Langages et Modèles à Objets

Mourad Oussalah

Professeur à l'Université de Nantes

(Petite) Bibliographie

➜ Livres de cours et d'application du cours sur C++

- ➢ Claude Delannoy - Programmer en langage C++ - 4ème édition revue et augmentée - Eyrolles 1998
- ➢ Claude Delannoy - Exercices en langage C++ - Programmation orientée objet - 2ème édition revue et augmentée - Eyrolles 1996

➜ Livre de référence sur C++
(plus difficile d'abord mais complet - grammaire de C++)

- ➢ Bjarne Stroustrup - Le langage C++ - 2ème édition Int'l Thomson publishing - Traduction française - 1996

(Petite) Bibliographie

- Livres plus généraux sur les objets
  - Les langages à objets - Langages de classes, langages de frames, langages d'acteurs - G. Masini, A. Napoli, D. Colnet, D. Léonard, K. Tombre - InterEditions - Collection iia - 1989.
  - Les objets - M. Bouzeghoub, G. Gardarin, P. Valduriez - Eyrolles - 1997.
  - Ingénierie objet - Concepts et techniques - C. Oussalah et alii - InterEditions - 1997.

Un peu d'histoire : langage à objets

Simula I 1966, destiné à la simulation, => regroupement des données et procédures dans une même entité.

Simula 1973 => formalisation des concepts d'objets, de classes, de hiérarchie, d'encapsulation.

Ada 1983 => paquetages, même idées, mais sans héritage.

Simula a servi de modèle à toute une lignée de langages : C++ 1986, Object Pascal 1986, Eiffel 1990, Smalltalk.

Smalltalk : - Smalltalk-72 1976, communication par envoie de message, - Smalltalk-76, 1978, introduit l'héritage, - Smalltalk-80, 1983, langage et un environnement de programmation

Deux écoles à l'origine :

- Difficulté de concevoir des logiciels de tailles importantes a imposé la conception structurée.

=> Ecole représentée par ALGOL : Algol 60 (1), Algol 68 (2), Pascal (3).

(1) Communications of the ACM, 6(1):1-17, 1963.
(2) Groupe Algol de l'AFCET. "Manuel Langage algorithmique Algol 68. Hermann, Paris, 1975.
N. Wirth "The Programming Language PASCAL". Acta Informatica, 1(1):25-68, 1971.

- Représentation des connaissances en regroupant les propriétés correspondants à un même concept.

=> Ecole représentée par les Frames (3)

(3) M. Minsky "A Framework for Representing Knowledge". In "The Psychology of Computer Vision", pages 211-281, McGraw-Hill, New York, 1975.

Classe = Structure + Comportement

Structure = attributs
(variables de classes,
variables d'instances)

comportement = méthodes

Comportement => encapsulation des méthodes dans la classe

Deux types de classes :

- Abstraites (pas d'instances et comportement)

- Concrètes (avec instances et comportement)

Exemple : Gestion du stock d'un magasin

Article

/* numéro sous lequel l'article est répertorié au catalogue du magasin */
référence : entier
/* texte décrivant l'article */
désignation : texte
/* prix unitaire hors taxe de l'article */
prixHT : réel
/* nombre d'articles de la catégorie décrite disponible dans le stock */
quantité : entier

---

prixTTC () : réel
    retourner(1.206 * prixHT)
prixTransport () : réel
    retourner(0.05 * prixHT)
retirer (q)
    quantité = quantité - q
ajouter (q)
    quantité = quantité + q

Les classes

Une classe est la description d'une famille d'objets qui partagent la même structure et le même comportement.

Une classe = composante statique (données, champs, attributs) + composante dynamique (opérations, fonctions ou procédures, méthodes).

Nom de classe
attribut : simple attribut : type donnée attribut : type_donnée=valeur_init ...
opération opération (liste_arg) : type_retour ...

Instance

Une instance est un objet qui respecte les propriétés d'une classe. Elle est le résultat du moulage de la classe.

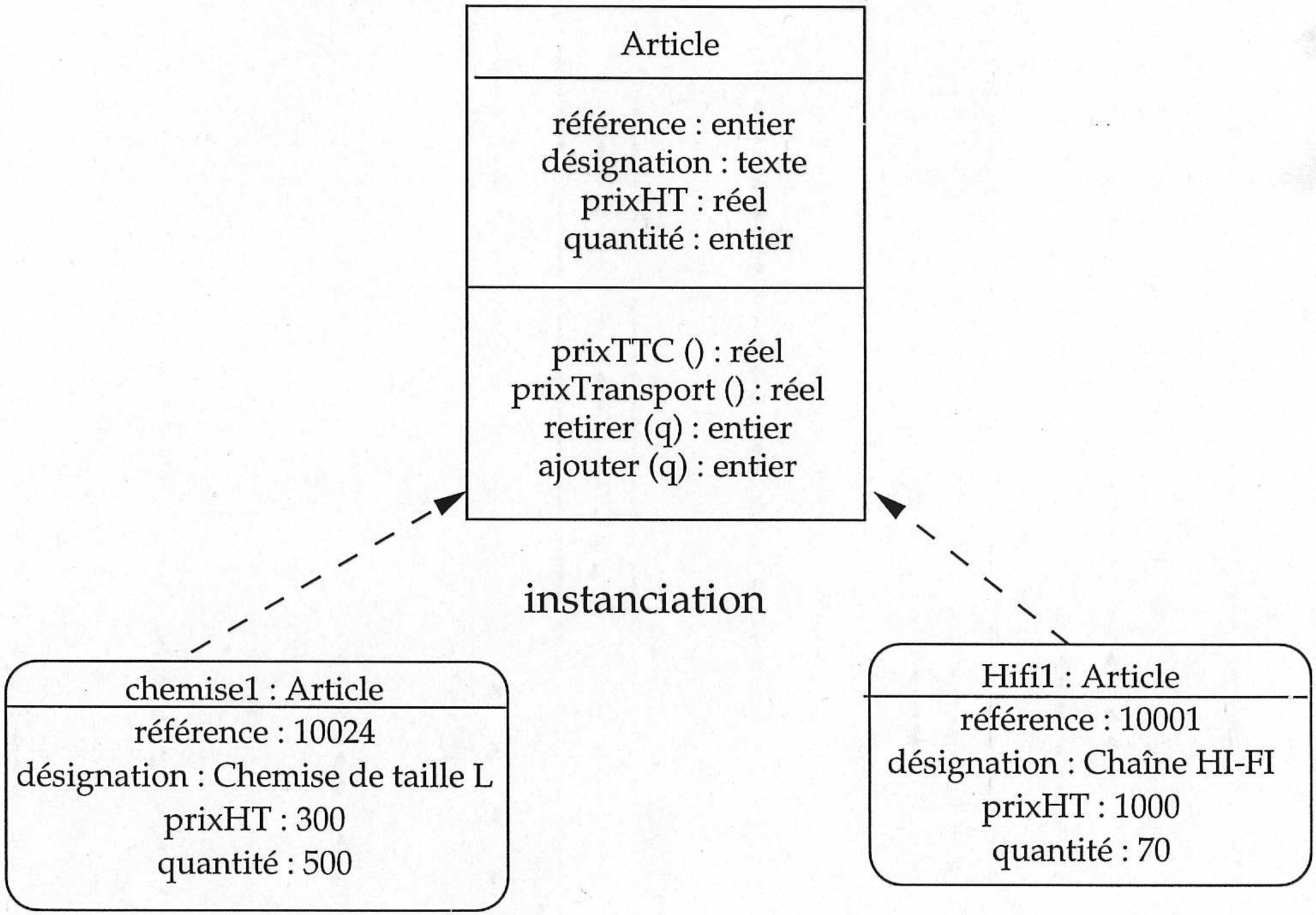
La classe est la représentation conceptuelle qui permet de construire un objet. Elle joue le rôle d'un moule.

Le moulage est partiel : une instance évalue les attributs d'une classe.

Cependant, les méthodes et un type d'attribut (variable de classe) ont des comportements communs pour l'ensemble des instances de la classe.

Deux types d'attributs : - attributs avec des comportements locaux (variables d'instance), - attributs avec des comportements partagés (variables de classes).

Instanciation de la classe Article



Communication par envoi de message

L'envoi de message (transmission) de message est le moyen de communication avec les objets.

Un objet est une entité indépendante dont la structure est connue de lui seul.

Il ne peut pas, en principe, agir sur un autre objet qu'à travers les méthodes appartenant à l'interface de cet autre objet.

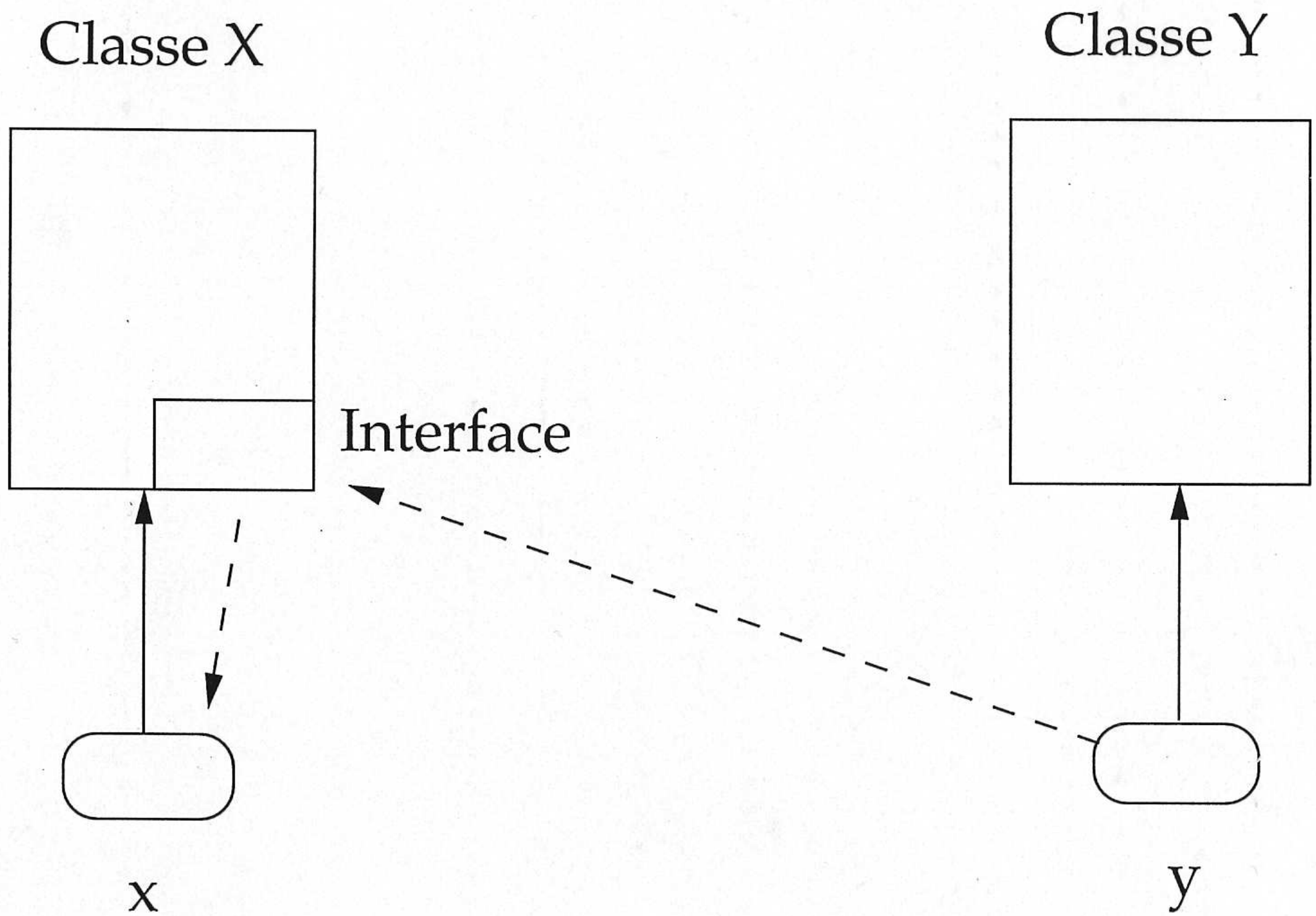
Agir sur une méthode appartenant à l'interface d'un objet revient à envoyer un message, qui demande l'exécution de la méthode en question.

