

MSIS Chapitre 5

Patrons de modélisation de données

Qu'est-ce qu'un patron?

Un patron (« ***pattern*** ») désigne souvent un modèle, une structure, un motif, un type, etc. réutilisable dans un contexte similaire.

C'est un instrument pour la réutilisation des meilleures pratiques, le transfert de connaissances et l'apprentissage.

Un patron constitue une base de savoir et de savoir-faire pour résoudre un problème récurrent dans un domaine. Il inclut :

- l'identification d'une **situation remarquable** – situation typique dans le domaine en question,
- l'identification d'un **problème récurrent à résoudre** dans cette situation,
- la caractérisation d'une **bonne solution** – un modèle à suivre ou imiter.

Patron de modélisation de données

- Situation :** une définition particulière d'un attribut, d'une classe, d'une association ou d'une composition de ces éléments
- Problème :** un problème de traitement de données, de cohérence d'information ou d'intégrité de données dans cette situation
- Solution :** un modèle générique qu'on peut imiter à chaque fois qu'on rencontre le problème

Dans ce cours

Règle générale d'unicité d'une classe

Les patrons de

- Normalisation des classes     
- Historisation  
- Transformation des associations     

Unicité d'une classe

Chaque concept ne doit être modélisé qu'une seule fois – n'apparaître qu'une seule fois dans le diagramme des classes.

Il faut identifier et éliminer les synonymes et les homonymes.

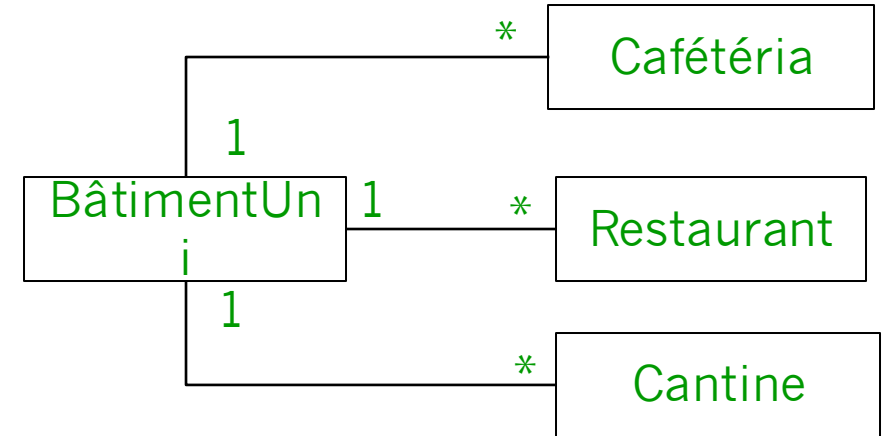
La présence d'homonymes ou de synonymes provient du fait que :

- on a attribué le même nom à deux concepts ayant une sémantique différente
→ compléter ou changer le nom d'une ou plusieurs classes
- on a modélisé deux fois le même objet sémantique
→ éliminer une des classes et restructurer le diagramme

Unicité d'une classe

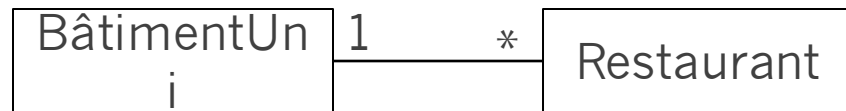
Exemple : gestion des lieux de restauration d'une université

Cantine, Restaurant et Cafétéria représentent les concepts ayant une sémantique similaire, voire identique.

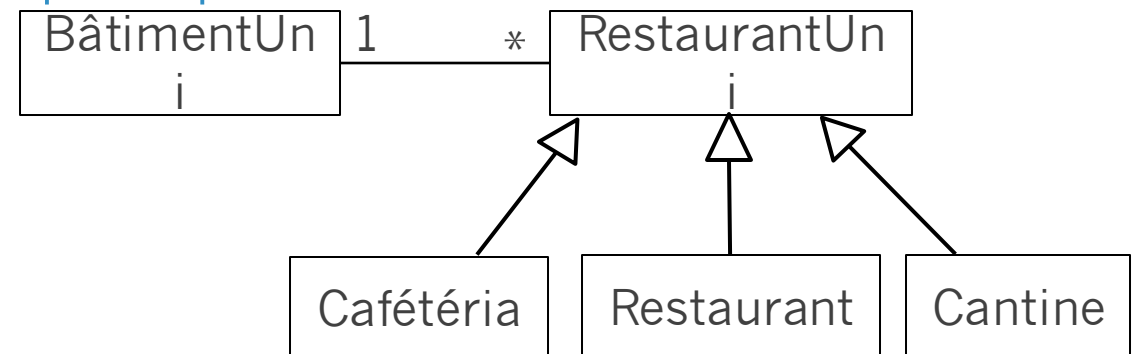


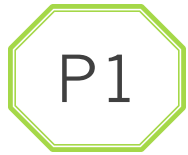
Solutions

1. Fusionner les trois classes, si aucune différence entre les trois concepts.



2. Créer une spécialisation des classes si chaque concept a des propriétés spécifiques.





Attribut composite

Classe A
attribut composite (a1 a2 a3 ...)

Situation : attribut composite d'une classe

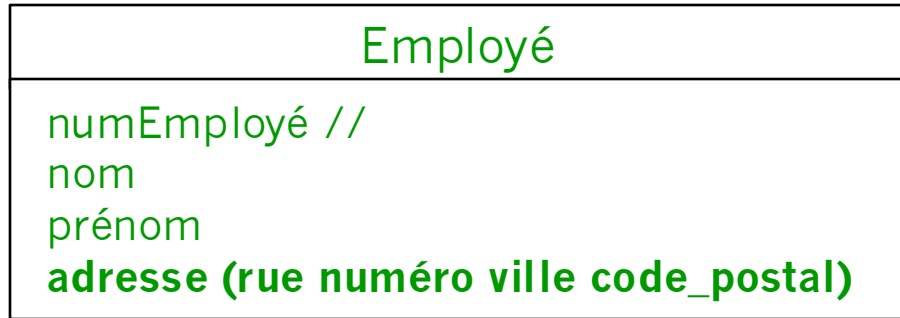
Problème : la classe n'est pas en forme normale

Solution : trois cas ci-dessous

	Opérations sur les composants de l'attribut	La valeur de l'attribut peut être partagée par plusieurs objets	Plusieurs valeurs pour l'attribut
1 Comme un attribut simple (de type Texte)	Non	Oui (mais répétition)	Non
2 Décomposition en plusieurs attributs	Oui	Oui (mais répétition)	Non
3 Création d'une nouvelle classe	Oui	Oui	Oui

P1

Modélisation d'un attribut composite



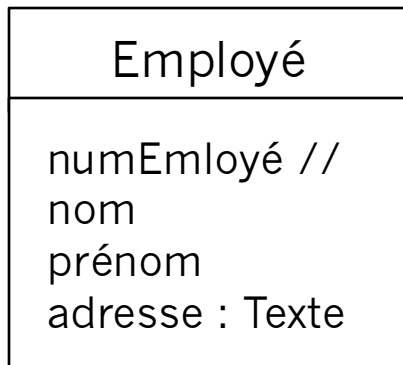
Questions à se poser :

Est-ce qu'il faudra rechercher les employés par ville ?

