Diagramme de Classes
&
Diagramme d'Objets

MCOO–Chapitre 3

Diagramme de Classes

- Le diagramme le plus important de la modélisation orientée objet
- Décrit la structure interne du système
- Fournit une représentation abstraite des objets du système qui interagissent ensemble pour réaliser le cas d'utilisation
- Vue statique : pas de facteur temporel
- Modélisation des classes du système et leurs relations
  - ✓ Indépendant du langage de programmation

Classes

- Description formelle d'un ensemble d'objets ayant une sémantique et des caractéristiques communes
- Un objet est une instance de classe
- Représentée par un rectangle divisé en 3 compartiments obligatoires et 2 optionnels
  - ✓ Nom de la classe
  - ✓ Attributs
  - ✓ Méthodes
  - ✓ Responsabilités (op) : ensemble des tâches devant être assurées par la classe mais pour lesquelles on ne dispose pas d'assez d'informations
  - ✓ Exceptions (op) : Situations exceptionnelles devant être gérées par les classes

Caractéristiques d'une Classe

- Visibilité
  - ✓ Public ou + :tout élément qui peut voir la classe courante peut également voir l'élément indiqué
  - ✓ Protected ou # : seul un élément situé dans la classe courante ou un de ses descendants peut voir l'élément indiqué.
  - ✓ Private ou – : seul un élément situé dans la classe courante peut voir l'élément.

Attributs

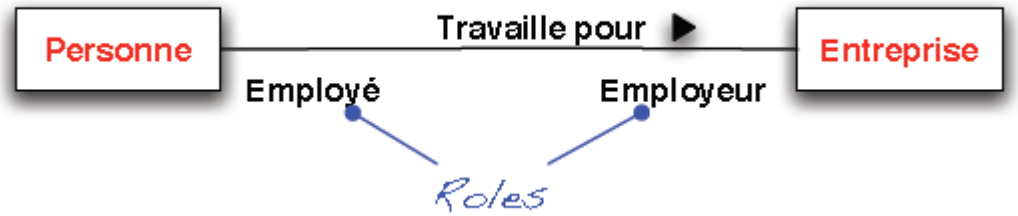
- Données encapsulées dans les objets de cette classe
- Définis par un nom, un type de données, et une visibilité
- A la forme: <visibilité> [/] <nom_attribut> : <type> ['['<multiplicité>']' [{<contrainte>}]] [= <valeur_par_déf.>]
- Exemple : + couleur : int [3] {list}
- Attribut de classe
  - ✓ Attribut propre à la classe, pas à l'instance(*static* en java)
  - ✓ Garde une valeur unique et partagée par toutes les instances de la classe
  - ✓ Graphiquement : souligné

Méthodes

- Décrit une fonctionnalité de la classe
- Doit contenir un nom, un type de retour et des paramètres
- A la forme :
  - ✓ <visibilité> <nom_méthode> ([<paramètre_1>, … , <paramètre_N>]) : [<type_renvoyé>] [{<propriétés>}]
- Un paramètre a la forme :
  - ✓ [<direction>] <nom_paramètre>:<type> ['['<multiplicité>']'] [=<valeur_par_défaut>]
- Exemple
  - ✓ + déplacer (in distance : int = 2) : void {abstract}

Relation d'Association

- Relation entre deux classes ou plus décrivant les connexions structurelles entre leurs instances
- Relie des classes au même niveau hiérarchique
- 4 décorations permettent de spécifier le lien entre objets :
  - ✓ Nom : nature des relations entre les objets
  - ✓ Direction : direction d'application du nom
  - ✓ Rôle : rôle spécifique de chacune des classes dans l'association
  - ✓ Cardinalité : nombre d'éléments affectés
- Exemple