Le message peut être vu comme une requête dont la satisfaction est à la charge de l'objet auquel elle est adressée.

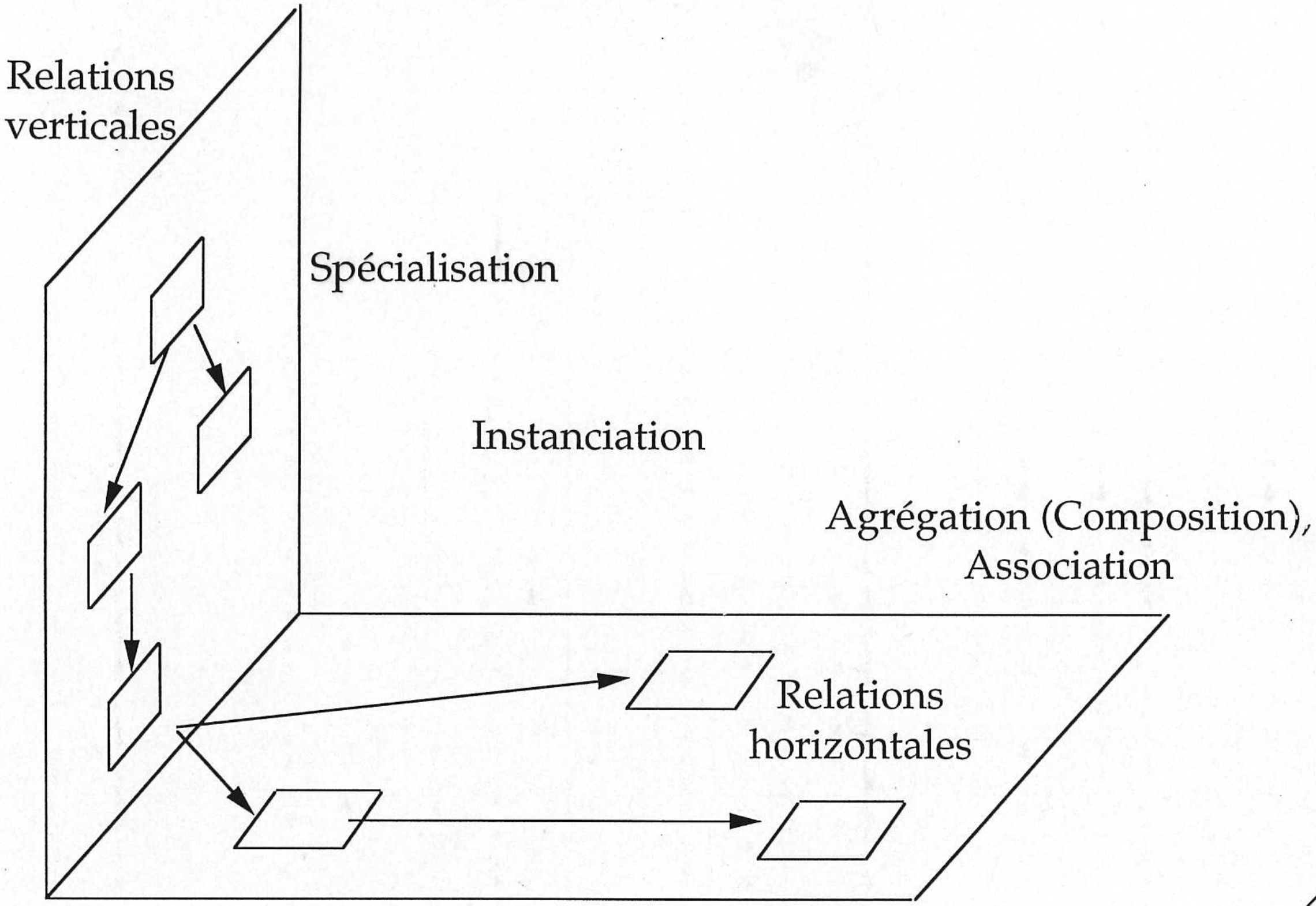
Communication par envoi de message

Message
=

receveur	sélecteur de méthode	arguments
	Exemples	
chemise231	prixTTC()	
télé43	prixTransport()	
braveheart	afficher	(coordonnées)


Classe X
Classe Y
Interface
x
y

Relations (Liens)

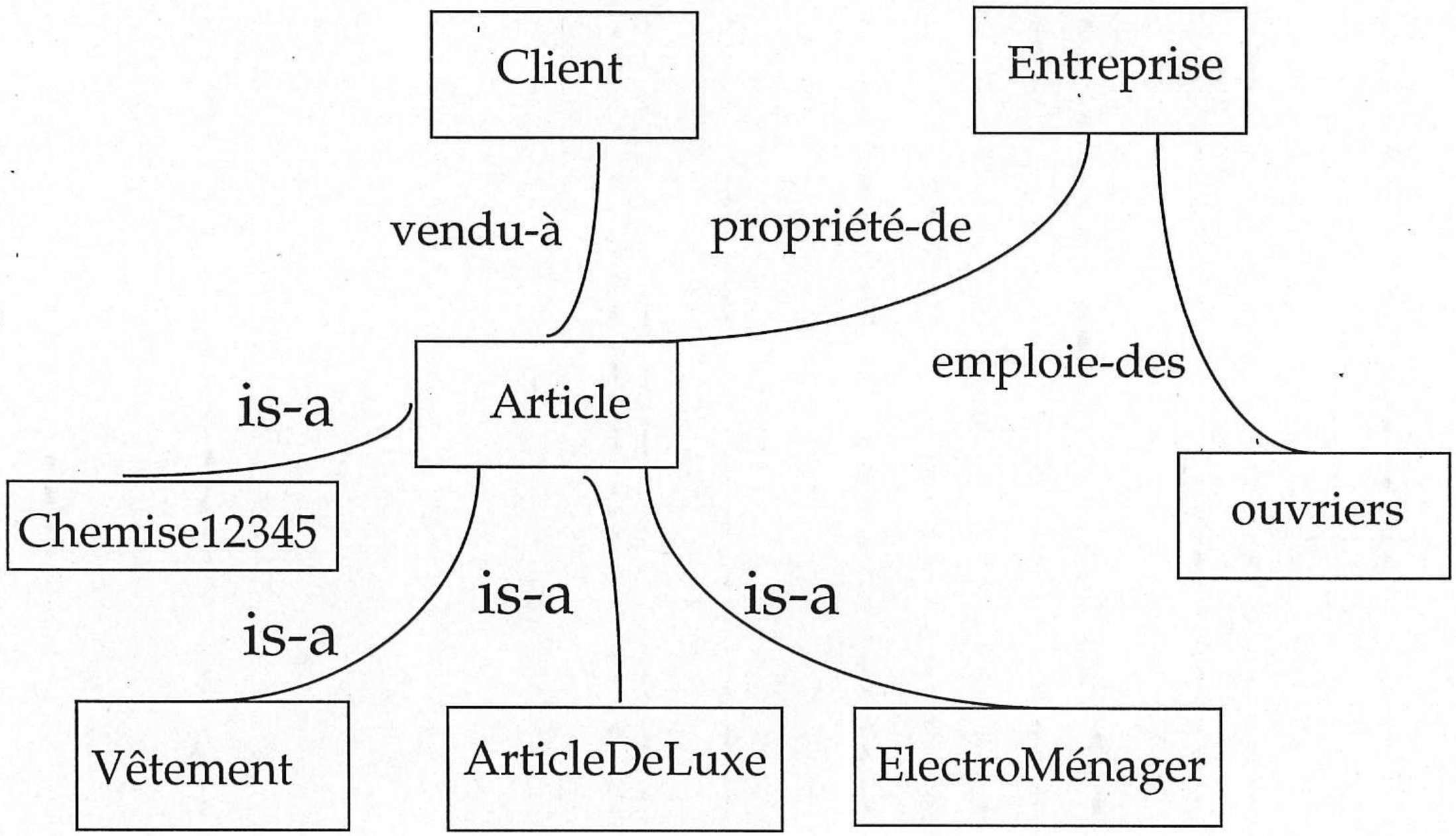


Relation verticale : Spécialisation (1)

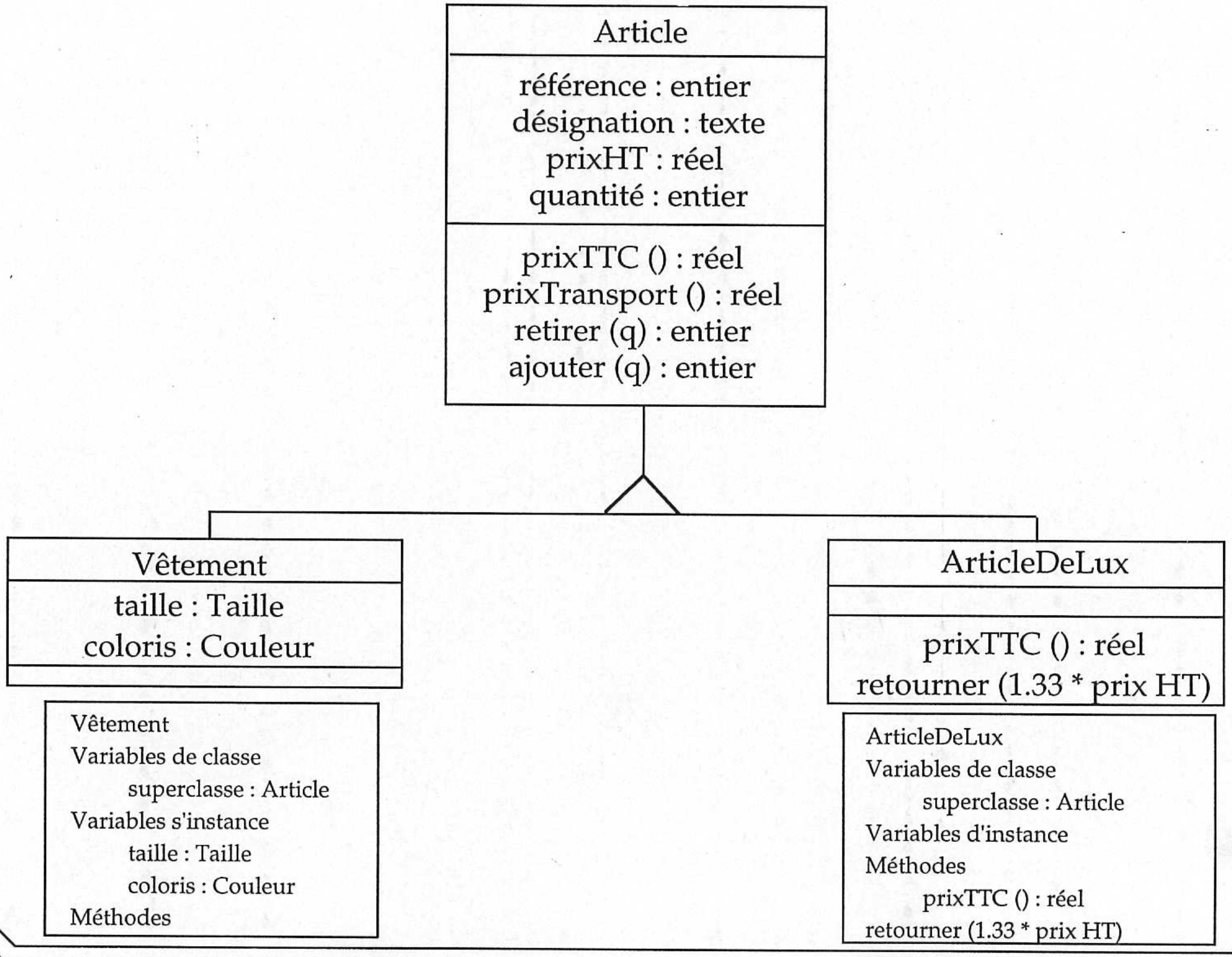
C'est un concept du modèle objet qui est à la base de la simplification de la réutilisation et de l'extensibilité des logiciel,
il permet de construire des classes d'objets au niveau le plus bas de la hiérarchie sans reconcevoir entièrement les structures et les comportements qui existent déjà dans les classes.