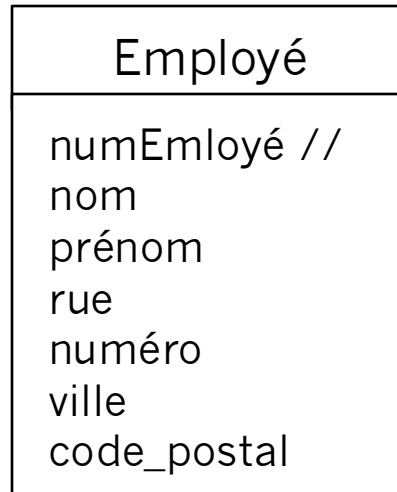
Est-ce qu'on veut connaître plusieurs adresses pour un employé ?

Est-ce qu'on veut garder la trace des changements d'adresse?

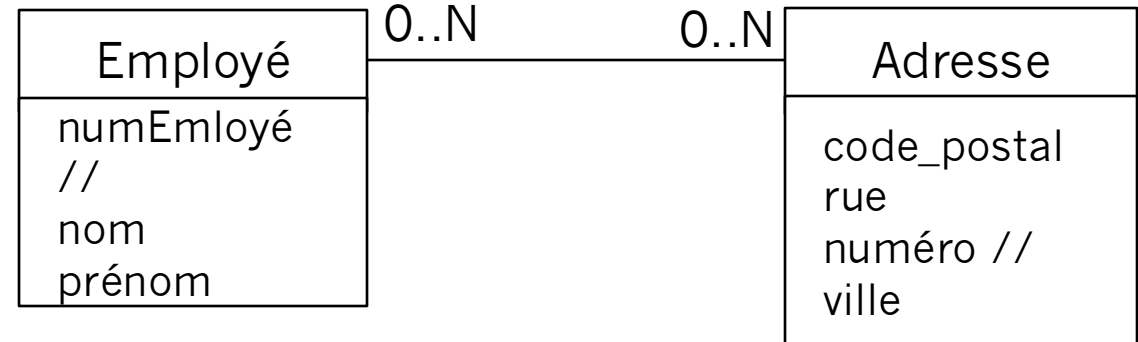
1



2



3



P1

Modélisation d'un attribut composite

1

Employé
numEmployé //
nom
prénom
adresse : Texte

En langage relationnel

Employé (numEmployé // nom, prénom, adresse)

numEmployé : Mot

nom : Texte

prénom : Texte

adresse : Texte

2

Employé
numEmployé //
nom
prénom
rue
numéro
ville
code_postal

Employé (numEmployé // nom, prénom, rue, numéro, ville, code-postal)

rue : Texte

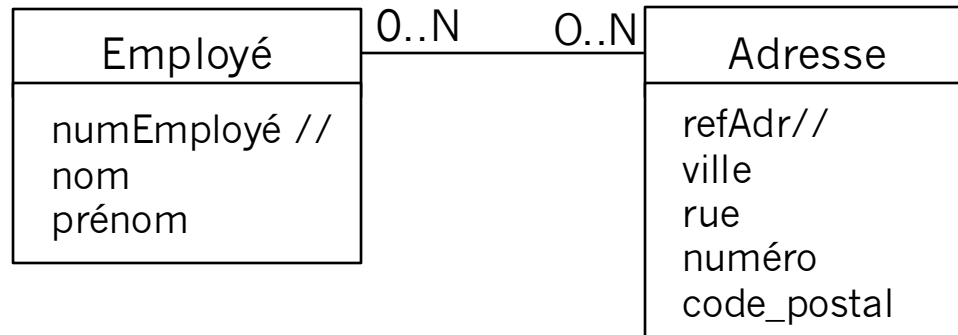
numéro : Mot

ville : Texte

code-postal : Mot

typeAdresse : Mot

3



Employé (numEmployé // nom, prénom)

Adresse (refAdr / code-postal rue numéro // ville)

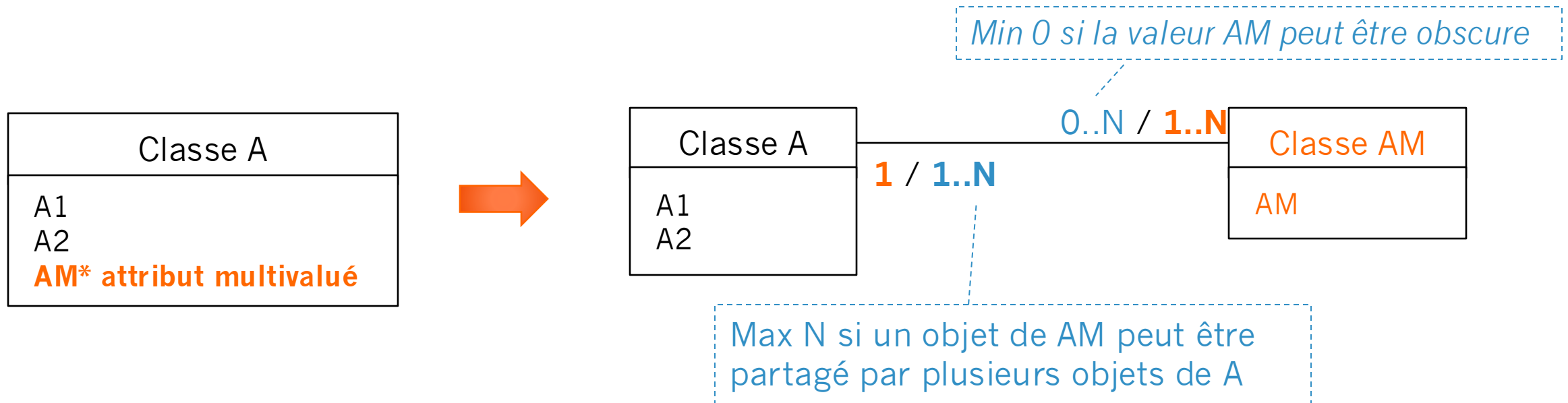
Employé-Adresse (numEmployé refAdr // typeAdresse)