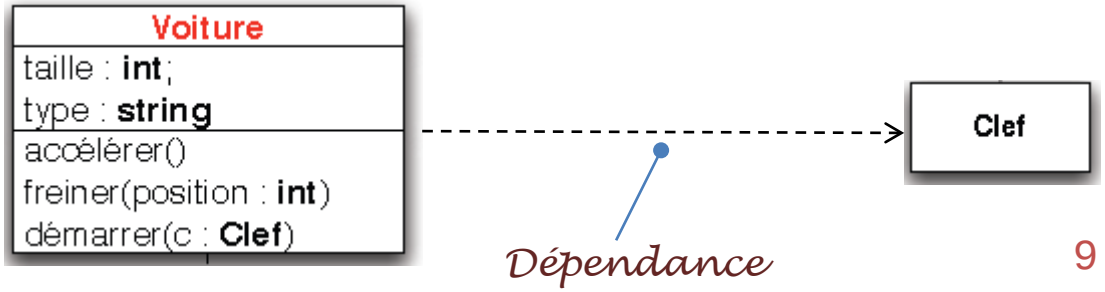


Relations d'Agrégation et de Composition

- Agrégation
  - ✓ Définit une relation hiérarchique entre les entités
  - ✓ Définit la relation : « se compose de » et modélise la notion de « possession » ou de « tout et partie »
- Composition
  - ✓ Définit une contenance structurelle entre les instances
  - ✓ La destruction de l'objet **composite** implique la destruction de ses **composants**
  - ✓ Une instance du composant appartient au plus à une instance du composite

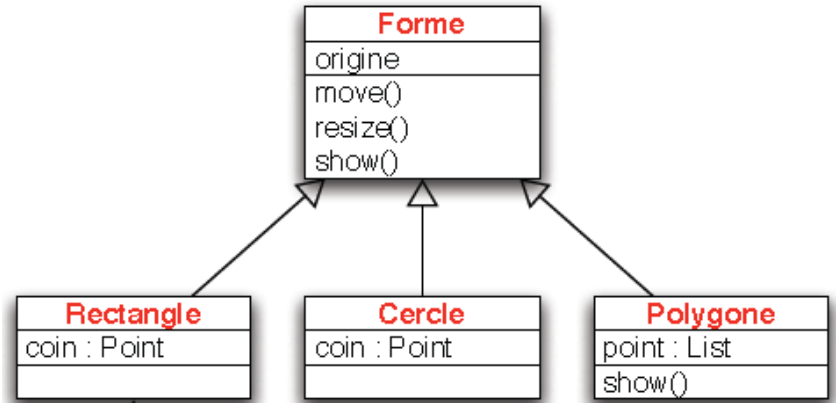
Relation de Dépendance

- La dépendance établit une relation d'utilisation entre 2 entités d'un même diagramme.
- La plupart du temps il s'agit d'une dépendance d'utilisation :
  - ✓ Argument d'une méthode par exemple
  - ✓ On parle alors de "relation d'utilisation"
- Cela permet d'identifier implications possibles des modifications à apporter dans une entité
  - ✓ Changement du comportement d'une des classes par exemple