La spécialisation trouve ses racines en Intelligence Artificielle. Ross et Quillian 1968 ont développé un modèle psychologique de la mémoire associative. Il avait introduit l'un des premiers modèles de réseaux sémantiques. Un réseau sémantique est composé de noeuds et d'arcs. Chaque arc a une sémantique particulière, par exemple l'héritage "is-a".

Exemple de réseau sémantique



Exemples


Article
référence : entier
désignation : texte
prixHT : réel
quantité : entier
prixTTC () : réel
prixTransport () : réel
retirer (q) : entier
ajouter (q) : entier
Vêtement
taille : Taille
coloris : Couleur
Vêtement
Variables de classe
superclasse : Article
Variables s'instance
taille : Taille
coloris : Couleur
Méthodes
ArticleDeLux
prixTTC () : réel
retourner (1.33 * prix HT)
ArticleDeLux
Variables de classe
superclasse : Article
Variables d'instance
Méthodes
prixTTC () : réel
retourner (1.33 * prix HT)

Relation verticale : Spécialisation (2)

Spécialisation = Héritage + Inclusion

Relation d'**Héritage** entre une classe X et une classe Y

<=>

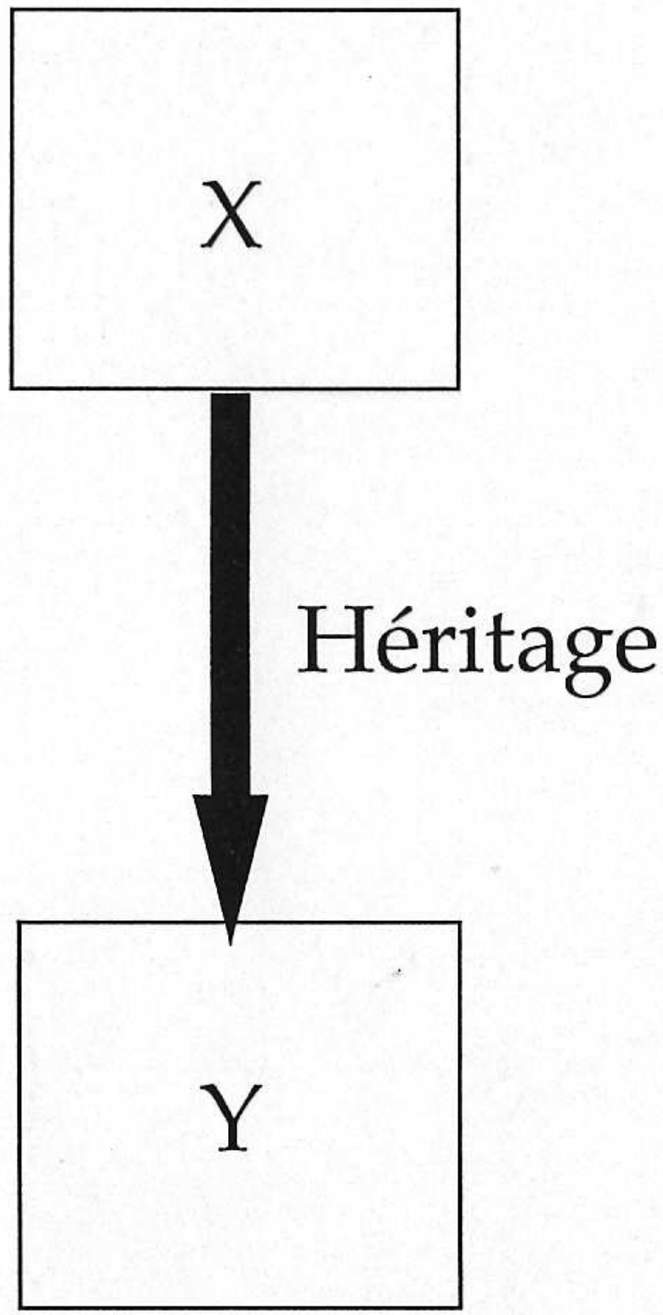
Diffusion des propriétés (structure + comportements) de X vers Y

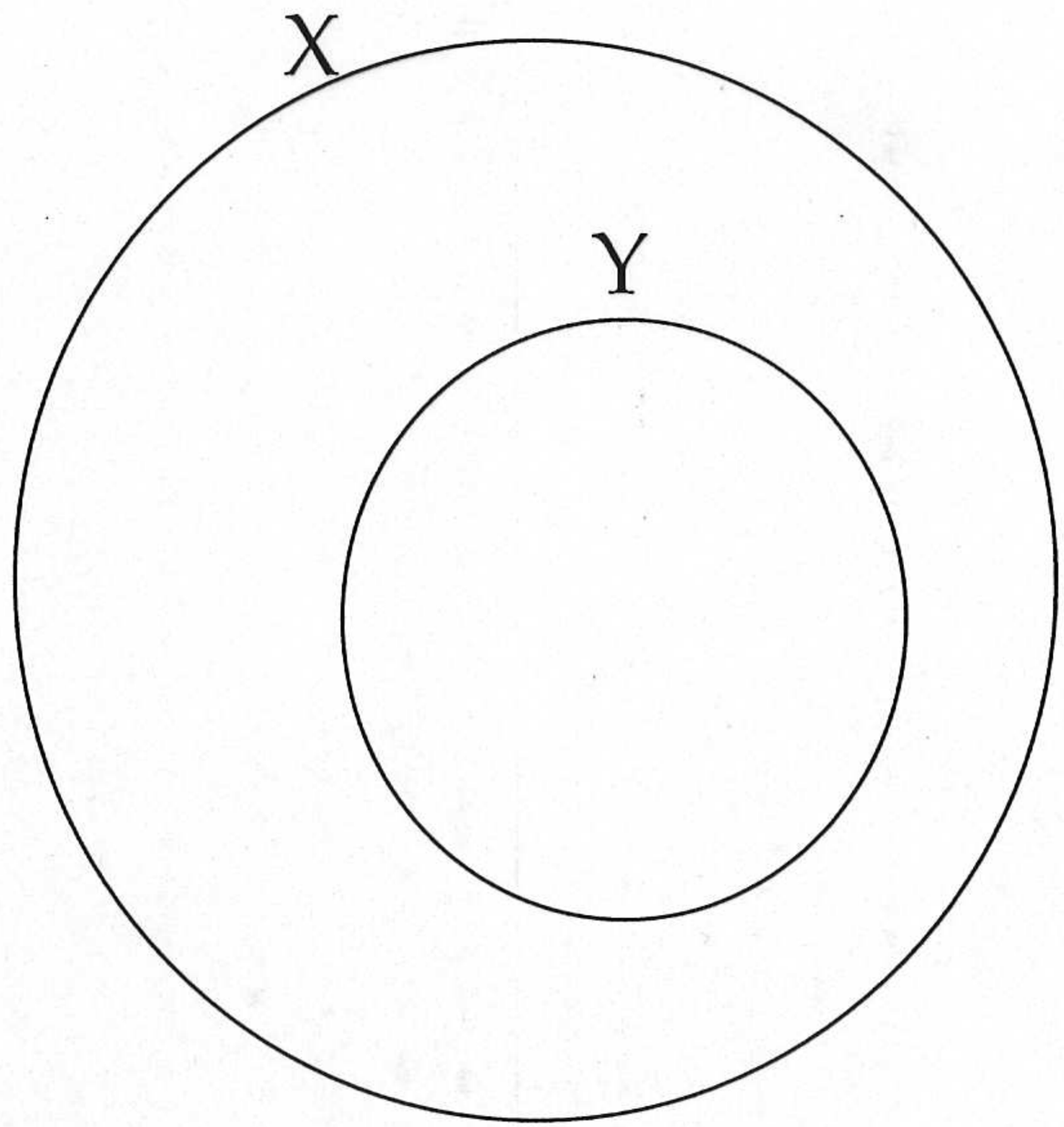
Y peut affiner les propriétés héritées de X en :
- ajoutant de nouvelles propriétés,
- affinant les domaines des attributs de X,
- en surchargeant les méthodes.

Relation d'**inclusion** entre une X et une classe Y

<=>

quelque soit y appartenant à Y, y appartient aussi à X
on dit que : Y inclus dans X, ou que Y est une sous-classe de X



