SAGIM 2024/2025

UML 2.5.1

DIAGRAMME DE CLASSE

SAGIM

Un **diagramme de classe** en **UML** (Unified Modeling Language) est un type de diagramme structurel utilisé pour décrire la structure d'un système en représentant les classes, leurs attributs, leurs méthodes et les relations entre elles.

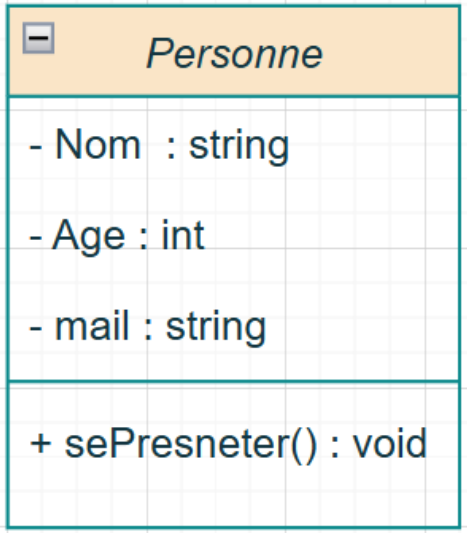
Représentation :

Une classe est représentée par un rectangle divisé en trois sections :

Nom de la classe : En haut, au centre.
Attributs : Dans la section du milieu. Ce sont les variables qui définissent l'état de la classe.
Méthodes : Dans la section du bas. Ce sont les opérations ou comportements que la classe peut exécuter.

CLASSE

Exemple d'une classe Personne :



SAGIM

Visibilité des membres :

+ **pour public** : accessible depuis n'importe où.

- **pour privé** : accessible uniquement depuis la classe elle-même.

pour protégé : accessible dans la classe et ses sous-classes.

CLASSE

SAGIM

Modificateurs d'accès des membres
Toutes les classes ont des niveaux d'accès différents, en fonction du modificateur d'accès (indicateur de visibilité). Voici les niveaux d'accès existants et les symboles qui leur sont associés :

Public (+)
Privé (-)
Protégé (#)
Paquetage (~)
Dérivé (/)
Statique (souligné)

Relations entre classes

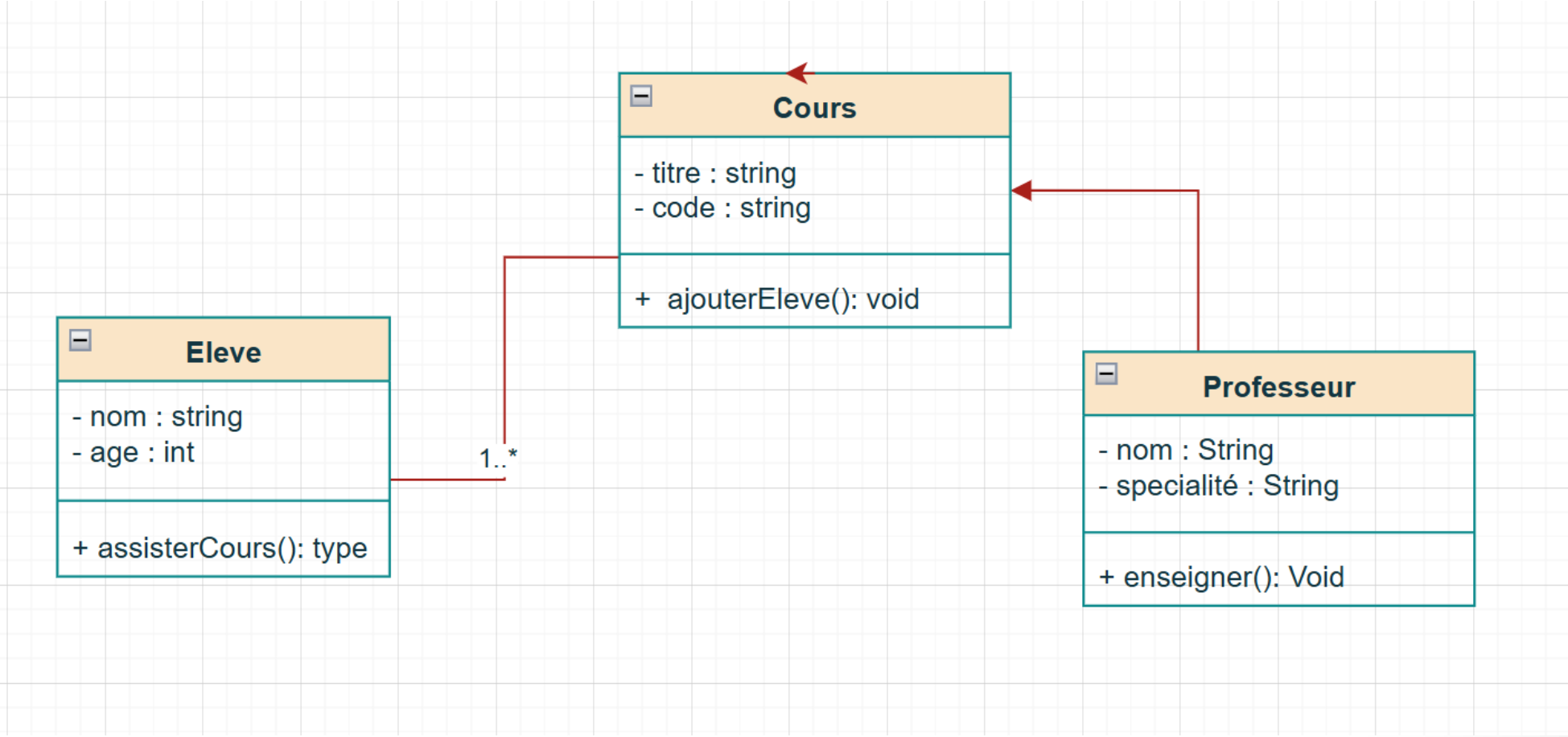
Association : Une ligne reliant deux classes avec ou sans multiplicité. La multiplicité indique combien **d'instances** d'une classe peuvent être **associées** à une autre.
Exemple : Une classe Auteur peut être associée à plusieurs **instances** de la classe Livre.
Héritage (Généralisation) : Représenté par une flèche creuse pointant vers la classe parente.
Exemple : Une classe Étudiant peut **hériter** d'une classe Personne.

Relations entre classes

SAGIM

Composition : Une forme plus forte d'association, représentée par une ligne avec un losange noir à une extrémité. Si la classe parente est **détruite**, les objets associés sont également **détruits**.
Exemple : Une classe **Voiture** pourrait être **composée** de plusieurs Pneus.
Agrégation : Une association moins forte que la composition, représentée par un losange vide à l'extrémité de la relation.
Exemple : Une classe **Équipe** pourrait avoir des Joueurs, mais les joueurs peuvent exister **indépendamment** de l'équipe.

