Programmation Orientée Objet

Rappel

Pourquoi un nouveau paradigme de programmation ?

1. Communauté GL

- Augmentation de la demande de nouveaux logiciels (applications)
- Augmentation de la complexité de ces logiciels.
- Maitrise de la complexité limitée.

Cette communauté doit résoudre les problèmes de :

- Développement
- maintenance et
- de la limite de la réutilisation

Pourquoi un nouveau paradigme de programmation ?

2, Communauté des programmeurs

- **Besoin d'encapsulation**

L'encapsulation est un mécanisme consistant à rassembler les données et les méthodes au sein d'une structure en cachant l'implémentation de l'objet, c.à.d. en empêchant l'accès aux données par un autre moyen que les services proposés. L'encapsulation permet donc de garantir l'intégrité des données contenues dans l'objet.

Convaincre tous ces problèmes consiste en la **modularité.**

Comme exemple de module on propose le package du langage **ADA**. Ce package est composé de deux parties : 1. **Partie interface** et 2. **Partie corps**

Problème de la programmation procédurale

- La modularité est basée sur les processus du système

Exemple: Développement d'une application Le système de bibliothèque

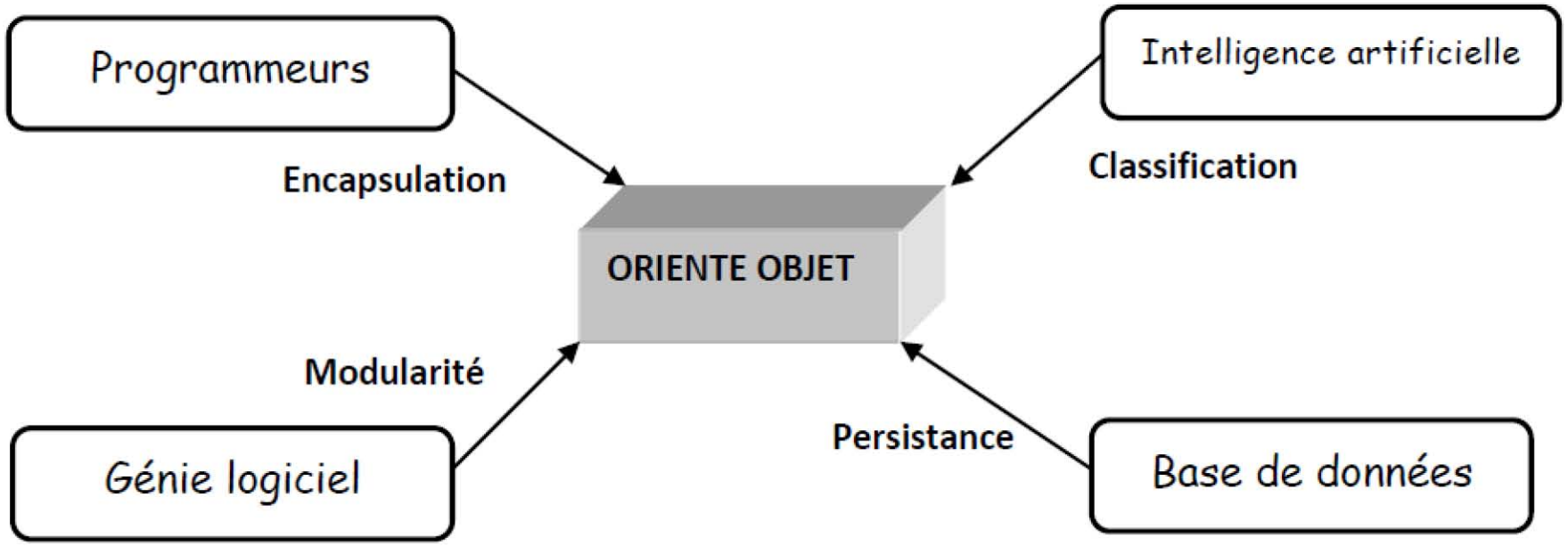
Dans cette exemple, on doit considérer les processus tel que :

1. Cataloguer les livres
2. Faire des réservations
3. Prêt et remise de livres
4. Etc.

La résolution du problème consiste à analyser ces processus en termes de taches procédurales à traiter et la production d'un système ou la représentation est basée sur un flux procédural de processus.