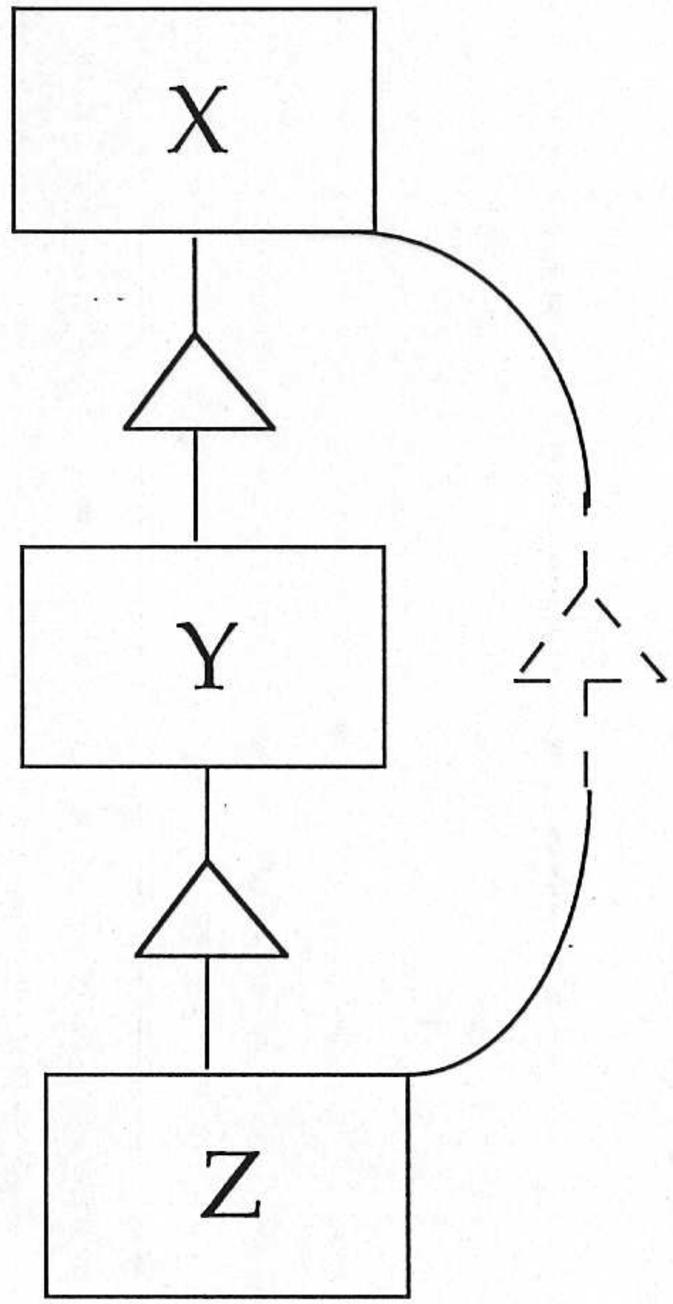


Inclusion

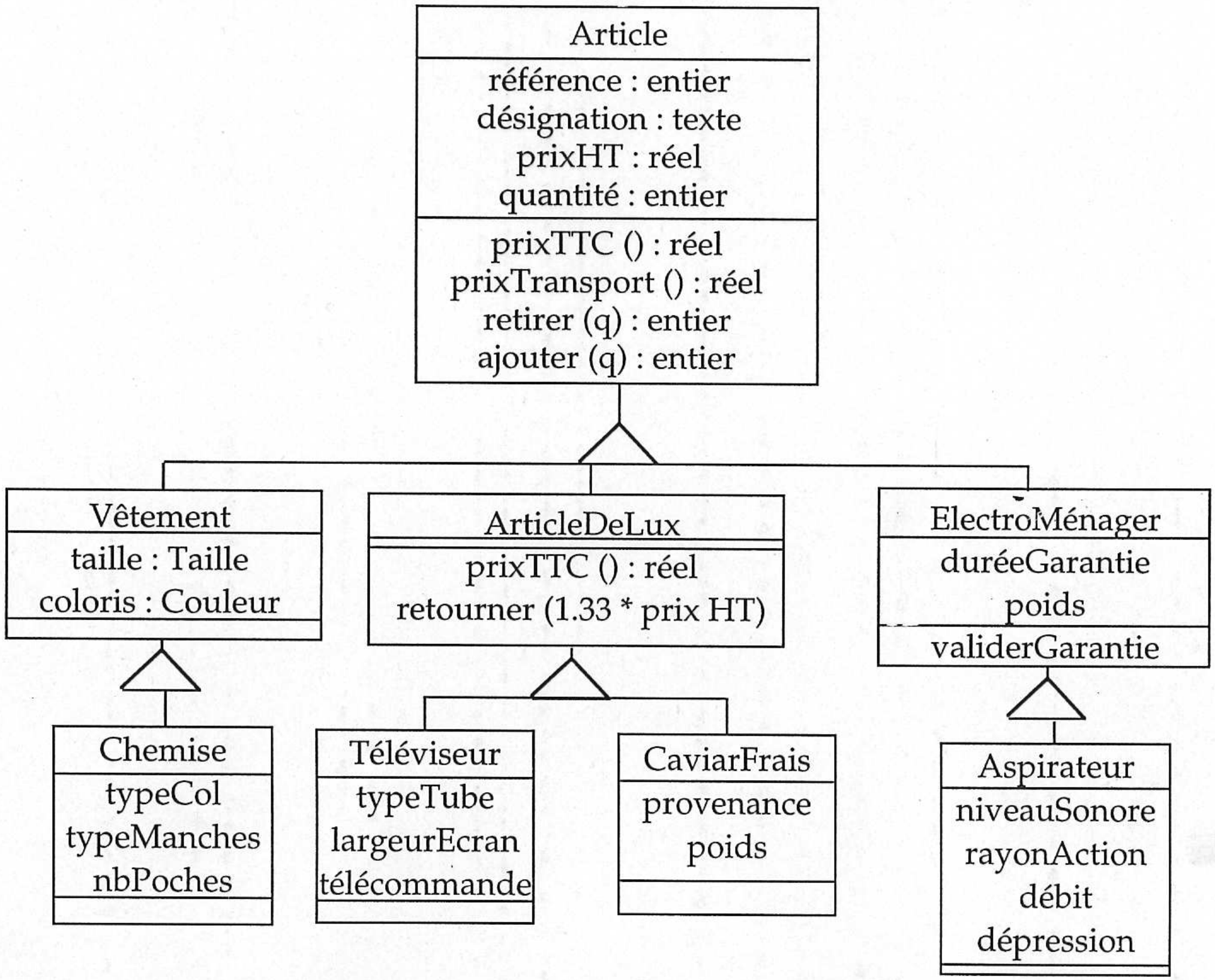
Relation de spécialisation est :

- Transitive

- Anti-symétrique



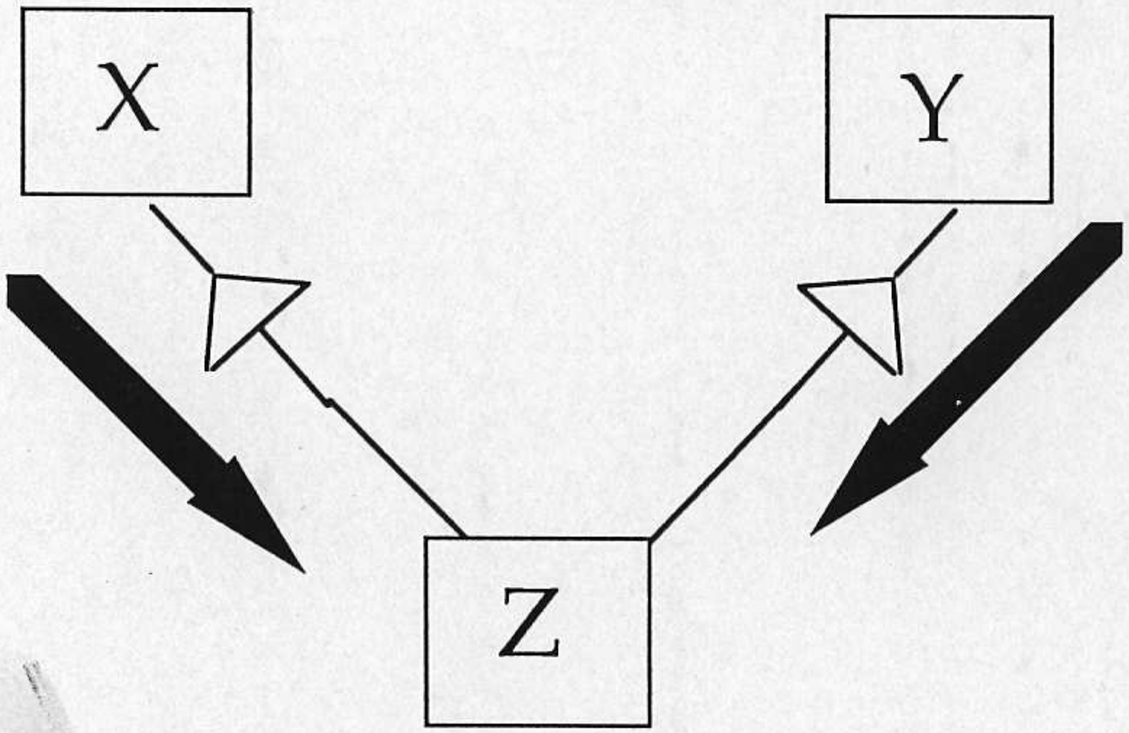
Exemple : Graphe de spécialisation



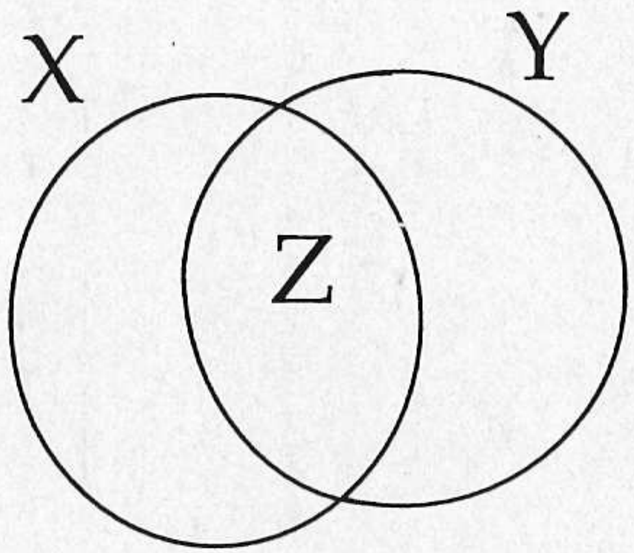
Spécialisation multiple

Spécialisation **simple** : une classe possède une superclasse unique, sauf pour la racine. => Arbre de spécialisation

Spécialisation **multiple** : une classe possède au moins une super-classe. => Graphe de spécialisation orienté sans circuit, appelé abusivement quelquefois "treillis d'héritage multiple"