Exemple de DC UML :



Exemple de DC UML :

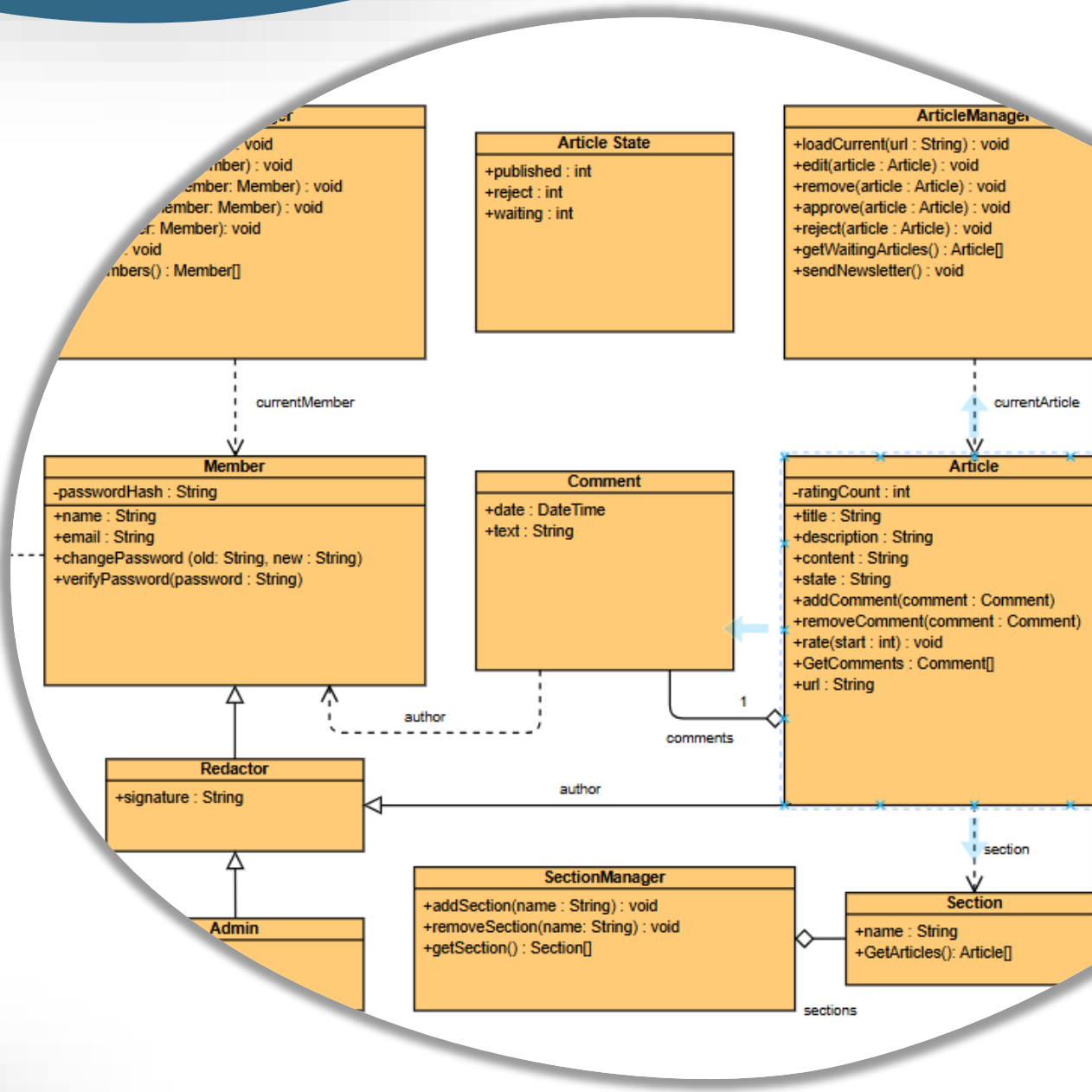
Dans cet exemple :

Un Élève peut assister à plusieurs Cours (indiqué par la multiplicité 1..*).

Un Professeur enseigne plusieurs Cours.

Diagramme De Classe

ASSOCIATION SIMPLE



Association (simple)

Une **association simple** relie deux classes et peut être annotée avec une **multiplicité** pour préciser le nombre **d'instances** d'une classe pouvant être liées à une autre.

Association (simple)

La multiplicité s'exprime ainsi :

1 : une instance de la classe est associée à **une seule** instance de l'autre classe.

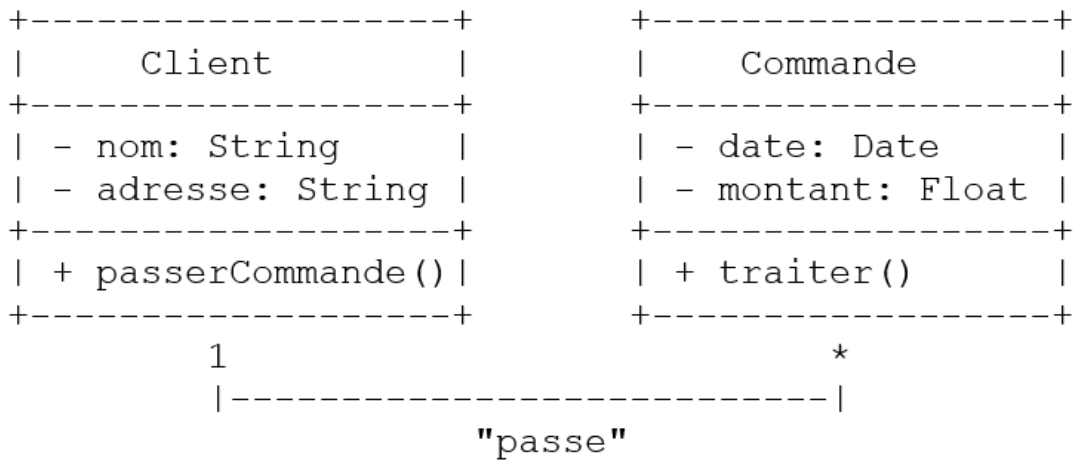
0..1 : une instance de la classe peut être associée à **zéro** ou **une instance** de l'autre classe.

***** : une instance **peut** être associée à **un nombre illimité** d'instances de l'autre classe.

1..* : **au moins** une instance **doit** être associée à une ou plusieurs instances.

Association (simple)

Exemple avec une classe **Commande** et une classe **Client** :



un **Client** peut passer **plusieurs Commandes**, mais chaque Commande est passée par **un seul** Client. C'est une association de type **1..***

TP de synthèse

Diagramme de classe en UML.

Gestion d'une bibliothèque

Une bibliothèque souhaite modéliser un système de gestion simple pour ses livres et ses auteurs.

Contexte :

Un livre est caractérisé par **un titre**, **un ISBN**, et **un nombre de pages**.
Un auteur est identifié par un **nom**, un **prénom**, et **une date de naissance**.
Un livre est écrit par **un ou plusieurs** auteurs.
Un auteur peut avoir **écrit un ou plusieurs** livres.

Objectif :

