Introduction à la modélisation conceptuelle de données avec UML

stph.scenari-community.org/bdd
mod1.pdf

Stéphane Crozat

12 janvier 2018

Table des matières

I - **Cours** 3

1. Notion de modèle 3

1.1. Exercice : Centre médical 4
1.2. Qu'est ce qu'un modèle ? 4
1.3. Qu'est ce qu'un modèle en informatique ? 4
1.4. Qu'est ce qu'un bon modèle ? 5

2. Introduction au diagramme de classes UML : classes et associations 6

2.1. Exercice : Lab I 6
2.2. Présentation d'UML 7
2.3. Classes 8
2.4. Attributs 9
2.5. Repérage des clés 10
2.6. Méthodes 11
2.7. Associations 11
2.8. Cardinalité 12
2.9. Classe d'association 14

II - **Exercices** 16

1. Exercice : Lire l'UML 17

1.1. Exercice : Tennis 17
1.2. Exercice : Journal 17
1.3. Exercice : Logistique 18

2. Exercice : Gestion d'une coopérative viticole 18

3. Exercice : Cours et intervenants 19

4. Exercice : Gestion méthodique du personnel 20

III - **Devoirs** 21

1. Exercice : Attrapez les tous ! 21

Contenus annexes 22

Questions de synthèse 27

Solutions des exercices 28

Glossaire 35

Abréviations 36

Bibliographie 37

Cours

I

La *modélisation conceptuelle* est l'étape *fondatrice* du processus de conception de BD*. Elle consiste à abstraire le problème réel posé pour en faire une reformulation qui trouvera une solution dans le cadre technologique d'un SGBD*.

Si le modèle dominant en conception de bases de données a longtemps été le modèle E-A*, le modèle UML* se généralise de plus en plus. Nous proposons ici une introduction au diagramme de classes à travers la représentation de classes et d'associations simples (il existe d'autres diagrammes UML, par exemple le diagramme de cas, et d'autres primitives de représentation dans le diagramme de classe, par exemple l'héritage).

1. Notion de modèle

Objectifs

Savoir ce qu'est un modèle

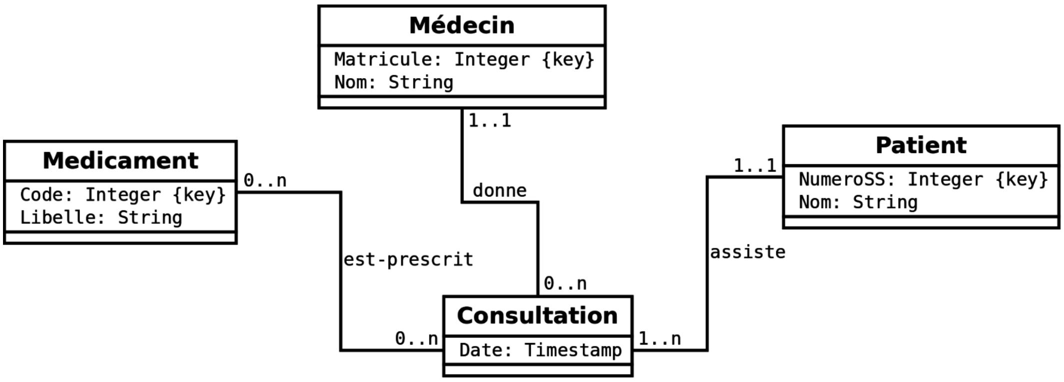
Savoir ce qu'est le langage UML

1.1. Exercice : Centre médical

[solution n°1 p.28*]*

[5 min]

Soit le modèle conceptuel suivant représentant des visites dans un centre médical. Quelles sont les assertions vraies selon ce schéma ?



- [ ] Un patient peut effectuer plusieurs visites.
- [ ] Tous les patients ont effectué au moins une consultation.
- [ ] Un médecin peut recevoir plusieurs patients pendant la même consultation.
- [ ] Un médecin peut prescrire plusieurs médicaments lors d'une même consultation.
- [ ] Deux médecins différents peuvent prescrire le même médicament.

1.2. Qu'est ce qu'un modèle ?

Définition : Modèle

« Modeling, in the broadest sense, is the cost-effective use of something in place of something else for some cognitive purpose. It allows us to use something that is simpler, safer or cheaper than reality instead of reality for some purpose. A model represents reality for the given purpose; the model is an abstraction of reality in the sense that it cannot represent all aspects of reality. » (Rothenberg, 1989*, cité par Arribe, 2014*)

« Système physique, mathématique ou logique représentant les structures essentielles d'une réalité et capable à son niveau d'en expliquer ou d'en reproduire dynamiquement le fonctionnement. » (TLFi)

Fondamental : Modèle

Un modèle est une représentation simplifiée de la réalité en vue de réaliser quelque chose.

1.3. Qu'est ce qu'un modèle en informatique ?

Définition : Modèle informatique

Un modèle informatique est une représentation simplifiée de la réalité en vue de réaliser un traitement avec un ordinateur.

Complément : Numérisation et abstraction : Toute information numérique a été codée selon un modèle donné

« Tout numérisation est une représentation de la réalité sous la forme d'une modélisation numérique. Cette modélisation procède d'une abstraction au sens où c'est une séparation d'avec le réel, au sens où c'est une construction destinée à la manipulation (algorithmique en l'occurrence) et au sens où c'est une simplification de la réalité. »

http://aswemay.fr/co/tropism-pres.html

1.4. Qu'est ce qu'un bon modèle ?

Attention

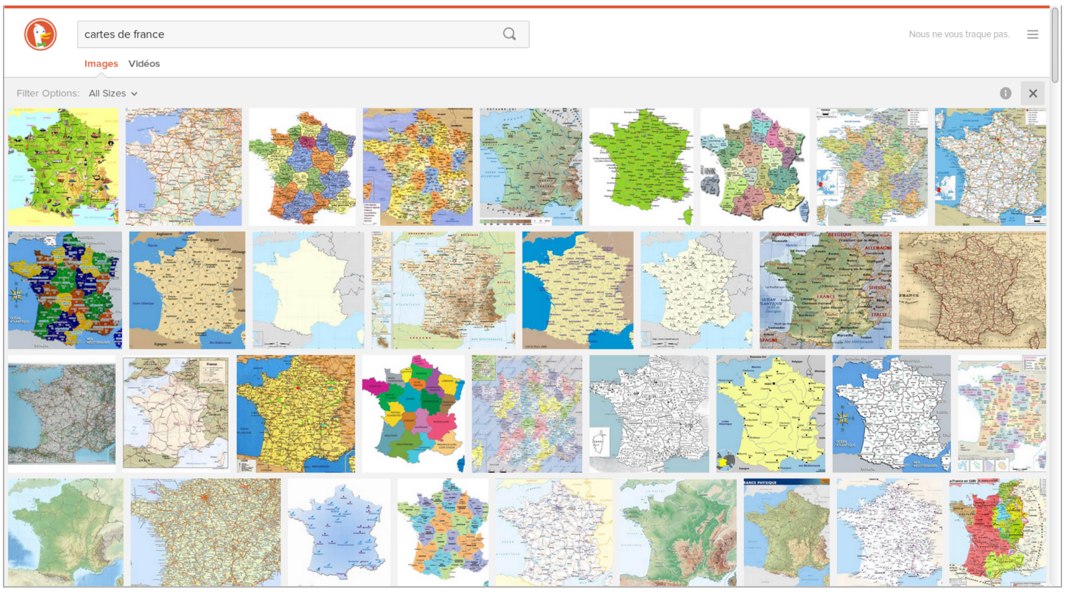
Un modèle est une abstraction, une simplification de la réalité, ce n'est pas la réalité, il n'est jamais complètement fidèle par construction.

Le seul modèle complètement fidèle à la réalité est la réalité elle-même, et ce n'est donc pas un modèle.

Exemple : La carte et le territoire

Une carte est un modèle d'un territoire. Elle est une représentation simplifiée destiné à un usage particulier :

- randonner à pied, en vélo ;
- se diriger en voiture sur des grands axes, sur des axes secondaires ;
- voler en avion de tourisme, en avion de ligne ;
- naviguer sur fleuve, sur mer ;
- étudier les frontières d'une région, d'un pays, de la terre ;
- étudier la démographie, l'économie... ;
- ...



Fondamental

À partir de cet exemple on notera que :

1. *Un modèle est orienté par un usage.*

Chacune de ces cartes est très différente selon ce que l'on veut faire.

2. *Un modèle ne cherche pas à être proche de la réalité.*
   Chacune de ces cartes est très différente de la réalité qu'elle représente.
3. *Un modèle adresse un niveau d'information qui existe mais qui n'est pas accessible dans la réalité.*
   Chacune de ces cartes permet quelque chose que ne permet pas l'accès direct à la réalité.

Méthode : Le rasoir d'Ockham : Entre deux modèles donnés le meilleur modèle est-il toujours le p fourni ?

La méthode de raisonnement connue sous le nom de rasoir d'Ockham (du nom du philosophe éponyme) consiste à préférer les solutions les plus simples aux plus complexes, lorsqu'elles semblent permettre également de résoudre un problème donné ; entre deux théories équivalentes, toujours préférer la plus simple.

Ce principe s'applique très bien à la modélisation : étant donné un objectif et plusieurs modèles possibles, il ne faut pas choisir a priori celui qui représente le plus de choses, mais préférer le plus simple dès qu'il couvre le besoin.