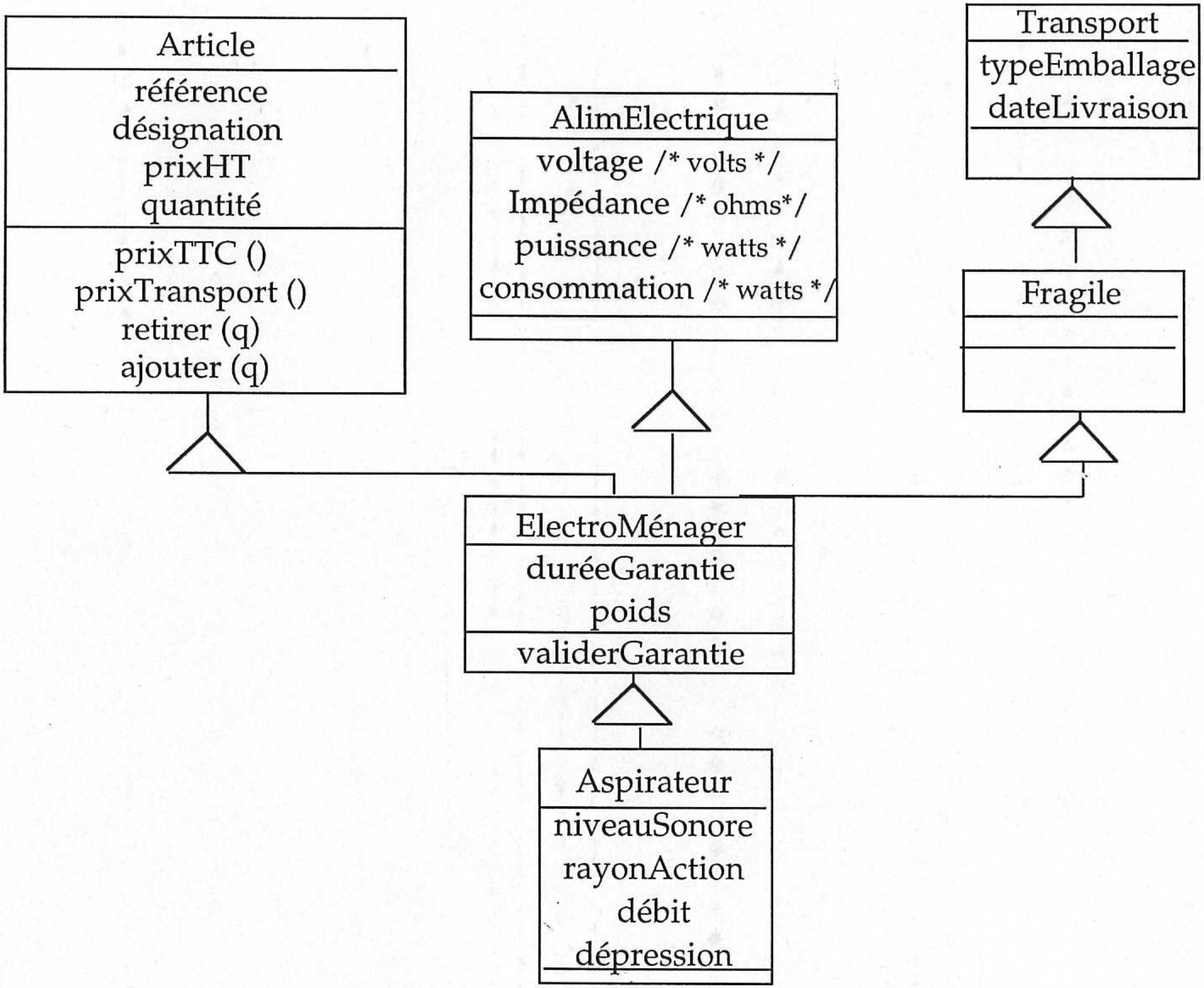


=>

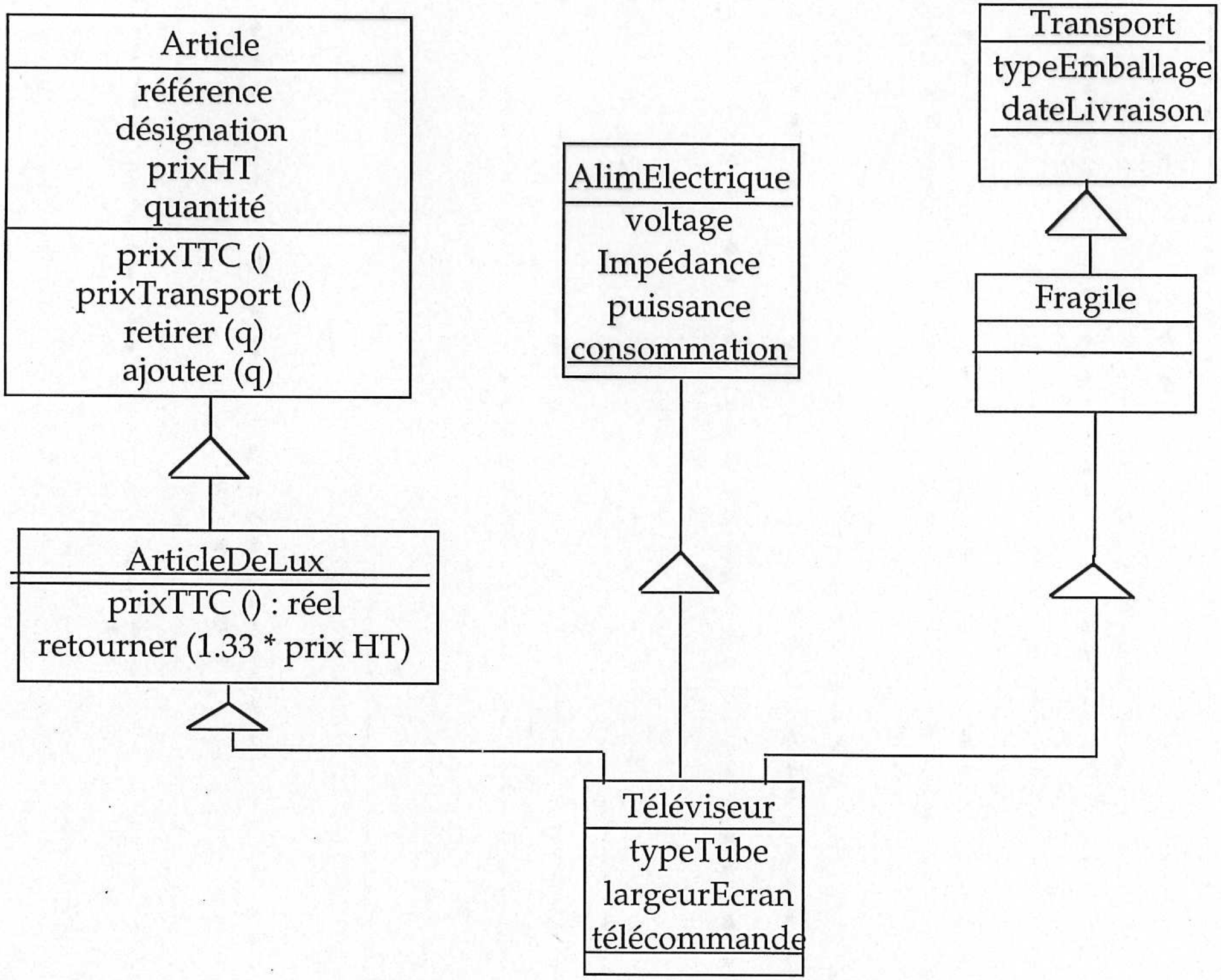


Avantage : accroissement de la modularité, réutilisation

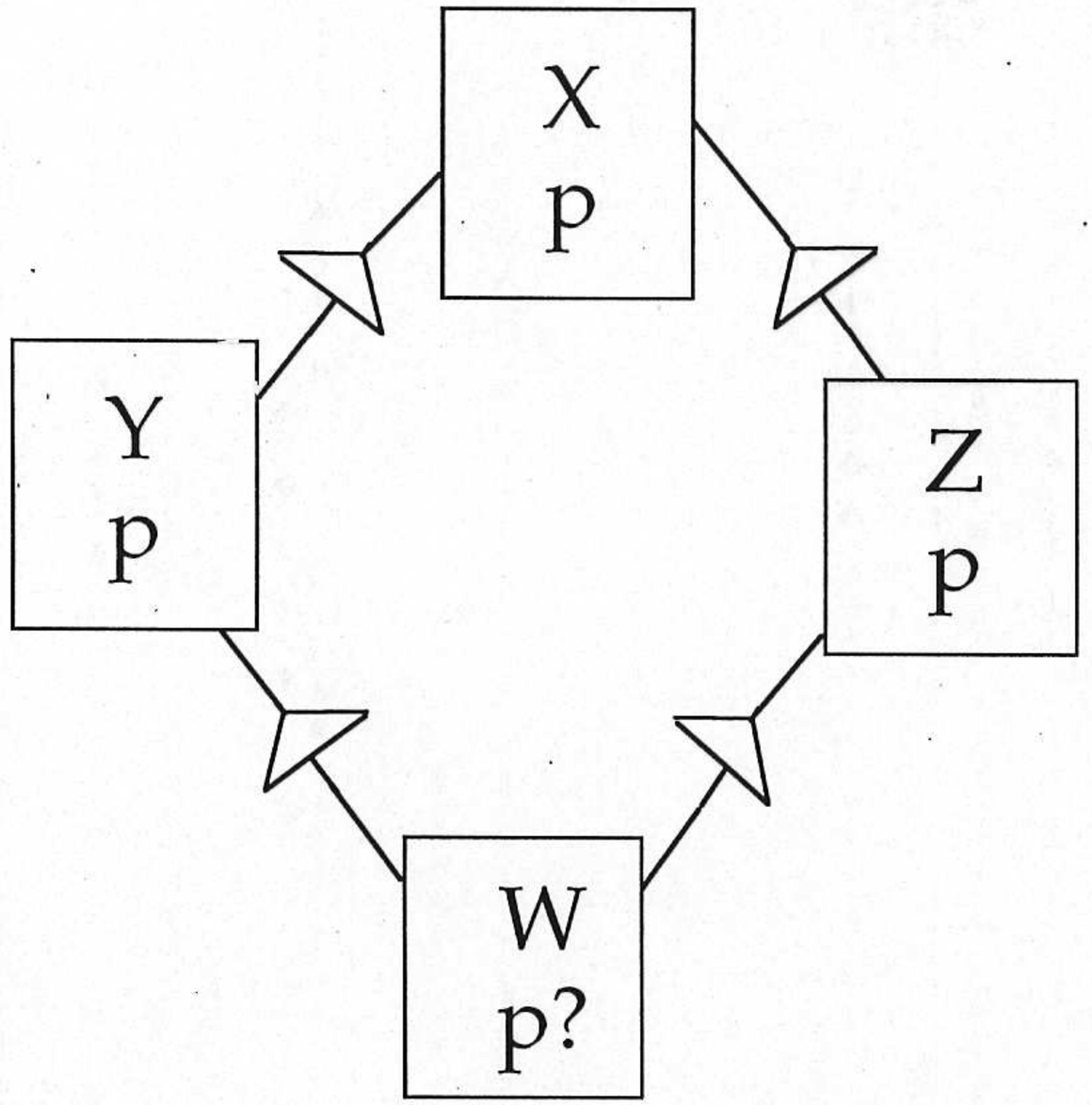
Exemples (1)

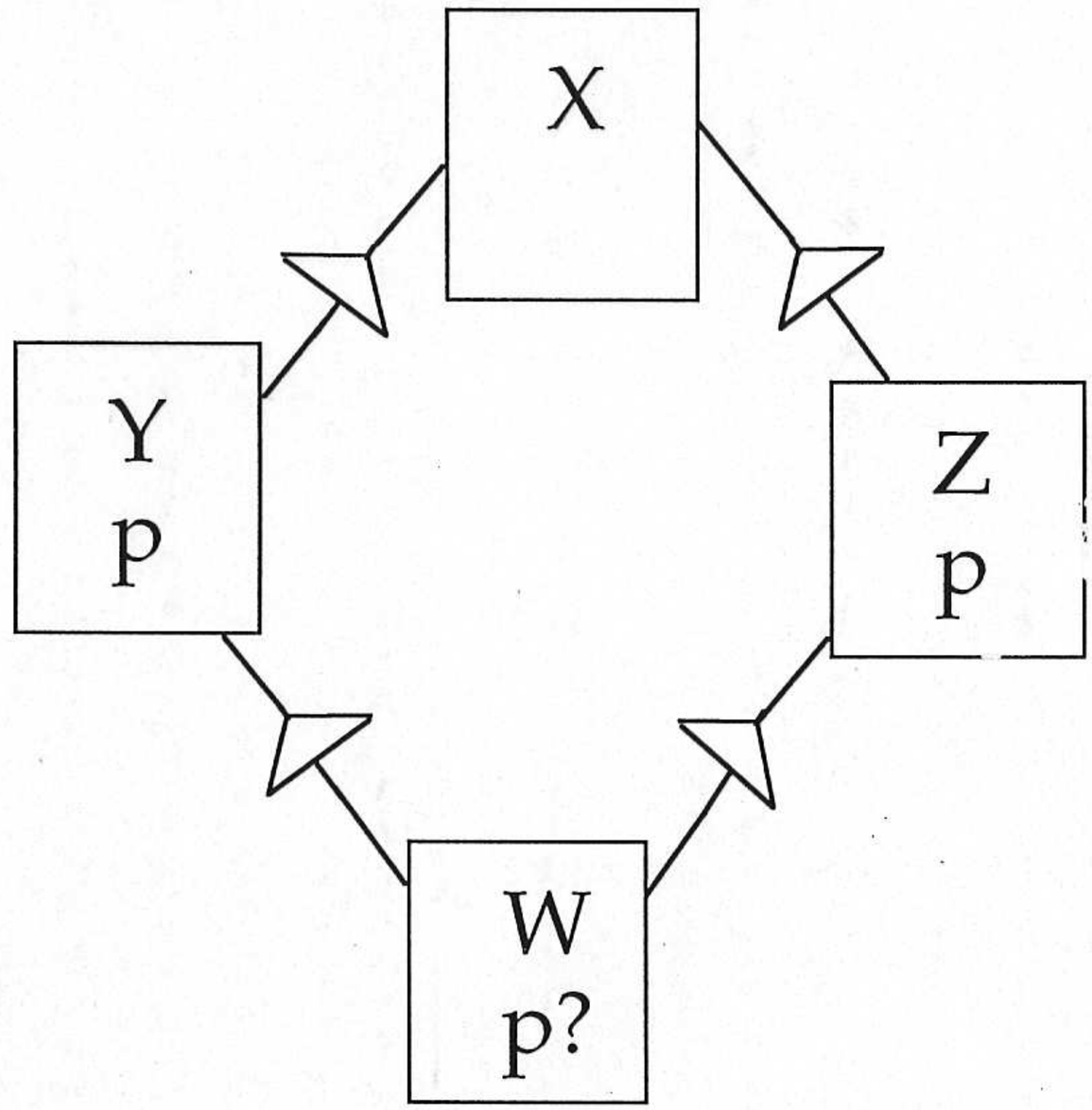

Article
référence
désignation
prixHT
quantité
prixTTC ()
prixTransport ()
retirer (q)
ajouter (q)
AlimElectrique
voltage /* volts */
Impédance /* ohms*/
puissance /* watts */
consommation /* watts */
Transport
typeEmballage
dateLivraison
Fragile
ElectroMénager
duréeGarantie
poids
validerGarantie
Aspirateur
niveauSonore
rayonAction
débit
dépression