De l'autre coté, la programmation orientée objet modélise les objets et leurs interactions dans l'espace du problème et la production d'un système basé sur ces objets et leurs interactions.

Origine de la programmation Orientée Objet



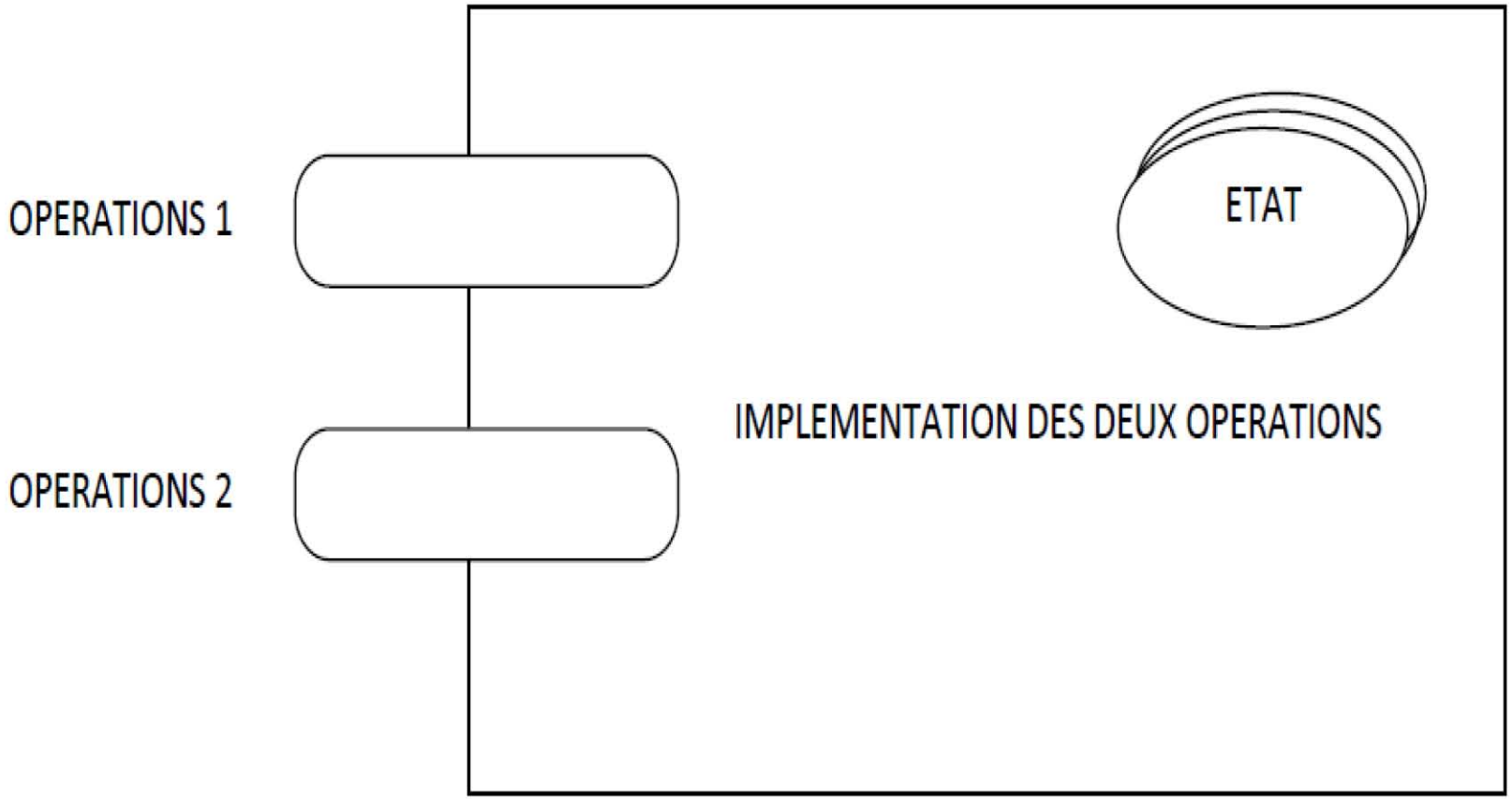
Le problème du monde réel est caractérisé par de objets et leurs interactions. Un système développé en utilisant la programmation orientée objet doit produire un système proche de la représentation du monde réel que celui de la programmation procédurale.