C'est un principe d'économie (il coûte moins cher à produire) et d'efficacité (car les éléments inutiles du modèle plus fourni nuiront à l'efficacité de la tâche).

Exemple

Ainsi, pour naviguer en voiture, il est plus simple de ne *pas* avoir sur la carte les chemins de randonnées qui ne sont pas praticables en voiture.

2. Introduction au diagramme de classes UML : classes et associations

Objectifs

Savoir faire un modèle conceptuel.

Savoir interpréter un modèle conceptuel.

2.1. Exercice : Lab I

Description du problème

[15 min]

Un laboratoire souhaite gérer les médicaments qu'il conçoit.

- Un médicament est décrit par un nom, qui permet de l'identifier. En effet il n'existe pas deux médicaments avec le même nom. Un médicament comporte une description courte en français, ainsi qu'une description longue en latin. On gère aussi le conditionnement du médicament, c'est à dire le nombre de pilules par boîte (qui est un nombre entier).
- À chaque médicament on associe une liste de contre-indications, généralement plusieurs, parfois aucune Une contre-indication comporte un code unique qui l'identifie, ainsi qu'une description. Une contre-indication est toujours associée à un et un seul médicament.

Exemple de données

Afin de matérialiser notre base de données, nous obtenons les descriptions suivantes :

- Le *Chourix* a pour description courte *« Médicament contre la chute des choux »* et pour description longue *« Vivamus fermentum semper porta. Nunc diam velit, adipiscing ut tristique vitae, sagittis vel odio. Maecenas convallis ullamcorper ultricies. Curabitur ornare. »*. Il est conditionné en boîte de 13.
  Ses contre-indications sont :
  - CI1 : Ne jamais prendre après minuit.
  - CI2 : Ne jamais mettre en contact avec de l'eau.
- Le *Tropas* a pour description courte *« Médicament contre les dysfonctionnements intellectuels »* et pour description longue *« Suspendisse lectus leo, consectetur in tempor sit amet, placerat quis neque. Etiam luctus porttitor lorem, sed suscipit est rutrum non. »*. Il est conditionné en boîte de 42.
  Ses contre-indications sont :
  - CI3 : Garder à l'abri de la lumière du soleil

Question 1

[solution n°2 p.28*]*

Réaliser le modèle conceptuel de données en UML du problème.

Question 2

[solution n°3 p.28*]*

Étendre le modèle conceptuel UML afin d'ajouter la gestion des composants. Un composant est identifié par un code unique et possède un intitulé. Tout médicament possède au moins un composant, souvent plusieurs. Tout composant peut intervenir dans la fabrication de plusieurs médicaments. Il existe des composants qui ne sont pas utilisés pour fabriquer des médicaments et que l'on veut quand même gérer.

2.2. Présentation d'UML

UML* est un langage de représentation destiné en particulier à la modélisation objet. UML est devenu une norme OMG* en 1997.

UML propose un formalisme qui impose de "penser objet" et permet de rester indépendant d'un langage de programmation donné. Pour ce faire, UML normalise les concepts de l'objet (énumération et définition exhaustive des concepts) ainsi que leur notation graphique. Il peut donc être utilisé comme un moyen de communication entre les étapes de spécification conceptuelle et les étapes de spécifications techniques.

Fondamental : Diagramme de classe

Le diagramme de classes est un sous ensemble d'UML qui s'attache à la description statique d'un modèle de données représentées par des classes d'objets.

Remarque

Dans le domaine des bases de données, UML peut être utilisé à la place du modèle E-A* pour modéliser le domaine. De la même façon, un schéma conceptuel UML peut alors être traduit en schéma logique (relationnel ou relationnel-objet typiquement).

2.3. Classes

Définition : Classe

Une classe est un type abstrait caractérisé par des propriétés (attributs et méthodes) communes à un ensemble d'objets et permettant de créer des instances de ces objets, ayant ces propriétés.

Syntaxe

Nom de la classe
Attributs
Méthodes()

Représentation UML d'une classe

Exemple : La classe Voiture

Voiture
Marque : string Type : string NbPortes : integer Puissance : integer Kilométrage : integer
Rouler()

Exemple de classe représentée en UML

Exemple : Une instance de la classe Voiture

L'objet V1 est une instance de la classe Voiture.

V1 : Voiture

- Marque : 'Citroën'
- Type : 'ZX'
- Portes : 5
- Puissance : 6
- Kilométrage : 300000

Complément

La modélisation sous forme de diagramme de classes est une modélisation statique, qui met en exergue la structure d'un modèle, mais ne rend pas compte de son évolution temporelle. UML propose d'autres types de diagrammes pour traiter, notamment, de ces aspects.

2.4. Attributs

Définition : Attribut

Un attribut est une information élémentaire qui caractérise une classe et dont la valeur dépend de l'objet instancié.

Un attribut est typé : Le domaine des valeurs que peut prendre l'attribut est fixé a priori.

- *Un attribut peut être multivalué* : Il peut prendre plusieurs valeurs distinctes dans son domaine.
- *Un attribut peut être dérivé* : Sa valeur alors est une fonction sur d'autres attributs de la classe
- *Un attribut peut être composé* (ou composite) : Il joue alors le rôle d'un groupe d'attributs (par exemple une adresse peut être un attribut composé des attributs numéro, type de voie, nom de la voie). Cette notion renvoie à la notion de variable de type `Record` dans les langages de programmation classiques.

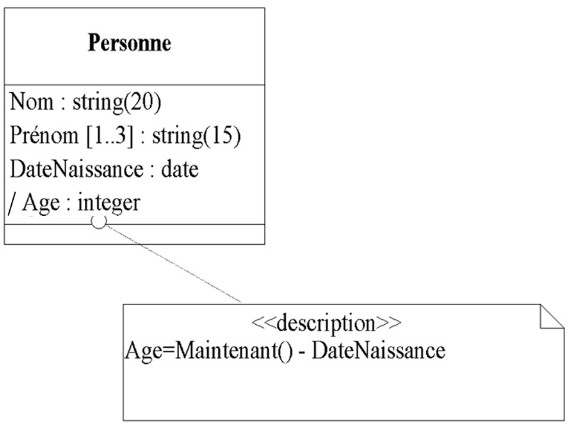
Attention : On utilise peu les attributs dérivés et composés en UML

- En UML on préfère l'usage de méthodes aux attributs dérivés. On utilisera toujours des méthodes dès qu la valeur de l'attribut dérivé dépend d'autres attributs extérieurs à sa classe.
- En UML on préfère l'usage de compositions aux attributs composés. On utilisera toujours des compositions pour les attributs composés et multivalués.

Syntaxe

```
attribut:type
attribut_multivalué[nbMinValeurs..nbMaxValeurs]:type
/attribut_dérivé:type
attribut_composé
   - sous-attribut1:type
   - sous-attribut2:type
   - ...
```

Exemple : La classe Personne



Représentation d'attributs en UML

Dans cet exemple, les attributs Nom, Prénom sont de type *string*, l'un de 20 caractères et l'autre de 10, tandis que DateNaissance est de type *date* et Age de type *integer*. Prénom est un attribut multivalué, ici une personne peut avoir de 1 à 3 prénoms. Age est un attribut dérivé, il est calculé par une fonction sur DateNaissance.

Complément : Voir aussi

Méthodes (cf. p.11)

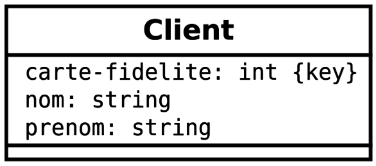
Composition (cf. p.22)

2.5. Repérage des clés

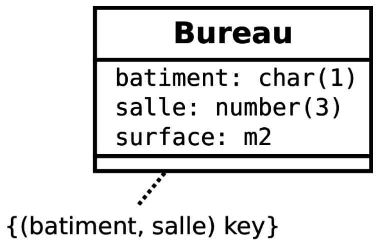
Un attribut ou un groupe d'attributs peut être annoté comme étant *clé* s'il permet d'identifier de façon unique un objet de la classe.

On ajoute le symbole `{key}` à côté du ou des attributs concernés.

Exemple



Clé en UML



Clé composée de deux attributs

Méthode

Le repérage des clés n'est pas systématique en UML (la définition des clés se fera essentiellement au niveau logique). On cherchera néanmoins à repérer les clés rendues évidentes par la phase de clarification.

Attention

On n'ajoutera jamais de clé artificielle* au niveau du MCD. Si aucune clé n'est évidente, on laisse la classe sans clé.

Attention : Attribut souligné et #

On trouvera dans ce cours des exemples d'attributs soulignés ou précédés de # pour exprimer l'unicité. Ce n'est pas une pratique standard et la notation `{key}` devrait lui être substituée.

- Un attribut souligné est normalement un attribut de classe, ou *static*, en UML,
- Un attribut précédé de # est normalement un attribut protégé en UML.

Mais les concepts d'attribut de classe et d'attribut protégé ne sont pas utilisés dans le cadre des bases de données.

2.6. Méthodes

Définition : Méthode

Une méthode (ou opération) est une fonction associée à une classe d'objet qui permet d'agir sur les objets de la classe ou qui permet à ces objets de renvoyer des valeurs (calculées en fonction de paramètres).