Exemples (2)

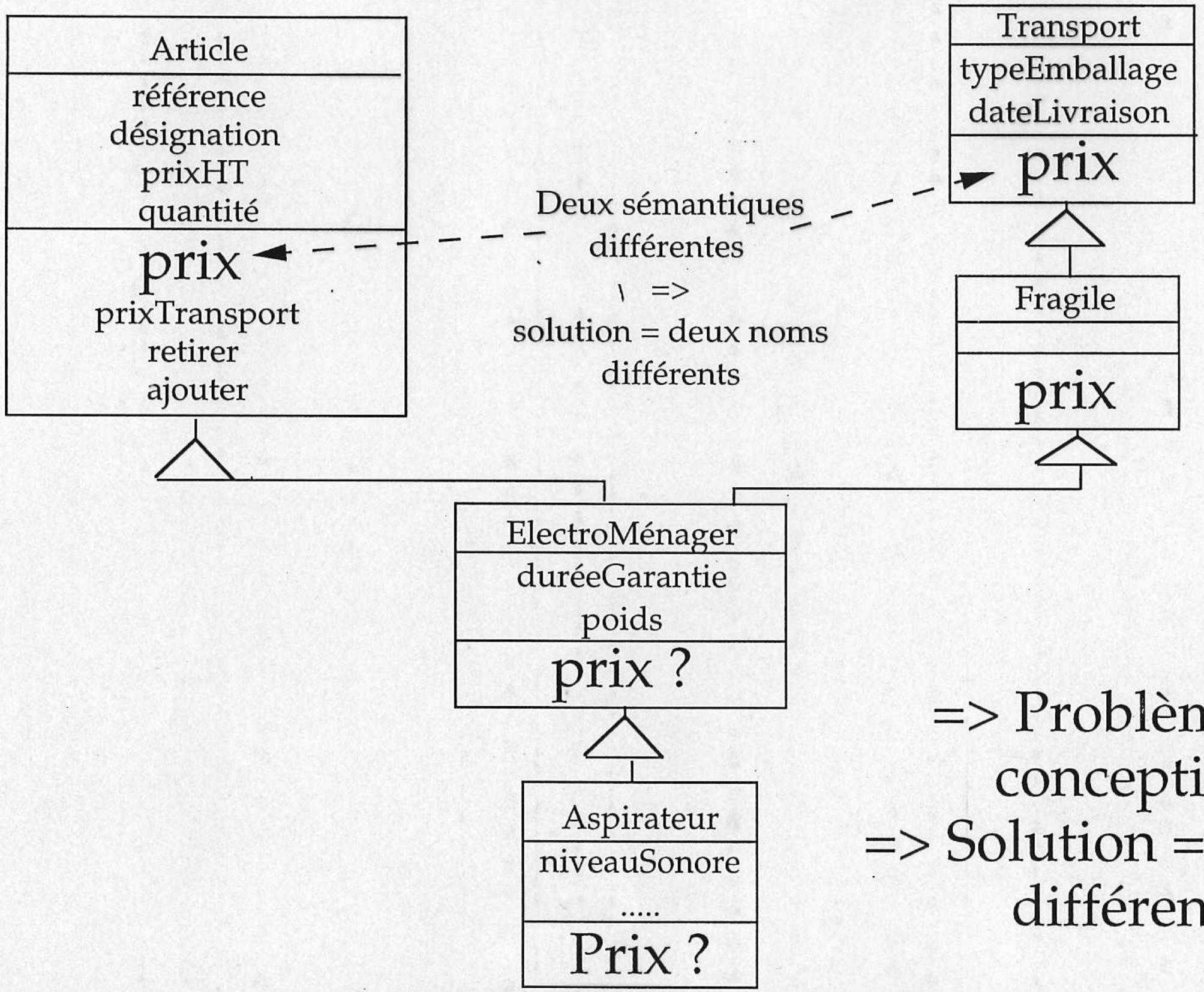

Article
référence
désignation
prixHT
quantité
prixTTC ()
prixTransport ()
retirer (q)
ajouter (q)
AlimElectrique
voltage
Impédance
puissance
consommation
Transport
typeEmballage
dateLivraison
Fragile
ArticleDeLux
prixTTC () : réel
retourner (1.33 * prix HT)
Téléviseur
typeTube
largeurEcran
télécommande

Spécialisation multiple : les conflits





Exemples (1)



