

Modélisation statique

Introduction :

Le diagramme de classes est sans doute le diagramme le plus important à représenter pour les méthodes d'analyse orientées objet. C'est le point central de tout développement orienté objet.

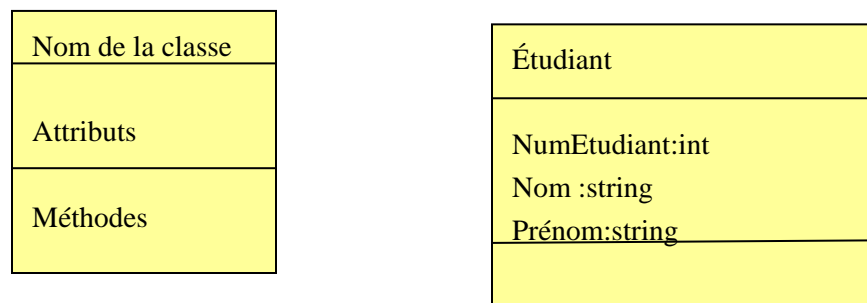
On peut voir le diagramme de classes à différents niveaux de développement. En analyse il permet de décrire la structure des entités manipulées par les utilisateurs. En conception, il permet de représenter un code orienté objet.

Définition : Le diagramme de classes

Un diagramme de classes est une collection d'éléments de modélisation statique qui montre la structure d'un modèle. Une classe représente la description d'un ensemble d'objets possédant les mêmes caractéristiques. Un diagramme de classes fait abstraction des aspects dynamiques et temporels du système.

Représentation d'une classe

Une classe est représentée par un rectangle séparé en trois parties:



Un attribut ou une méthode peut être de type :

Protégé, il est précédé du symbole #, visible aux sous classes de la classe

Privé, il est précédé du symbole -, visible à la classe seule

Public, il est précédé du symbole +, visible à tous les clients de la classe

Liens entre classes ou association entre classes.

L'association est la relation la plus évidente qui existe entre classes, elle exprime une connexion sémantique bidirectionnelle entre deux classes.

Une association est une relation entre deux classes qui décrit les connexions structurelles entre leurs instances. Une association indique donc qu'il peut y avoir des liens entre des instances des classes associées.

Exemple1, association simple :

Comment représenter une personne qui travaille dans une seule compagnie? Ou encore dans une compagnie, nous avons plusieurs employés?

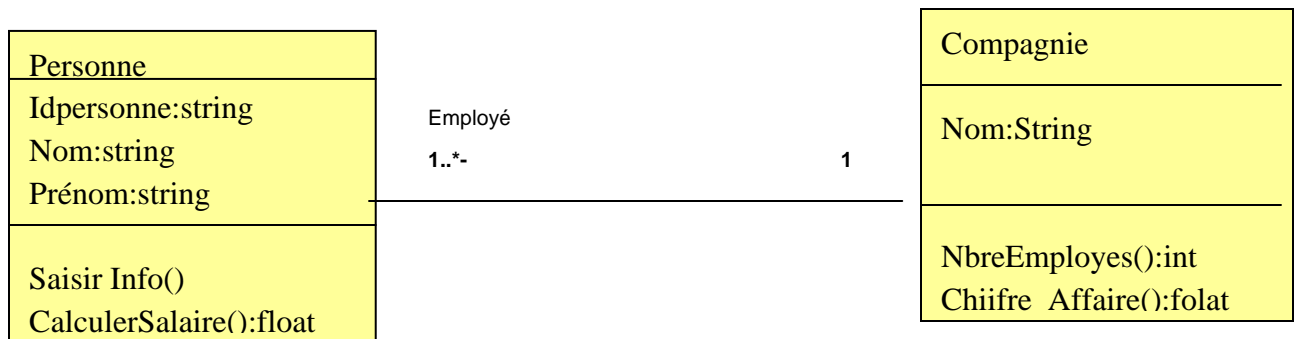


Figure 1: Diagramme de classes (représentation 1)

Dans ce diagramme, le **nom de l'association** est **Employé**, le nom du **rôle** de la classe **Personne** pour l'association est: Employé.