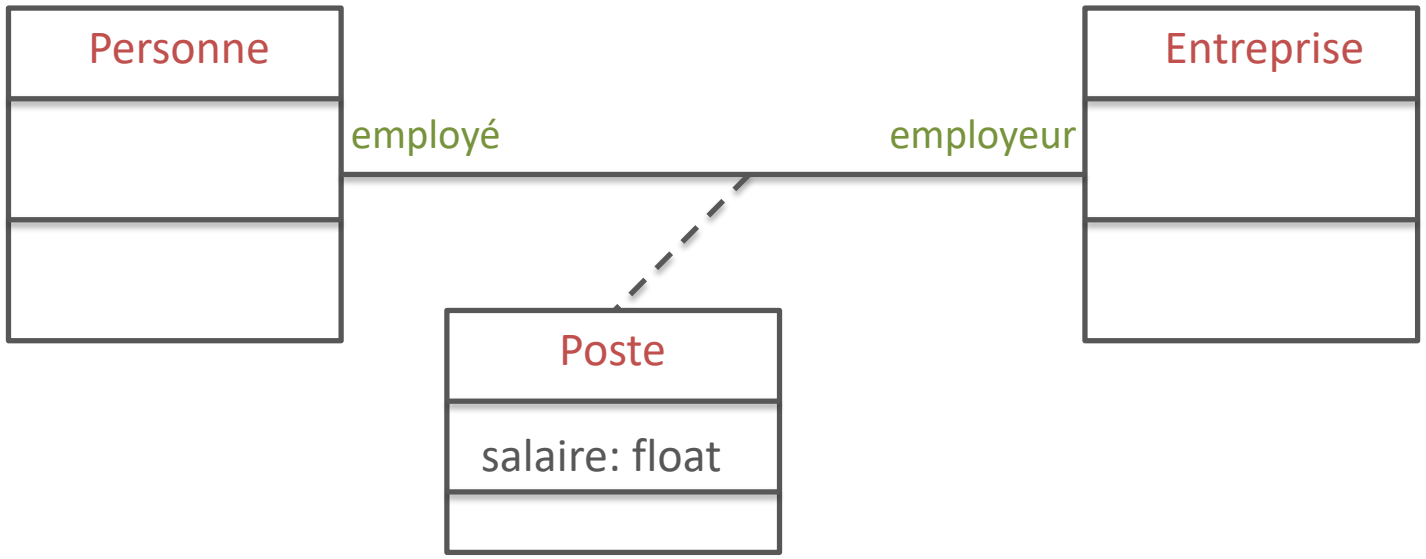
Relation de Généralisation

- Modélise la relation d'héritage
  - ✓ La généralisation correspond à la notion "est une sorte de"
  - ✓ Modélisation des relations parents / enfants
- Les entités issues d'une généralisation sont utilisables partout ou leur classe mère peut l'être (mais pas l'inverse)
- Cette relation est modélisée par une flèche pointant sur la classe mère