Attribut multivalué

Situation : attribut multivalué d'une classe

Problème : la classe n'est pas en forme normale

Solution : création d'une **nouvelle classe** pour un attribut multivalué



P2

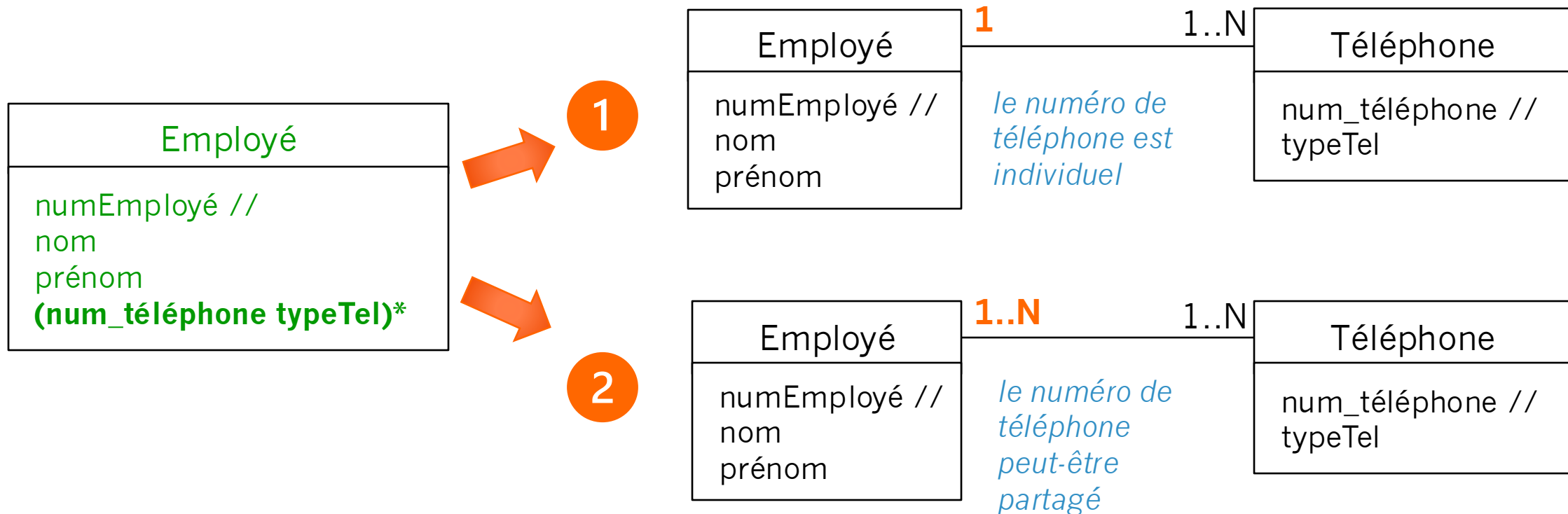
Modélisation d'un attribut multivalué

Exemple : tout employé peut posséder plusieurs numéros de téléphone (bureau, maison, mobile,...)

Questions à se poser :

Est-ce qu'un numéro peut être partagé par plusieurs employés ?

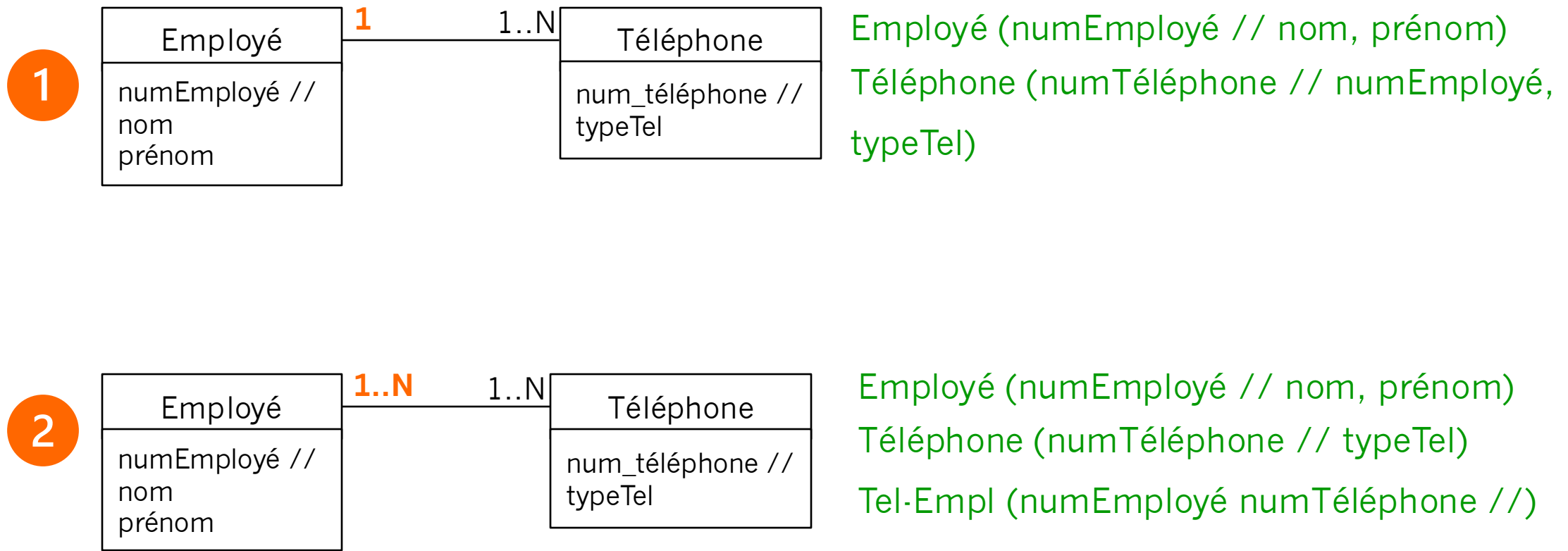
Est-ce qu'un numéro de téléphone peut exister sans être associé à au moins un employé ?