Syntaxe

```
methode(paramètres):type
```

Remarque : Méthodes et modélisation de BD

Pour la modélisation des bases de données, les méthodes sont surtout utilisées pour représenter des données calculées (à l'instar des attributs dérivées) ou pour mettre en exergue des fonctions importantes du système cible. Seules les méthodes les plus importantes sont représentées, l'approche est moins systématique qu'en modélisation objet par exemple.

Remarque : Méthodes, relationnel, relationnel-objet

Lors de la transformation du modèle conceptuel UML en modèle logique relationnel, les méthodes *ne seront généralement pas implémentées*. Leur repérage au niveau conceptuel sert donc surtout d'aide-mémoire pour l'implémentation au niveau applicatif.

Au contraire, un modèle logique relationnel-objet permettra l'implémentation de méthodes directement associées à des tables. Leur repérage au niveau conceptuel est donc encore plus important.

Complément

Transformation des méthodes par des vues *(cf. p.24)*

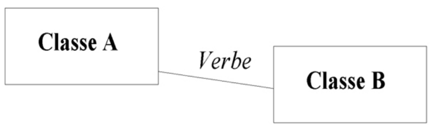
2.7. Associations

Définition : Association

Une association est une relation logique entre deux classes (association binaire) ou plus (association n-aire) qui définit un ensemble de liens entre les objets de ces classes.

Une association est *nommée*, généralement par un verbe. Une association peut avoir des propriétés (à l'instar d'une classe). Une association définit le nombre minimum et maximum d'instances autorisée dans la relation (on parle de cardinalité).

Syntaxe



Notation de l'association en UML

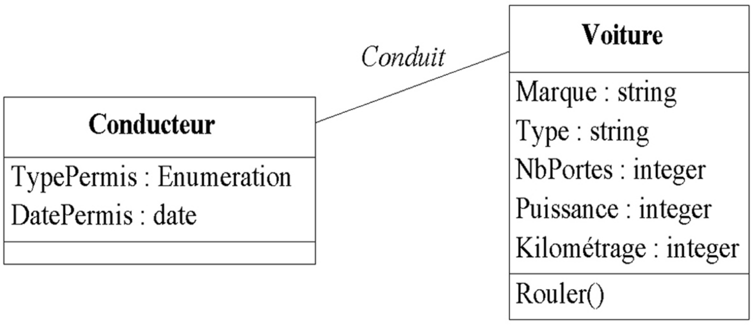
Attention

Le nom de l'association (verbe qui la décrit) est obligatoire, au même titre que le nom d'une classe ou d'un attribut.

Remarque

Une association est généralement bidirectionnelle (c'est à dire qu'elle peut se lire dans les deux sens). Les associations qui ne respectent pas cette propriété sont dites unidirectionnelles ou à navigation restreinte.

Exemple : L'association Conduit



Représentation d'association en UML

L'association Conduit entre les classes Conducteur et Voiture exprime que les conducteurs conduisent des voitures.

Complément : Voir aussi

- *Cardinalité (cf. p.12)*
- *Explicitation des associations (cf. p.24)*
- *Associations ternaires (cf. p.24)*
- *Contraintes sur les associations (cf. p.25)*

2.8. Cardinalité

Définition : Cardinalité d'une association

La cardinalité d'une association permet de représenter le nombre minimum et maximum d'instances qui sont autorisées à participer à la relation. La cardinalité est définie pour les deux sens de la relation.

Syntaxe

Si min_a (resp. max_a) est le nombre minimum (resp. maximum) d'instances de la classe A autorisées à participer à l'association, on note sur la relation, à côté de la classe A : $min_a..max_a$.

Si le nombre maximum est indéterminé, on note n ou *.

Attention

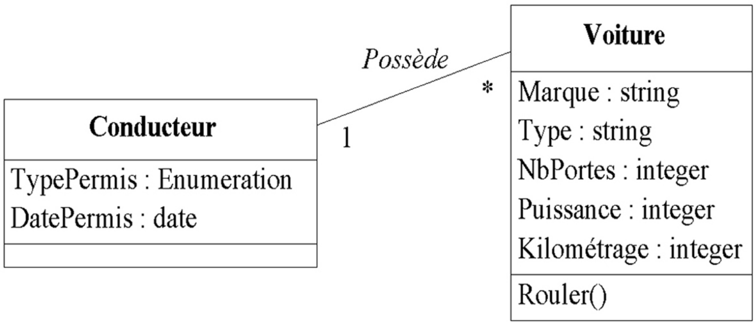
La notation de la cardinalité en UML est opposée à celle adoptée en E-A. En UML on note à gauche (resp. à droite) le nombre d'instances de la classe de gauche (resp. de droite) autorisées dans l'association. En E-A, on note à gauche (resp. à droite) le nombre d'instances de la classe de droite (resp. de gauche) autorisées dans l'association.

Remarque

Les cardinalités les plus courantes sont :

- 0..1 (optionnel)
- 1..1 ou 1 (un)
- 0..n ou 0..* ou * (plusieurs)
- 1..n ou 1..* (obligatoire)

Exemple : La cardinalité de l'association Possède



Représentation de cardinalité en UML

Ici un conducteur peut posséder plusieurs voitures (y compris aucune) et une voiture n'est possédée que par un seul conducteur.

Fondamental : Terminologie

- On appelle association 1:1 les associations de type :
  - 0..1:0..1
  - 0..1:1..1
  - 1..1:0..1
  - 1..1:1..1
- On appelle association 1:N les associations de type :
  - 0..1:0..N
  - 0..1:1..N
  - 1..1:0..N
  - 1..1:1..N
- On appelle association N:M (ou M:N) les associations de type :
  - 0..N:0..N
  - 0..N:1..N
  - 1..N:0..N

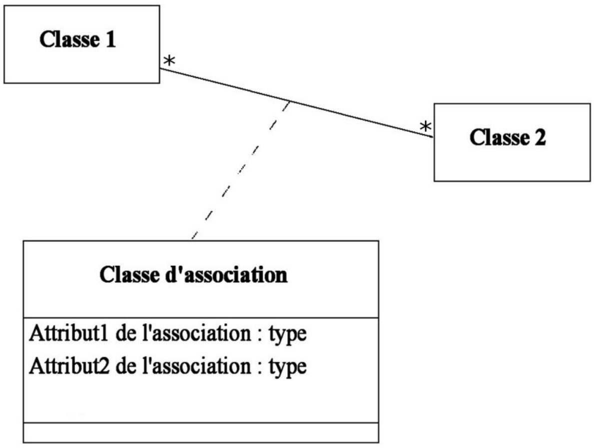
- 1..N:1..N

2.9. Classe d'association

Définition : Classe d'association

On utilise la notation des classes d'association lorsque l'on souhaite ajouter des propriétés à une association.

Syntaxe : Notation d'une classe d'association en UML



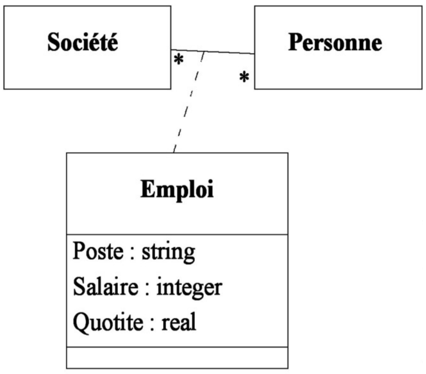
Notation d'une classe d'association en UML

Méthode

On réserve en général les classes d'association aux associations N:M.

Il est toujours possible de réduire une classe d'association sur une association 1:N en migrant ses attributs sur la classe côté N, et c'est en général plus lisible ainsi.

Exemple : Exemple de classe d'association



Emplois

Conseil

Selon le standard UML une classe d'association est une classe et à ce titre elle peut être mobilisée dans d'autres associations ou dans des héritages. Nous déconseillons néanmoins ces notations qui ont tendance à complexifier la lecture et la transformation du diagramme.

Nous conseillons donc de ne jamais associer une classe d'association.