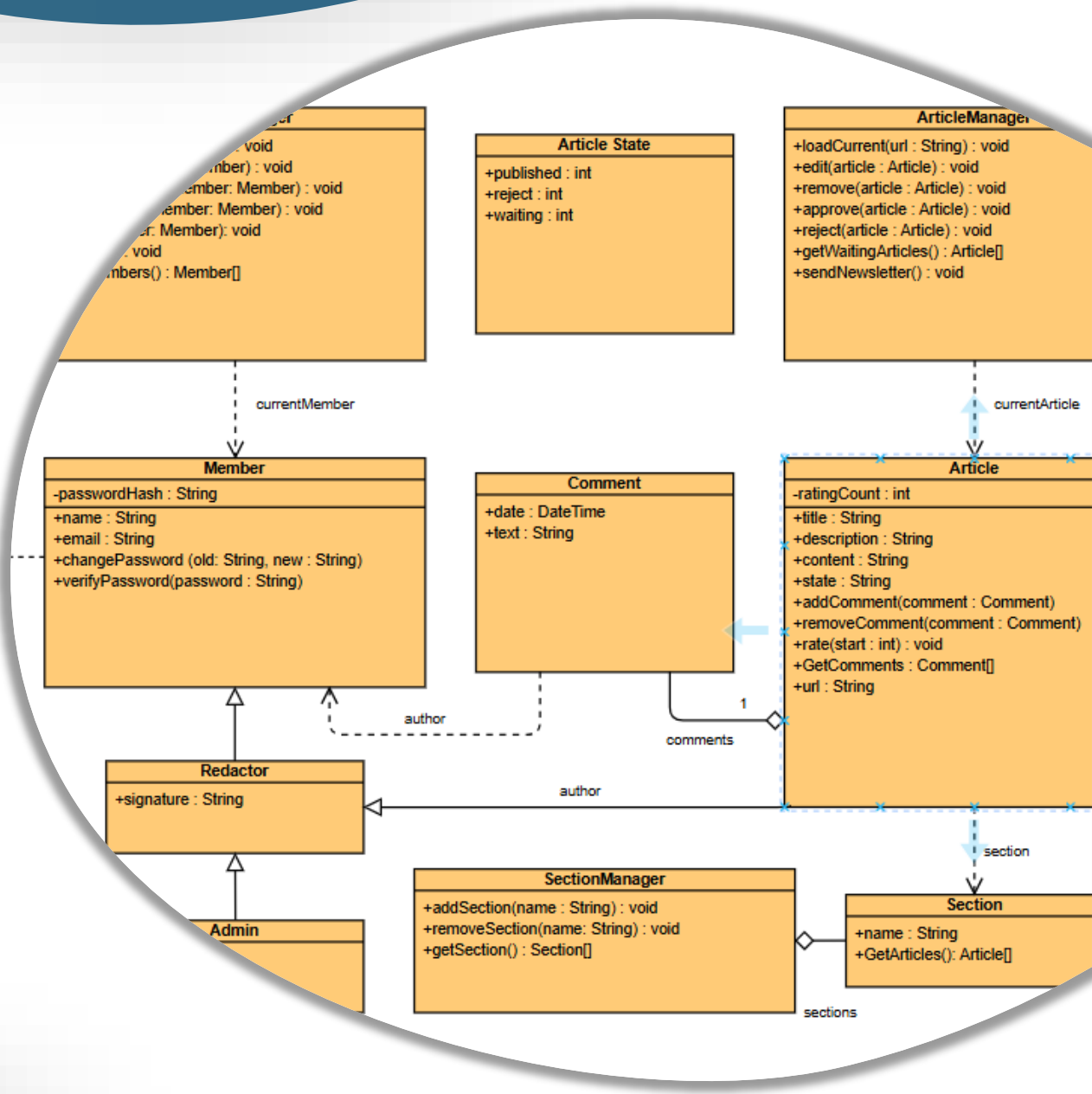
Créez le diagramme de classes correspondant à ce système en respectant les relations ci-dessous :
Une association entre les classes "**Livre**" et "**Auteur**".
Indiquez les multiplicités.

Gestion d'une bibliothèque

SAGIM

Diagramme De Classe

Héritage
(ou Généralisation)



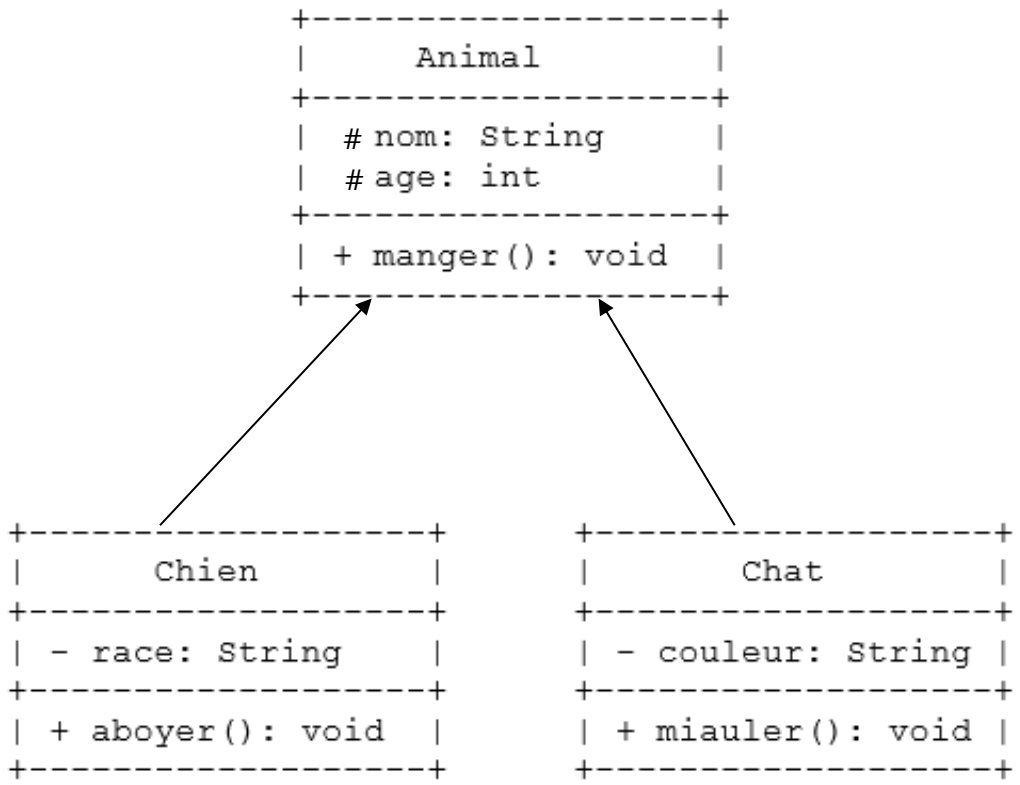
Héritage (ou Généralisation)

L'héritage est un mécanisme essentiel dans la modélisation orientée objet. Il permet de créer des sous-classes qui héritent des caractéristiques et des comportements de la classe parente. L'héritage est représenté par une flèche vide qui pointe de la classe enfant vers la classe parente.

Héritage (ou Généralisation)

SAGIM

Exemple avec les classes Animal, Chien et Chat :



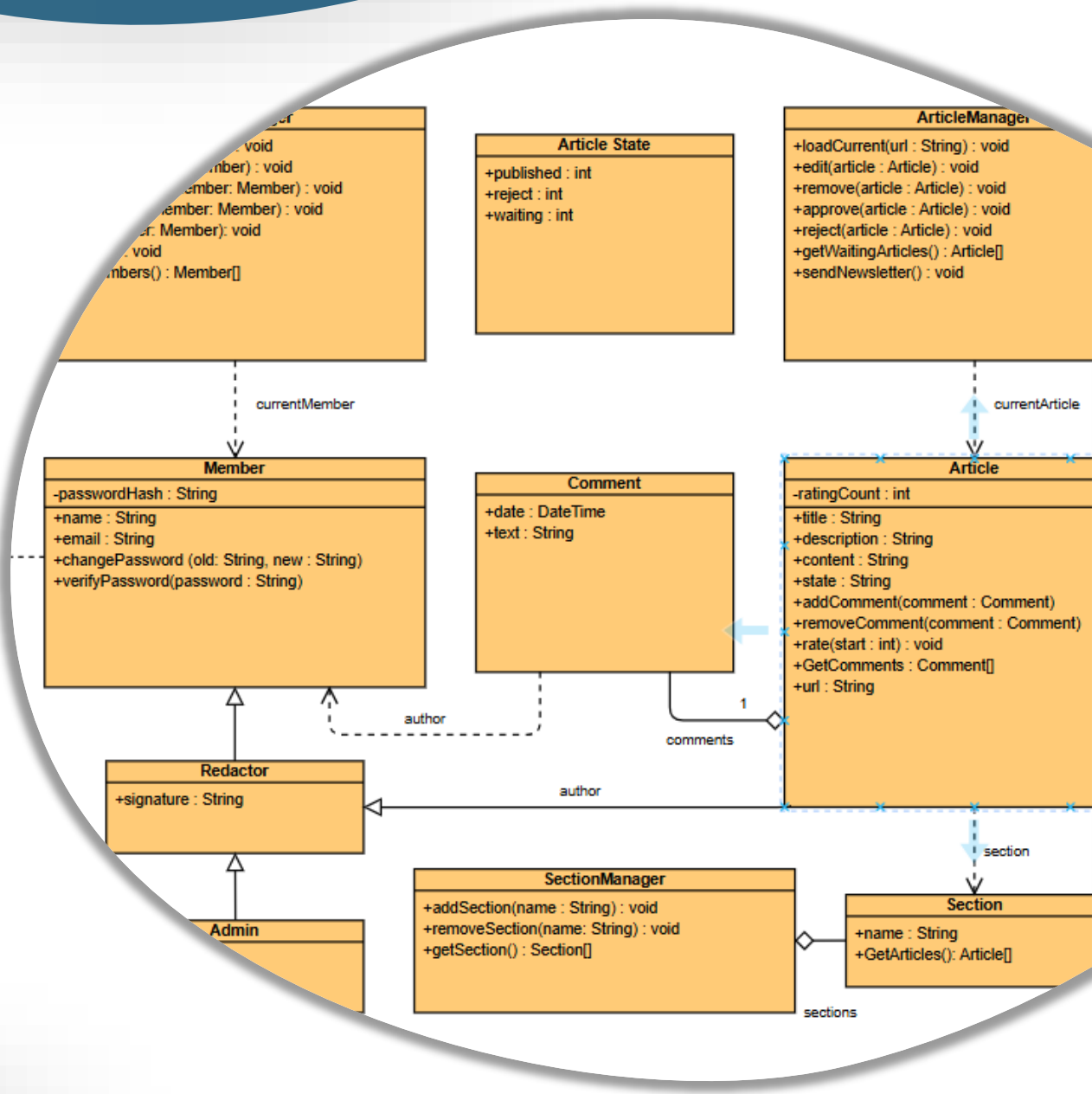
Héritage (ou Généralisation)