P2

Modélisation d'un attribut multivalué

Traduction en langage relationnel



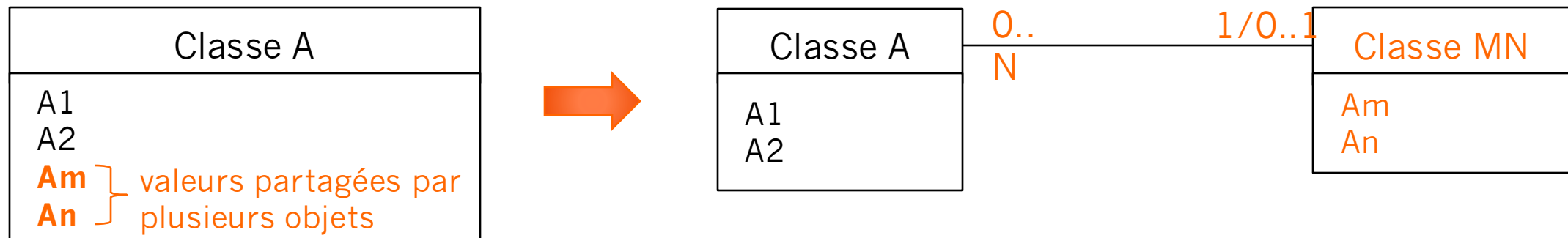
Décomposition d'une classe

Situation :

- plusieurs objets partagent les mêmes valeurs pour un ou plusieurs attributs de la classe et/ou
- un ou plusieurs attributs expriment un concept du monde réel représentable sous forme d'une classe

Problème : redondance d'information

Solution : extraire ces attributs dans une (ou plusieurs) nouvelle(s) classe(s)

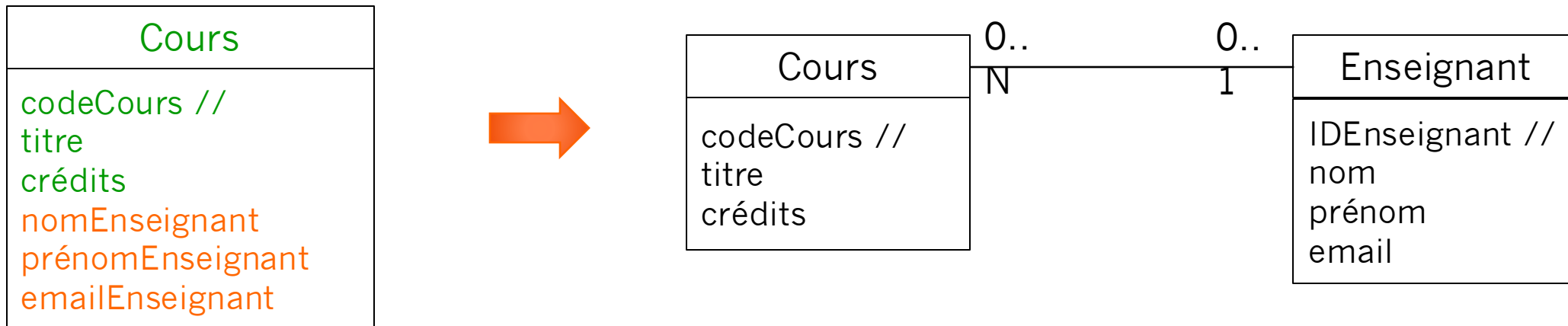


Décomposition d'une classe

Exemple : plusieurs cours ont le même enseignant.

Problème : répétitions de l'information concernant l'enseignant pour tous les cours concernés.

Solution : extraire les attributs concernant l'enseignant dans une autre classe.

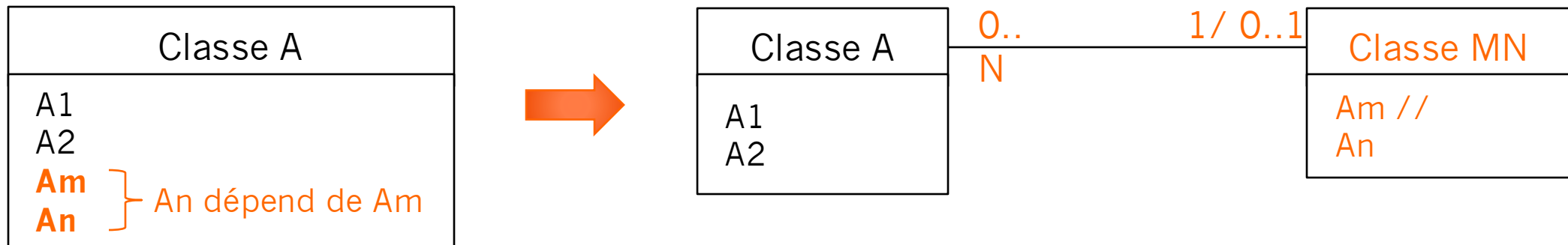


Décomposition d'une classe

Situation : il y a une dépendance fonctionnelle entre les attributs de la classe (nous verrons le concept de dépendance fonctionnelle dans le cours de bases de données)

Problème : redondance d'information

Solution : extraire ces attributs dans une nouvelle classe

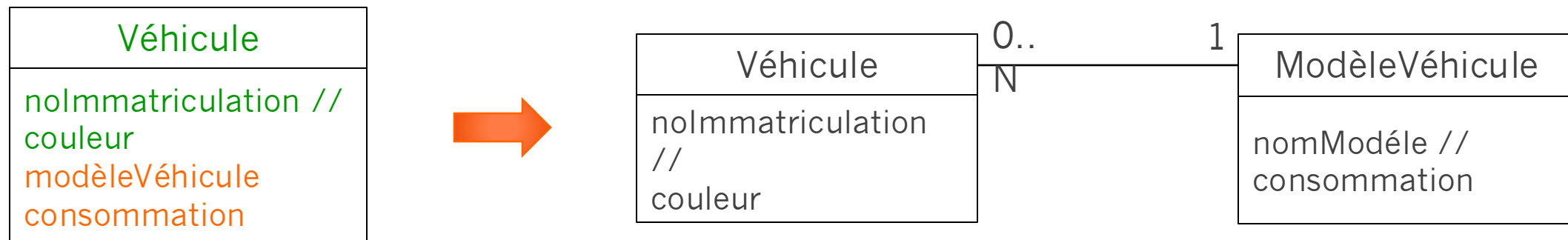


Décomposition d'une classe

Exemple : la consommation d'un véhicule dépend du modèle et non du nomImmatriculation.

Problème : répétitions de l'information concernant la consommation pour tous les véhicules de même modèle.

Solution : extraire les attributs concernant le modèle dans une autre classe.