Exercices

II

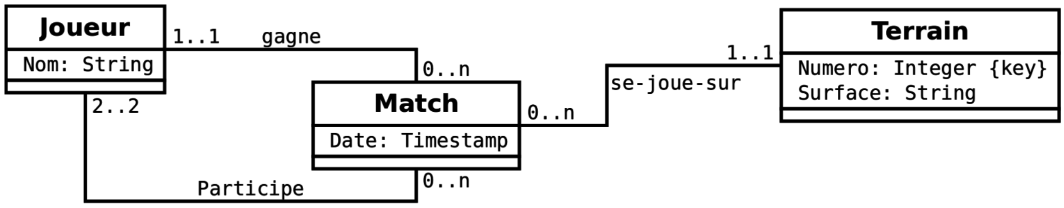
1. Exercice : Lire l'UML

[solution n°4 p.29]

[15 min]

1.1. Exercice : Tennis

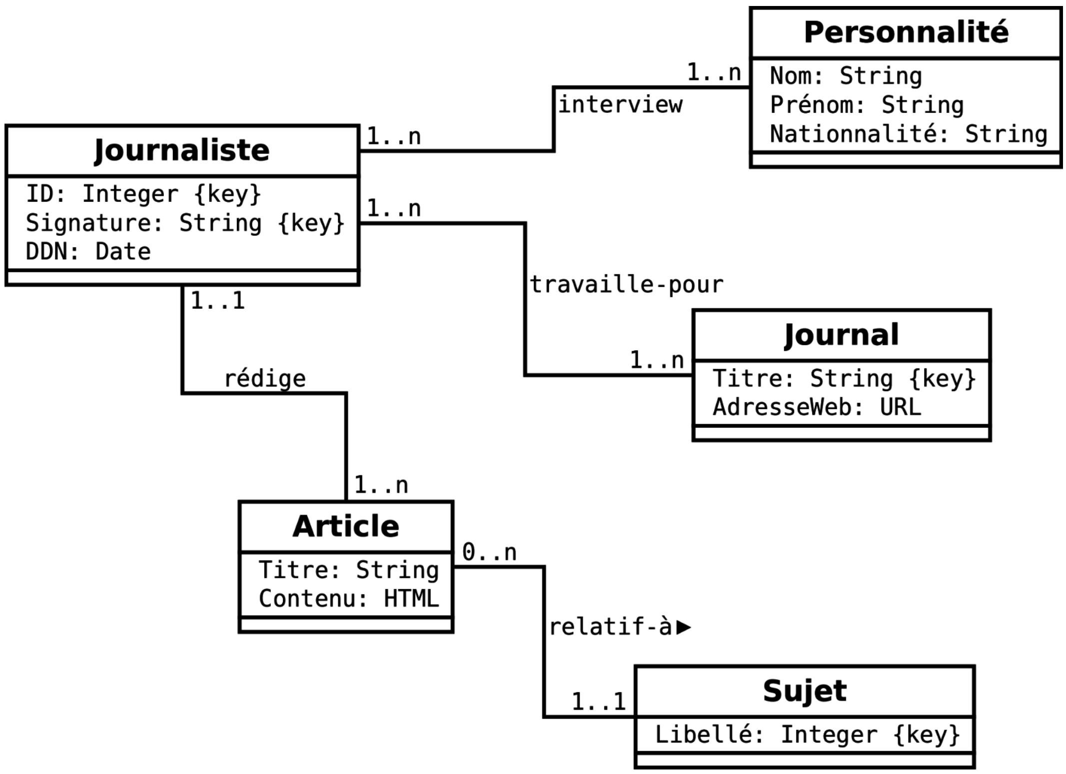
Le schéma suivant représente les rencontres lors d'un tournoi de tennis. Quelles sont les assertions vraies selon ce schéma ?



- [ ] On peut jouer des matchs de double.
- [ ] Un joueur peut gagner un match sans y avoir participé.
- [ ] Il peut y avoir deux matchs sur le même terrain à la même heure.
- [ ] Connaissant un joueur, on peut savoir sur quels terrains il a joué.

1.2. Exercice : Journal

Voici le schéma conceptuel du système d'information (très simplifié) d'un quotidien. Quelles sont les assertions vraies selon ce schéma ?

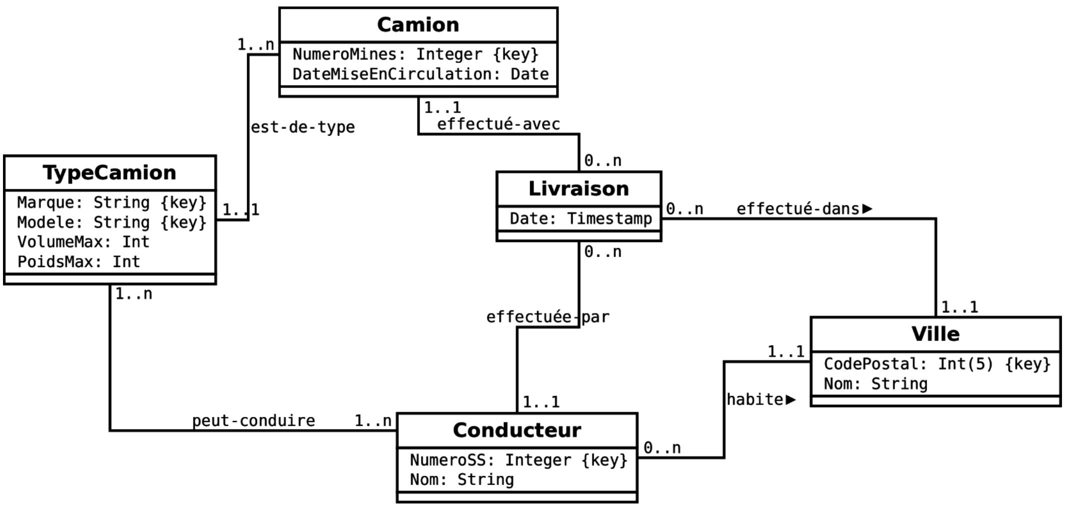


- ☐ Un article peut être rédigé par plusieurs journalistes.
- ☐ Un article peut être publié plusieurs fois dans le même journal.
- ☐ Un article peut être publié dans un journal par un journaliste qui ne travaille pas pour ce journal.
- ☐ Il peut y avoir plusieurs articles sur le même sujet.
- ☐ Un journaliste peut interviewer une personnalité sans faire d'article à ce propos.

1.3. Exercice : Logistique

Une société de transport routier veut installer un système d'information pour rendre plus efficace sa logistique. Embauché au service informatique de cette compagnie, vous êtes donc chargé de reprendre le travail déjà effectué (c'est à dire le schéma suivant).

Quelles sont les assertions vraies selon ce schéma ?



- ☐ Un conducteur peut conduire plusieurs camions.
- ☐ Un conducteur peut conduire un camion sans y être autorisé.
- ☐ Il peut y avoir plusieurs conducteurs pour le même camion.
- ☐ Un conducteur peut livrer sa propre ville

2. Exercice : Gestion d'une coopérative viticole

[20 minutes]

Cet exercice a été inspiré par Bases de données : objet et relationnel*.

On considère une base "Coopérative" qui possède les caractéristiques suivantes :

- Un vin est caractérisé par un numéro entier unique `nv`, un cru, une année de production et un degré.
- Un viticulteur est caractérisé par un numéro entier unique `nvt`, un nom et une ville.
- Un viticulteur produit plusieurs vins, chaque vin n'est produit que par un viticulteur.

- Les buveurs sont caractérisés par un numéro de buveur `nb`, un nom, prénom et une adresse (limitée à la ville pour simplifier).
- Un buveur consomme des vins et peut passer des commandes pour acheter des vins.

Question 1

[solution n°5 p.31*]*

Lister tous les types d'objet à considérer, les attributs associés et les domaines de valeurs de ces attributs. Repérer les éventuelles clés.

Question 2

[solution n°6 p.31*]*

Lister toutes les associations à considérer et indiquer leurs cardinalités.

Question 3

[solution n°7 p.32*]*

Donner le diagramme UML de cette situation.

3. Exercice : Cours et intervenants

[20 min]

On souhaite réaliser une base de données pour gérer les cours dispensés dans une école d'ingénieur, ainsi que les personnes qui interviennent dans ces cours.

Chaque cours est identifié par une année et un numéro. Chaque cours a donc un numéro unique localement à chaque année. Un cours possède un titre et un type ('C' pour Cours, 'TD' ou 'TP'). Un cours possède également une date de début, et une date de fin, qui est toujours de 5 jours après la date de début.

Chaque intervenant est identifié par son nom (deux intervenants ne peuvent pas avoir le même nom). Il a un prénom, un bureau, un ou plusieurs numéros de téléphones (jusqu'à trois numéros) et des spécialités. Un bureau est défini par un centre ('R' pour Royallieu, 'BF' pour Benjamin Franklin et 'PG' pour Pierre Guillaumat), un bâtiment (une lettre de A à Z), et un numéro (inférieur à 1000). Les spécialités sont des couples de chaînes de caractères désignant un domaine (par exemple 'BD') et une spécialité (par exemple 'SGBDRO').

Chaque cours est donné par un unique intervenant.

Voici un exemple : Le cours 'Machines universelles', n°21 de l'année 2014 est donné par Alan Turing entre le 05/01/2014 et le 10/01/2014. C'est un cours de type 'C'. Alan Turing a le bureau 666 au bâtiment X de PG. Il a les numéros de téléphone 0666666666 et 0766666666. Il possède les spécialités suivantes :

- Domaine : Mathématique ; Spécialité : Cryptographie
- Domaine : Informatique ; Spécialité : Algorithmie
- Domaine : Informatique ; Spécialité : Intelligence Artificielle

Question

[solution n°8 p.32*]*

Réaliser le modèle UML de la base de données. Préciser les clés et les types des attributs.

4. Exercice : Gestion méthodique du personnel

[30 minutes]

Le service de gestion du personnel d'une entreprise désire s'équiper d'un outil lui permettant de gérer informatiquement ses employés, leurs salaires et leurs congés. Les spécifications suivantes ont pu être mises en exergue par une analyse des besoins réalisée auprès des utilisateurs du service du personnel.

Généralités :

- tout employé est identifié par un nom, un prénom et une date de naissance ;
- tout employé remplit une fonction et appartient à un service ;
- pour chaque employé on gère la date d'embauche et la quotité (c'est à dire le pourcentage de temps travaillé par rapport au temps plein, en cas de travail à temps partiel).

