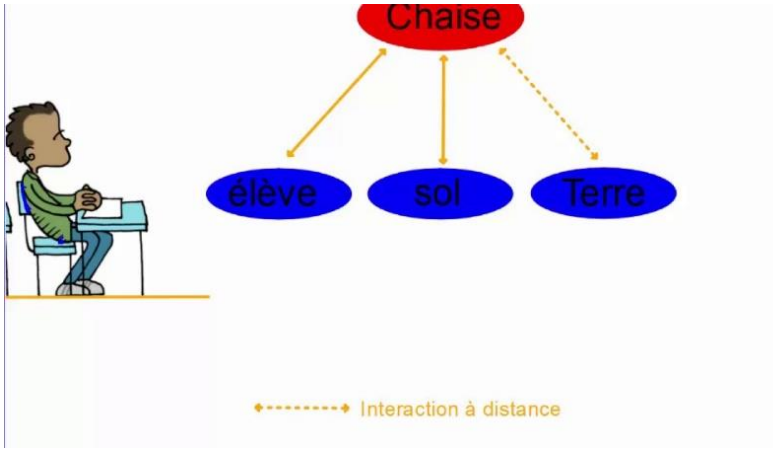
II. Approche objet et système d'information.

III. Les principes et concepts objets.

IV. UML.

II. Approche objet et système d'information

OBJET:



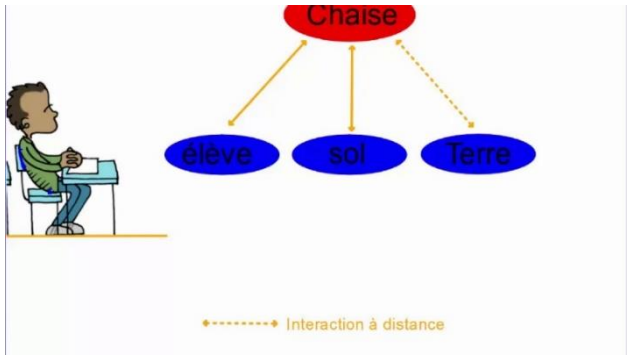
L'approche objet est une manière de représenter les choses d'une part et une technologie pour développer des systèmes d'information d'autre part.

II. Approche objet et système d'information

Une approche naturelle.

Chacun se représente les choses à sa manière, en fonction de:

- Ce qu'elles sont.
- Ce qu'elles font.
- Ce qu'on en fait.



II. Approche objet et système d'information

On commence par percevoir les choses comme un tout, à travers leurs apparences ou leurs comportements.

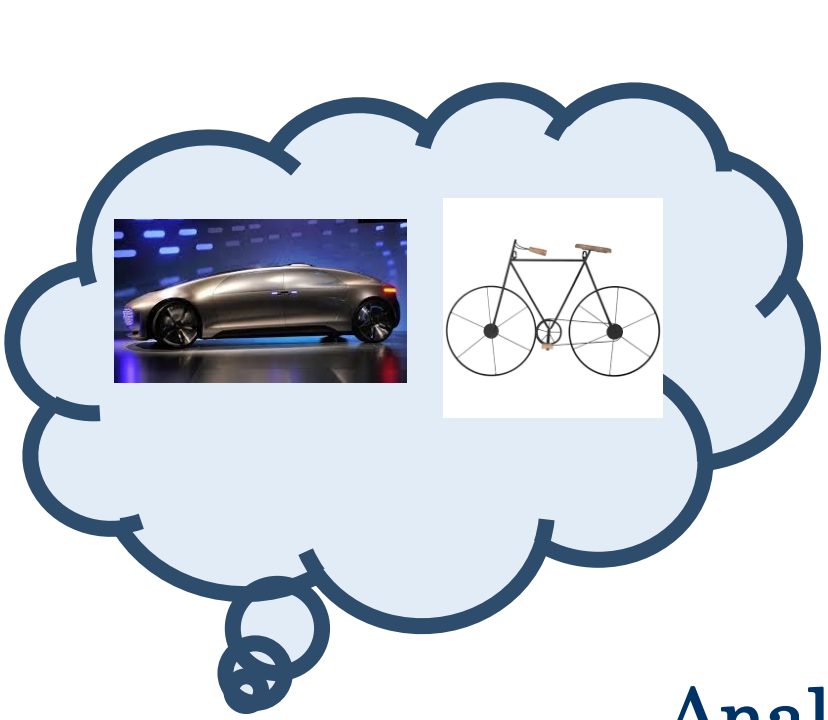
II. Approche objet et système d'information

Ensuite on les décompose, ou au contraire on regroupe des objets qui sont proches, que l'on utilise ensemble ou que l'on remarque en même temps.

Associations

II. Approche objet et système d'information

Ensuite on les <u>décompose</u>, ou au contraire on <u>regroupe</u> des objets qui sont proches, que l'on utilise ensemble, ou que l'on remarque en même temps



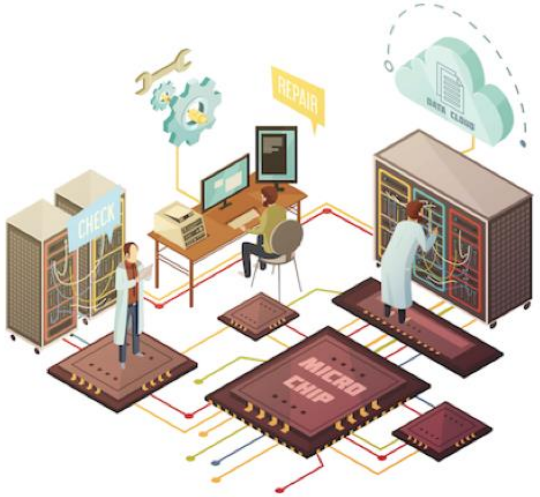
Analogies

II. Approche objet et système d'information

SYSTÉMES D'INFORMATION:

Un système d'information se définit comme une boîte noire caractérisée par:

- Le domaine.
- Les acteurs et interfaces.
- Les services.



II. Approche objet et système d'information

SYSTÉMES D'INFORMATION:

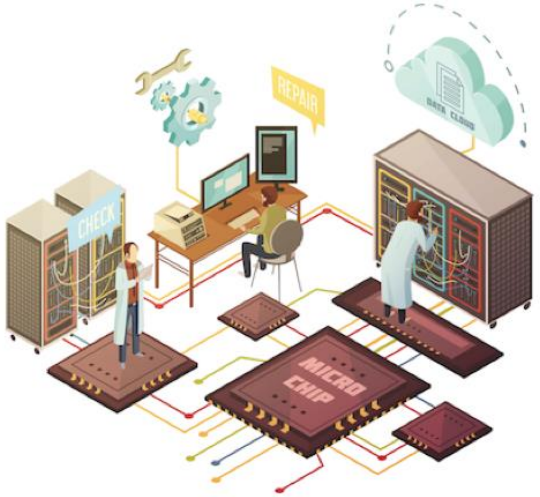
- **Le domaine:** c'est l'ensemble des éléments de l'environnement que le système doit **« connaître »** pour tenir son rôle. Ces éléments peuvent être matériels ou organisationnels. Ils doivent être **« représentés »** par le système.
- **Les acteurs et interfaces:** se sont les **interlocuteurs** du système. Ils assurent la communication entre le domaine et le système.
- **Les services:** définis en terme de situations, événements et actions.

II. Approche objet et système d'information

SYSTÉMES D'INFORMATION:

Un système d'information se définit comme une boîte noire caractérisée par:

- Le domaine.
- Les acteurs et interfaces.
- Les services.



II. Approche objet et système d'information

Objets du domaine:

Le domaine d'un système d'information regroupe les objets que le système doit gérer pour supporter les services demandés.

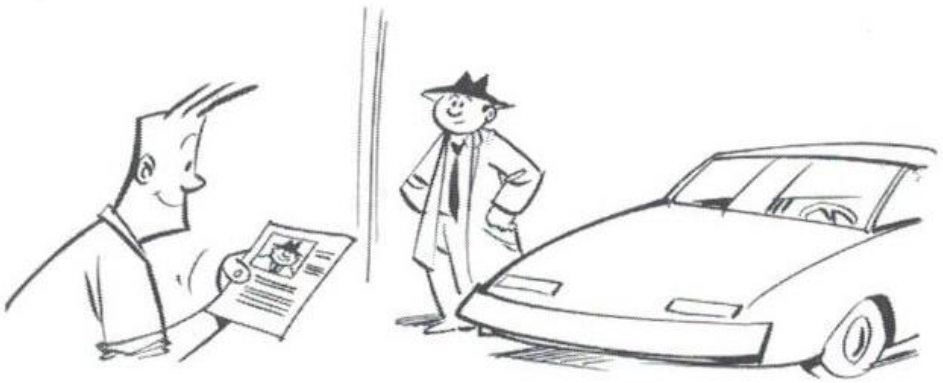
Cela implique l'***identification*** et l'***état*** des objets.

Système d'information

II. Approche objet et système d'information

L'identification des objets:

Le système doit assurer la correspondance entre les objets du domaine et leur représentation, ainsi que les activités qui les concernent.

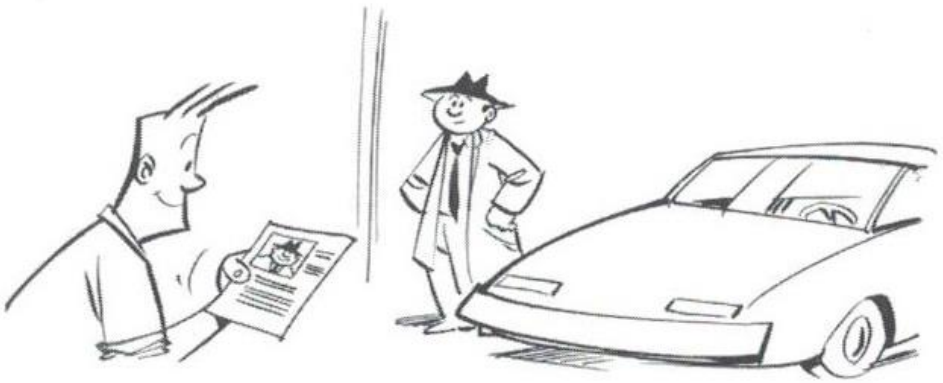


Représentation des objets du domaine

II. Approche objet et système d'information

L'identification des objets:

L'approche Orientée objet considère le logiciel comme une collection d'objets ***identifiés*** et possédant des caractéristiques ***structurelles*** et ***comportementales***.



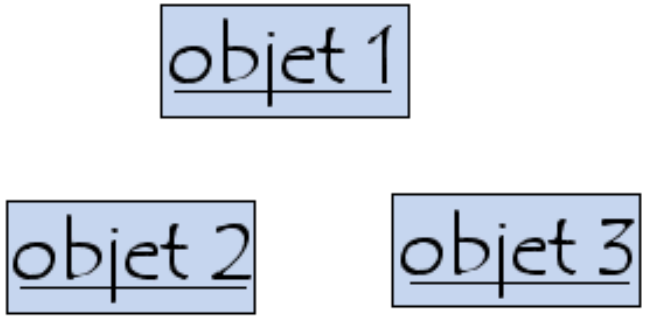
Représentation des objets du domaine

II. Approche objet et système d'information

Identité:

L'objet possède une identité qui permet de le distinguer des autres objets, indépendamment de son état.

Code barre



Représentation des objets

II. Approche objet et système d'information

Caractéristiques structurelles:

Les informations caractérisant l'objet. Se sont les attributs (variables) stockant des données relatives à l'état de l'objet.

MaVoiture:Voiture
Audi 10CV 2L

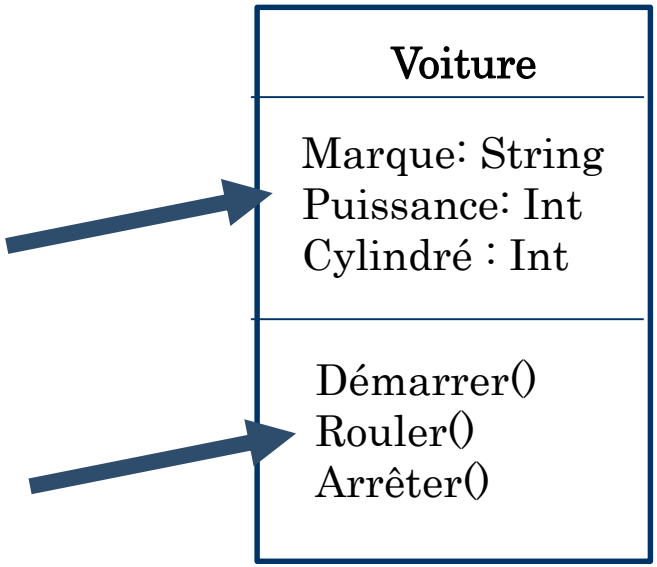
Représentation des objets

II. Approche objet et système d'information

Caractéristiques comportementales:

Elles se traduisent par des méthodes (fonctions) qui spécifient le comportement de l'objet.