SAGIM

Ici, Chien et Chat héritent tous deux de la classe Animal, et ils ajoutent leurs propres caractéristiques spécifiques (comme race et couleur) et comportements (comme aboyer et miauler).

Diagramme De Classe

La Composition



Composition

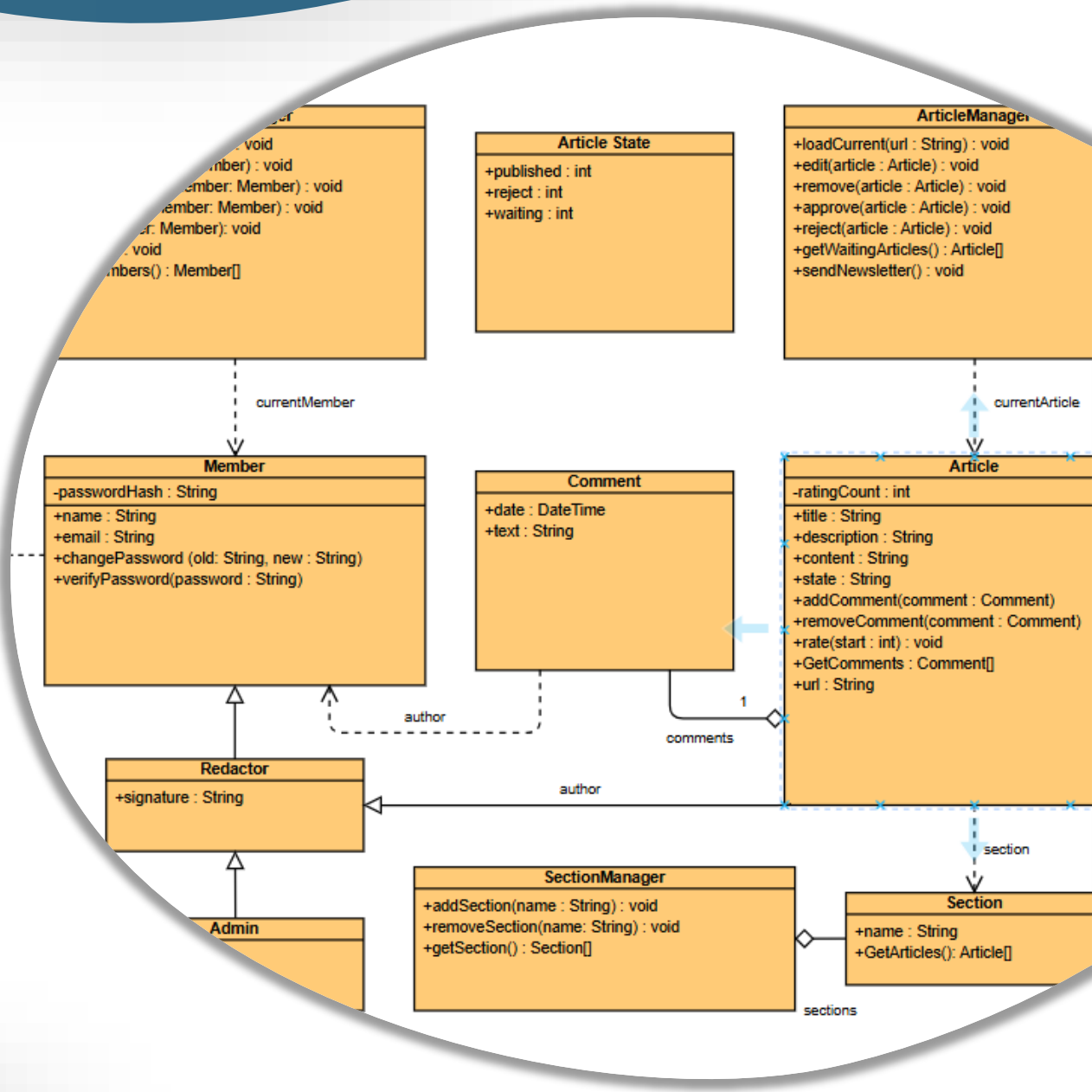
La composition représente une relation "partie-tout" très forte, où la classe "partie" ne peut pas exister sans la classe "tout".
Cela signifie que si la classe "tout" est détruite, les objets de la classe "partie" sont également détruits.
Cela se représente par un losange noir à l'extrémité de la ligne.

Composition

La composition représente une relation "partie-tout" très forte, où la classe "partie" ne peut pas exister sans la classe "tout".
Cela signifie que si la classe "tout" est détruite, les objets de la classe "partie" sont également détruits.
Cela se représente par un losange noir à l'extrémité de la ligne.

Diagramme De Classe

Dépendance



Dépendance

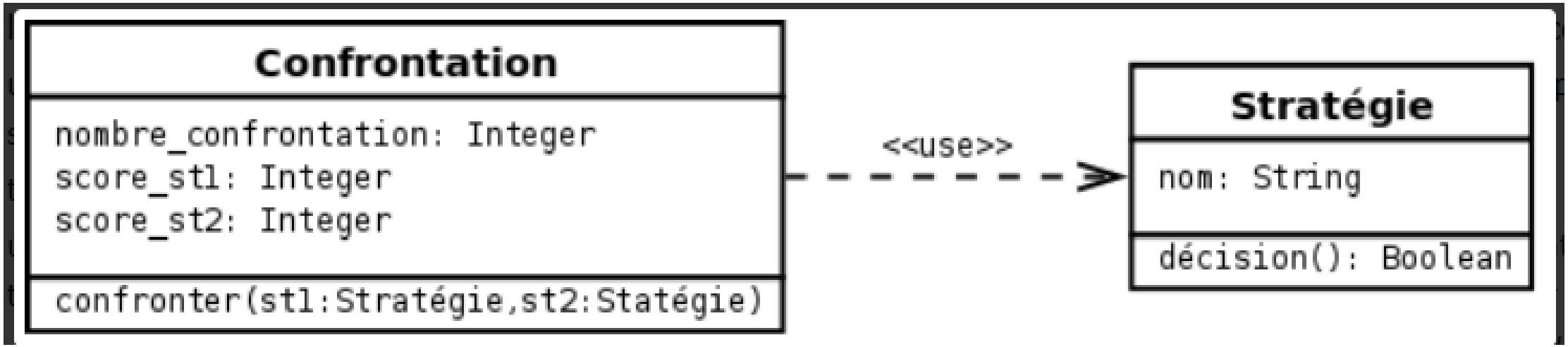
Une dépendance est une relation unidirectionnelle exprimant une dépendance sémantique entre des éléments du modèle. Elle est représentée par un trait discontinu orienté.