Le diagramme se lit comme suit :

Une personne travail pour une seule entreprise. Dans une entreprise il y 'a une à plusieurs personnes:



Figure 2: Diagramme de classes: Représentation2

Remarque:

Entre la représentation de la figure 1 et la représentation de la figure 2, pour la représentation des associations entre les classes UML opte pour la représentation de la figure1

Notion de cardinalité ou multiplicité

Une cardinalité est le nombre de fois minimum et maximum qu'une instance d'une classe participe à l'association. Ou encore c'est le nombre d'instances de l'association pour une instance de la classe. On l'appelle également la multiplicité

Nous distinguons :

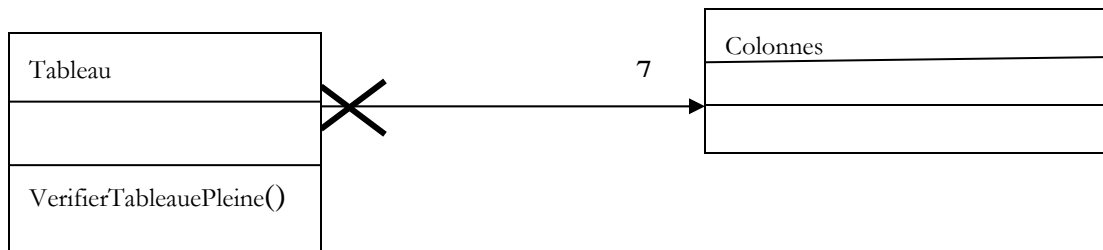
1	un et un seul
0..1	Zéro ou un
m..n	De m à n (entier)
*	Plusieurs
0..*	De zéro à plusieurs
1..*	d'un à plusieurs

Question 1 : Comment représenter qu'une pièce a une seule forme?

Réponse :

Notion de rôle : un rôle spécifie la fonction d'une classe pour une association donnée

Relation navigable:



Sources UML 2 par la pratique Étude de cas et exercices corrigés. Pascal Roques Editions Eyrolles

<http://uml.free.fr/>

On représente graphiquement la navigabilité par une flèche du côté de la terminaison navigable et on empêche la navigabilité par une croix du côté de la terminaison non navigable

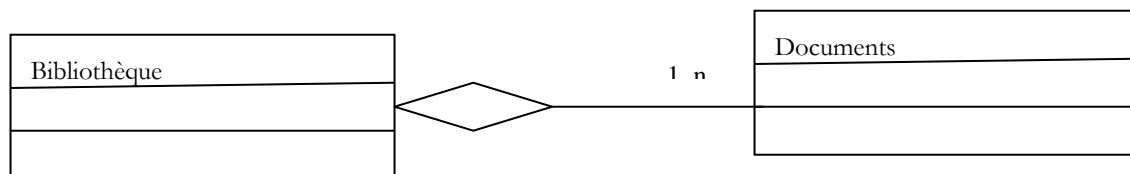
Par exemple, sur la figure la terminaison du côté de la classe *Tableau* n'est pas navigable : cela signifie que les instances de la classe *Colonnes* ne stockent pas de liste d'objets du type *Tableau*. Inversement, la terminaison du côté de la classe *Colonnes* est navigable : chaque objet *Tableau* contient 7 colonnes.


Par défaut, une association est navigable des deux côtés.

Agrégation : est un type d'association, elle est symbolisé par 

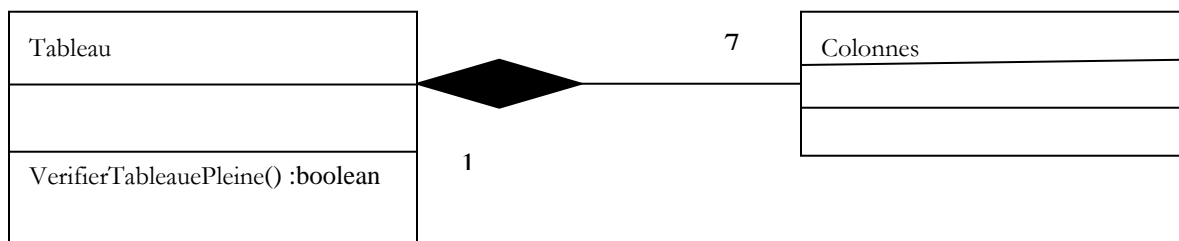
Elle définit la relation « partie de ».