Les actions que l'objet est à même de réaliser.



Représentation des objets

II. Approche objet et système d'information

Un objet est responsable de ses comportements.
Si la voiture reçoit un ordre Alors elle peut :

- ✓ **Exécuter** l'ordre.
- ✓ **Refuser** l'ordre(envoi message d'erreur).

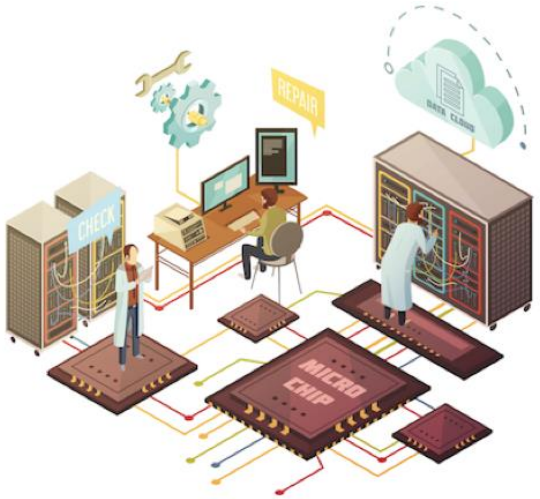
Représentation des objets

II. Approche objet et système d'information

SYSTÉMES D'INFORMATION:

Un système d'information se définit comme une boîte noire caractérisée par:

- Le domaine.
- Les acteurs et interfaces.
- Les services.



II. Approche objet et système d'information

Acteurs et interfaces:

Acteurs:

Les acteurs sont les seuls interlocuteurs reconnus par le système, mais pas nécessairement identifiés.

Le système échange des messages sans connaître l'identité de ses interlocuteurs.

Trois catégories d'acteurs:

- Les personnes.
- Les équipements.
- Les autres systèmes.

II. Approche objet et système d'information

Acteurs et interfaces:

Interfaces:

Un système d'information peut communiquer de trois manières avec les acteurs:

- Analogique.
- Numérique.
- Syntaxique.

II. Approche objet et système d'information

Acteurs et interfaces:

Interfaces:

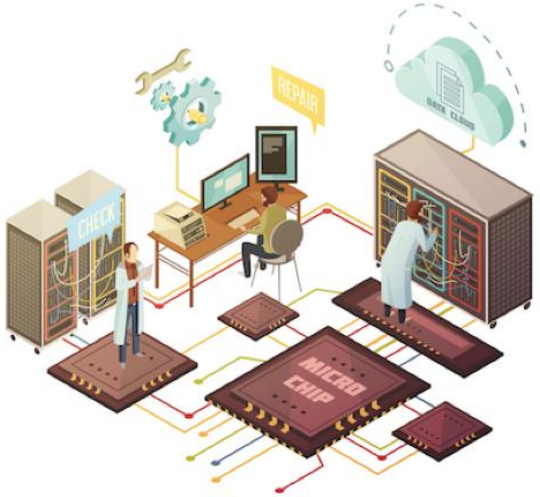
- Analogique: les messages échangés se résument à un signal (chaleur, poids, pression, etc.). C'est le mode de communication utilisé avec les machines/équipements.
- Numérique: les messages échangés sont déjà codés pour être directement exploitables. C'est le mode de communication utilisé avec les autres systèmes informatiques.
- Syntaxique: les messages échangés doivent être interprétés avant d'être exploités. C'est le mode de communication utilisé avec l'homme.

II. Approche objet et système d'information

SYSTÉMES D'INFORMATION:

Un système d'information se définit comme une boîte noire caractérisée par:

- Le domaine.
- Les acteurs et interfaces.
- Les services.



II. Approche objet et système d'information

Services:

Les fonctions du système d'information se définissent en termes de services:

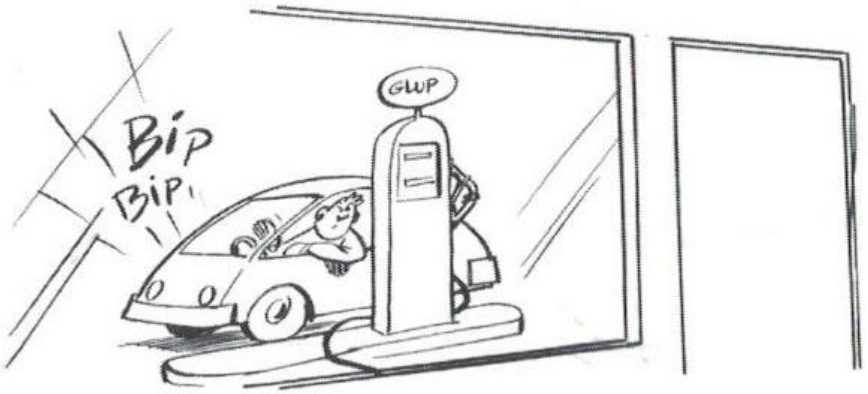
- Gestion des informations nécessaires au déroulement des activités;
- Traitement de ces informations;
- Contrôle du déroulement des activités.

Pour définir ces services, il faut connaître les **événements**, les **situations** et les **actions**

II. Approche objet et système d'information

Les événements:

Les services fournis en réponse à un événement externe au système. Cet événement peut être calendaire ou directement associé à l'intervention d'un acteur.

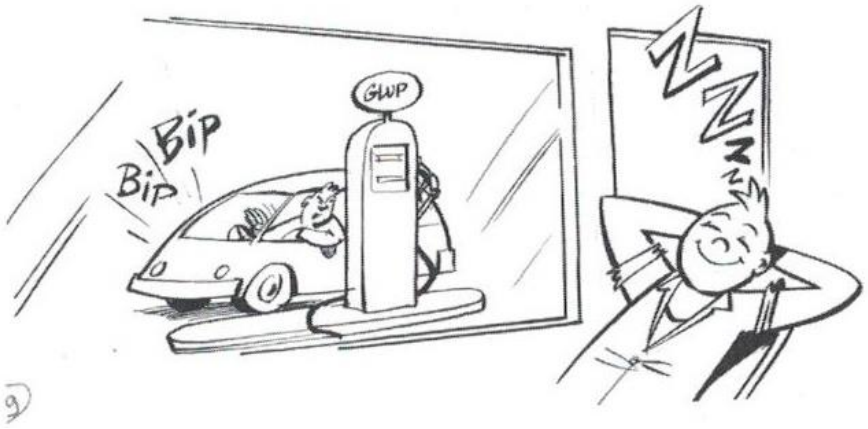


Evènement externe

II. Approche objet et système d'information

Les situations:

Les événements ne déclenchent les actions que sous certaines conditions. Ne sont concernées que les ***situations*** qui traduisent les attentes respectives du système et des acteurs.



Situation

II. Approche objet et système d'information

Les actions:

Un service se définit par les ***actions*** que le système est supposé entreprendre en fonction des situations et des événements. On ne considère que les actions significatives au niveau du domaine.

Actions

Analyse Orientée Objet

I. Introduction

II. Approche objet et système d'information.

III. Les principes et concepts objets.

IV. UML.

Système d'information et approche objet .

II. Système d'information et approche objet

Système d'information et approche objet:

Le développement d'un système d'information comporte deux grandes phases:

1. **La définition des besoins** (objets du domaine, service, conditions d'utilisation).
2. **Spécification du système** (architecture et composants).

II. Système d'information et approche objet

Représentation des objets du domaine:

La modélisation porte sur les **objets** dont **l'identité**, les **caractéristiques** et le **comportement** dépendent uniquement du domaine et peuvent donc être définis indépendamment de tout système d'information.

L'approche objet fournit les concepts et la démarche nécessaires.

II. Système d'information et approche objet

Une question de point de vue:

Pour modéliser il faut ***un point de vue*** c'est-à-dire une position et une préoccupation.

La **position** donne le **champ de vision**, les **perceptions**

La **préoccupation** détermine ce qui est **nécessaire**.

L'objet est une représentation, toute représentation est **motivée**, donc il n'y a pas d'objet sans motivation.

Une question de point de vue

II. Système d'information et approche objet

Une question de distinction:

Toute représentation commence par une ***distinction***: avant toute chose il faut pouvoir distinguer les objets les uns des autres, donc les ***identifier***.

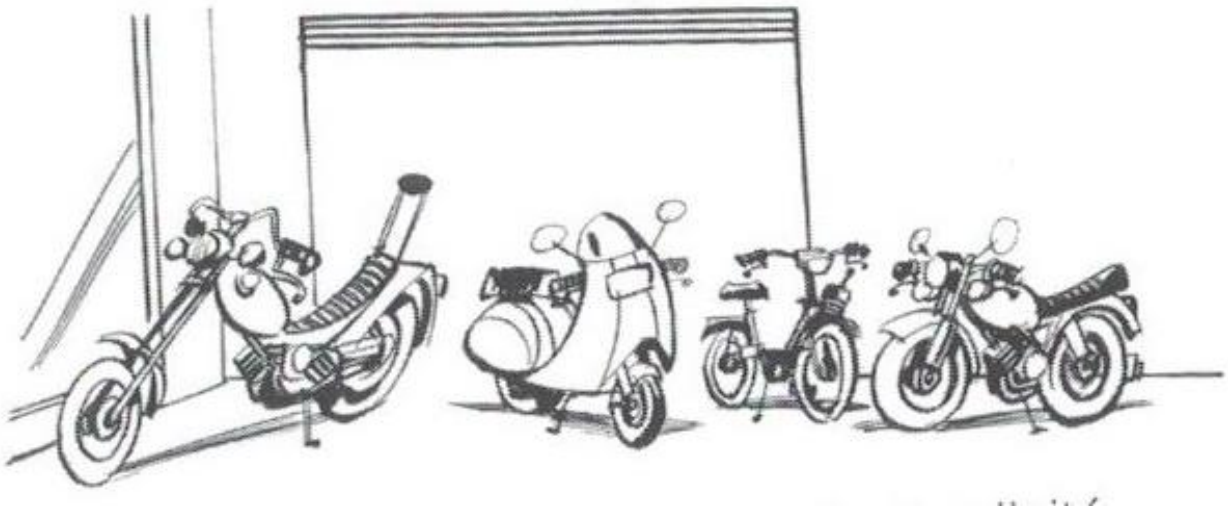
Il faut ensuite ***caractériser*** les objets.

Une question de distinction

II. Système d'information et approche objet

Une question de simplicité:

Une représentation doit réduire la complexité du problème, c'est-à-dire y mettre un peu d'ordre: Trouver des similitudes, des associations, définir des règles, des contraintes.

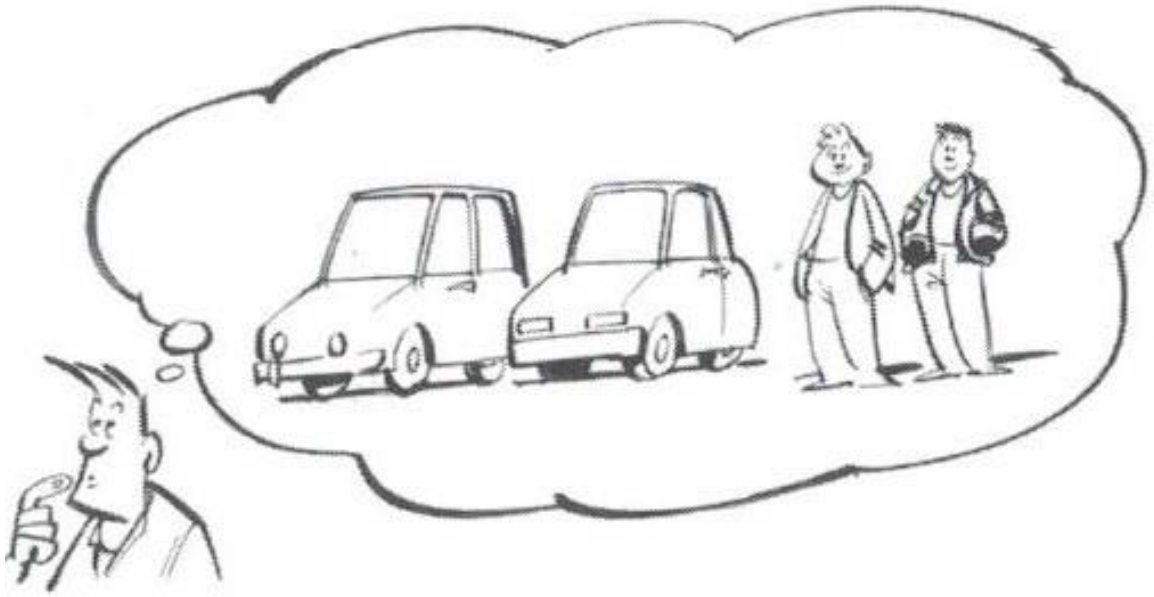


Une question de simplicité

II. Système d'information et approche objet

Une question d'efficacité:

Selon les points de vue, les représentations (modèles) seront différentes, et pour un même point de vue certaines seront plus ou moins exhaustives, cohérentes, efficaces.