Elle indique que la modification de la cible peut impliquer une modification de la source.

La dépendance est souvent stéréotypée pour mieux expliciter le lien sémantique entre les éléments du modèle

Dépendance

SAGIM



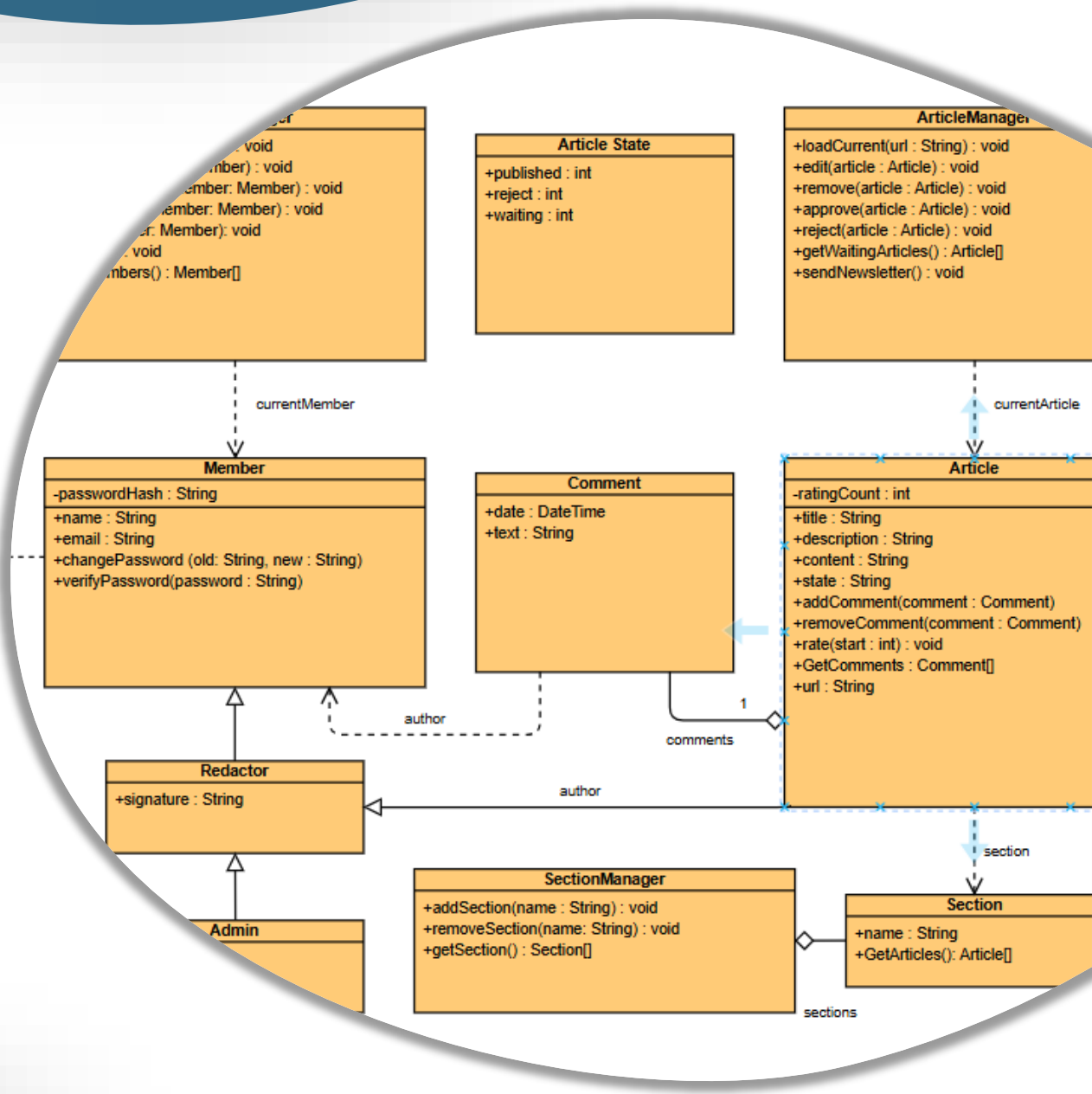
Dépendance

SAGIM

On utilise souvent une dépendance quand une classe en utilise une autre comme argument dans la signature d'une opération. Par exemple, le diagramme montre que la classe Confrontation utilise la classe Stratégie, car la classe Confrontation possède une méthode confronter dont deux paramètres sont du type Stratégie.
Si la classe Stratégie, notamment son interface, change, alors des modifications devront également être apportées à la classe Confrontation.

Diagramme De Classe

Les interfaces



INTERFACES

Le rôle de ce classeur, stéréotypé << interface >>, est de regrouper un ensemble de propriétés et d'opérations assurant un service cohérent.

L'objectif est de diminuer le couplage entre deux classeurs. La notion d'interface en UML est très proche de la notion d'interface en Java.

Une interface est représentée comme une classe excepté l'absence du mot-clef abstract (car l'interface et toutes ses méthodes sont, par définition, abstraites) et l'ajout du stéréotype << interface >>