Gestion des salaires :

- pour chaque employé on gère l'historique de ses salaires dans l'entreprise, chaque salaire étant affecté à une période de temps ;
- un salaire est composé d'un salaire brut, de charges patronales et de charges salariales ;
- on cherchera à partir des ces données à obtenir également le salaire chargé (brut + charges patronales), e le salaire net (brut - charges salariales), et ce en particulier pour le salaire en cours (celui que touche actuellement le salarié).

Gestion des congés :

- pour chaque employé, on mémorise chaque congé pris (posant qu'un congé concerne toujours une ou plusieurs journées entières) ;
- chaque employé a le droit aux jours de congés suivants :
    - 25 jours (pour une quotité de 1) et 25 x quotité sinon,
    - chaque fonction ouvre les droits à un certain nombre de jours de RTT,
    - chaque service ouvre les droits à un certain nombre de jours de RTT,
    - chaque tranche de 5 ans passés dans l'entreprise donne droit à 1 jour supplémentaire,
    - les employés de plus de 40 ans ont un jour supplémentaire, et ceux de plus de 50 ans deux.
- pour chaque employé on cherchera à connaître le nombre total de jours de congés autorisés, le nombre de jours pris et le nombre de jours restants sur l'année en cours.

Question

[*solution* n°9 p.33]

Réaliser le diagramme de classes permettant de modéliser ce problème.

Indice :

On fera apparaître les méthodes pertinentes étant donné le problème posé.

Méthodes *(cf. p.11)*

Devoirs

III

1. Exercice : Attrapez les tous !

[30 minutes]

La Ligue Pokémon souhaite que vous conceviez la prochaine génération du Pokédex. Le Pokédex est une base de données répertoriant différentes espèces de petits monstres combattants appelés Pokémon. Il devra également garder trace des plus grands combats de Pokémon de l'Histoire.

Pokémons

Une *espèce de Pokémon* est identifiée par un nom unique. Elle est décrite par différentes caractéristiques morphologiques (couleur et taille moyenne), et un ou deux (jamais zéro) *types* (feu, eau, acier, ...). Il existe naturellement plusieurs espèces de même type. Une espèce peut être l'*évolution* d'une autre espèce. Par exemple, *Raichu* est l'évolution de *Pikachu.* On considère qu'une espèce ne peut évoluer, au plus, qu'en une seule autre espèce, et que deux espèces différentes ne peuvent jamais évoluer en une même espèce.

Un *Pokémon* est un monstre d'une espèce précise. Il est identifié par un numéro unique distribué par la Ligue Pokémon. Il est caractérisé par un surnom affectif (pas nécessairement unique), un niveau (un entier compris entre 1 et 100 qui mesure ses aptitudes au combat), et un nombre de points de pouvoir (un entier positif qui limite ses capacités magiques).

Un Pokémon connaît des *attaques*. Une attaque est identifiée par un nom unique. Elle est également caractérisée par sa valeur de puissance (un entier positif), et un ou deux (jamais zéro) types. Chaque attaque requière un nombre donné de points de pouvoir pour être lancée. On notera qu'un Pokémon peut apprendre des attaques d'un type différent du sien.

Dresseurs de Pokémon et tournois

Chaque *dresseur* de Pokémon est identifié par un numéro unique distribué par la Ligue Pokémon. Il est caractérisé par un nom, un prénom, et un *centre* de rattachement. Il peut dresser plusieurs Pokémon, d'espèces et de niveaux différents. Un Pokémon n'a qu'un seul dresseur (son maître).

Un centre est identifié par un numéro de certification unique, également distribué par la Ligue Pokémon.

Un *combat de Pokémon* est la rencontre de deux dresseurs Pokémon. Il a lieu à une date précise, dans un centre certifié par la Ligue (les combats de rue sont interdits). Un vainqueur est proclamé à l'issu de chaque combat. Les combats sont parfois organisés dans le cadre de *tournois*. Chaque tournoi est identifié par un numéro d'homologation unique distribué par la Ligue. Ils sont organisés par un centre certifié (mais tous les combats d'un tournoi n'ont pas forcément lieu dans le centre organisateur).

Question

Réalisez un diagramme UML répondant aux besoins de la Ligue Pokémon.

Contenus annexes

> Composition

Définition : Association de composition

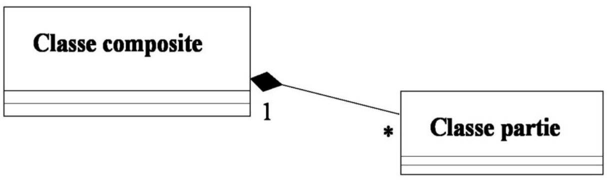
On appelle composition une association particulière qui possède les propriétés suivantes :

- La composition associe une classe composite et des classes parties, tel que tout objet partie appartient à un et un seul objet composite. C'est donc une association 1:N (voire 1:1).
- La composition n'est pas partageable, donc un objet partie ne peut appartenir qu'à un seul objet composite à la fois.
- Le cycle de vie des objets parties est lié à celui de l'objet composite, donc un objet partie disparaî quand l'objet composite auquel il est associé disparaît.

Remarque

- La composition est une association particulière (binaire de cardinalité contrainte).
- La composition n'est pas symétrique, une classe joue le rôle de conteneur pour les classes liées, elle prend donc un rôle particulier a priori.
- La composition est une agrégation avec des contraintes supplémentaires (non partageabilité et cycle de vie lié).

Syntaxe : Notation d'une composition en UML



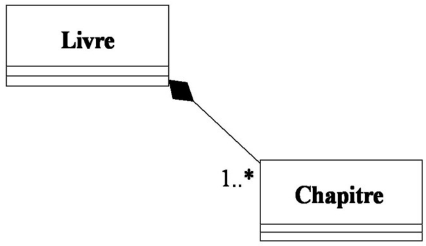
Notation de la composition en UML

Attention : Composition et cardinalité

La cardinalité côté composite est toujours de exactement 1.

Côté partie la cardinalité est libre, elle peut être 0..1, 1, * ou bien 1..*.

Exemple : Exemple de composition



Un livre

On voit bien ici qu'un chapitre n'a de sens que faisant partie d'un livre, qu'il ne peut exister dans deux livres différents et que si le livre n'existe plus, les chapitres le composant non plus.

Remarque : Composition et entités faibles

La composition permet d'exprimer une association analogue à celle qui relie une entité faible à une entité identifiante en modélisation E-A*. L'entité de type faible correspond à un objet partie et l'entité identifiante à un objet composite.

Conseil : Composition et attribut multivalué

- Une composition avec une classe partie dotée d'un seul attribut peut s'écrire avec un attribut multivalué.
- Un attribut composé et multivalué peut s'écrire avec une composition.

Rappel : Voir aussi

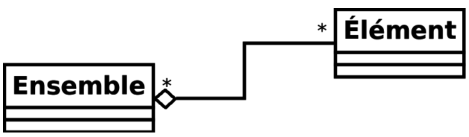
- *Attributs* *(cf. p.9)*
- *Agrégation* *(cf. p.23)*

> Agrégation

Définition : Association d'agrégation

L'agrégation est une association particulière utilisée pour préciser une relation tout/partie (ou ensemble /élément), on parle d'association *méréologique*.

Syntaxe



Notation de l'agrégation en UML

La cardinalité, peut être exprimée librement, en particulier les instances de la classe Élément peuvent être associées à plusieurs instances de la classe Ensemble, et même de plusieurs classes.

Attention

L'agrégation garde toutes les propriétés d'une association classique (cardinalité, cycle de vie, etc.), elle ajoute simplement une terminologie un plus précise via la notion de tout/partie.

> Transformation des méthodes par des vues

Méthode

Lorsqu'une méthode est spécifiée sur un diagramme UML pour une classe C, si cette méthode est une fonction relationnelle (elle renvoie une unique valeur et elle peut être résolue par une requête SQL), alors on crée une vue qui reprend les attributs de la classe C et ajoute des colonnes calculées pour les méthodes.

Remarque : Attributs dérivés

Les attributs dérivés étant apparentés à des méthodes, ils peuvent également être gérés par des vues.

> Explicitation des associations (sens de lecture et rôle)

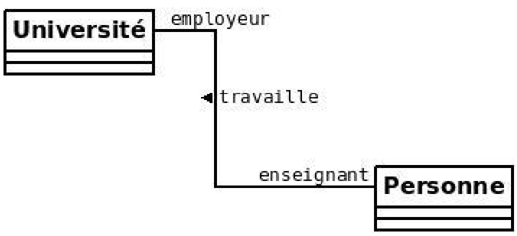
Syntaxe : Sens de lecture

Il est possible d'ajouter le sens de lecture du verbe caractérisant l'association sur un diagramme de classe UML, afin d'en faciliter la lecture. On ajoute pour cela un signe < ou > (ou un triangle noir) à côté du nom de l'association

Syntaxe : Rôle

Il est possible de préciser le rôle joué par une ou plusieurs des classes composant une association afin d'en faciliter la compréhension. On ajoute pour cela ce rôle à côté de la classe concernée (parfois dans un petit encadré collé au trait de l'association.