Une question d'efficacité

II. Système d'information et approche objet

Trois mots:

exhaustives

efficaces ***cohérentes***

- Un modèle ***exhaustif*** doit prendre en compte de manière complète et compréhensible l'ensemble des éléments.
- Un modèle ***cohérent*** doit garantir la compatibilité de ces éléments.
- Un modèle ***efficace*** doit réduire la complexité, c'est-à-dire le nombre de concepts utilisés pour décrire le domaine.

II. Système d'information et approche objet

Conclusion:

- Les systèmes informatiques modernes sont composés d'objets qui collaborent pour assurer les services requis.
- Plus encore, dans les systèmes distribués, des composants totalement étrangers doivent pouvoir collaborer sans même se connaître.
- La conception du système se ramène donc à la spécification de ses composants, de leurs caractéristiques, de leurs comportements.

II. Système d'information et approche objet

Conclusion:

Le système: les objets du domaine + les classeurs

LA MÉTHODE

II. Système d'information et approche objet

LA MÉTHODE:

La méthode permet d'aller du problème à la solution par une démarche raisonnée.

La démarche: construire le modèle puis construire le système

II. Système d'information et approche objet

LA MÉTHODE:

1990-1995 Emergence des méthodes objet.

Comment structurer un système sans concentrer l'analyse uniquement aux **données** ou aux **traitements** mais aux deux à la fois ?

Analyse Orientée Objet

I. Introduction

II. Approche objet et système d'information.

III. Les principes et concepts objets.

IV. UML.

III. Les principes et concepts objets

1. Objet et classe.
2. Encapsulation.
3. Héritage.
4. Polymorphisme.

III. Les principes et concepts objets

1. Objet et classe.
2. Encapsulation.
3. Héritage.
4. Polymorphisme.

III. Les principes et concepts objets

1. Objet et classe:

Sens commun: Identifier un objet revient à le distinguer des autres.

III. Les principes et concepts objets

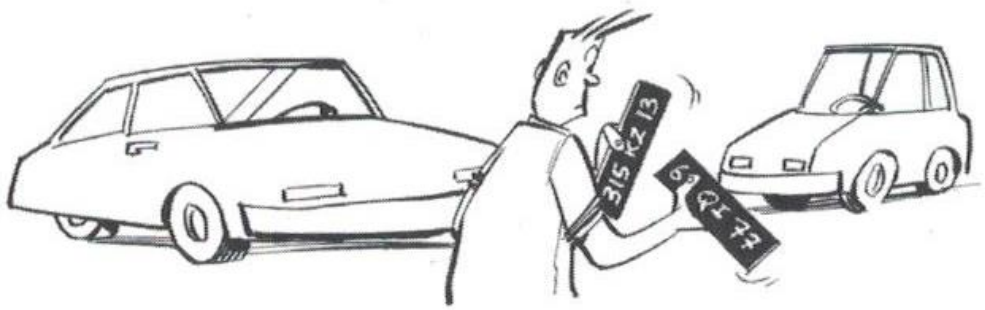
1. Objet et classe:

Informaticien: Identifier un objet c'est choisir, parmi ses caractéristiques, celles qui permettront de le retrouver.

III. Les principes et concepts objets

1. Objet et classe:

Chaque objet possède une identité, qu'il reçoit à sa création et qu'il conserve jusqu'à sa disparition.
L'identité ne dépend pas des caractéristiques de l'objet. Les caractéristiques peuvent toutes être modifiées sans que l'identité de l'objet ne soit affectée.



Le changement de plaque n'affecte pas l'identité du véhicule

III. Les principes et concepts objets

1. Objet et classe:

L'objet est défini à la fois par des informations et des comportements.

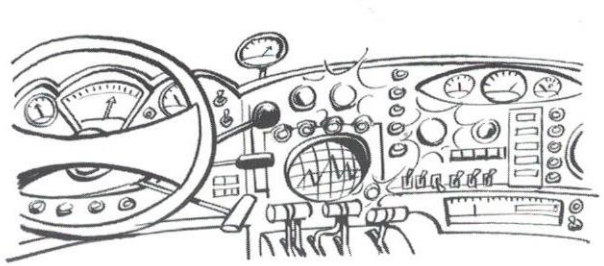
- ✓ Les **informations** sont les données (attributs, données…),
- ✓ Les **comportements** sont les traitements (méthodes, opérations…).

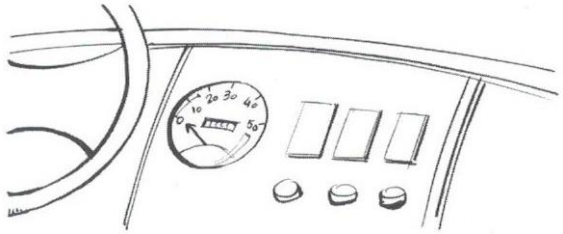
III. Les principes et concepts objets

1. Objet et classe:

Définition:

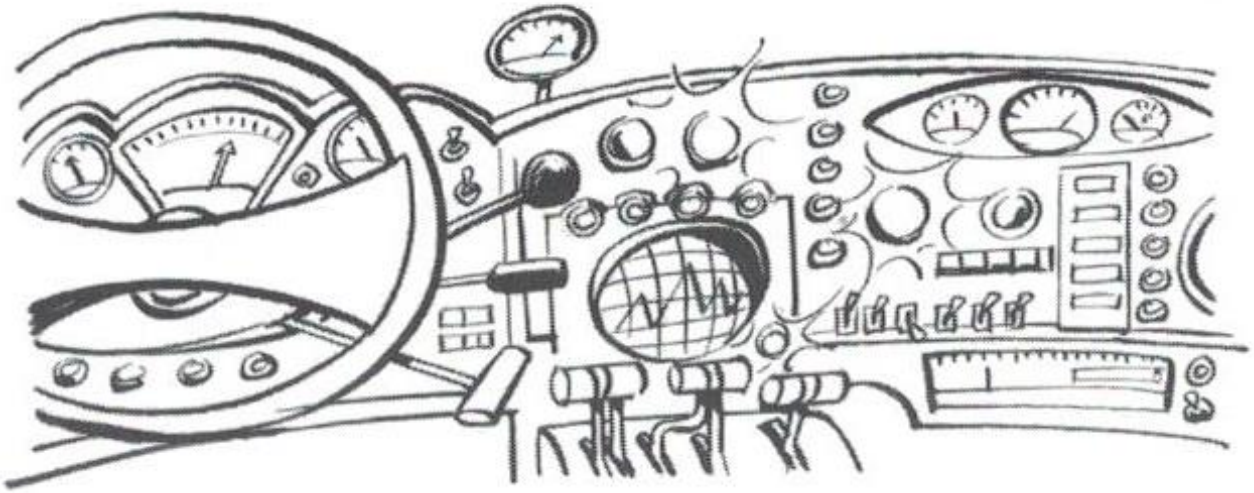
Un objet est une entité du monde réel qui se caractérise par un ensemble de **propriétés**, des **états significatifs** et un **comportement**.





III. Les principes et concepts objets

1. Objet et classe:

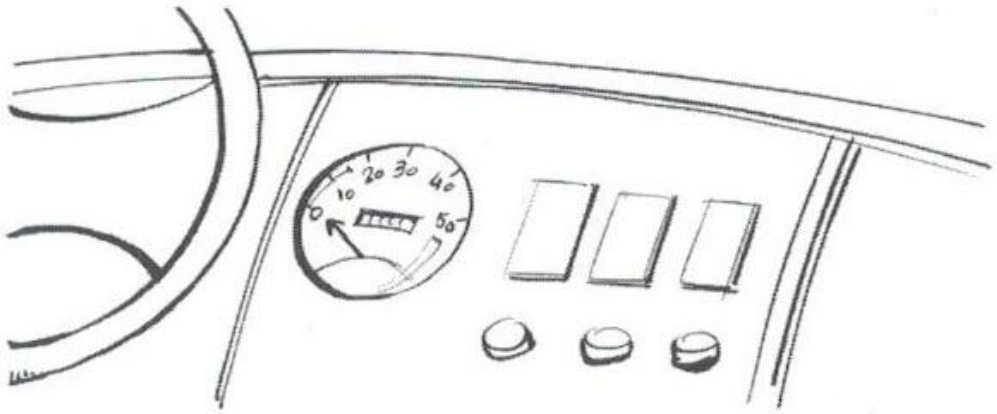


Les propriétés de l'objet VOITURE

III. Les principes et concepts objets

1. Objet et classe:

Il faut demander à l'objet dans quel état il se trouve et quels sont ses liens avec les autres objets.
L'état de l'objet correspond aux valeurs de tous les attributs à un instant donné.

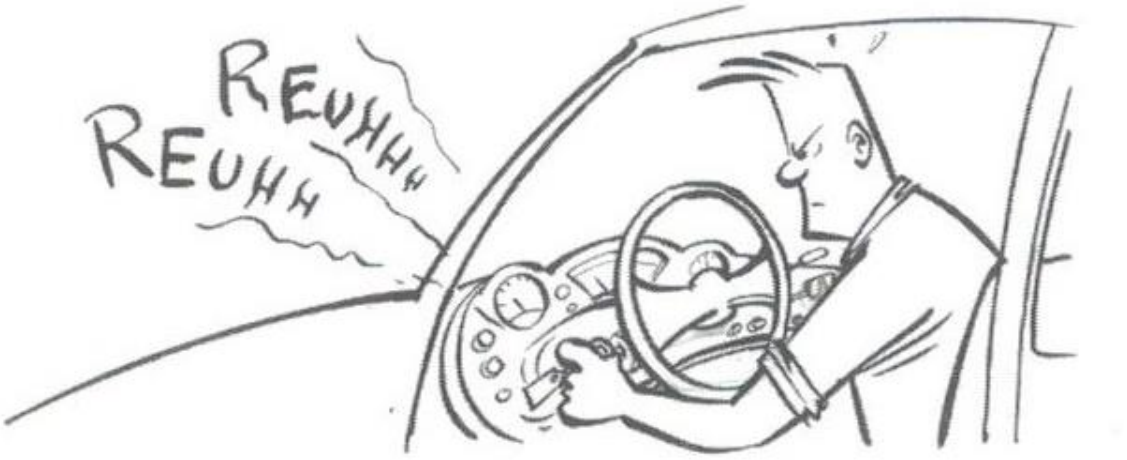


L'état de l'objet VOITURE

III. Les principes et concepts objets

1. Objet et classe:

Chaque objet a des responsabilités bien définies.
Le comportement d'un objet est défini par l'ensemble des opérations qu'il peut exécuter en réaction à des messages envoyés par d'autres objets.

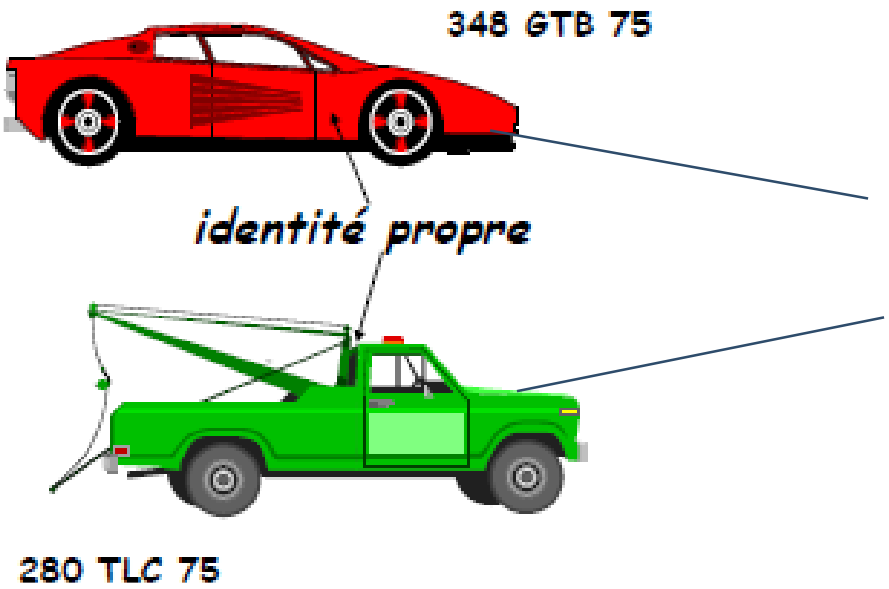


Le comportement de l'objet VOITURE

III. Les principes et concepts objets

1. Objet et classe:

Une classe est l'abstraction d'un ensemble d'objets qui possèdent une structure identique (attributs) et un même comportement (opérations).