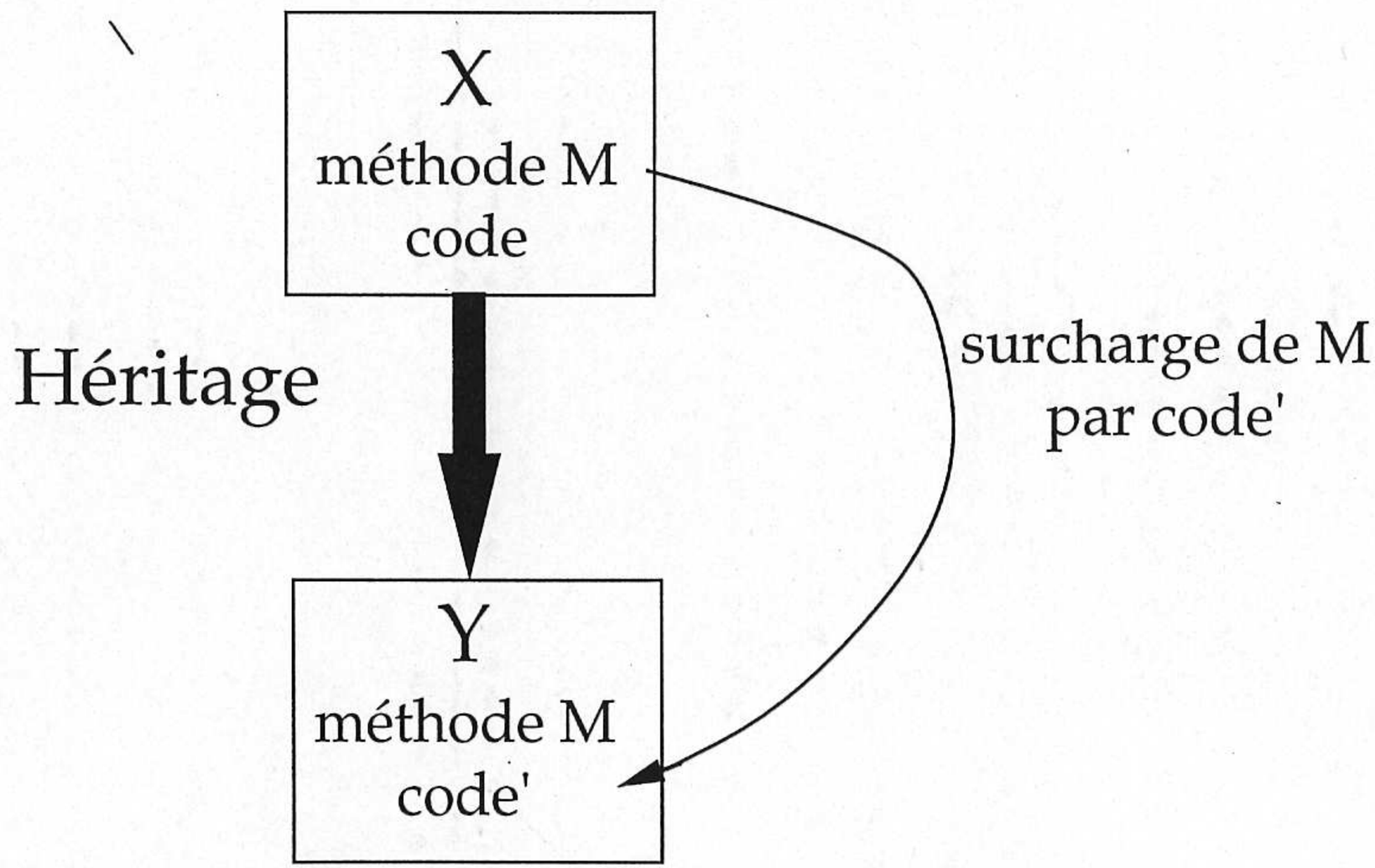
=> Problème de conception

=> Solution = 2 noms différents

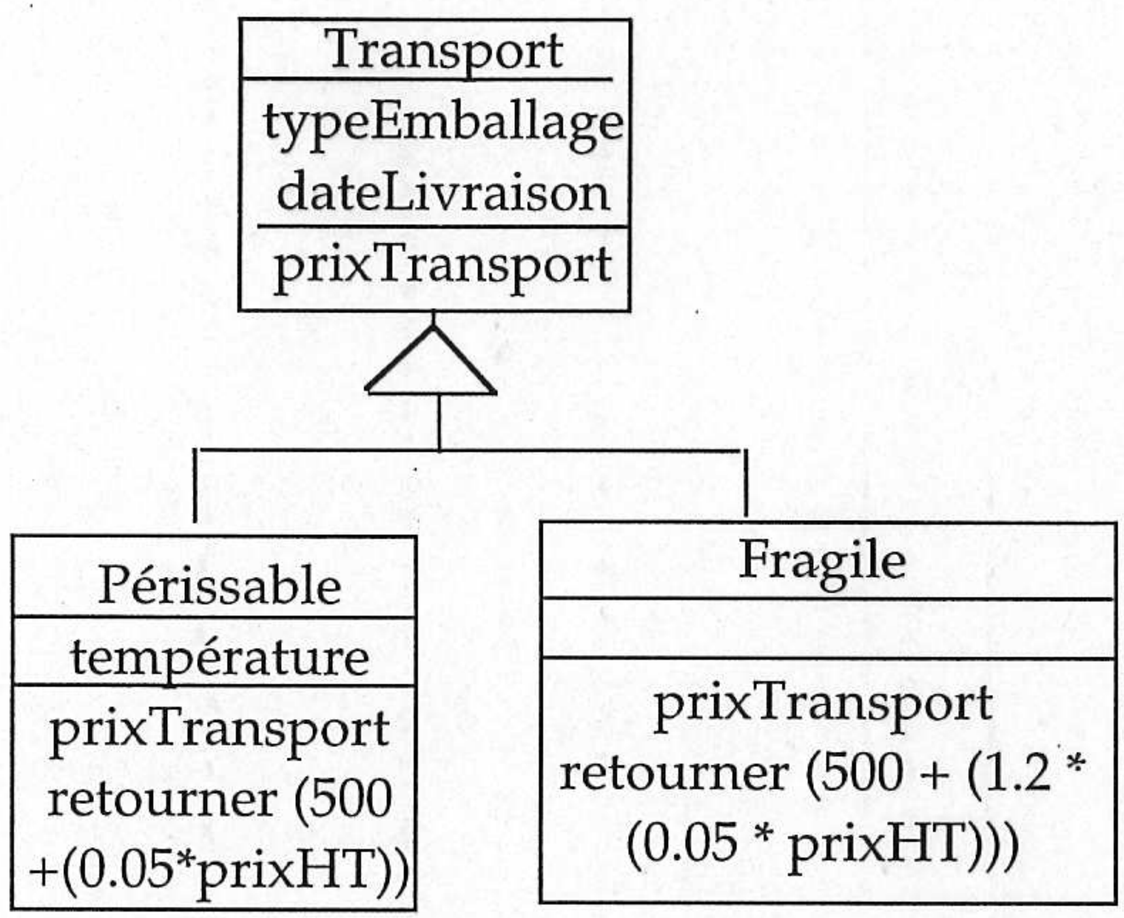
Surcharge

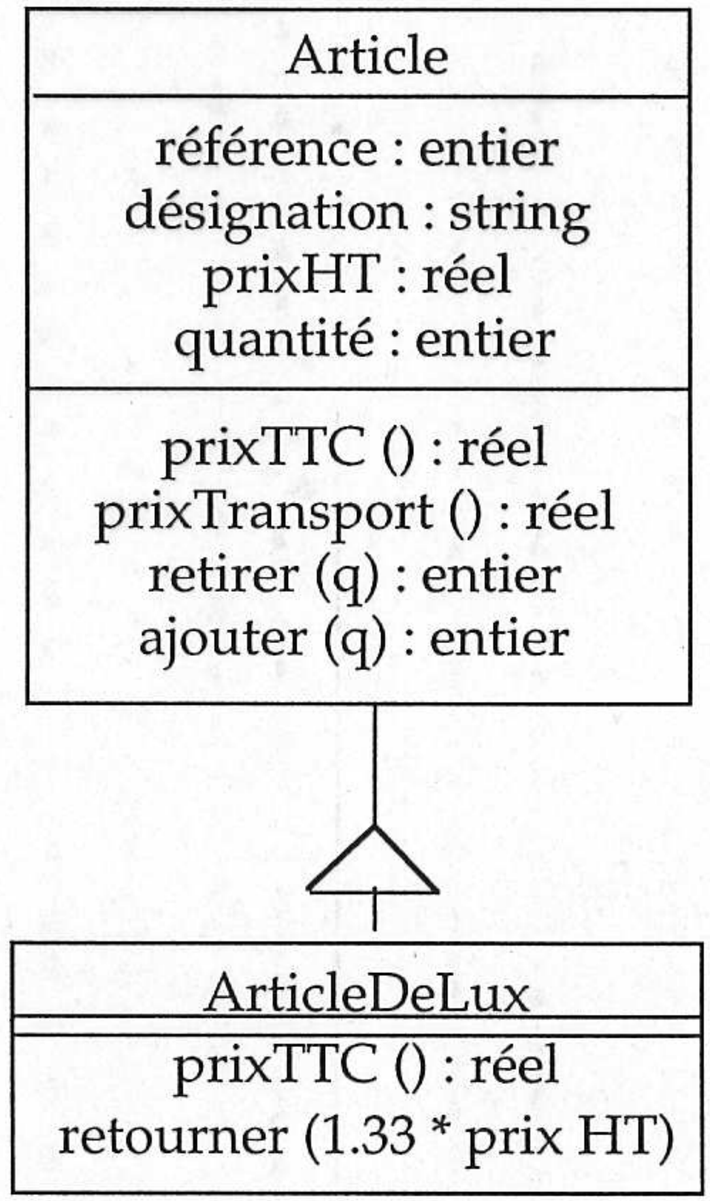
Surcharge = modification du code d'une méthode héritée.
permet des opérations de même noms avec des sémantiques et des implantations différentes pour des types d'objets différents

=> Extensibilité



Exemples


Transport
typeEmballage
dateLivraison
prixTransport
Périssable
température
prixTransport
retourner (500 +(0.05*prixHT))
Fragile
prixTransport
retourner (500 + (1.2 * (0.05 * prixHT)))


Article
référence : entier
désignation : string
prixHT : réel
quantité : entier
prixTTC () : réel
prixTransport () : réel
retirer (q) : entier
ajouter (q) : entier
ArticleDeLux
prixTTC () : réel
retourner (1.33 * prix HT)

Liaison dynamique
(dynamic binding ou late binding)

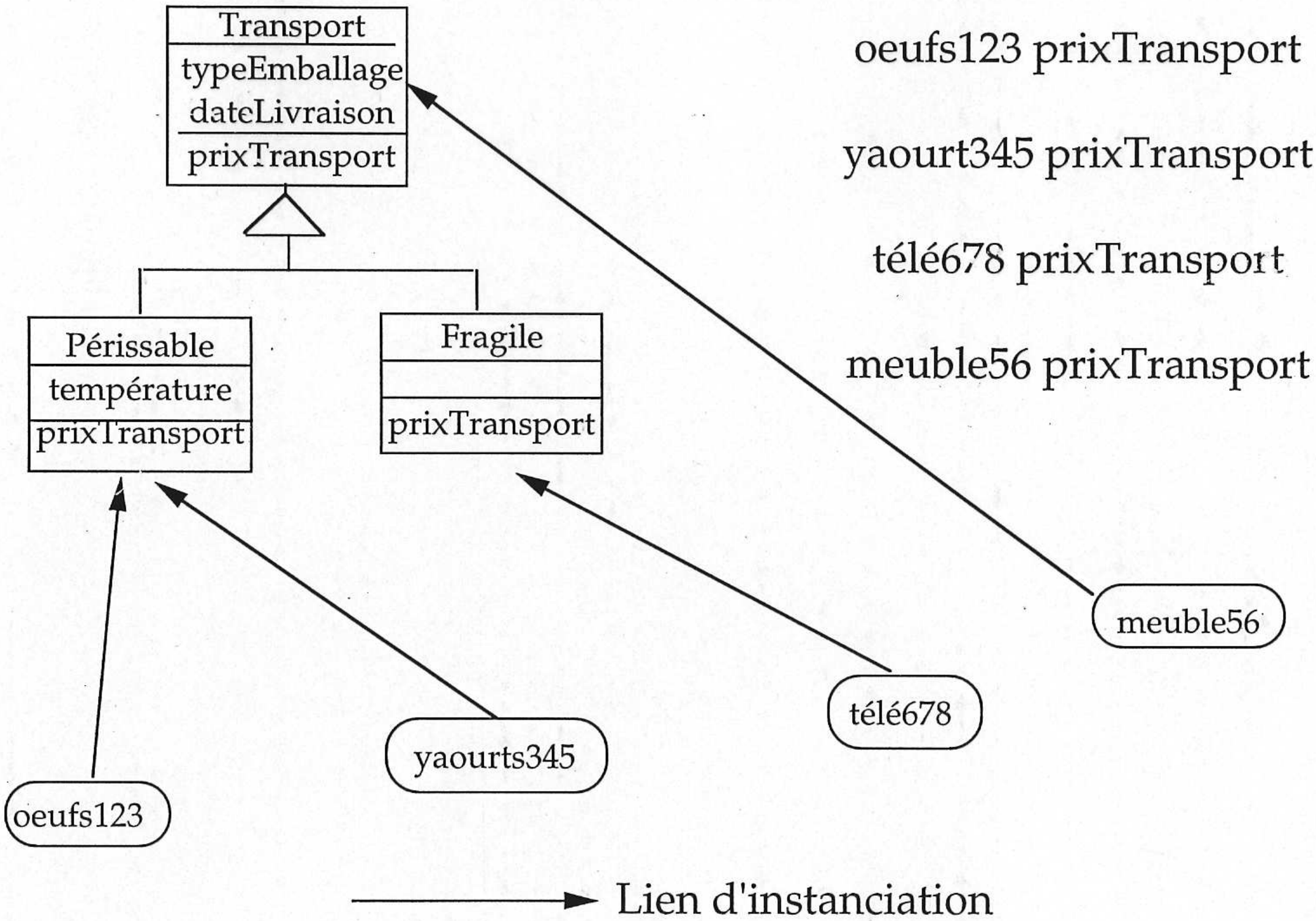
La liaison dynamique est étroitement liée à la surcharge, elle constitue avec la surcharge, l'un des concepts les plus puissants et les plus fréquemment utilisés du modèle à objets.

Liaison dynamique signifie que le ~~le~~ système met en liaison les sélecteurs de messages et les méthodes qui les implantent lors de l'exécution.

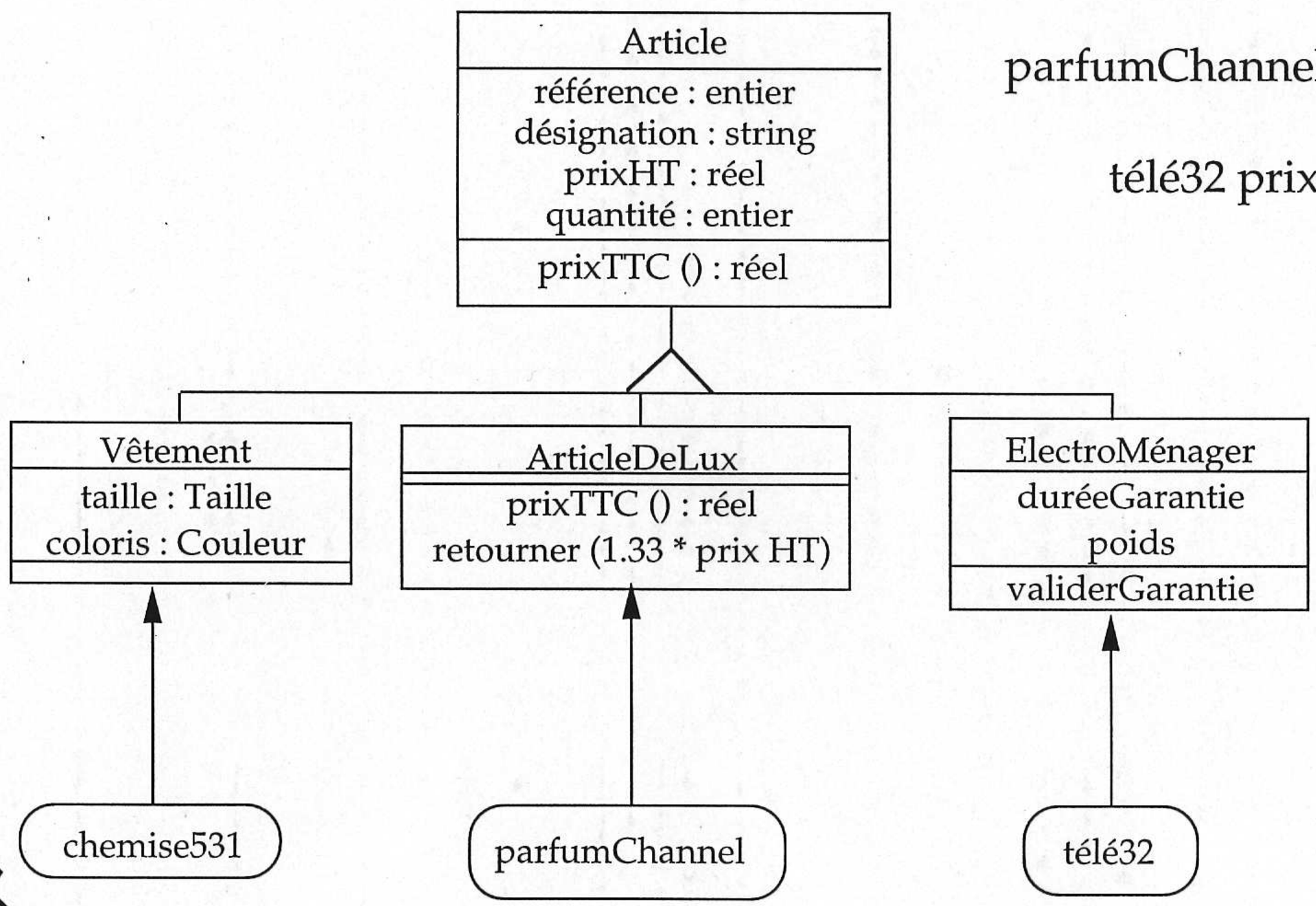
=> Une méthode se comportera différemment suivant le type d'objets qui recevra le message.

=> Extensibilité du modèle à objets qui ne remet pas en cause les programmes développés sur la base de ce modèle.