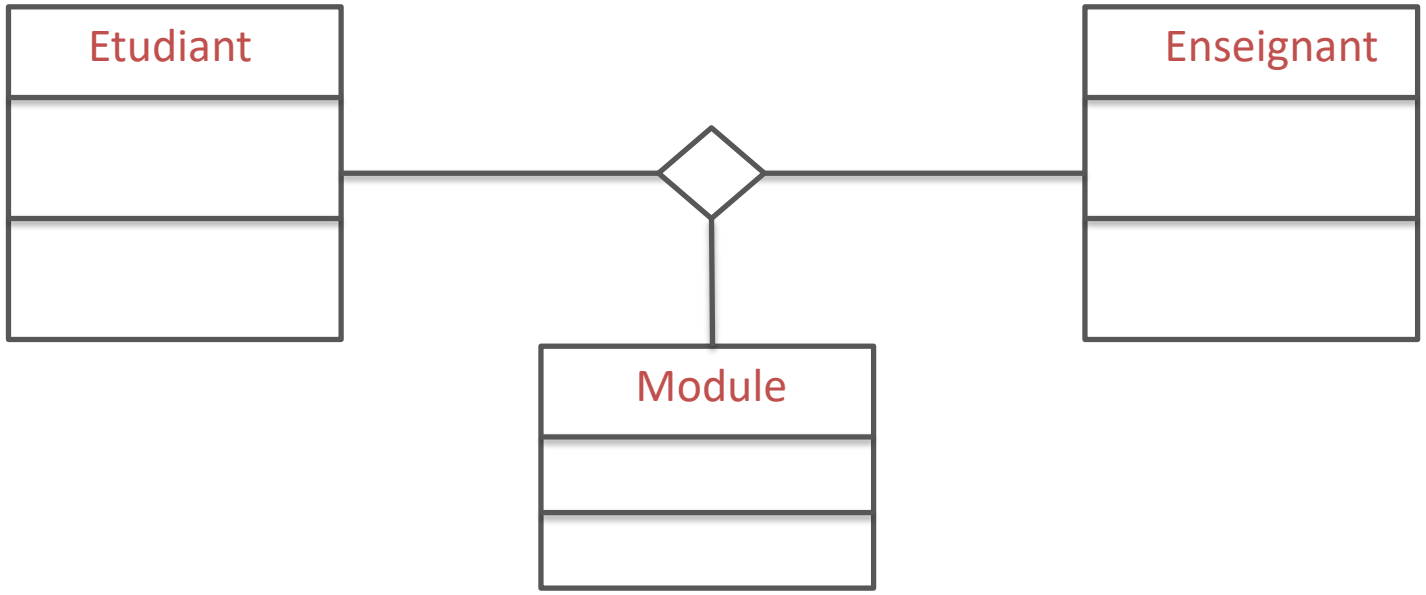
Classe-Association

- Une association peut avoir des propriétés, qui ne sont disponibles dans aucune des classes qu'elle lie
  - ✓ On définit alors une classe-association



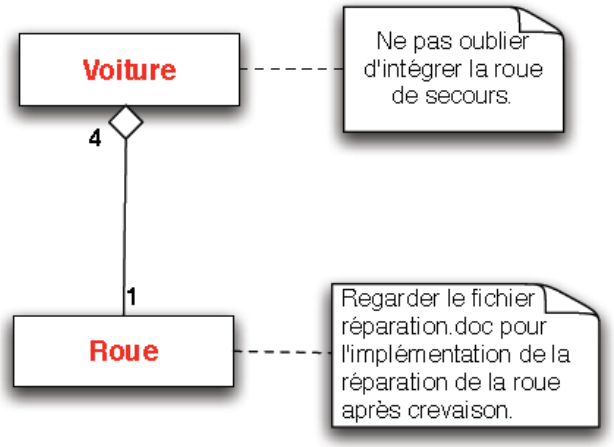
Association n-aire

- Association qui lie plus que 2 classes
- Peu utilisée
  - ✓ Gestion des multiplicités délicate



Les Notes (Commentaires)

- Lors des phases de modélisation et de spécification, il est nécessaire de documenter ses modèles :
  - ✓ Contraintes matérielles
  - ✓ Contraintes de performance
  - ✓ Choix techniques réalisés
  - ✓ Références à d'autres docs
  - ✓ Explications techniques, etc.