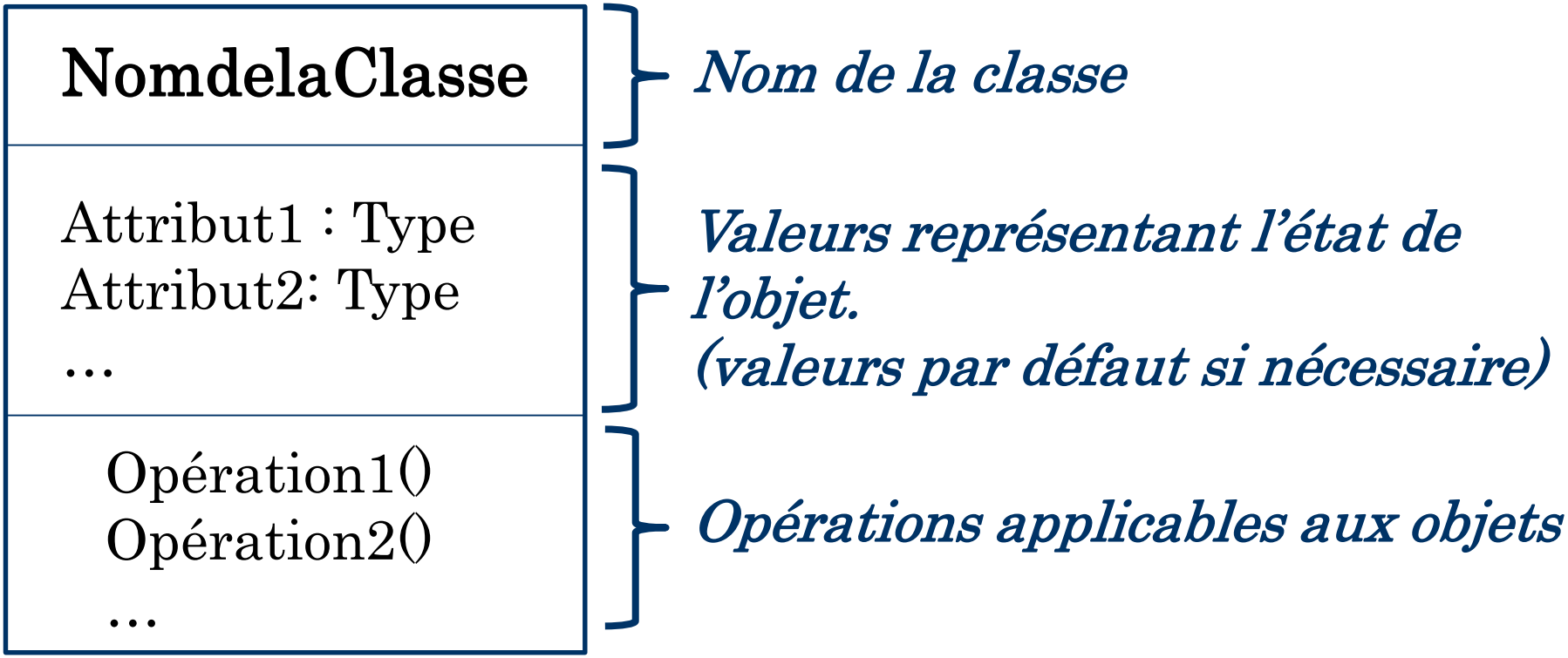
Objet Voiture

NomdelaClasse
Attribut1 : Type Attribut2: Type …
Opération1() Opération2() …

Classe Voiture

III. Les principes et concepts objets

Représentation d'une classe:



III. Les principes et concepts objets

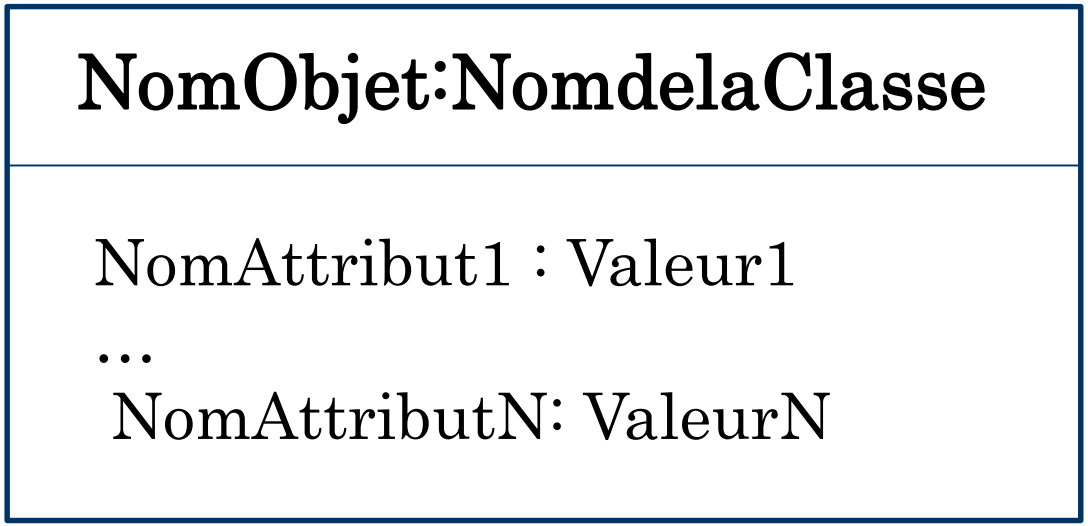
1. Objet et classe:

Exemple de classe

Voiture
Marque: String Puissance: Int Cylindré : Int
Démarrer() Rouler() Arrêter()

III. Les principes et concepts objets

Représentation d'un objet:



III. Les principes et concepts objets

Exemple d'objet:

MaVoiture:Voiture
Audi 10CV 2L

Un objet est responsable de ses comportements.
Si la voiture reçoit un ordre Alors elle peut :

- ✓ **Exécuter** l'ordre.
- ✓ **Refuser** l'ordre(envoi message d'erreur).

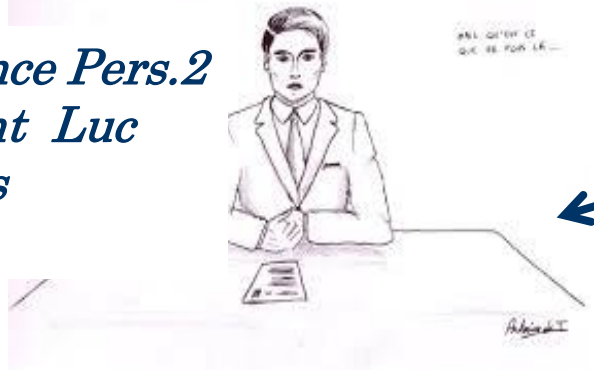
III. Les principes et concepts objets

Autre exemple:

Tous les objets de la classe sont des **instances**.

Instance Pers.1
Dupont Alain
40 ans
HEPL

Instance Pers.2
Durant Luc
42 ans
IBM



Instanciation

Instanciation

Classe Personne

Attributs
Nom
Age
Société

Méthodes
NomEnMajuscule()
ChangerAge()
ChangerSociété()

III. Les principes et concepts objets

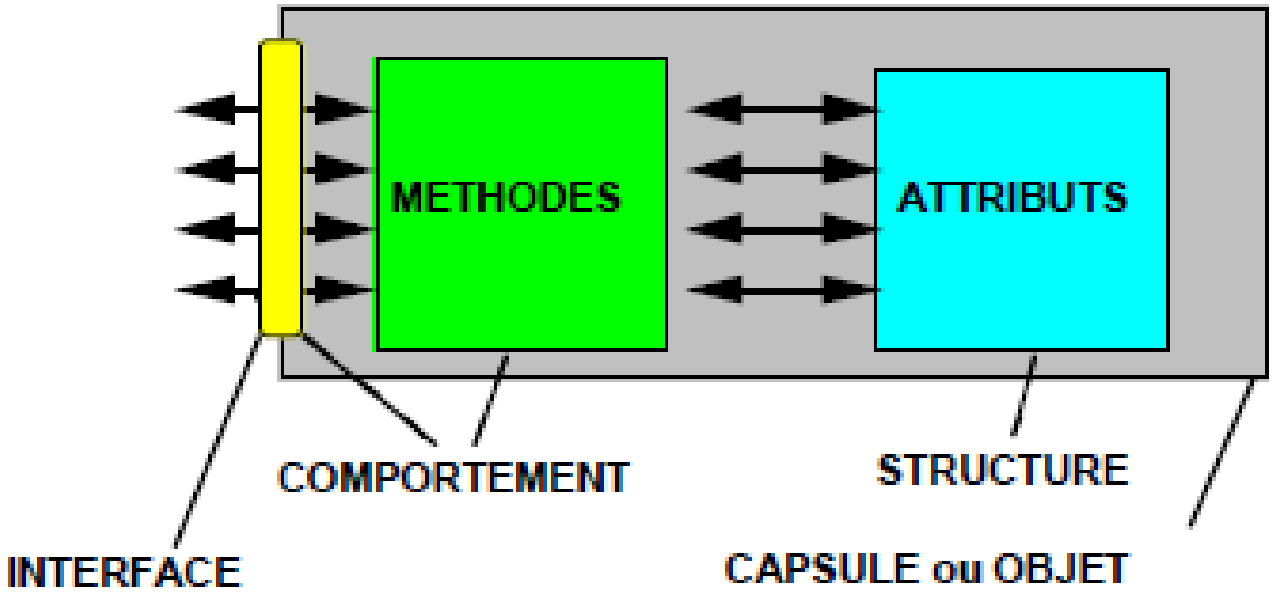
1. Objet et classe.
2. **Encapsulation.**
3. Héritage.
4. Polymorphisme.

III. Les principes et concepts objets

2. Encapsulation:

Définition:

L'encapsulation est le **regroupement** dans une même classe de la description des **attributs** et des **opérations**.



III. Les principes et concepts objets

2. Encapsulation:

- Les **objets d'une même classe** portent tous les attributs de la classe.
- Les **instances d'une même classe** partagent les comportements définis dans la classe.

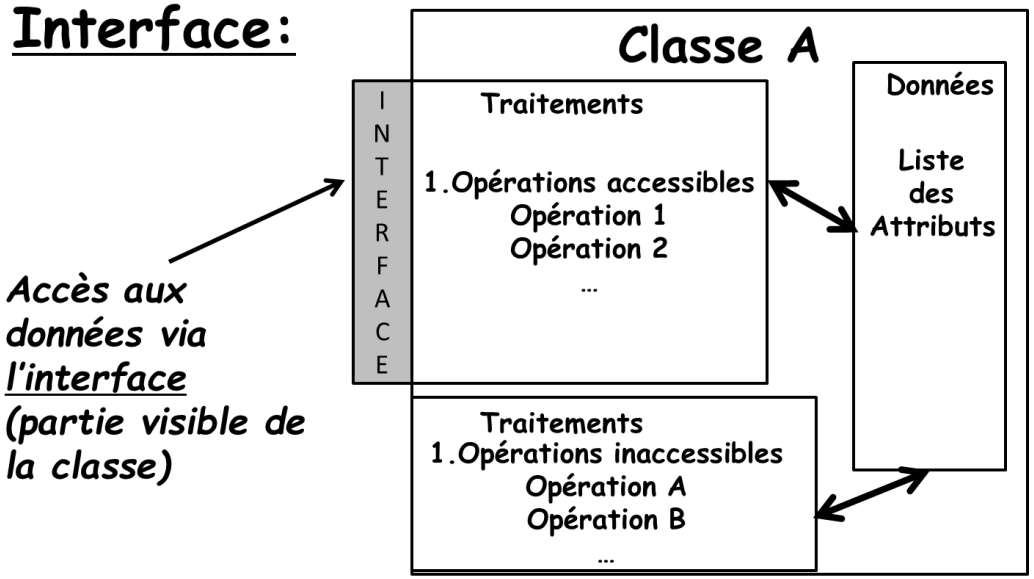
III. Les principes et concepts objets

2. Encapsulation:

- Permet d'interdire ou de contrôler l'accès direct aux attributs ou méthodes des objets.

- Consiste à masquer les détails d'implémentation d'un objet en définissant une interface.