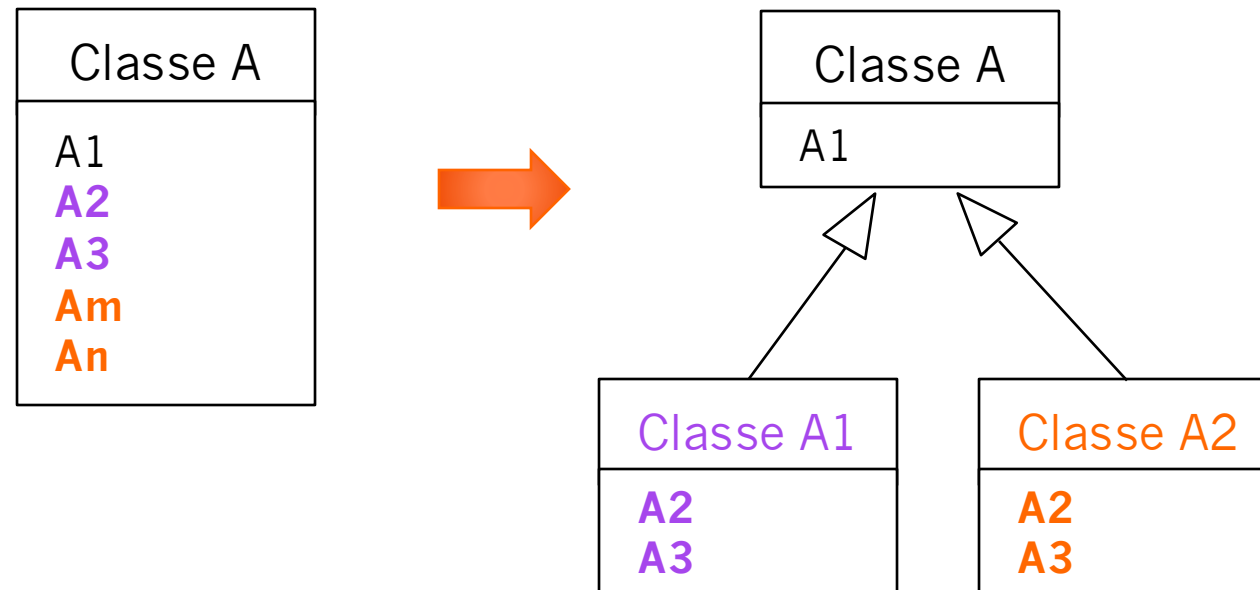
Valeur inexistante des attributs

Situation : une classe contient des attributs prenant la valeur « inexistante » en fonction de la valeur des autres attributs

Problème : clarté de la sémantique des attributs

Solution : transformer la classe en une structure de généralisation/spécialisation

Si les valeurs de A2 et A3 sont renseignées, alors Am et An ne peuvent pas avoir de valeurs et inversement



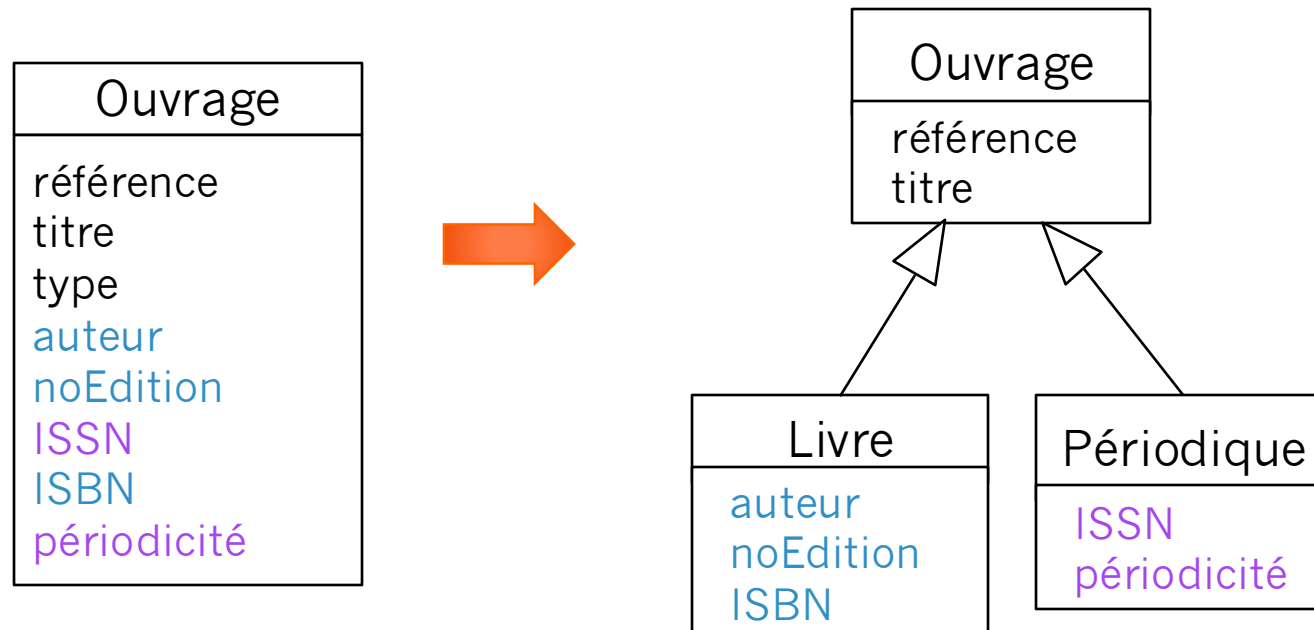
Valeur inexistante des attributs

Exemple : une bibliothèque possède deux types d'ouvrage : livres et périodiques.

Les objets de type livre ne peuvent pas avoir de valeur des attributs ISSN et périodicité.

Les objets de type périodique ne peuvent pas avoir de valeur des attributs auteur, noEdition, ISBN.

Solution : spécialiser la classe Ouvrage en Livres et Périodiques.