INTERFACES

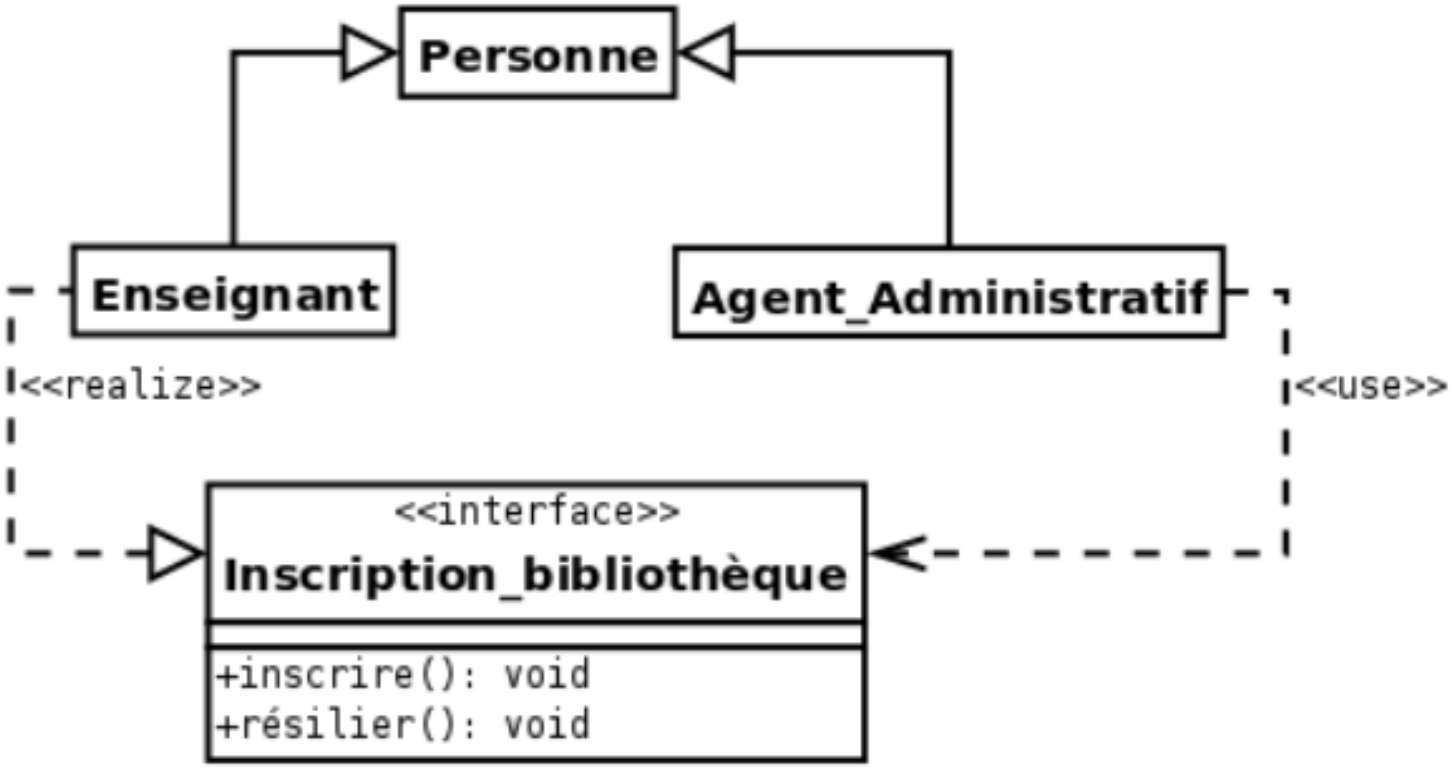
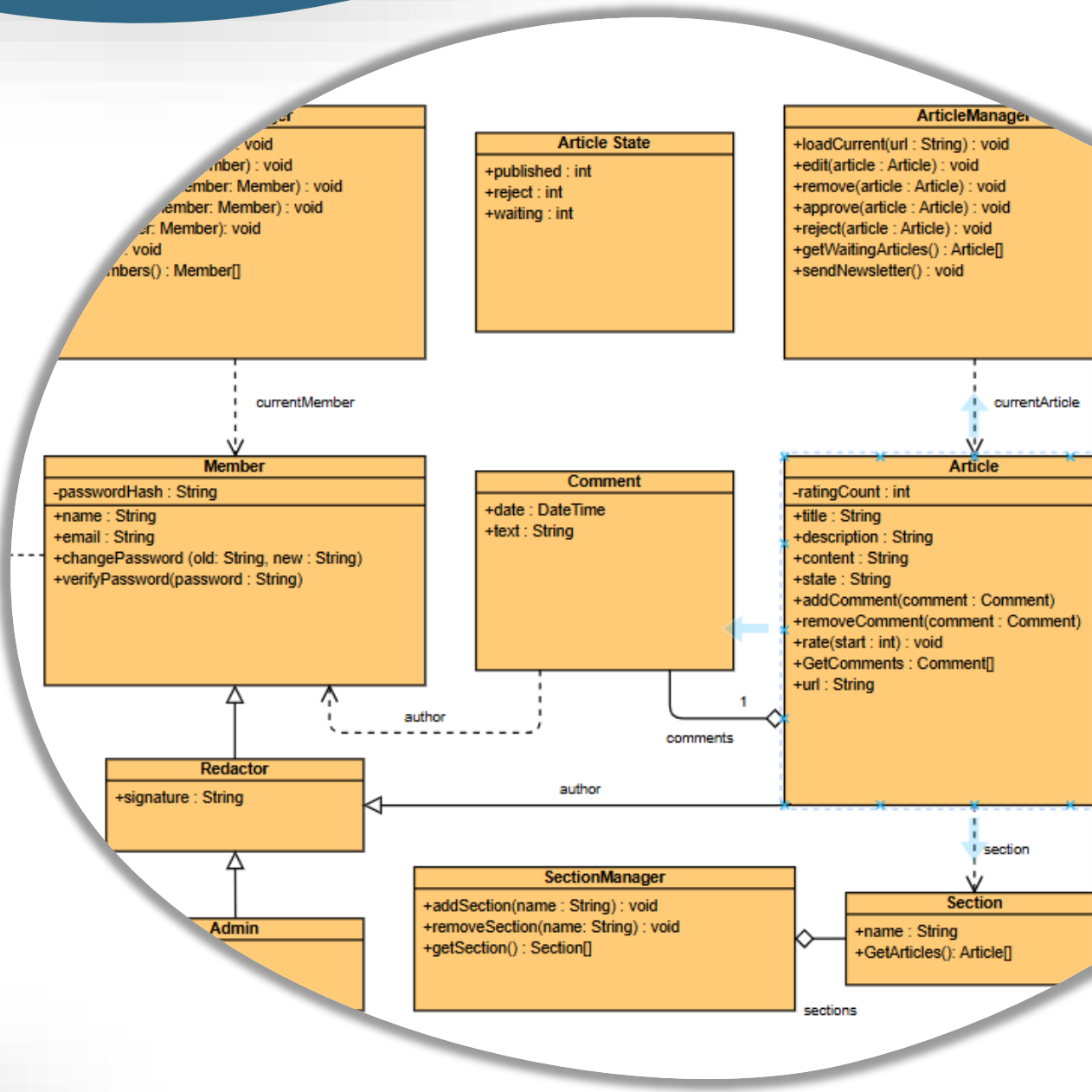


Figure 3.22 : Exemple de diagramme mettant en œuvre une interface.

Diagramme De Classe

Réalisation



REALISATION

Dans les diagrammes UML, une relation de réalisation d'interface est un type spécialisé de relation d'implémentation entre un discriminant et une interface fournie.
La relation de réalisation d'interface spécifie que le discriminant réalisant doit se conformer au contrat spécifié par l'interface fournie.
Généralement, les relations de réalisation d'interface n'ont pas de nom. Si vous nommez une réalisation d'interface, son nom est affiché à côté du connecteur dans le diagramme.

REALISATION

SAGIM

REALISATION

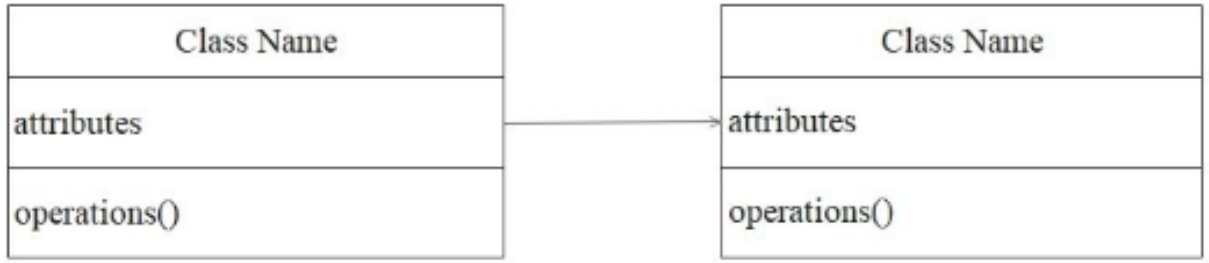
Comme le montre la figure suivante, une relation de réalisation d'interface est affichée dans l'éditeur de diagramme comme une ligne tiretée avec une pointe de flèche vide.
La réalisation d'interface est dirigée du discriminant vers l'interface fournie.

Résumé

Diagramme de classe en UML.