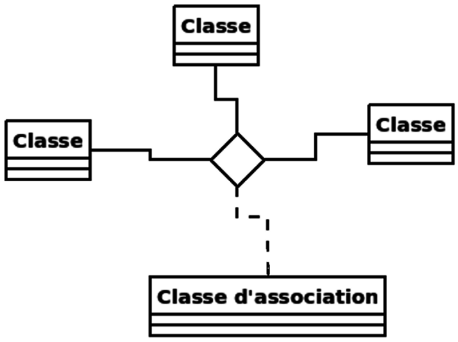
Exemple



Rôle et sens de lecture sur une association

> Associations ternaires

Syntaxe



Notation d'une association ternaire

Conseil : Ne pas abuser des associations ternaires

Il est toujours possible de réécrire une association ternaire avec trois associations binaires, en transformant l'association en classe.

Conseil : Pas de degré supérieur à 3

En pratique on n'utilise jamais en UML d'association de degré supérieur à 3.

> Transformation des agrégations

Rappel : Agrégation

Les associations de type agrégation se traitent de la même façon que les associations classiques.

Agrégation 1:N

```
Classe1(#a,b)
Classe2(#c,d,a=>Classe1)
```

Agrégation N:M

```
Classe1(#a,b)
Classe2(#c,d)
```

```
Assoc(#a=>Classe1,#c=>Classe2)
```

Questions de synthèse

Quels sont les principaux éléments du diagramme de classes UML ?

....
....
....
....
....
....
....

Quelles sont les différences et points communs entre la diagramme de classe UML et le modèle E-A étendu ?

....
....
....
....
....
....
....

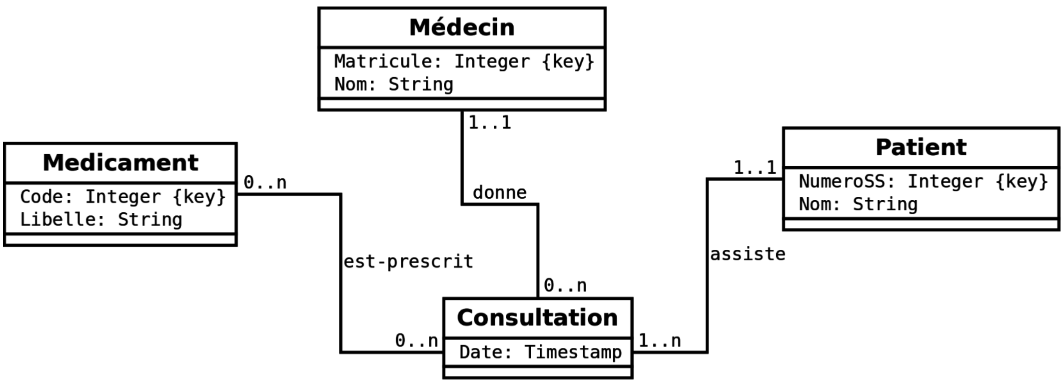
Solutions des exercices

> **Solution** n°1

Exercice p. 4

[5 min]

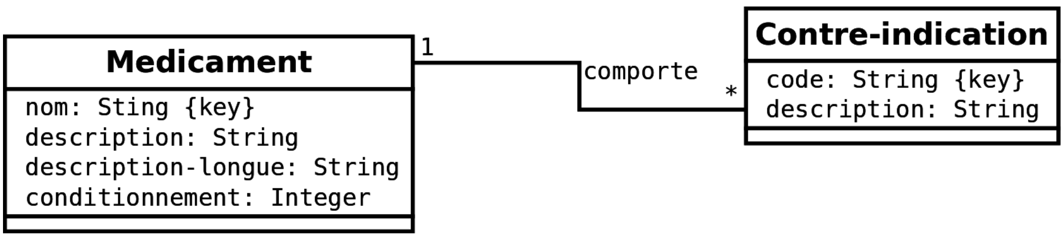
Soit le modèle conceptuel suivant représentant des visites dans un centre médical. Quelles sont les assertions vraies selon ce schéma ?



- [x] Un patient peut effectuer plusieurs visites.
- [x] Tous les patients ont effectué au moins une consultation.
- [ ] Un médecin peut recevoir plusieurs patients pendant la même consultation.
- [x] Un médecin peut prescrire plusieurs médicaments lors d'une même consultation.
- [x] Deux médecins différents peuvent prescrire le même médicament.

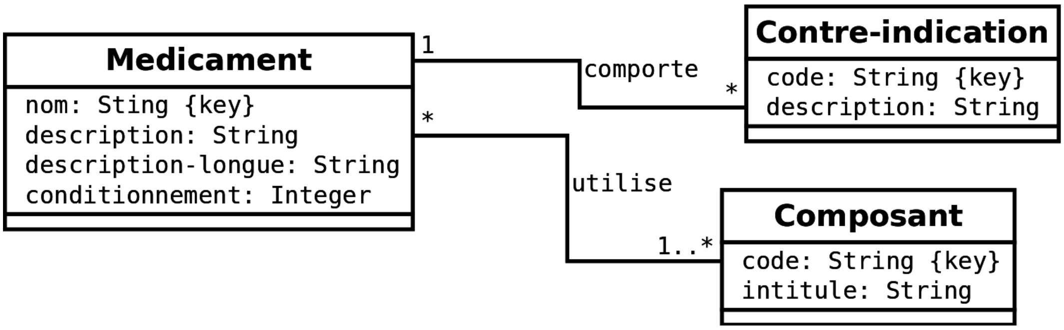
> **Solution** n°2

Exercice p. 7



> **Solution** n°3

Exercice p. 7

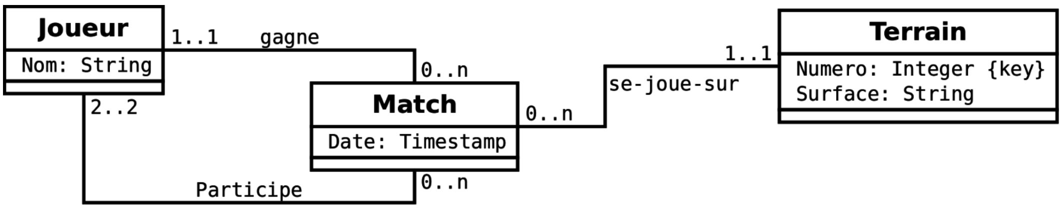


> **Solution** n°4

Exercice p. 17

Exercice : Tennis

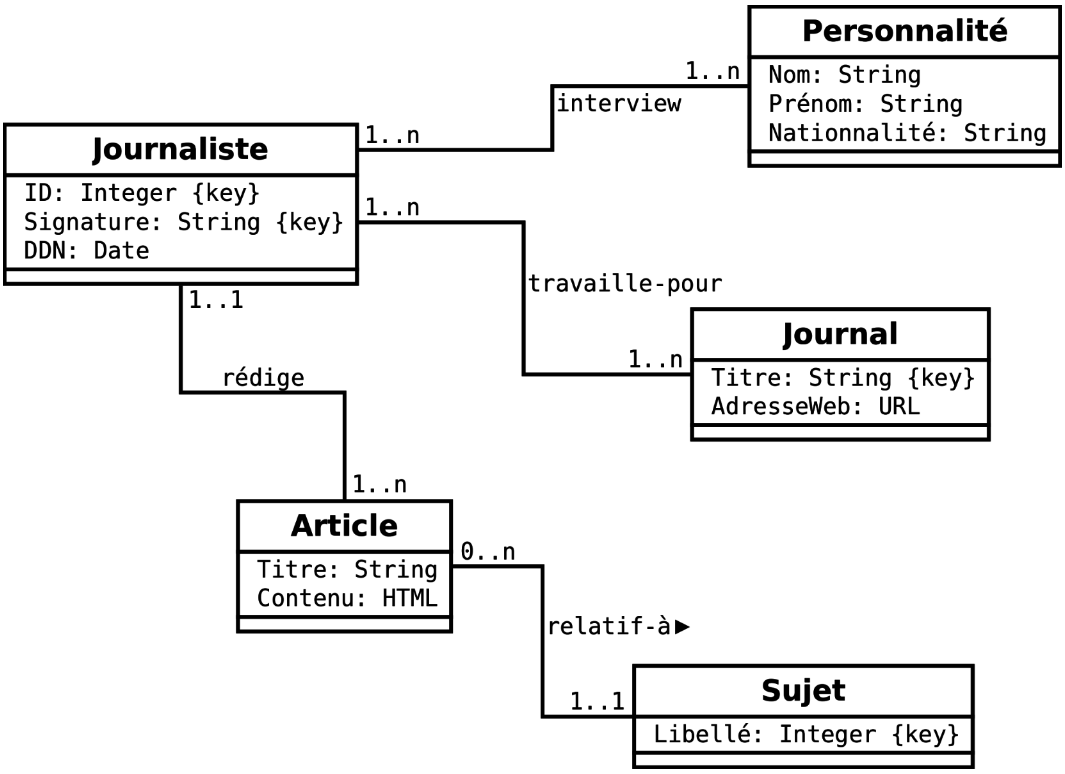
Le schéma suivant représente les rencontres lors d'un tournoi de tennis. Quelles sont les assertions vraies selon ce schéma ?



- ☐ On peut jouer des matchs de double.
- ☑ Un joueur peut gagner un match sans y avoir participé.
- ☑ Il peut y avoir deux matchs sur le même terrain à la même heure.
- ☑ Connaissant un joueur, on peut savoir sur quels terrains il a joué.

Exercice : Journal

Voici le schéma conceptuel du système d'information (très simplifié) d'un quotidien. Quelles sont les assertions vraies selon ce schéma ?