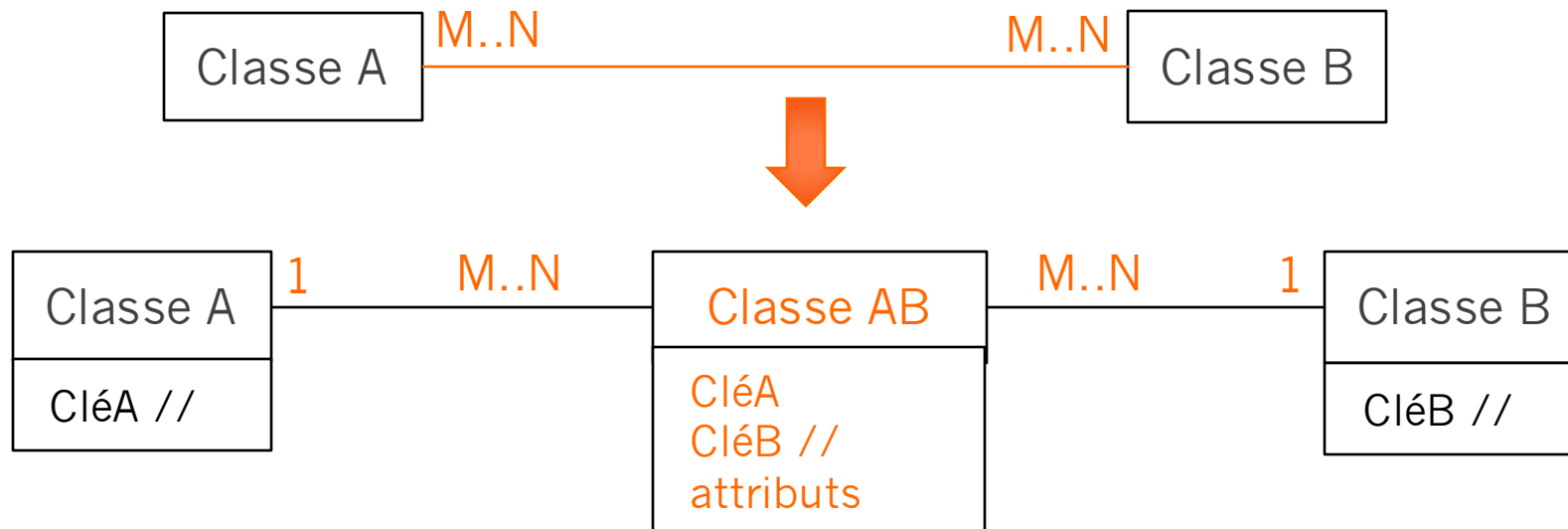
Association multivaluée (M..N)

Situation : Les deux rôles de l'association ont la cardinalité M..N.

Problèmes :

- Ajout des attributs à l'association
- Evolution du modèle – besoin de lier cette association à d'autres classes ou de la spécialiser
- Garantie des règles d'intégrité (RI)

Solution : transformer l'association M-N en classe (d'association).





Transformation d'une association M-N en classe

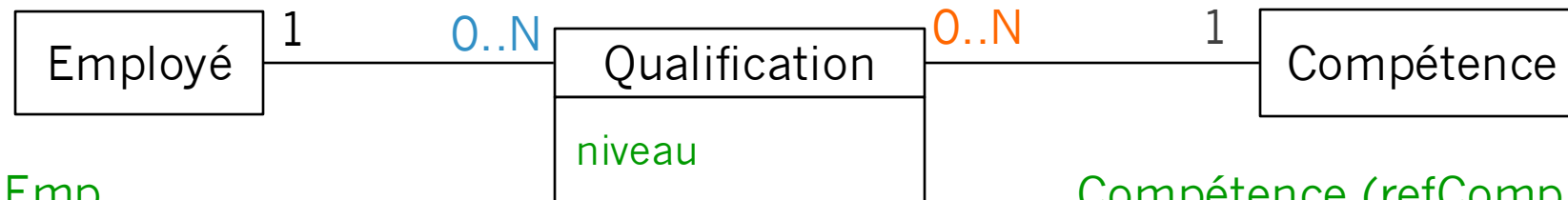
Cas simple : Gestion des compétences des employés

On veut connaître les compétences des employés.

Les compétences sont génériques et partagées par plusieurs employés.



Pour chaque compétence d'un employé on veut connaître son **niveau** (basique, moyen, avancé, expert).



Employé (idEmp
//nom,...)

Qualification (idEmp refComp //niveau)

Compétence (refComp //intitulé, ...)

P6

Transformation d'une association M-N en classe

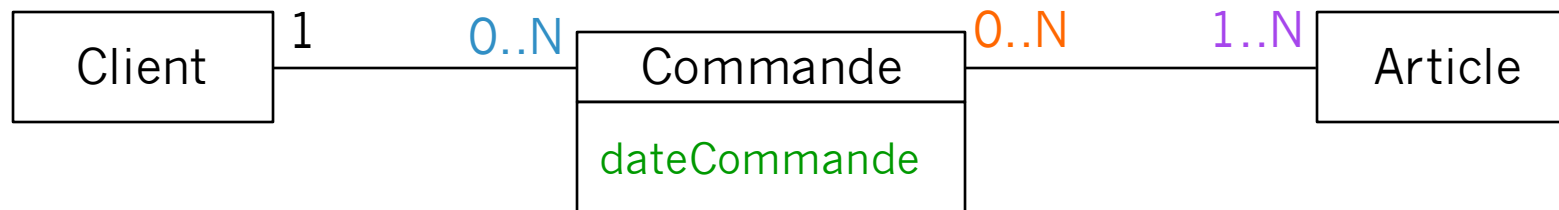
Cas plus complexe : Gestion des commandes

Un client peut commander plusieurs articles à la fois.

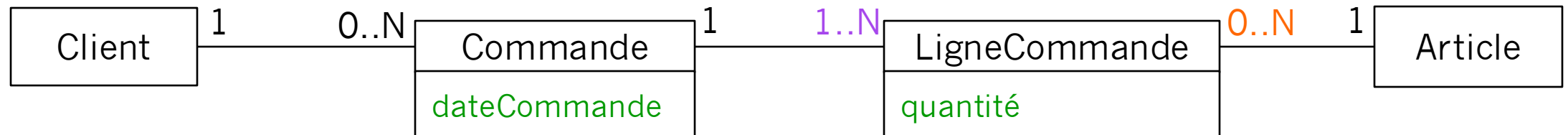


On veut connaître la **date** de commande et la **quantité** commandée pour chaque article.

①



②



P7

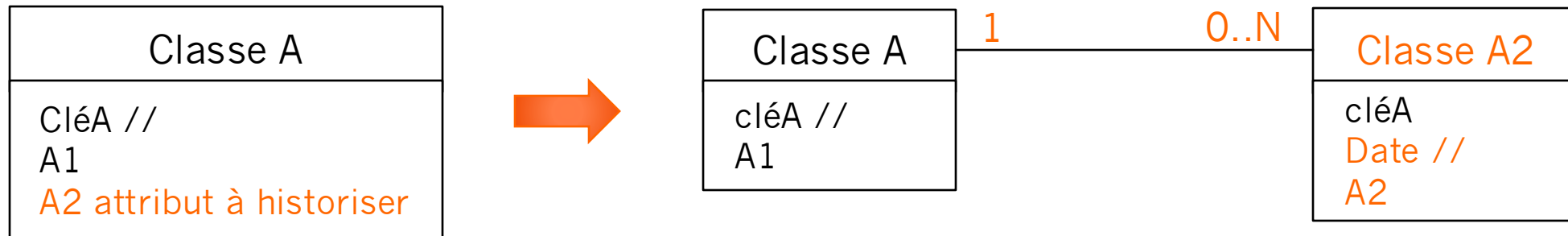
Historisation des valeurs d'un attribut

Situation : la valeur d'un attribut est sujet au changement.