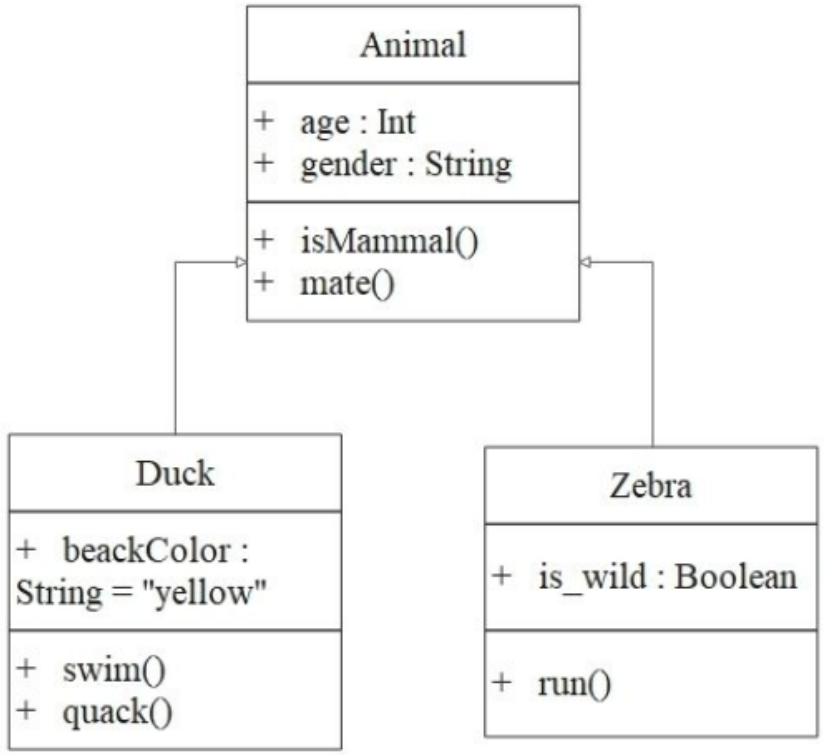
Association bidirectionnelle	 Une association bilatérale est représentée par une ligne droite reliant deux classes. Elle démontre simplement que les classes sont conscientes de leur relation l'une avec l'autre.

Association unilatérale



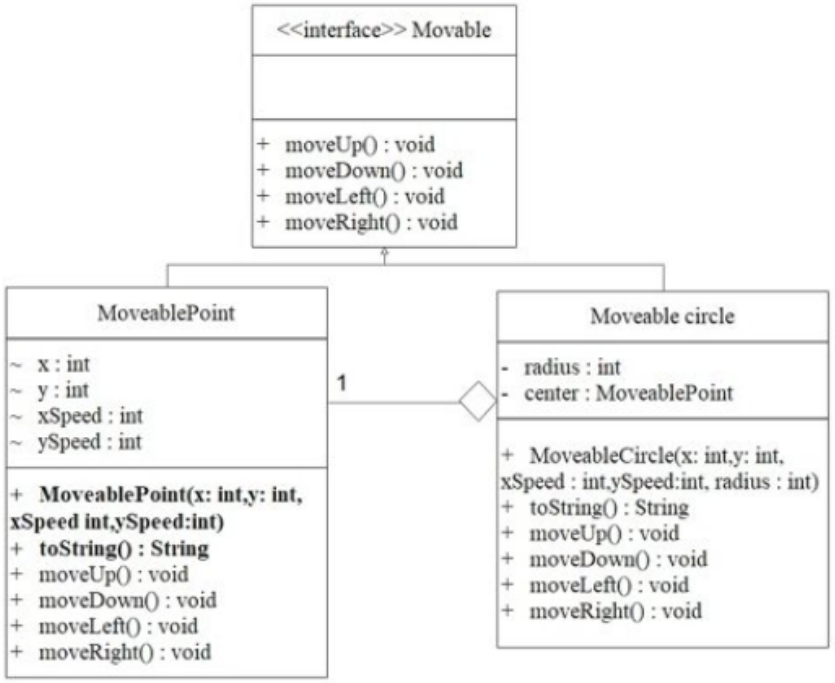
Une association unilatérale est représentée par une pointe de flèche ouverte reliant une classe à une autre. Elle montre qu'une classe est consciente de sa relation avec une autre classe.

Héritage

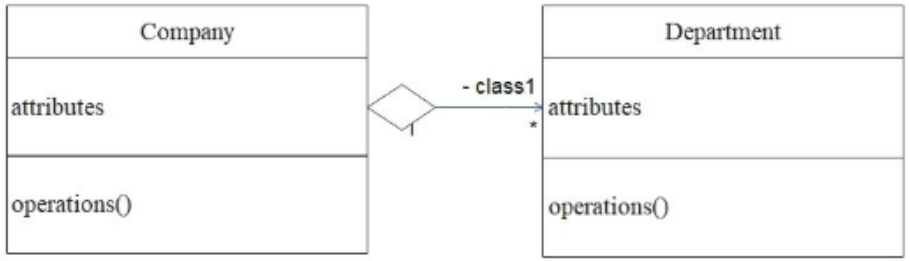


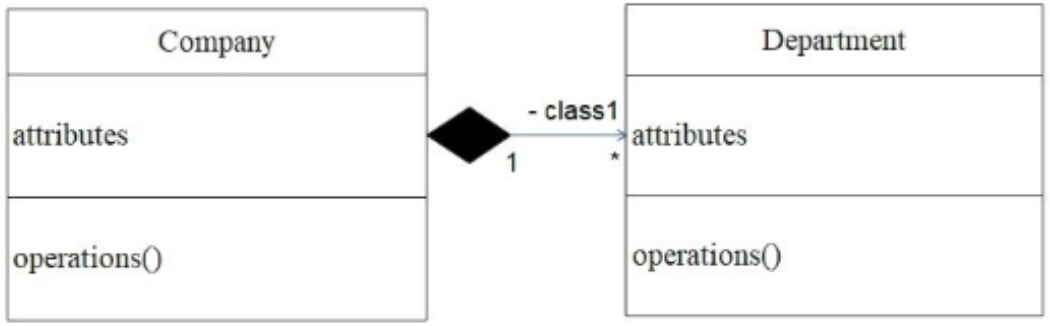
Indique une relation "enfant-parent" entre les classes. La classe enfant est une classe spécialisée, une sous-classe du parent.

Réalisation/Implémentation

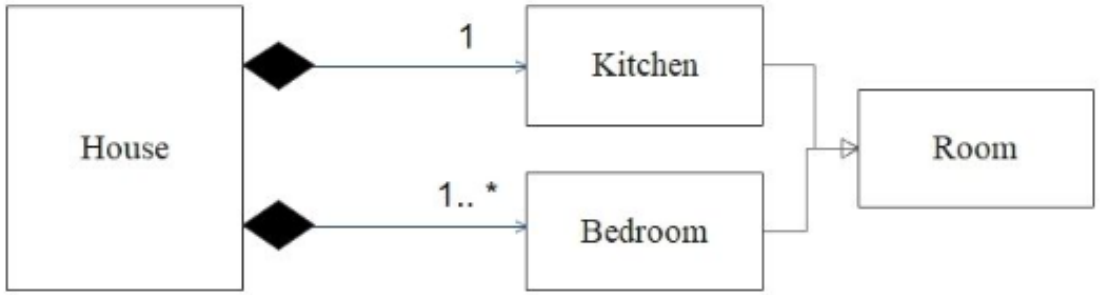


Une classe met en œuvre le comportement spécifié par une autre classe.

Dépendance	Comme son nom l'indique, une classe dépend d'une autre. C'est ce que montre une flèche en pointillés.
Aggrégation	 Cela représente une relation unilatérale entre les classes. Une classe fait partie d'une autre, ou lui est subordonnée. Dans ce cas, les classes enfant et parent peuvent exister indépendamment.

Composition	 il s'agit d'une forme d'agrégation où une classe est dépendante d'une autre. Une classe est une partie d'une autre. Dans ce cas, les classes enfants et les classes parents ne peuvent pas exister indépendamment.