Exemples (1)


Transport
typeEmballage
dateLivraison
prixTransport
Périssable
température
prixTransport
Fragile
prixTransport
oeufs123 prixTransport
yaourt345 prixTransport
télé678 prixTransport
meuble56 prixTransport
meuble56
télé678
yaourts345
oeufs123
Lien d'instanciation

Exemples (2)

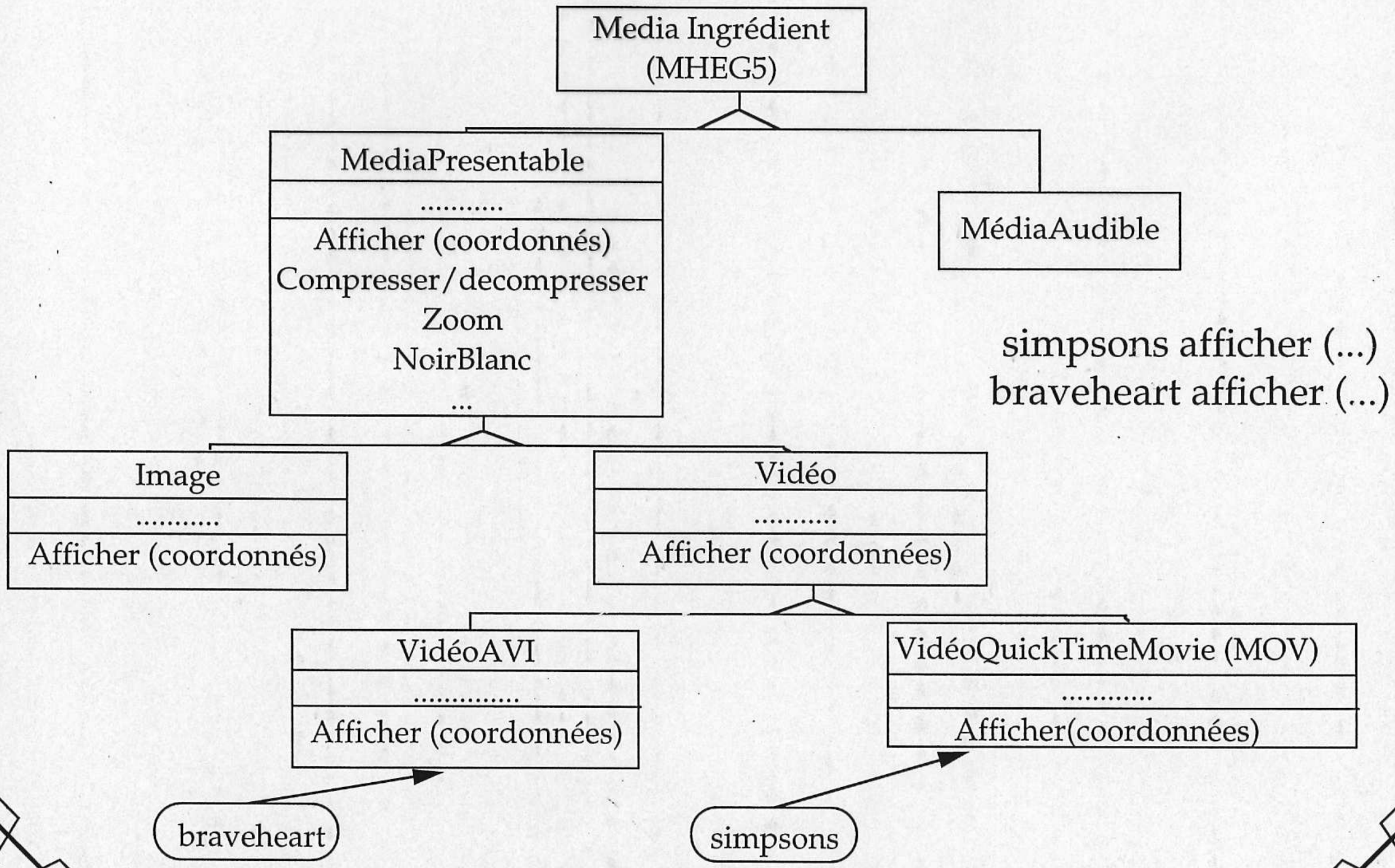


chemise531 prixTTC

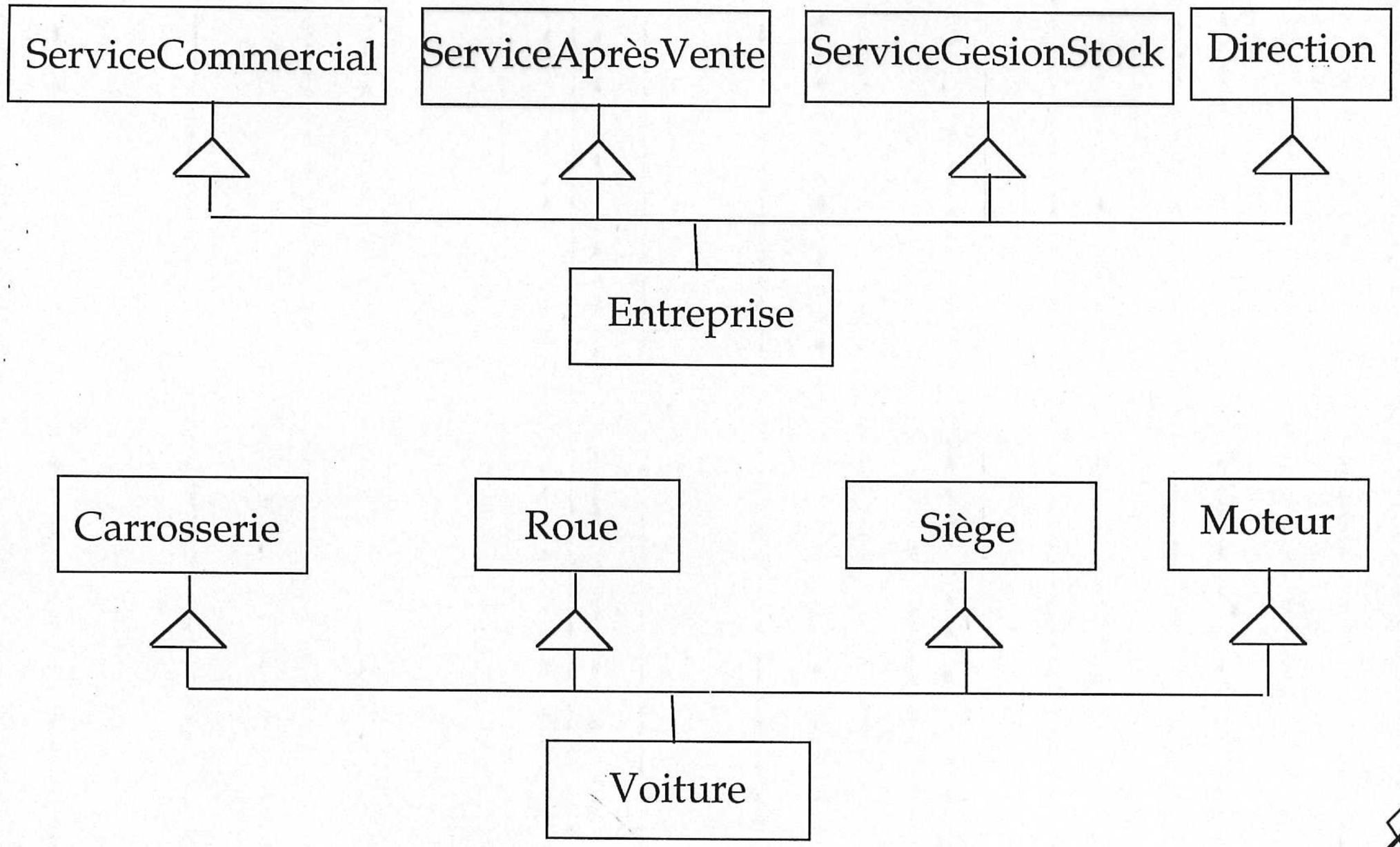
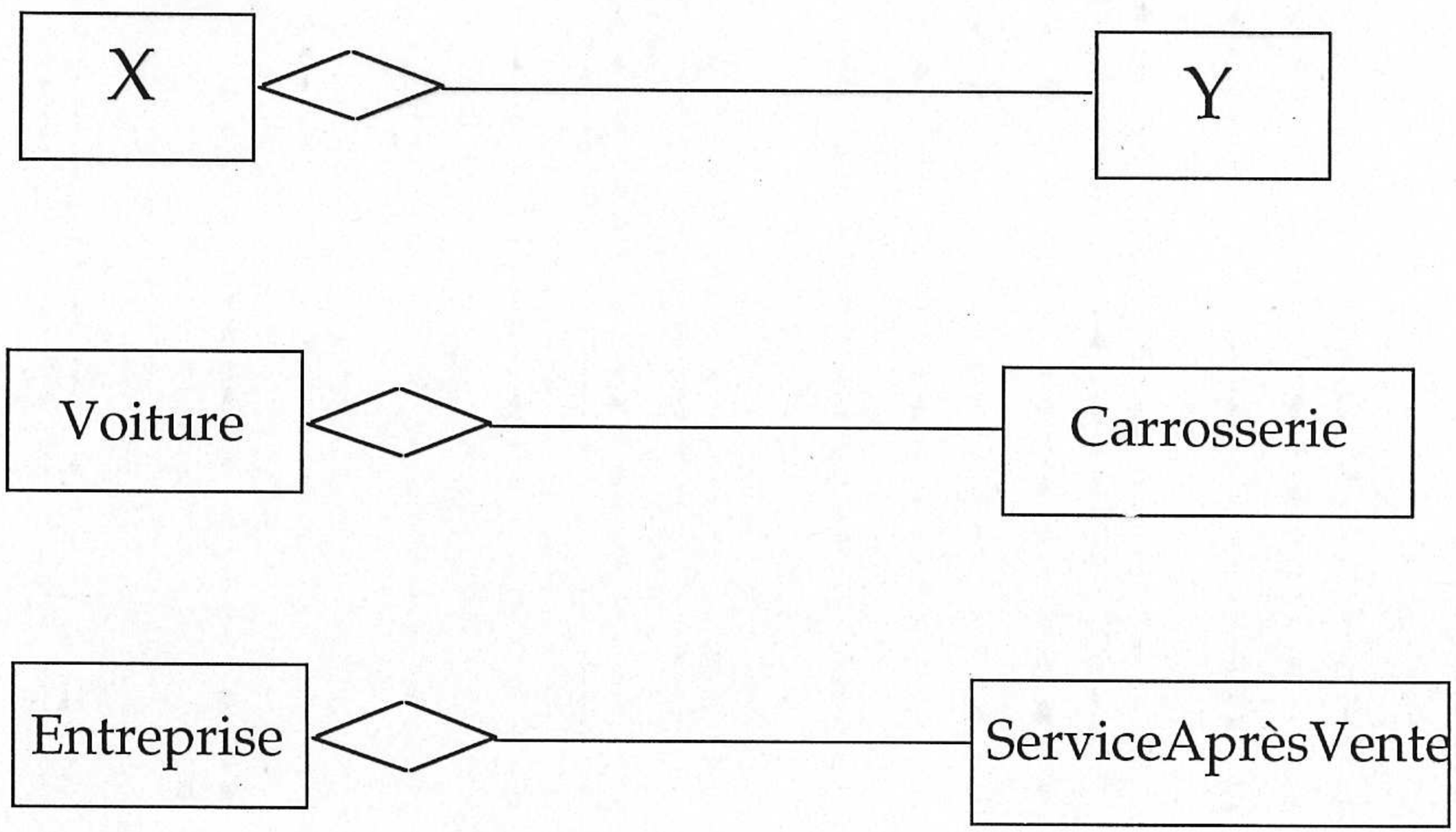
parfumChannel prixTTC

télé32 prixTTC

Exemples (3)


Media Ingrédient
(MHEG5)
MediaPresentable
.....
Afficher (coordonnés)
Compresser/decompresser
Zoom
NoirBlanc
...
MédiaAudible
simpsons afficher (...)
braveheart afficher (...)
Image
.....
Afficher (coordonnés)
Vidéo
.....
Afficher (coordonnées)
VidéoAVI
.....
Afficher (coordonnées)
VidéoQuickTimeMovie (MOV)
.....
Afficher(coordonnées)
braveheart
simpsons

Erreurs à éviter



Relation horizontale : Composition (Agrégation)