Problème : il est nécessaire de garder la trace de tous les changements de sa valeur.

Solution :

- Extraire l'attribut concerné dans une nouvelle classe.
- Ajouter un attribut de type date (ou autre type d'historisation, ex. numéro d'ordre).
- L'attribut date fait partie de la clé de la nouvelle classe.



P7

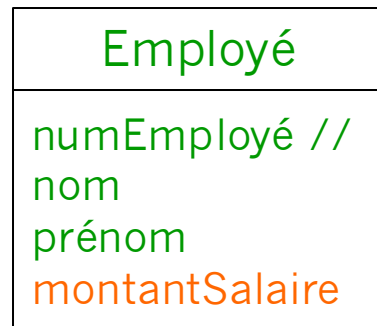
Historisation des valeurs d'un attribut

Exemple : Gestion des employés

On veut garder la trace des changements de salaire.

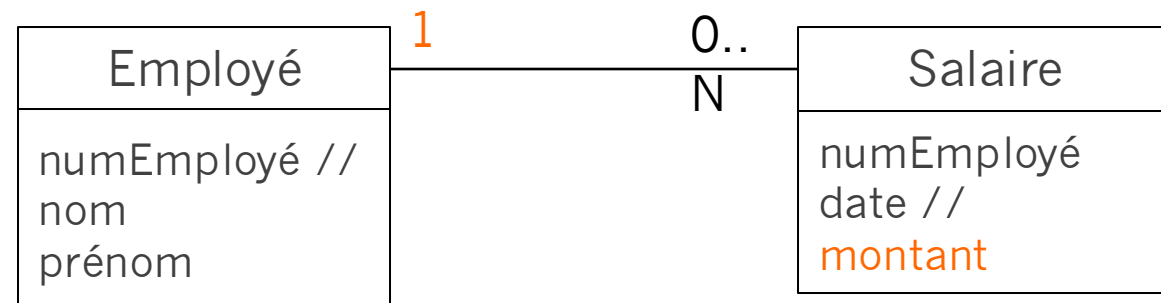
Sans historisation

on ne garde que le montant
du salaire en cours



Avec historisation

on garde tous les salaires de l'employé



Si le salaire change on
ne saura plus quel était
le montant précédent.

La classe Salaire dépend existentiellement de la classe Employé.
Le salaire reçu par un employé n'appartient qu'à cet employé.

Historisation d'une association

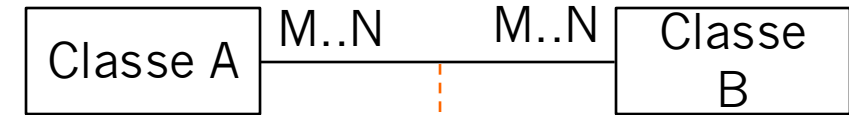
Situation : l'association multivaluée a une sémantique temporelle

Cas particulier: un objet de la classe A peut être associé au même objet de la classe B plusieurs fois à des moments différents de son existence.

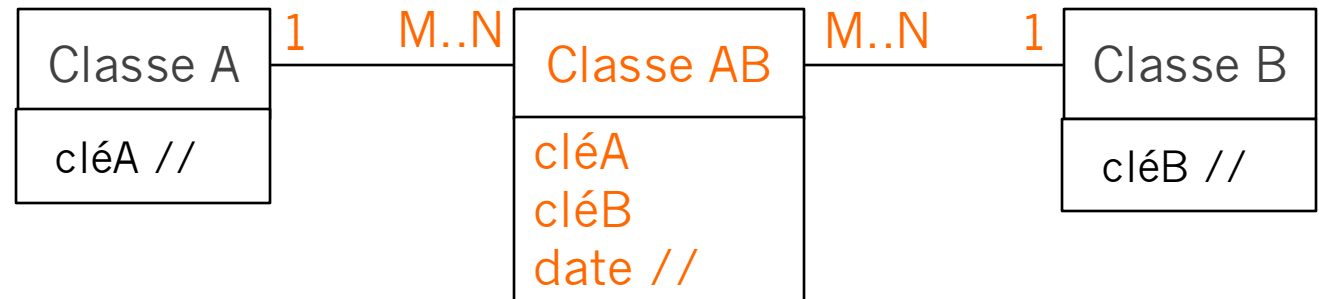
Problème : il est nécessaire de connaître la dimension temporelle de chaque lien.

Solution :

- Transformer l'association en une nouvelle classe.
- Ajouter un attribut de type date dans la clé de la nouvelle classe.