Objet

- l'objet est une collection d'opérations qui partagent un état. Les opérations déterminent les messages (appels) auxquels l'objet doit répondre. L'état partagé est caché de l'extérieur est accessible uniquement aux opérations de l'objet. Les variables représentant l'état interne de l'objet sont appelées **variables d'instances** et ses opérations sont appelées **méthodes**. La collection de méthodes détermine son interface et son comportement.
- Un objet est une abstraction d'une donnée caractérisée par un identifiant unique et invariant, un état représenté par une valeur simple ou structurée et une classe d'appartenance

Objet = Identité + Etat + Comportement

Identificateur : OID (objet Identifier)

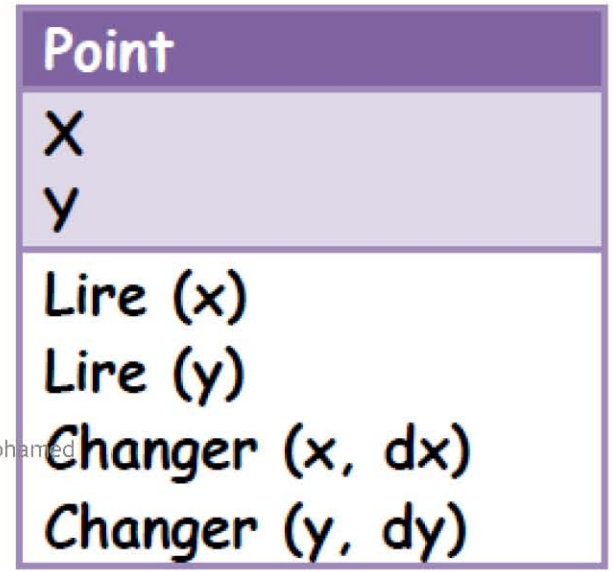

OPERATIONS 1
OPERATIONS 2
ETAT
IMPLEMENTATION DES DEUX OPERATIONS

Classe

- c'est un mécanisme qui permet de regrouper sous une même définition des objets similaires.
- Pour le partage de code et de propriétés ce qui nous permet de faire une économie de mémoire et d'éviter le problème de cohérence de copies. La description des attributs et le code des méthodes sont factorisés au niveau de la classe.

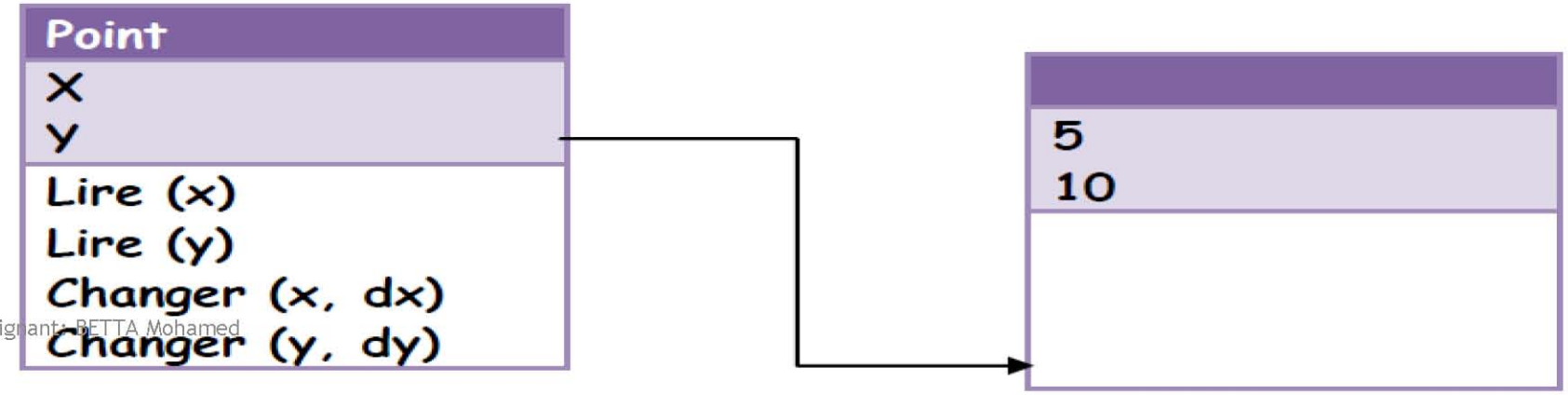
La classe sert comme moule pour créer des objets