Exemple2 : un document fait partie d'une bibliothèque



Une **composition** est une agrégation forte, elle est représentée par le symbole 

Exemple3 : un tableau est composé de colonnes



La composition, également appelée agrégation composite, décrit une contenance structurelle entre instances. Ainsi, la destruction de l'objet composite implique la destruction de ses composants

Dans les relations de composition ou agrégation, nous parlons d'objet composite et d'objet composant. Ainsi une instance de la classe **Tableau est dit objet composite** et une instance de la classe **colonne est dit objet composant**

Différences entre une agrégation et une composition :

Pour la relation de composition :

- La destruction de l'objet composite implique la destruction de ses composants.
- La multiplicité du côté composite ne doit pas être supérieure à 1 (*i.e.* 1 ou 0..1).
- La composition est une agrégation non partagée.

Question 2 : comment représenter «un livre est composé de chapitres»?

- **Réponse**

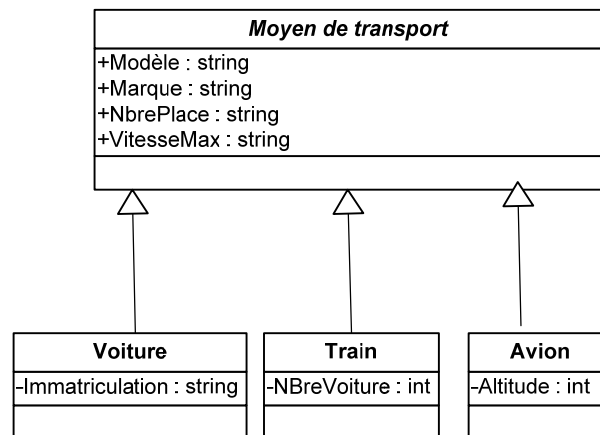
Généralisation, super-classe, sous-classe

La généralisation : définit une relation de classification entre une classe plus générale (super-classe) et une classe plus spécifique (sous-classe). Il s'agit de prendre des classes existantes (déjà) mise en évidence et de créer de nouvelles classes qui regroupent leurs parties communes, il faut aller du plus spécifiques au plus général, c'est une démarche ascendante.

La spécialisation : il s'agit de sélectionner des classes existantes (déjà) identifiées et d'en dériver de nouvelles classes plus spécialisées en spécifiant simplement les différences. Il s'agit d'une démarche descendante

Une classe abstraite est une classe qui ne s'instancie pas directement, mais qui représente une simple abstraction afin de factoriser les propriétés communes des sous-classes. Elle se note en italique.

Exemple4 : Comment représenter qu'un moyen de transport peut être un avion, un train ou une voiture?



La classe *Moyen de transport* est une classe abstraite.

Voici quelques propriétés de la relation d'héritage :

- La sou- classe possède toutes les caractéristiques des ses super-classes , mais elle ne peut accéder aux caractéristiques privées de cette dernière.
- Toutes les associations de la super-classe s'appliquent aux sous-classes dérivées.
- Une instance d'une classe peut être utilisée partout où une instance de sa super-classe est attendue. Par exemple, toute opération acceptant un objet d'une classe *Moyen de transport* doit accepter un objet de la classe avion.
- Une classe peut avoir plusieurs parents, on parle alors d'héritage multiple

En UML, la relation d'héritage n'est pas propre aux classes. Elle s'applique à d'autres éléments du langage comme, les acteurs ou les cas d'utilisation

Question3 : Comment représenter la situation suivante :

- Des documents sont soit des journaux soit des volumes ou des BD (bandes dessinées)
- Les volumes sont soit des dictionnaires soit des livres
- Les documents ont un titre et un numéro.