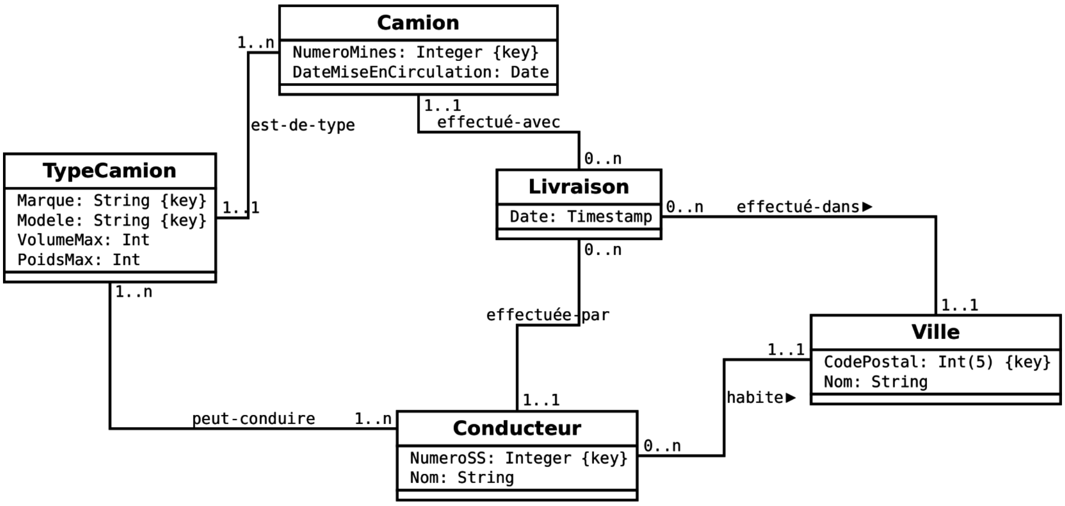


- ☐ Un article peut être rédigé par plusieurs journalistes.
- ☑ Un article peut être publié plusieurs fois dans le même journal.
- ☑ Un article peut être publié dans un journal par un journaliste qui ne travaille pas pour ce journal.
- ☑ Il peut y avoir plusieurs articles sur le même sujet.
- ☑ Un journaliste peut interviewer une personnalité sans faire d'article à ce propos.

Exercice : Logistique

Une société de transport routier veut installer un système d'information pour rendre plus efficace sa logistique. Embauché au service informatique de cette compagnie, vous êtes donc chargé de reprendre le travail déjà effectué (c'est à dire le schéma suivant).

Quelles sont les assertions vraies selon ce schéma ?



- ☑ Un conducteur peut conduire plusieurs camions.
- ☑ Un conducteur peut conduire un camion sans y être autorisé.
- ☑ Il peut y avoir plusieurs conducteurs pour le même camion.
- ☑ Un conducteur peut livrer sa propre ville

> **Solution** n°5

Exercice p. 19

- *Buveurs*
  Nb:Entier (clé), Nom:Char, Ville:Char)
- *Vins*
  Nv:Entier (clé), Cru:Char, Millésime:Entier(4), Degré:Décimal
- *Viticulteurs*
  Nvt:Entier (clé), Nom:Char, Région:Char

> **Solution** n°6

Exercice p. 19

Entité	Association	Entité	Cardinalité
Buveur	Consommation	Vin	N:M
Viticulteur	Production	Vin	1:N
Buveur	Commande	Vin	N:M

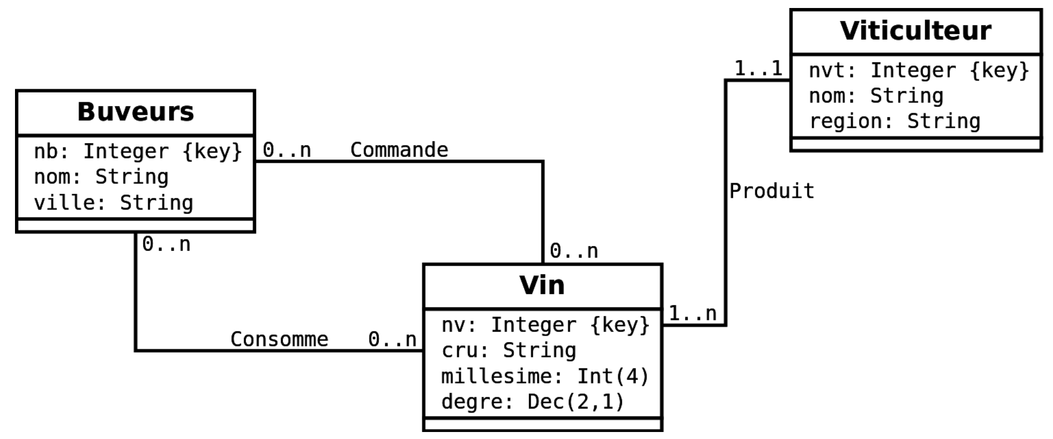
Remarque

Concernant l'association `Production`, la cardinalité minimale (coté Vins) peut être aussi bien 0 ou 1 en fonction de la conception de la base : il se peut qu'il y ait des Vins importés et que l'entité Viticulteur ne concerne que les producteurs français, ou plus généralement qu'il y ait des vins pour lesquels on ne connaît pas le producteur et réciproquement des viticulteurs pour lesquels on ne connaît pas les vins produits.

En d'autres termes les cardinalités dépendent fortement des hypothèses considérées concernant la situation qu'on doit modéliser.