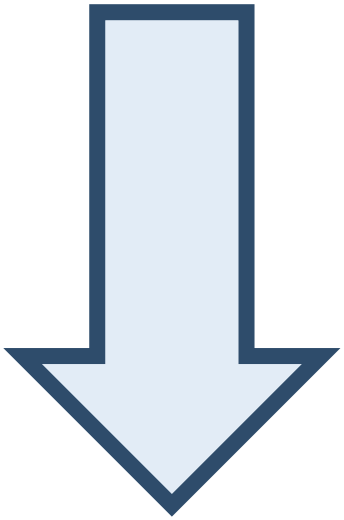
Interface:



III. Les principes et concepts objets

2. Encapsulation:

- Permet d'interdire ou de contrôler l'accès direct aux attributs ou méthodes des objets.



Visibilité

III. Les principes et concepts objets

2. Encapsulation:

Visibilité

La **visibilité** permet de définir de quelle manière un **attribut** ou une **méthode** d'un objet sera **accessible** dans le programme.

Une classe doit rendre **visible** ce qui est nécessaire pour manipuler ses instances et rien d'autre.

L'implémentation d'une méthode doit utiliser ce qui est nécessaire aux traitements et rien d'autre.

III. Les principes et concepts objets

2. Encapsulation: Intérêt

A. *Meilleur sécurité et lisibilité:*

Permet de protéger le contenu des classes d'une manipulation maladroite et/ou mal intentionnée.

B. *Meilleur modularité (interface, privée):*

Permet de modifier la structure des données internes sans modifier l'interface de celle-ci et donc sans pénaliser les utilisateurs.

III. Les principes et concepts objets

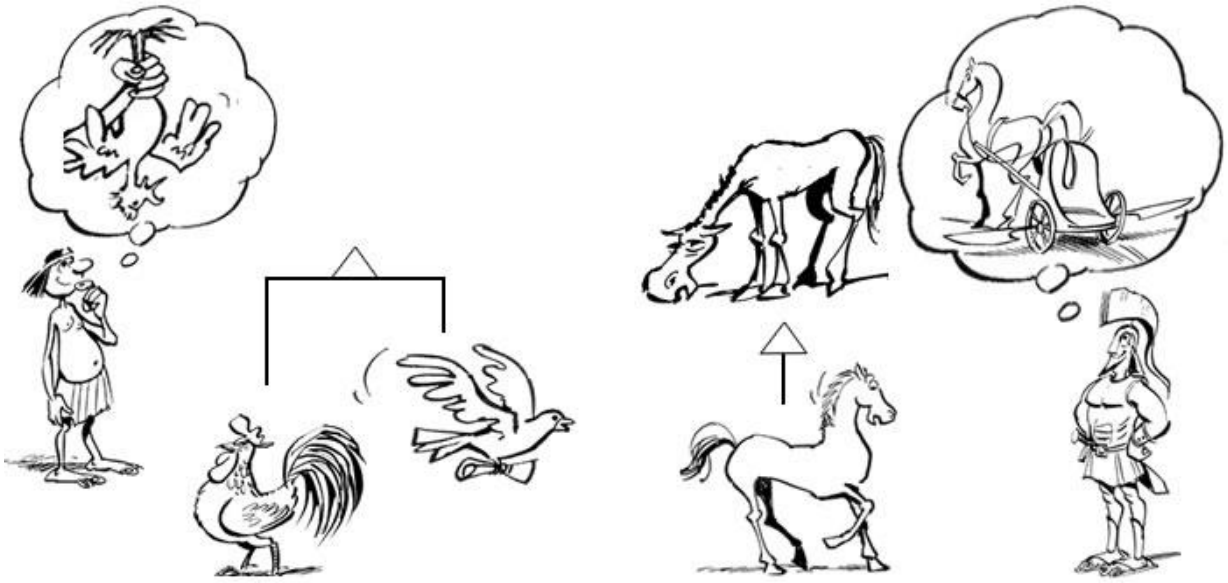
1. Objet et classe.
2. Encapsulation.
3. **Héritage.**
4. Polymorphisme.

III. Les principes et concepts objets

3. Héritage:

L'héritage permet de partager entre les classes des attributs et des méthodes.

=> Permet de spécialiser des classes en réutilisant ***les attributs*** *et* ***comportements*** *d'une autre classe.*

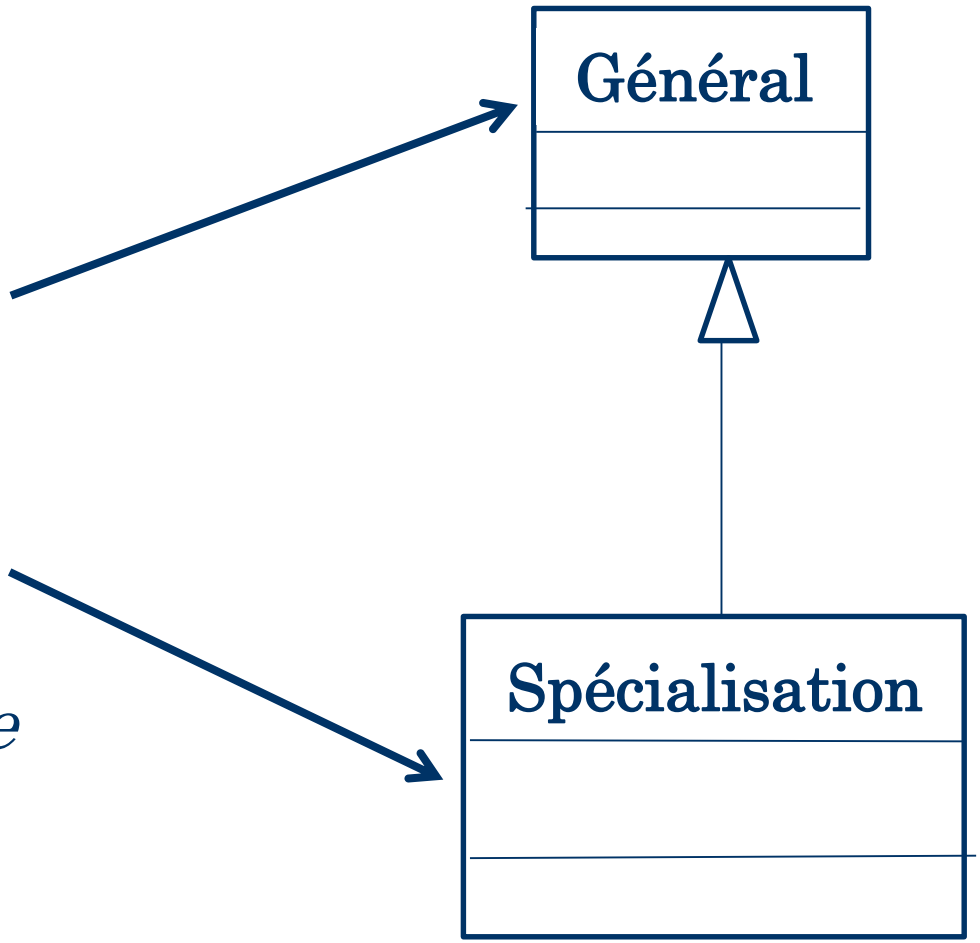


III. Les principes et concepts objets

3. **Héritage**:

Classe Mère
Super-Classe

Sous-Classe
Classe Fille
Classe Dérivée

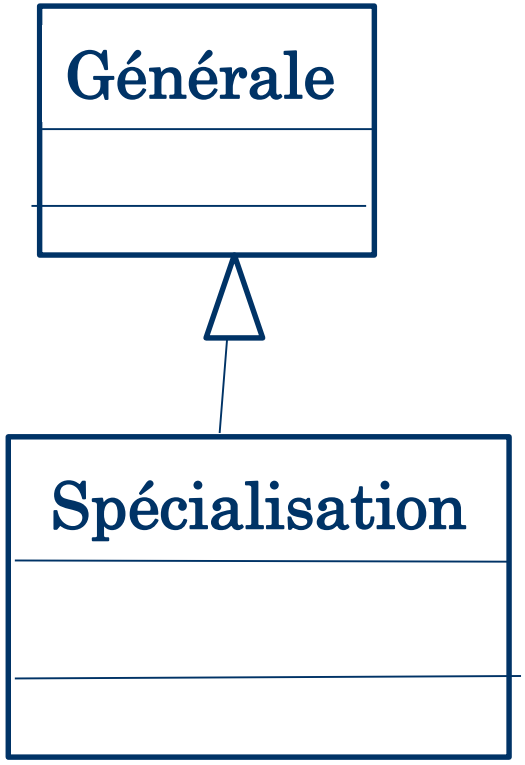


III. Les principes et concepts objets

3. Héritage:

Intérêt:

L'héritage permet de **découper**, **factoriser** et **réutiliser** du code.

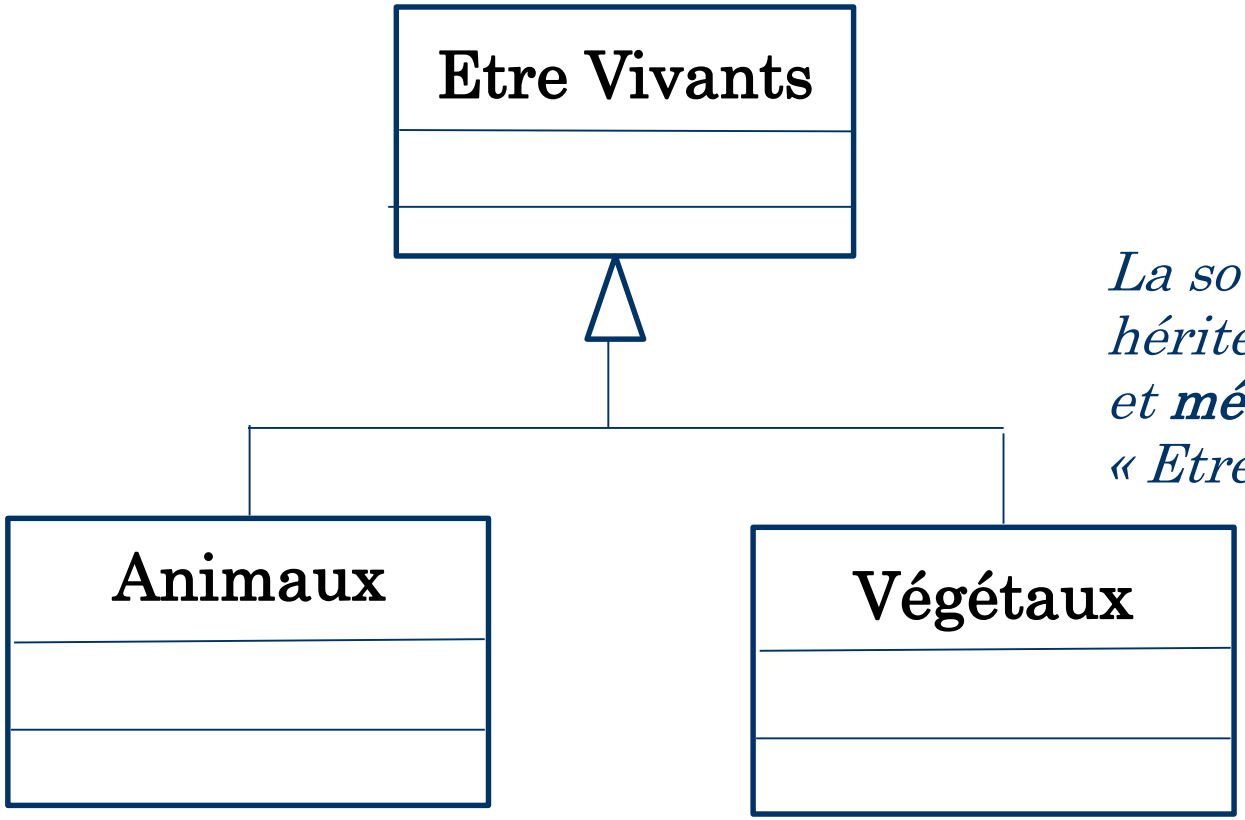


*La classe « spécialisation » hérite des **méthodes** et **attributs** de la classe « générale».*

*Dans la classe « **spécialisation** » on peut ajouter des **attributs** et des **méthodes** en plus de ceux de la classe « générale ».*