- En UML, on utilise les *Notes*

Exemple : Banque

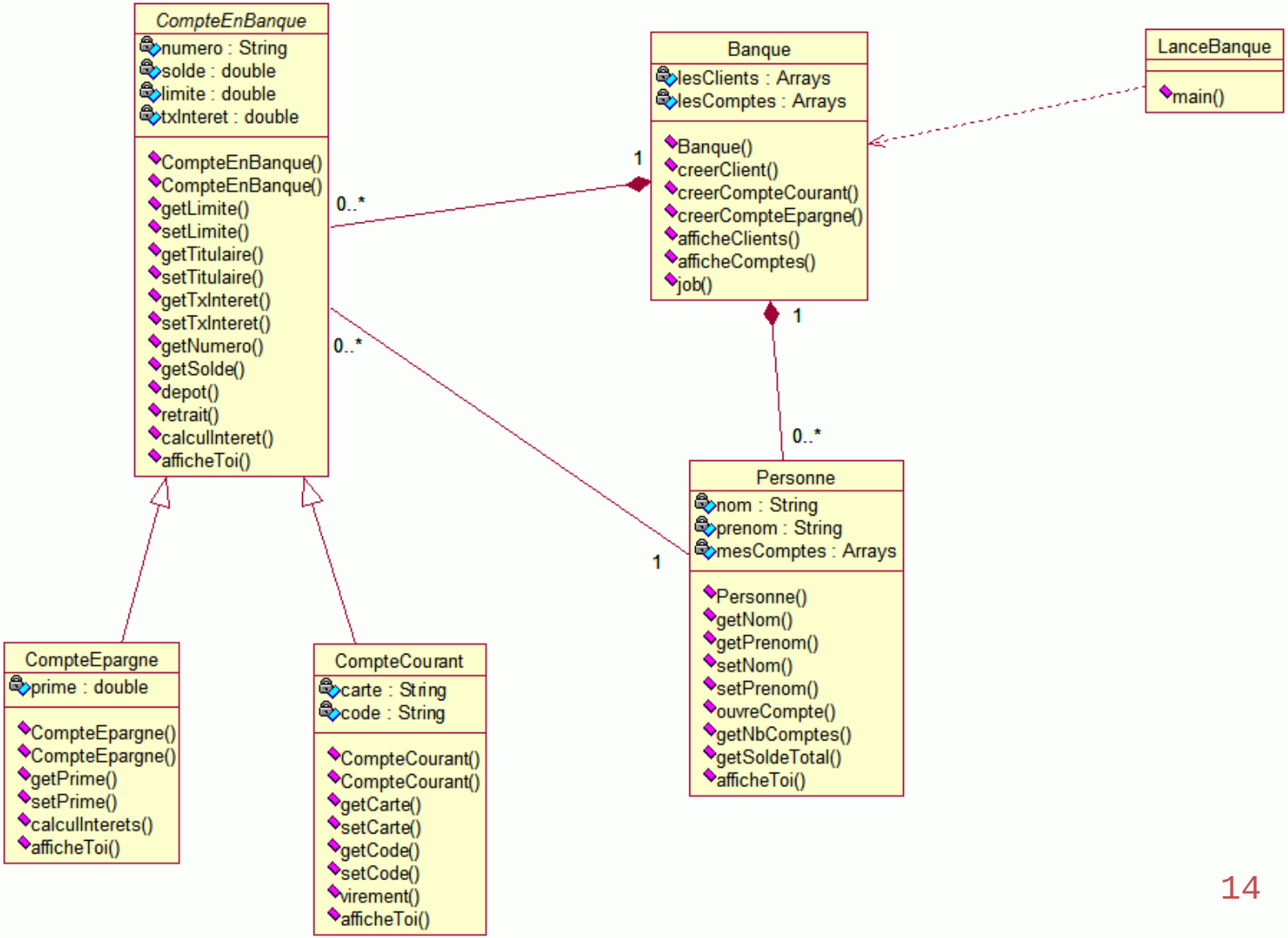


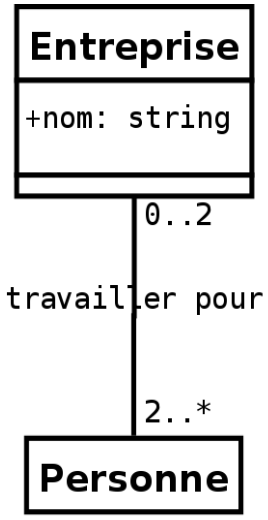
Diagramme d'Objets

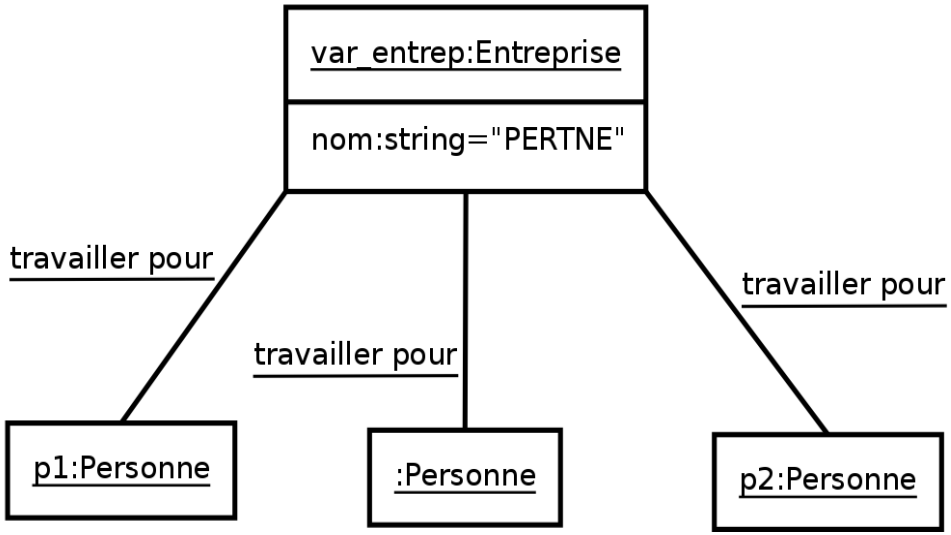
- Représente des objets (i.e. instances de classes) et leurs liens (i.e. instances de relations) pour donner une vue figée de l'état d'un système à un instant donné
- Peut être utilisé pour
  - ✓ illustrer le modèle de classes en montrant un exemple qui explique le modèle
  - ✓ préciser certains aspects du système en mettant en évidence des détails imperceptibles dans le diagramme de classes
  - ✓ exprimer une exception en modélisant des cas particuliers ou des connaissances non généralisables qui ne sont pas modélisés dans un diagramme de classe
- **Le diagramme de classes modélise les règles et le diagramme d'objets modélise des faits**