- La classe est caractérisée par trois aspects : Instanciation, Attributs et Opérations
- C'est un type abstrait de données caractérisé par des **propriétés** (attributs + opérations) communes de ses objets et un mécanisme permettant de créer des objets ayant ces propriétés



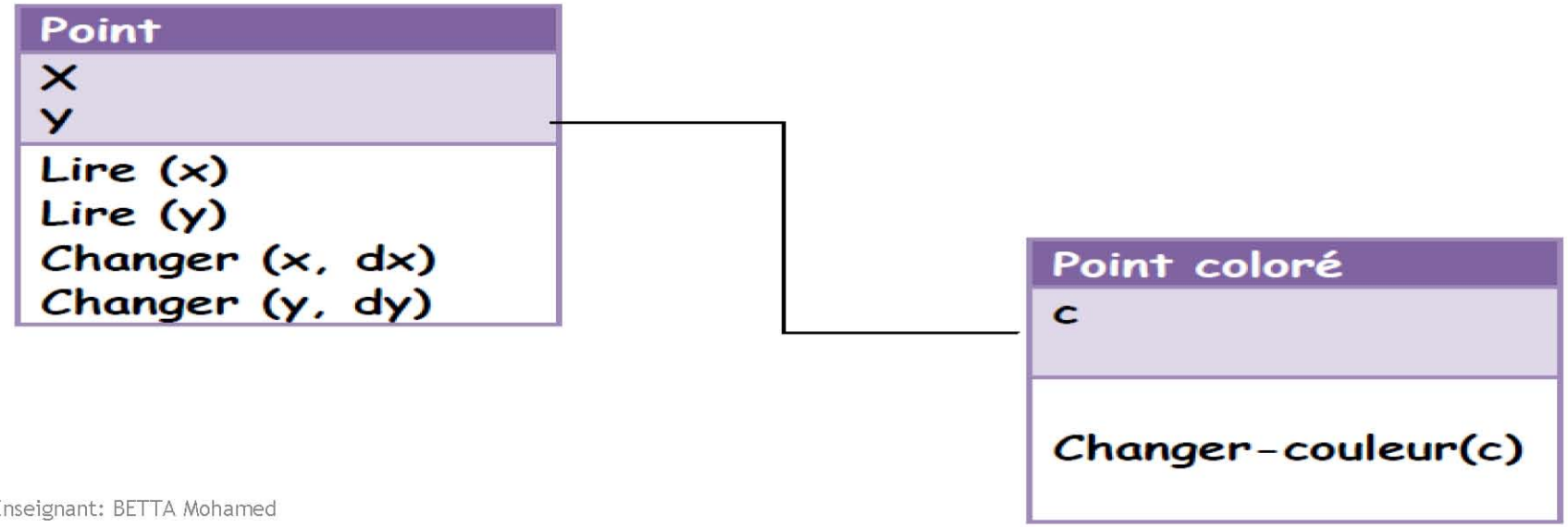
Différence entre classe et objet

- La différence se situe dans la façon dont les attributs et méthodes sont traités dans les objets et les classes.
- 1. Une classe est une définition sur les objets, les attributs et méthodes dans la classe sont des déclarations et ne contiennent pas de valeurs.
- 2. Les objets sont traités comme instance de classe.
- 3. Les valeurs de l'ensemble des attributs décrivent l'état des objets



Héritage

L'héritage permet de réutiliser le comportement d'une classe dans la définition de nouvelles classes. La sous-classe d'une classe hérite les opérations de la classe parente et doit avoir de nouvelles opérations et de nouvelles variables d'instances



Pourquoi l'héritage

- Structurer l'univers du discours : avoir une meilleure lisibilité, maitrise et compréhension
- Réutilisation de propriétés, de code et de comportement + partage de code.

La réutilisation est acquise grâce au mécanisme d'héritage

Formes d'héritage

1. Héritage pur (pas de redéfinition du code des méthodes)
2. Héritage avec surcharge (overloading)
3. Héritage simple (hiérarchie d'héritage = arbre)
4. Héritage multiple (hiérarchie d'héritage = graphe

Un programme OO = {O} + interaction entre objets

L'interaction est effectuée par l'envoi de messages

La structure d'un message = Object-receveur méthode [arguments]