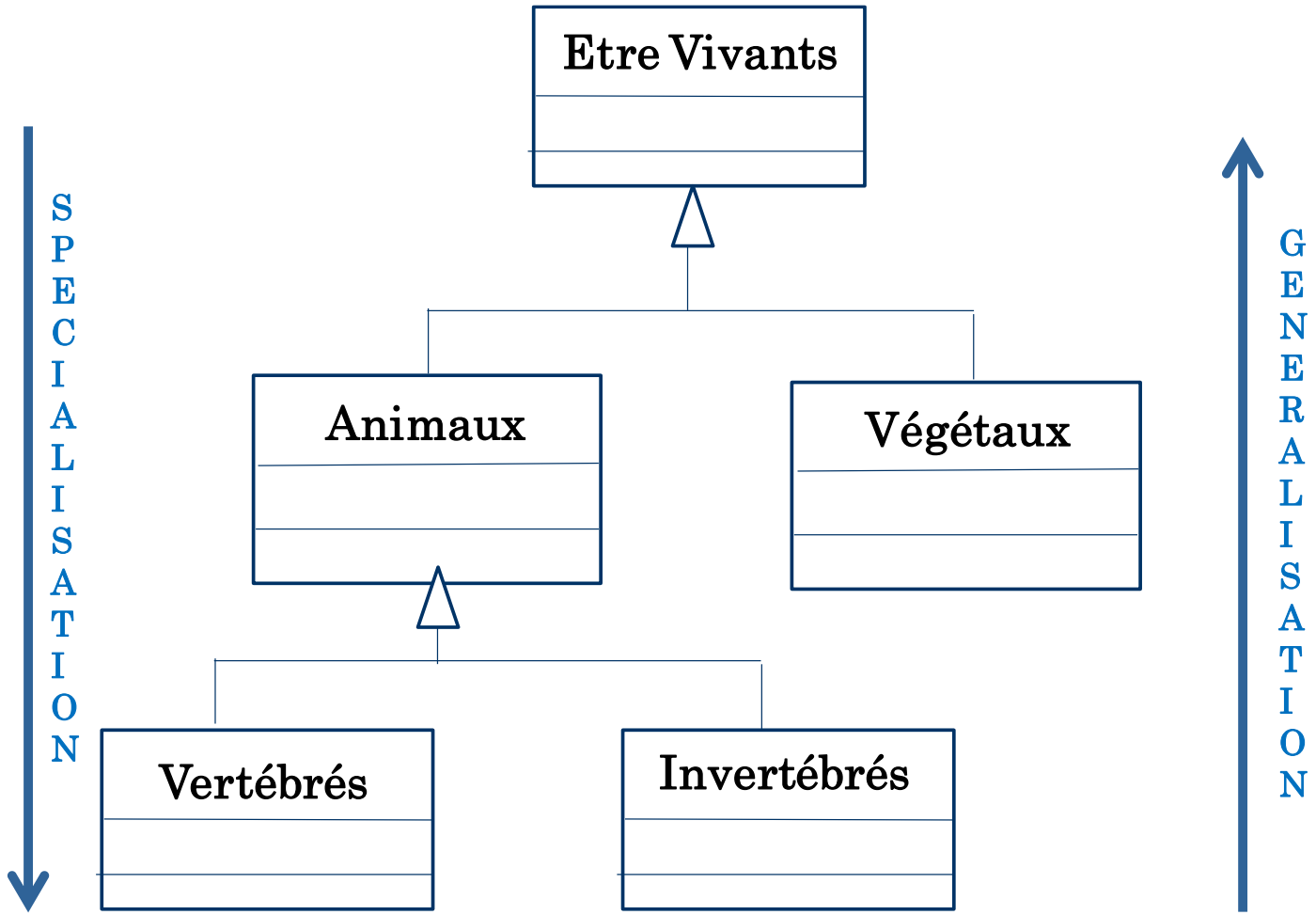
III. Les principes et concepts objets

3. Héritage:



*La sous-classe « Végétaux » hérite de tous les **attributs** et **méthodes** de la classe « Etre vivants ».*

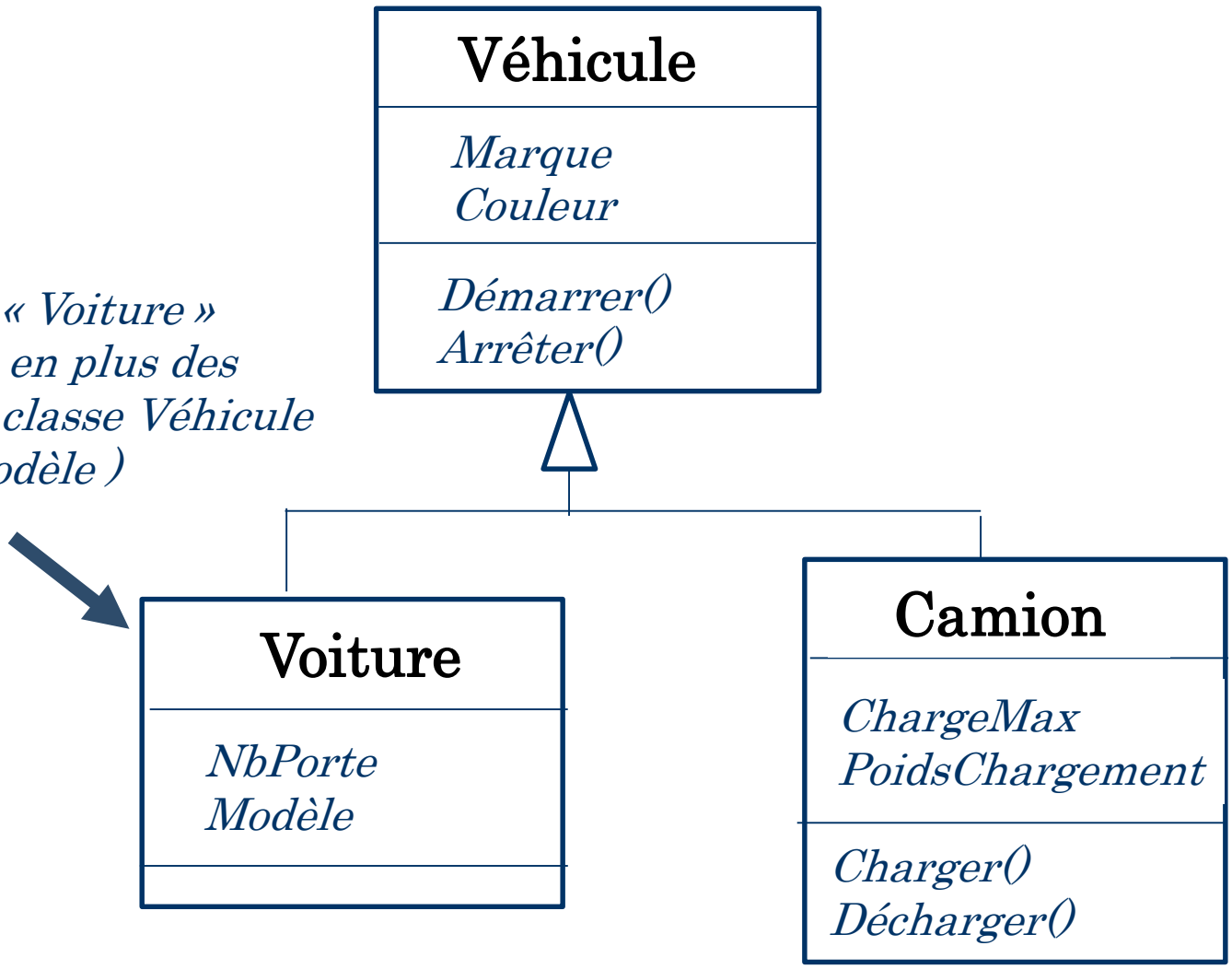
III. Les principes et concepts objets



III. Les principes et concepts objets

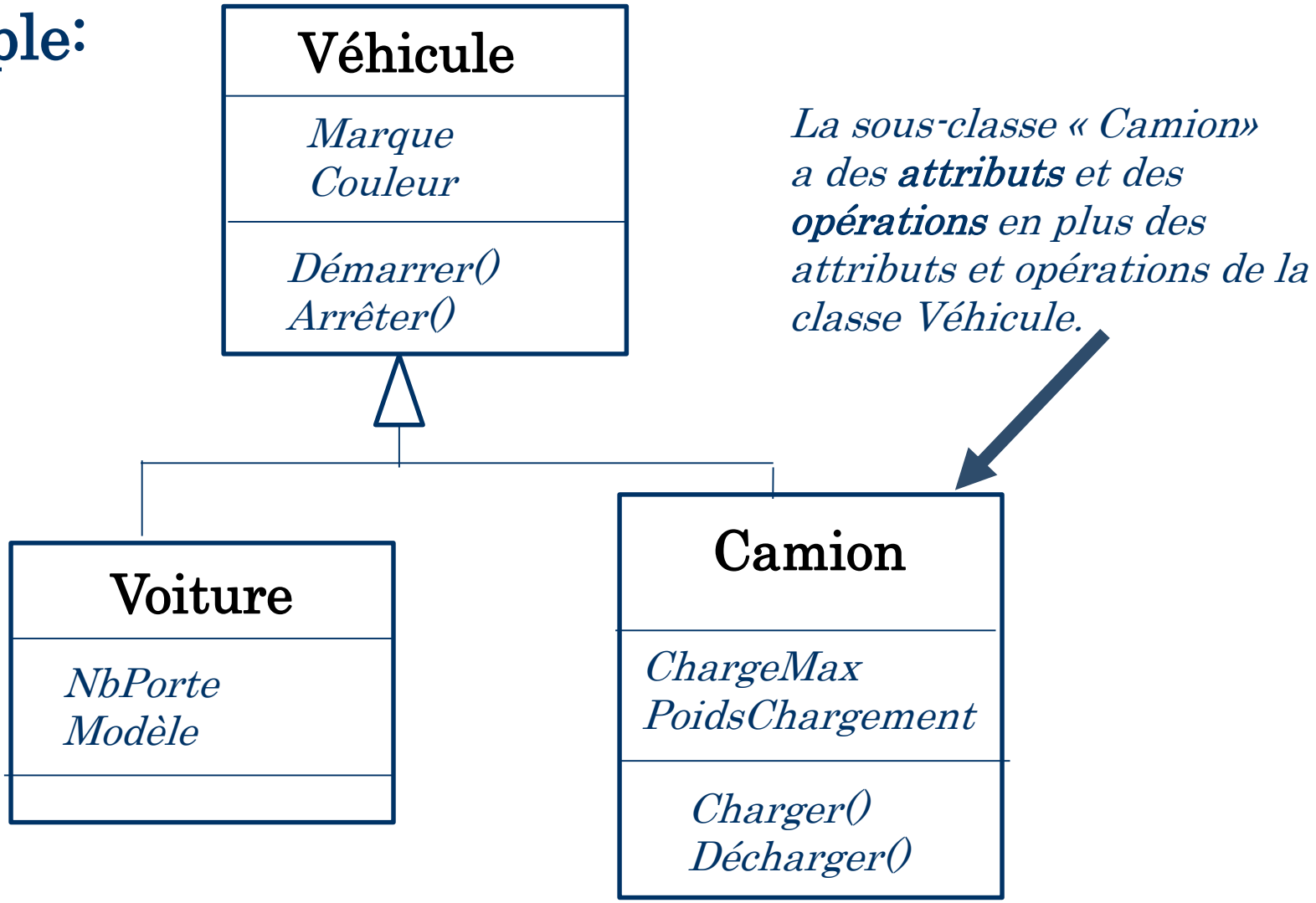
Exemple:

*La sous-classe « Voiture » a des **attributs** en plus des attributs de la classe Véhicule (NbPorte et Modèle)*



III. Les principes et concepts objets

Exemple:



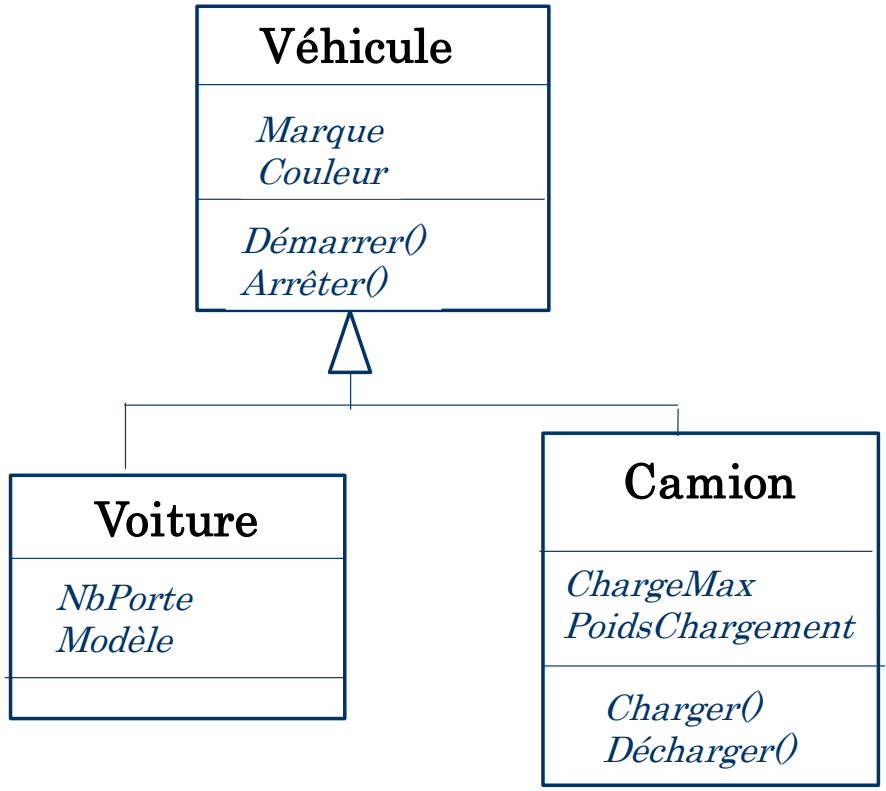
*La sous-classe « Camion» a des **attributs** et des **opérations** en plus des attributs et opérations de la classe Véhicule.*

Héritage simple

Les classes « Camion » et « Voiture » dérivent et peuvent utiliser les attributs et les méthodes de la classe « Véhicule ».