- Les volumes ont en plus un auteur, les BD ont en plus le nom du dessinateur, les journaux ont une périodicité.

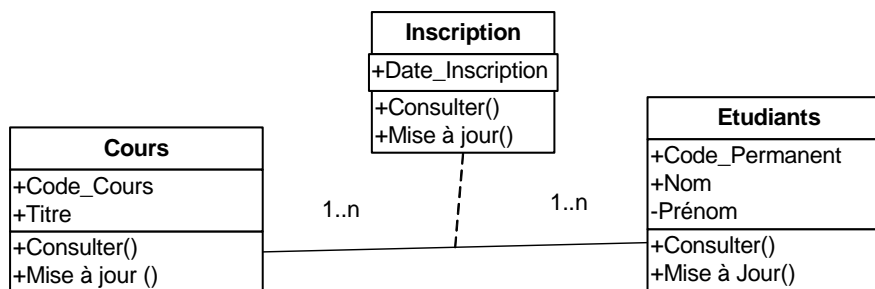
Réponse



Classe d'association

Il s'agit d'une classe qui réalise la navigation entre les instances d'autres classes. Elle sert connecter les classes entre elles

Exemple 5



Parfois, une association doit posséder des propriétés. Par exemple, l'association *Inscription* entre un élève et un cours possède comme propriété la date

d'inscription. Cette propriété n'appartient ni aux cours, ni aux étudiants Il s'agit donc bien de propriétés de l'association *Inscription*

Les associations ne pouvant posséder de propriété, il faut introduire un nouveau concept pour modéliser cette situation : celui de *classe-association*.

Une classe-association possède les caractéristiques des associations et des classes : elle se connecte à deux ou plusieurs classes et possède également des attributs et des opérations.

Une classe-association est caractérisée par un trait discontinu entre la classe et l'association qu'elle représente

Question4

Reprendre l'énoncé de la question précédente et représenter la situation suivante :

- Seuls les livres sont empruntables.
- Un adhérent peut emprunter plusieurs livres. On gardera la date de prêt et la date prévue pour le retour.

Réponse



Notion de package

Le package est un mécanisme regroupant plusieurs éléments d'UML (peut regrouper des classes, des cas d'utilisation, des interfaces...).

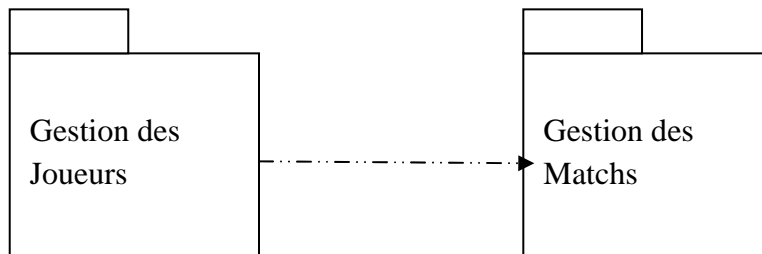
Le diagramme de package sert à :

- Avoir une vision globale des différents sous systèmes du système à l'étude.
- Représenter l'architecture globale du système
- Aider à organiser du code (java ou C#) .
- modulariser les diagrammes (UML) les plus complexes.

Le découpage d'un modèle (de classes ou de cas d'utilisation) est une activité délicate. Il faudra regrouper les classes d'un point de vue sémantique c'est-à-dire :

- Les classes d'un même package doivent rendre des services de même nature aux utilisateurs.
- Minimiser les dépendances entre les packages.
- Les dépendances entre packages doivent refléter des relations internes au système.
- Il ne doit pas y avoir de dépendance cyclique entre des packages

Exemple 6:



Chaque package doit être décrit par son diagramme de classes.

Question5

Reprendre l'énoncé des questions3 et 4, puis donner le des packages

Réponse