Multiplicité



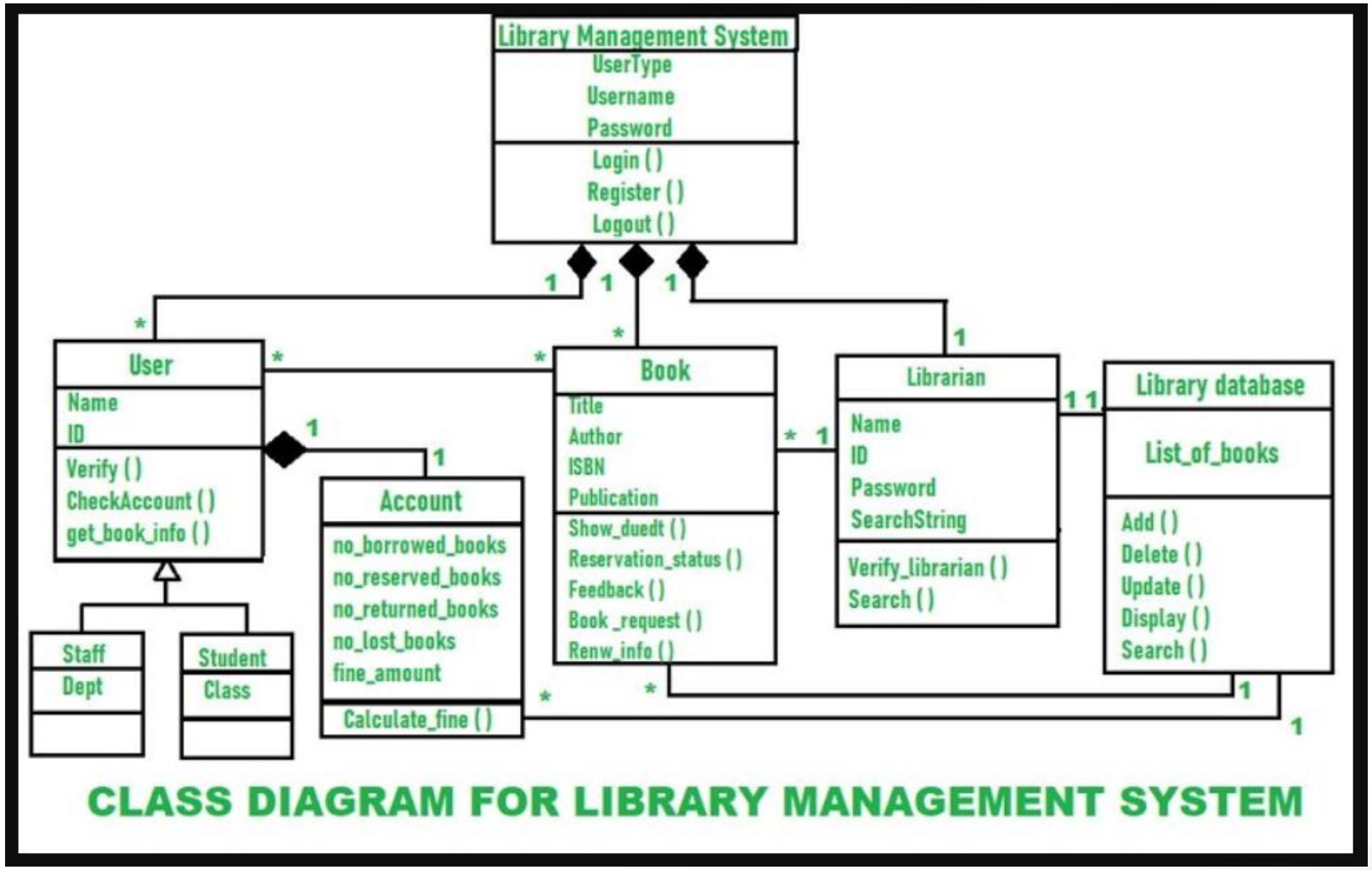
La multiplicité est utilisée pour déterminer combien de fois un attribut apparaît. Dans cet exemple, cette maison possède exactement une cuisine et au moins une chambre.

EXEMPLES

Diagramme de classe en UML.

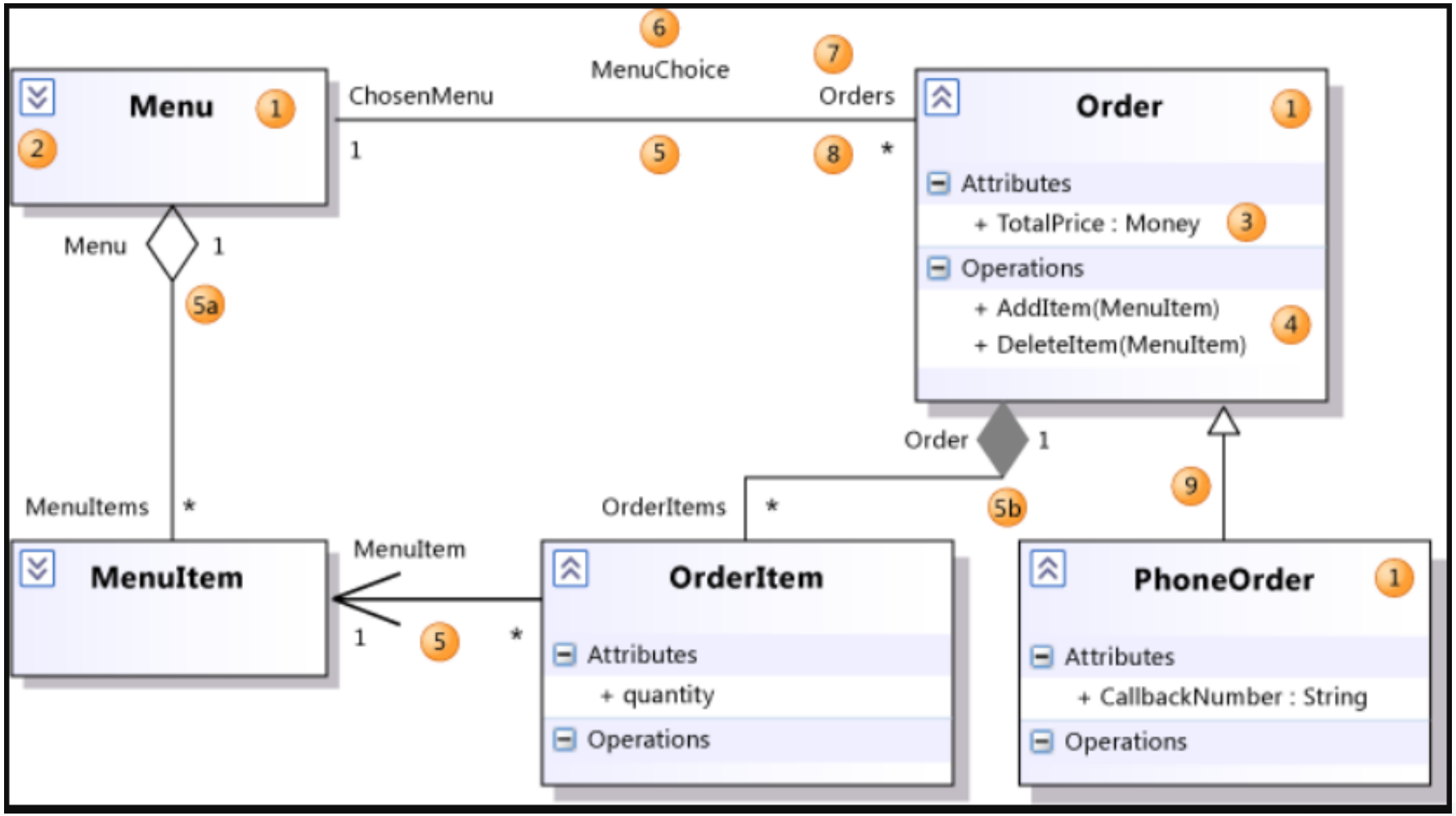
EXEMPLE 1

SAGIM



EXEMPLE 2

SAGIM



EXEMPLE 3

SAGIM