On parle d'**Héritage simple**.

« La classe ne peut hériter que d'une seule superclasse ».



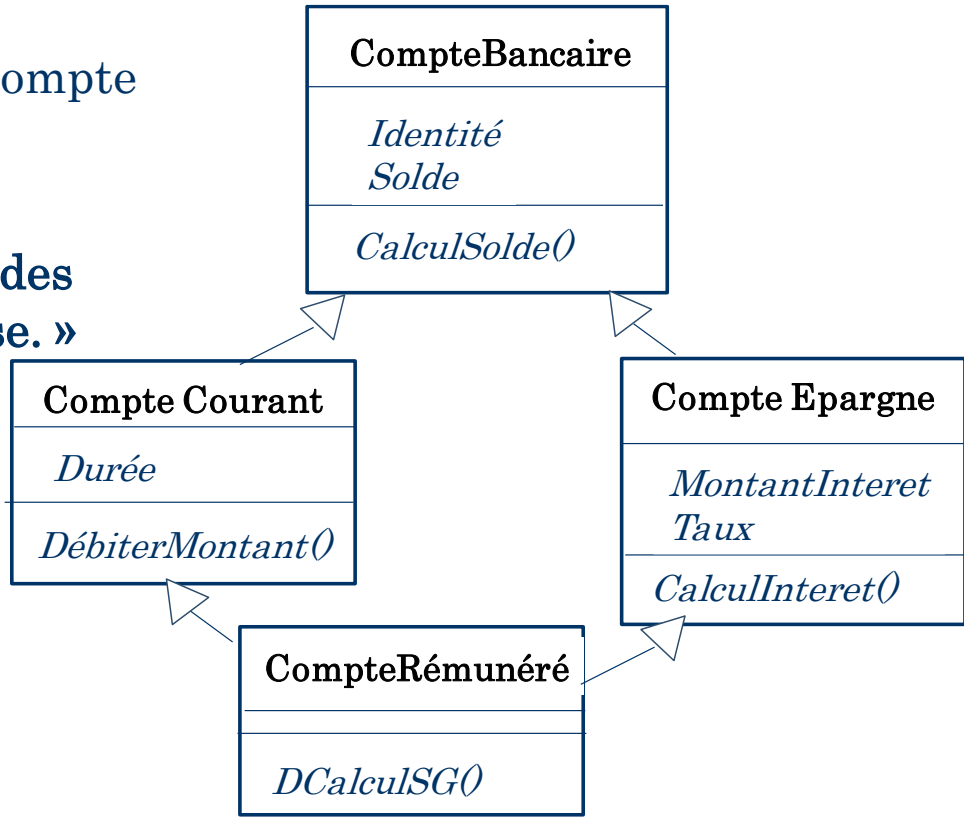
Héritage multiple

La classe « Compte Rémunéré » va hériter deux fois:

- de la classe « Compte Bancaire »
- des classes « Compte Courant » et « compte Epargne».

On parle d'**Héritage multiple.**

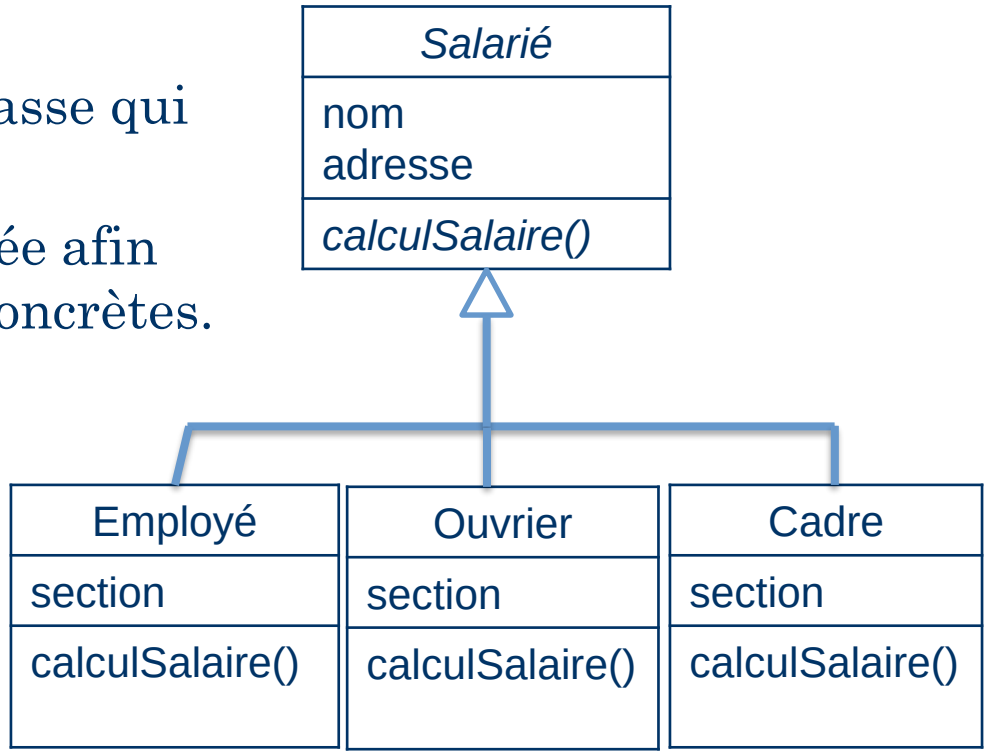
« La classe peut hériter des attributs et des comportements de plus d'une superclasse. »



III. Les principes et concepts objets

Classe Abstraite

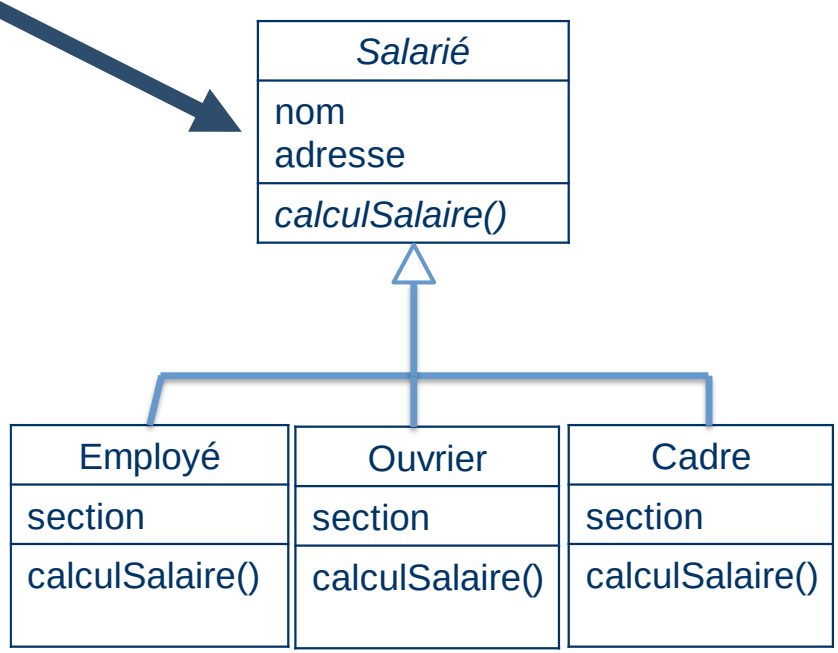
- Une classe abstraite est une classe qui n'est pas **instanciable**.
- Une classe abstraite est déclarée afin d'être dérivée par des classes concrètes.



Classe Abstraite

Héritage **Total** et **Disjoint**
Le salarié est obligatoirement Cadre, Employé ou Ouvrier.

Salarié en italique.
OU
Salarié {Abstract}



III. Les principes et concepts objets

Intérêt de l'héritage

- L'héritage permet la **réutilisation** du code.
  Lorsqu'on va instancier une classe spécialisée, le code des attributs et méthodes de la classe ne seront pas implémentés à nouveau.
- L'héritage permet une **organisation hiérarchique** des classes.
  La maintenance d'une bibliothèque de classe sera plus aisé pour une équipe de développement.

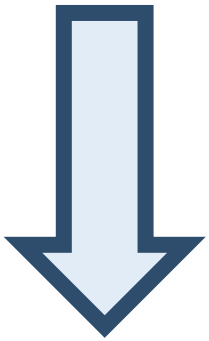
Inconvénient

- L'héritage multiple n'est pas implémenté par tous les langages de programmation O.O (Ex: C++ mais pas Java, C#, PHP).

III. Les principes et concepts objets

Comment résoudre le problème de l'héritage multiple?

- L'héritage multiple n'est pas implémenté par tous les langages de programmation O.O (Ex: C++ mais pas Java, C#, PHP).



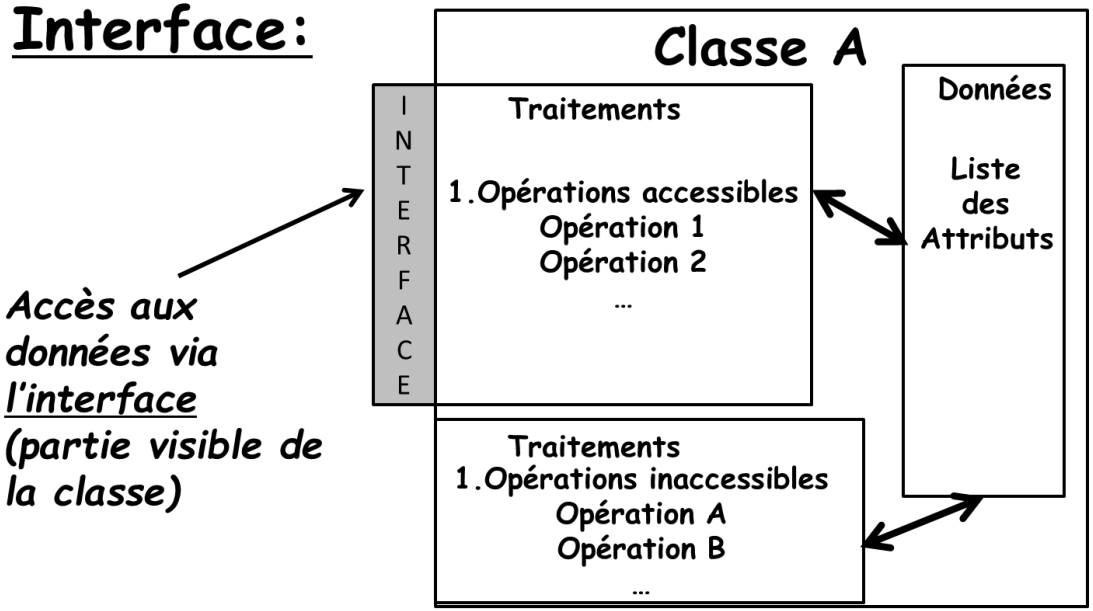
Interface

III. Les principes et concepts objets

Comment résoudre le problème de l'héritage multiple?

Une **Interface** est comme une classe abstraite dans laquelle aucune méthode ne serait **implémentée**.

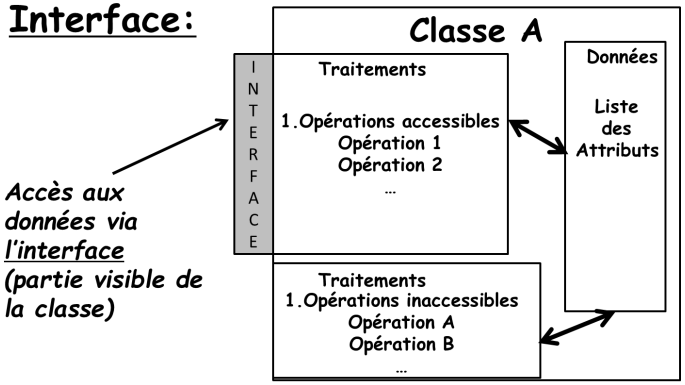
Les **méthodes** sont seulement **déclarées**.