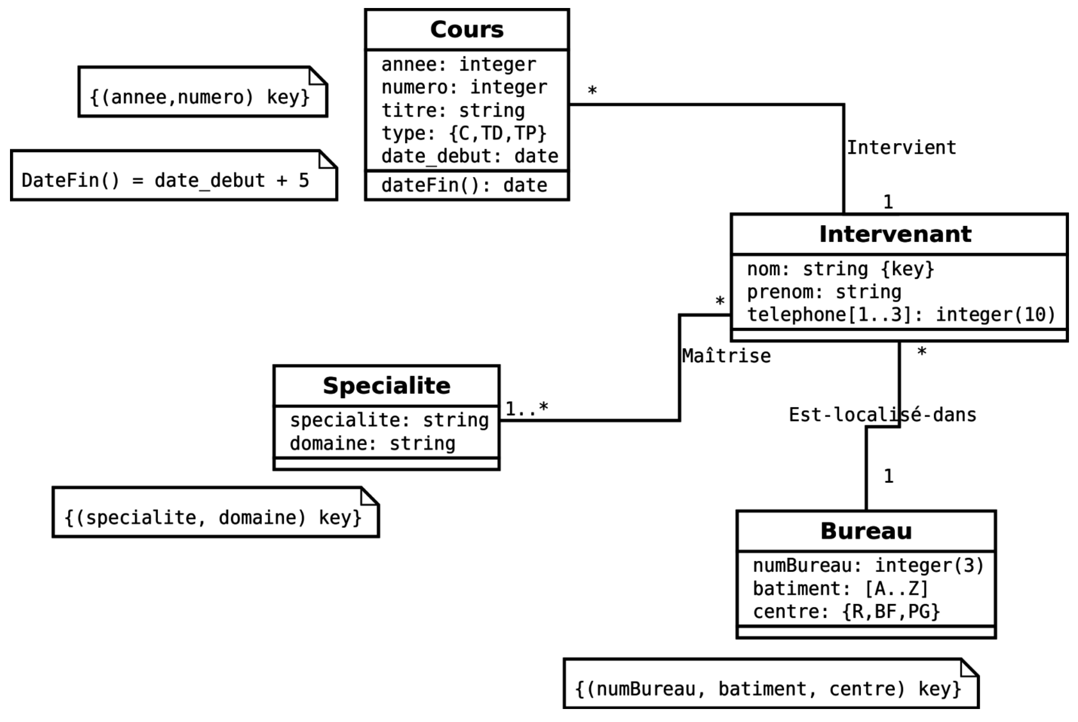
> **Solution** n°7

Exercice p. 19



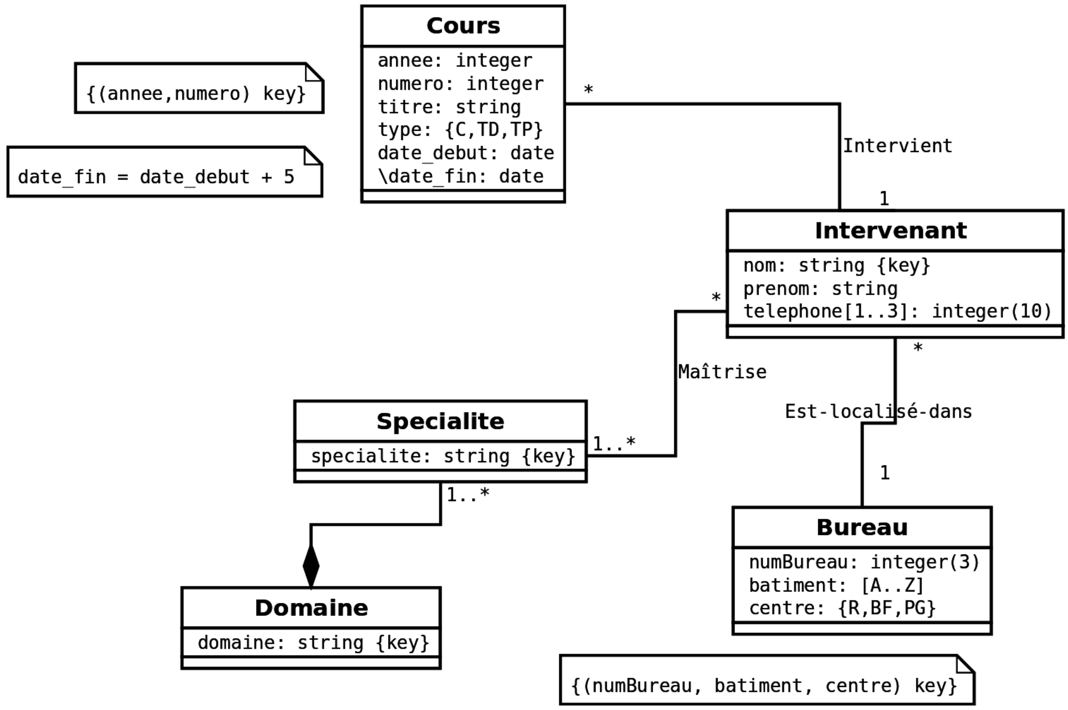
> **Solution** n°8

Exercice p. 19



- Certaines cardinalités peuvent être discutées, par exemple, rien n'indique que les bureaux sont partagés, e donc on pourrait adopter une association 1:1.
- La proposition d'une unique classe `Specialite` est alternative au couple de classes `Domaine-Specialite`.
- La date de fin peut être posée en attribut dérivé ou en méthode.

Complément : Solution alternative

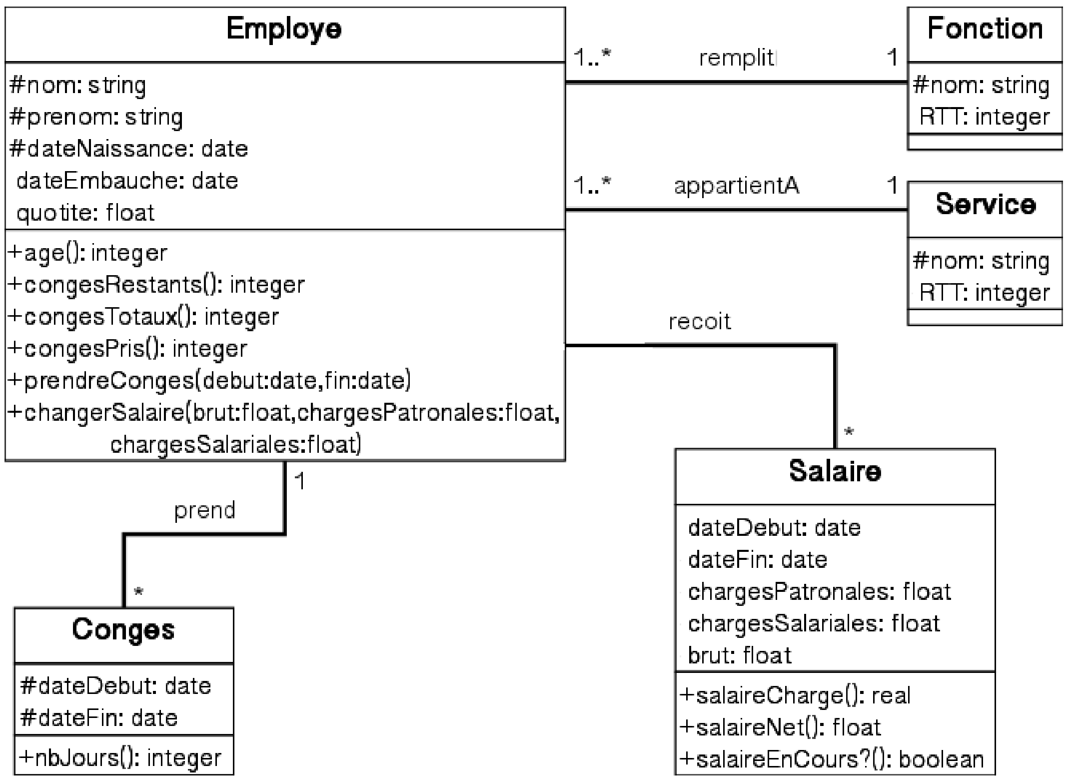


- *Composition (cf. p.)*

> **Solution** n°9

Exercice p. 20

Exemple



Conseil

On préférera la notation *{key}* plutôt que # pour repérer les clés.

Remarque

On remarquera les méthodes *prendreConges* et *changerSalaire* qui modélisent des opérations courantes dont la complexité (elles concernent plusieurs tables) et l'importance dans l'énoncé justifient une mise en exergue.

Bien entendu, il n'existe pas de solution unique à ce problème. Par exemple si seul le salaire en cours est réellement intéressant, on pourra affecter les méthodes *salaireCharge* et *salaireNet* à la classe *Employe* plutôt que *Salaire*.

Méthode : Pourquoi représenter des méthodes en UML en base de données ?

- Pour rendre l'UML plus explicite, la méthode est alors une technique d'annotation
- Pour montrer qu'elles sont importantes et s'en souvenir au moment du développement de l'application (on met ainsi en exergue certaines fonctions énoncées lors de la clarification)
- Parce qu'elle peuvent donner lieu à des *vues* intéressantes en relationnel
- Parce que si on fait une BD en relationnel-objet (par exemple) on pourra implémenter ces méthodes dans le modèle.

Complément : Exemple avec composition et explicitation des associations

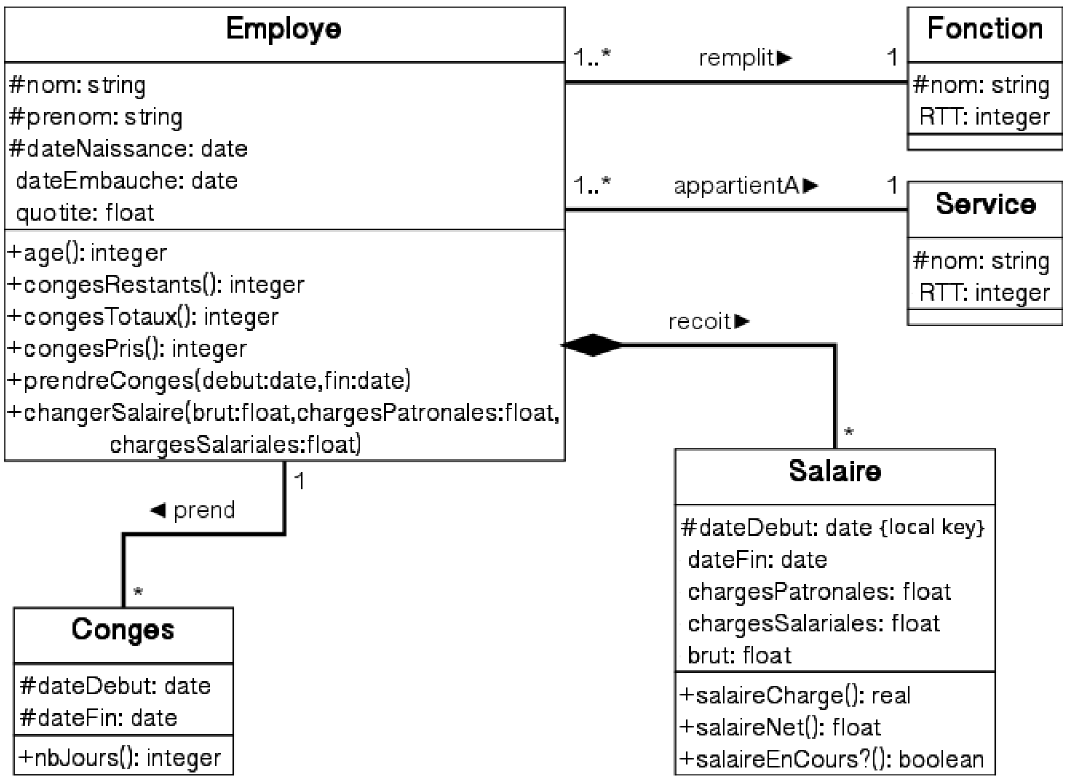


Diagramme de classes de gestion du personnel

- *Explicitation des associations (cf. p.)*
- *Composition (cf. p.)*

Glossaire

Clé artificielle (surrogate key)

En relationnel une clé artificielle est un attribut artificiel (qui ne représente aucune donnée) ajouté à la relation pour servir de clé primaire. On mobilise cette technique lorsque l'on n'a pas pu ou voulu choisir une clé naturelle pour clé primaire.

Abréviations

BD : Base de Données

E-A : Entité-Association

OMG : Object Management Group

SGBD : Système de Gestion de Bases de Données

UML : Unified Modeling Language

Bibliographie

Arribe Thibaut. 2014. *Conception des chaînes éditoriales : documentariser l'activité et structurer le graphe documentaire pour améliorer la maîtrise de la rééditorialisation*. Université de Technologie de Compiègne, Mémoire de Doctorat. http://ics.utc.fr/~tha.

Gardarin Georges. *Bases de données : objet et relationnel*. Eyrolles, 1999
.

Rothenberg Jeff, Widman Lawrence E, Loparo Kenneth A, Nielsen Norman R. 1989. *The nature of modeling*
. Rand. vol.3027.