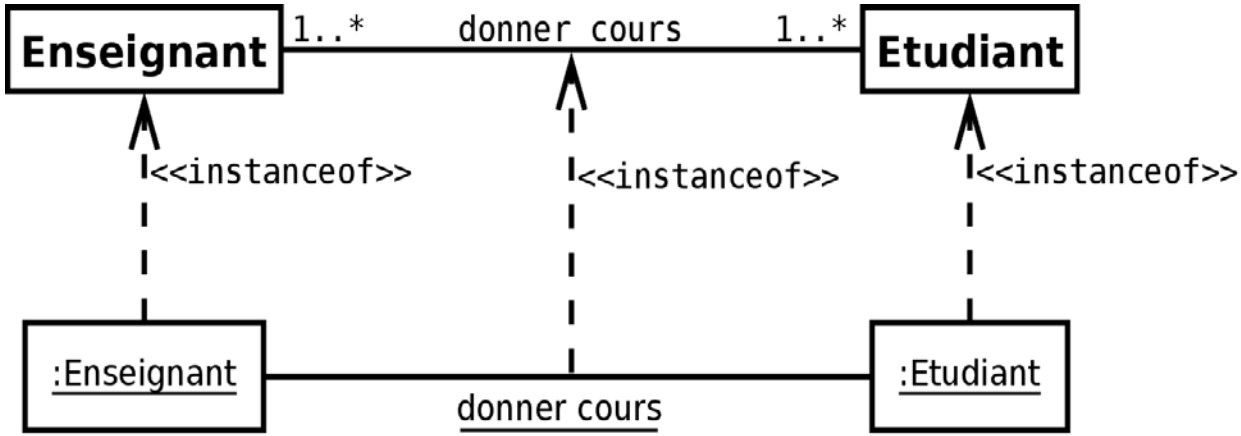
Graphiquement

- Un objet est représenté comme une classe, mais le compartiment des méthodes n'est pas indiqué
- Le nom de l'objet est composé du nom de l'instance, suivi de celui de la classe, et est souligné
- Les attributs reçoivent des valeurs
  - ✓ Si certaines valeurs ne sont pas renseignées, l'objet est **partiellement défini**
- La relation de généralisation n'est **jamais** représentée
- Les multiplicités ne sont pas représentées
- On peut représenter la dépendance d'instanciation

Exemples


Entreprise
+nom: string
0..2
travailler pour
2..*
Personne


var_entrep:Entreprise
nom:string="PERTNE"
travailler pour
travailler pour
travailler pour
p1:Personne
:Personne
p2:Personne


Enseignant
1..*
donner cours
1..*
Etudiant
<<instanceof>>
<<instanceof>>
<<instanceof>>
:Enseignant
:Etudiant
donner cours

Activité 2: Bibliothèque

- Une bibliothèque compte les exemplaires des titres suivants parmi les livres dont elle dispose : « Histoire de la 2eme guerre mondiale », « Les Aventures de Robin Hood », et deux exemplaires de « Harry Potter».
- Felix et Alain sont des utilisateurs abonnés. Alain a emprunté «Les Aventures de Robin Hood» tandis que Felix a emprunté deux livres: « Histoire de la 2eme guerre mondiale » et un exemplaire de « Harry Potter ».
- Représenter le diagramme de classes et le diagramme d'objets modélisant ce cas de figure.