III. Les principes et concepts objets

Les classes abstraites et les interfaces ont chacune une fonction bien distincte:

Salarié{Abstract}
nom adresse
calculSalaire()



Les classes abstraites servent à factoriser du code.

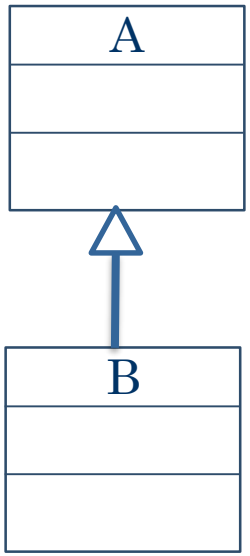
Les interfaces servent à définir des contrats de services c'est-à-dire à réutiliser les objets.

Classe abstraite ou Interface?

Dans la plupart des langages actuels (C#, Java, PHP), il est possible pour une classe d'hériter que d'une seule classe parente (abstraite ou non), mais d'implémenter plusieurs interfaces.

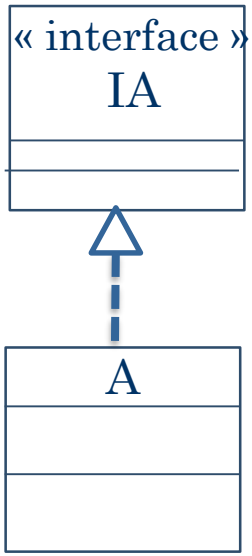
III. Les principes et concepts objets

Héritage simple




```
public class A {
.....
}
public class B extends A {
......
}
```

Réalisation d'une interface par une classe




```
public interface IA {
.....
}
public class B implements IA {
......
}
```

III. Les principes et concepts objets

1. Objet et classe.
2. Encapsulation.
3. Héritage.
4. Polymorphisme.

III. Les principes et concepts objets

4. Polymorphisme:

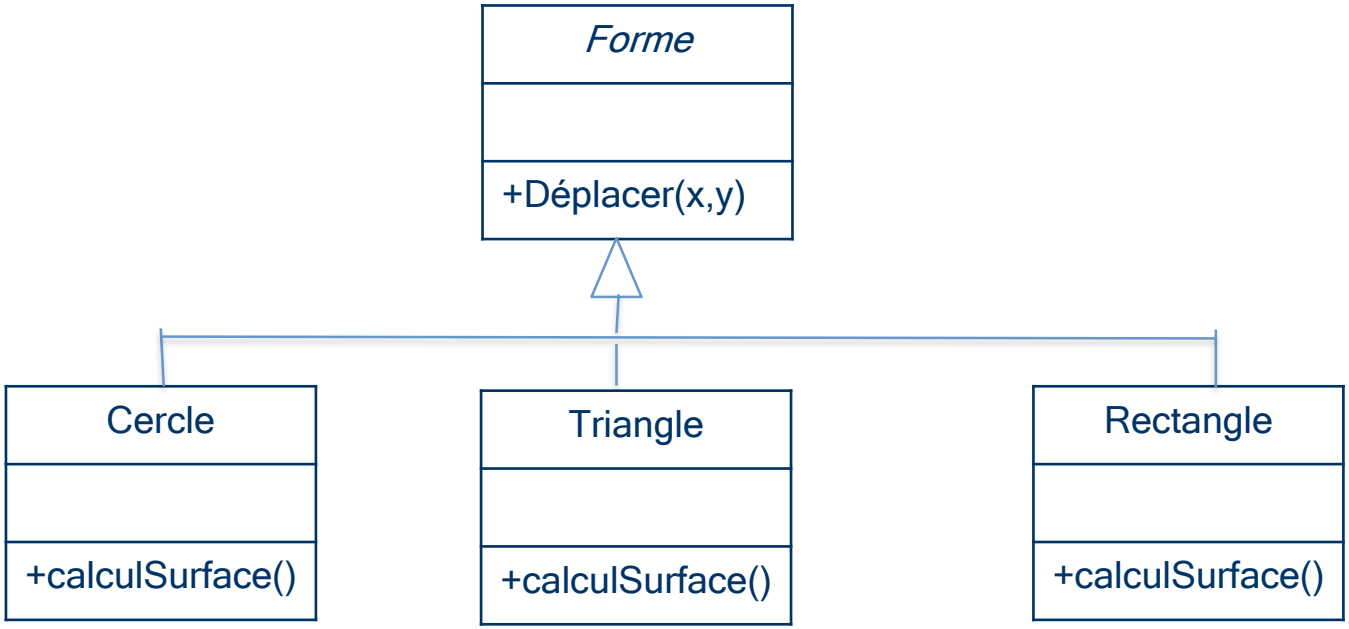
Le nom **polymorphisme** vient du grec et signifie « qui peut prendre plusieurs formes ».

Le **polymorphisme** est la possibilité pour un même **message** de déclencher des traitements différents, suivant les objets auxquels il est adressé.

Le mécanisme de polymorphisme permet donc de donner le même nom à des traitements différents.

III. Les principes et concepts objets

4. Polymorphisme:



La méthode « CalculSurface » est polymorphe.

III. Les principes et concepts objets

Conclusion:

L'approche objet permet:

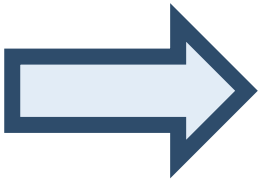
- **D'obtenir un code réutilisable**:
  Les types d'objets créés peuvent servir de base pour d'autres objets en ne développant que les évolutions nécessaires.
- **D'offrir un code plus compréhensible**:
  Chaque objet va contenir ses propriétés et ses méthodes.
- **D'intégrer la modularité**:
  Chaque objet ne communique avec les autres que par des interfaces connues et définies.

III. Les principes et concepts objets

Conclusion:

MAIS, la programmation orientée objet est **moins intuitive** que la programmation fonctionnelle.

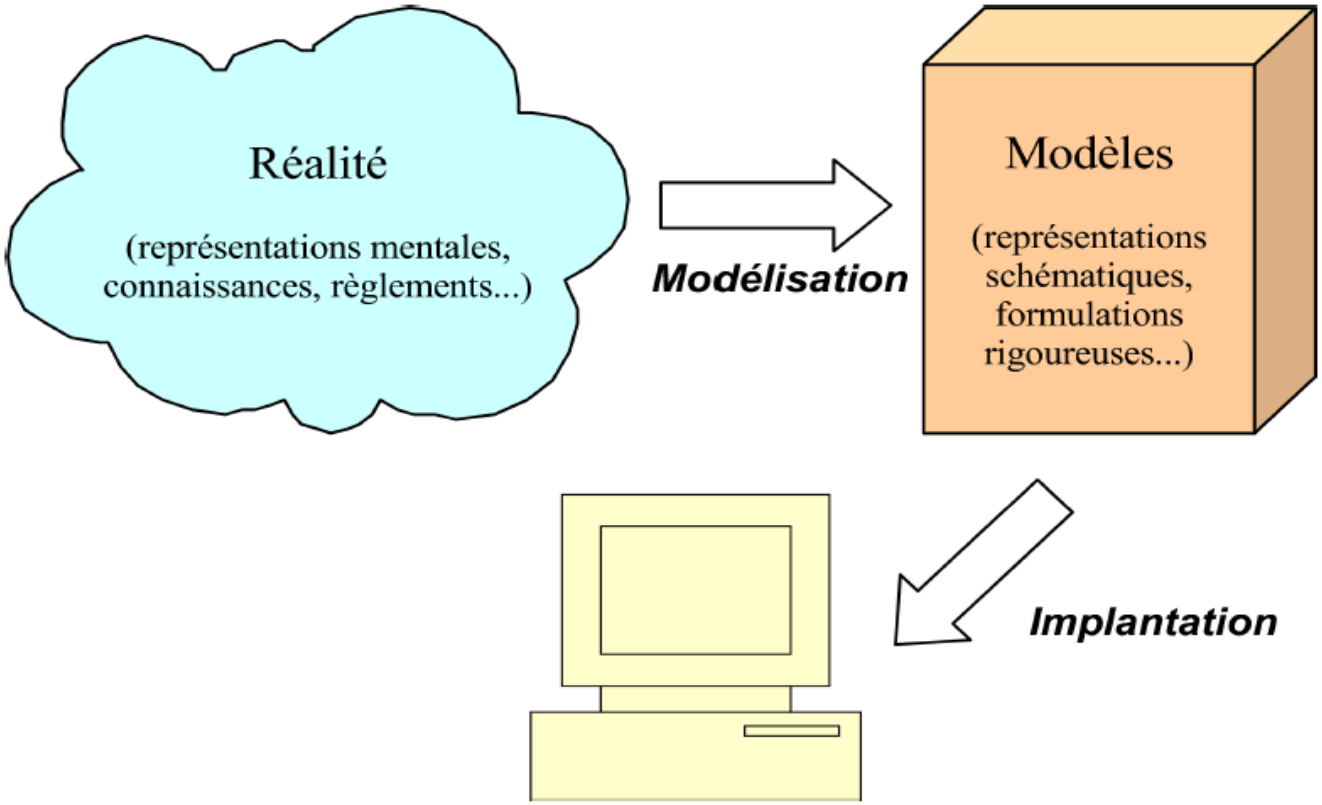
Il est plus naturel de **décomposer** les problèmes informatiques en terme de **fonctions** qu'en terme d'ensembles d'**objets en interaction**.



L'approche objet requiert de modéliser avant de concevoir

III. Les principes et concepts objets

Modéliser avant de concevoir



Pourquoi modéliser?

Pour une simplification de la réalité:

- **Simplifier la problématique posée par le client.**
- **Visualiser le système comme il est ou comme il devrait l'être.**
- **Valider le modèle vis-à-vis des clients**

Pourquoi modéliser?

Avantage:

- Minimiser au maximum les erreurs lors de la phase de programmation.
- Assurer la cohérence et éviter la redondance des données.
- Assurer la conformité du logiciel aux exigences du client.

III. Les principes et concepts objets

Modéliser avant de concevoir

Apparition des méthodes objet.

Au milieu des années 90, plusieurs dizaines de méthodes objet sont disponibles, mais aucune ne prédomine.

III. Les principes et concepts objets

De 1990 à 1995 émergence des méthodes objet.

Les méthodes prédominantes de la modélisation OO

- **OMT** de James Rumbaugh (General Electric) qui fournissait une représentation graphique des aspects statique, dynamique et fonctionnel du système;
- **OOD** de Grady Booch (department of Defense) qui introduisait le concept de paquetage (package);
- **OOSE** d'Ivar Jacobson (Ericsson) qui fondait son analyse sur la description des besoins des utilisateurs (cas d'utilisation ou use cases)

III. Les principes et concepts objets



Les trois gourous « ***the Amigos*** » se mirent d'accord pour définir une méthode commune qui fédérait leurs apports respectifs.

Naissance d'UML (**U**nified **M**odeling **L**anguage)
Ou **Langage de Modélisation Unifié.**

UML est bien un langage et non une méthode

Bibliographie

1. Benoît CHARROUX, Aomar OSMANI, Yann THIERRY-MIEG. UML2 Pratique de la modélisation. 3ème édition. PEARSON.
2. Laurent AUDIBERT. UML2 de l'apprentissage à la pratique. 2ème édition. ELLIPSES.
3. Christian SOUTOU. Modélisation des bases de données (UML et les modèles entité-association). 3ème édition. EYROLLES.
4. Chantal MORLEY, Jean HUGUES, Bernard LEBLANC. 4ème édition. UML 2 pour l'analyse d'un système d'information. DUNOD.
5. Hugues BERSINI. L'orienté objet. 3ème édition. EYROLLES.
6. Laurent DEBRAUWER, Fien VAN DER HEYDE. UML 2.5. 4ème édition. ENI Editions.
7. Jean-Luc HAINAUT. Bases de données concepts, utilisation et développement. DUNOD.
8. Gilles ROY. Conception de bases de données avec UML. Presses de l'université du Québec.
9. Craig LARMAN. UML2 et les design patterns. 3ème édition. PEARSON Education.
10. Frank BARBIER. UML 2 et MDE. DUNOD.
11. Laurent DEBRAUWER, Naouel KARAM. UML 2 entraînez-vous à la modélisation. Seconde édition. ENI Editions.
12. Corine COSTA. Cours Projets et bureau d'études.