date

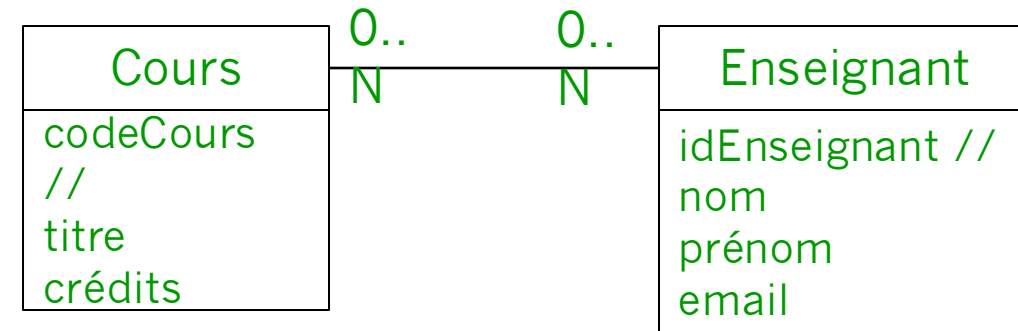


Historisation d'une association

Exemple : Responsables des cours

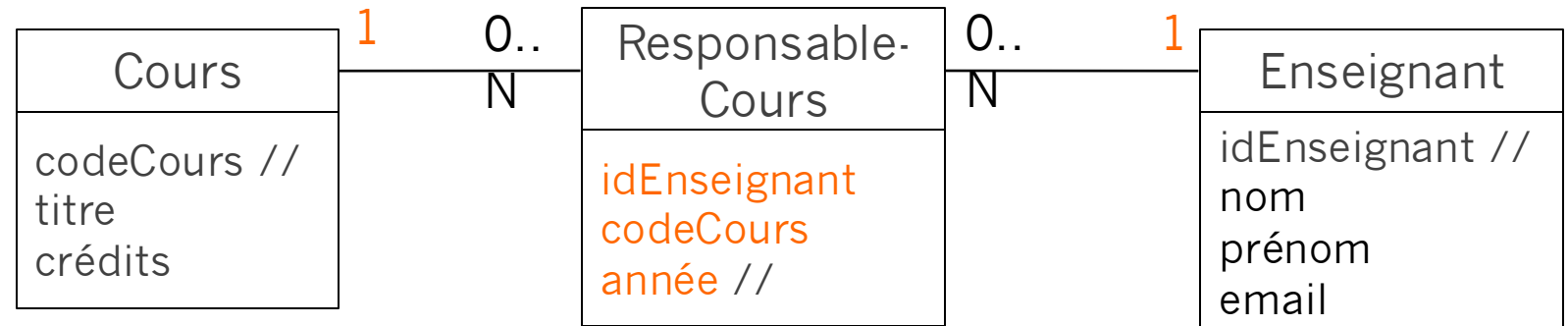
Sans historisation

On sait seulement qu'un enseignant est ou était responsable d'un cours mais on ne sait pas de quelle année s'agit-il.



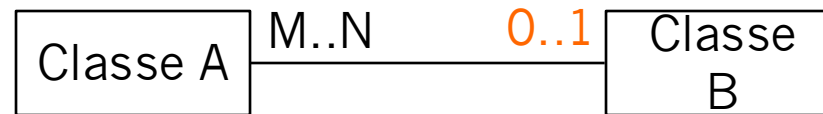
Avec historisation

On veut garder la trace de quel enseignant était responsable de quel cours d'une année académique à l'autre.



Association (0..1)

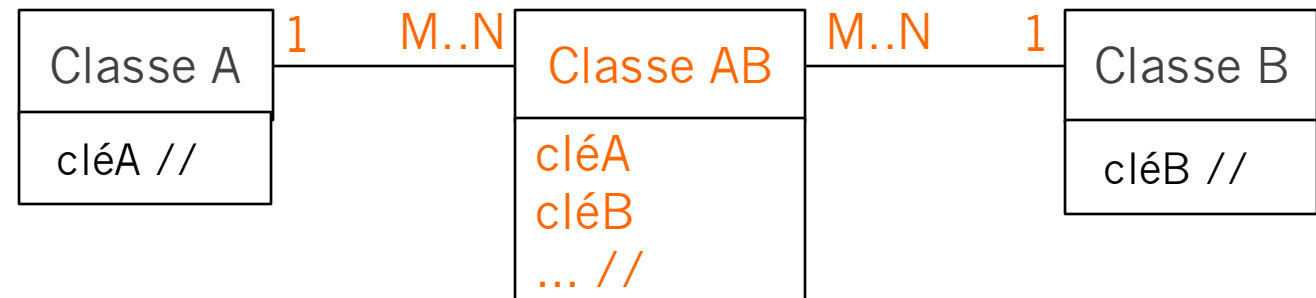
Situation : Dépendance référentielle entre la classe A et la classe B permet d'associer un objet de A à au plus d'un objet de B (cardinalité 0..1).



Problèmes :

- Cohérence d'information – si l'objet de B disparaît, l'objet de A ne peut plus être relié à l'objet de B.
- Evolution du modèle – changement des règles de gestion et des cardinalités est impossible.
- Historisation de l'association est impossible.

Solution : Transformer l'association en classe avec des dépendances existentielles

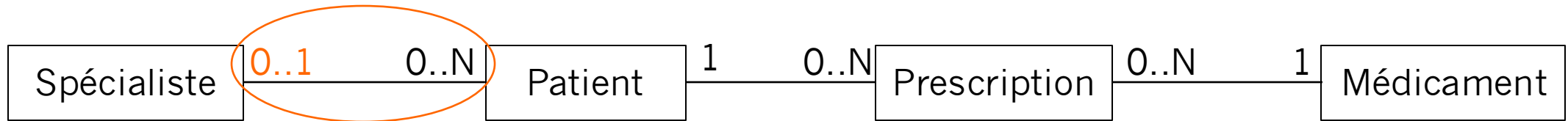


P9

Transformation d'une association 0..1

Exemple : Gestion des patients

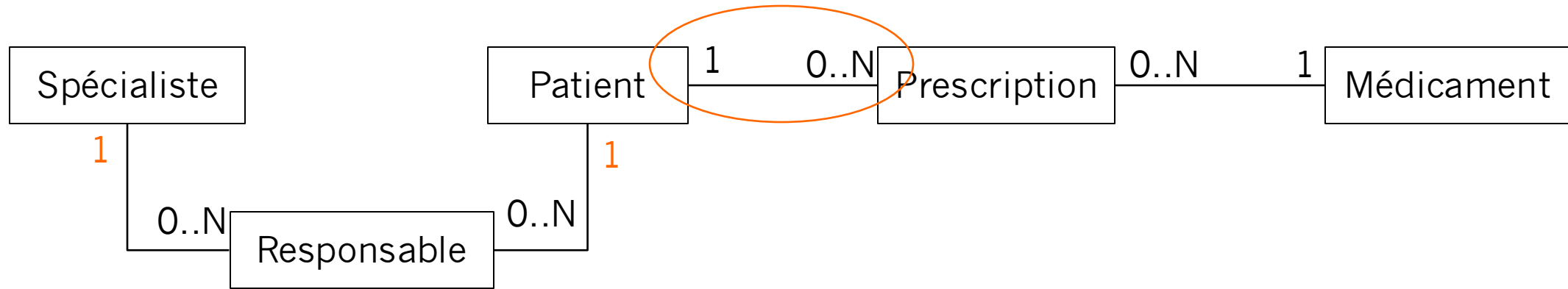
Un patient $p1$ est sous la responsabilité d'un seul spécialiste $s1$. Le patient a des médicaments à prendre prescrits par son spécialiste.



1. Si le patient $p1$ change de spécialiste en $s2$, que deviennent les prescriptions faites par $s1$? $s2$ les prend-il sous sa responsabilité?
2. Changement de règle de gestion: un patient peut maintenant avoir plusieurs spécialistes responsables ! Comment faire ?

P9

Transformation d'une association 0..1

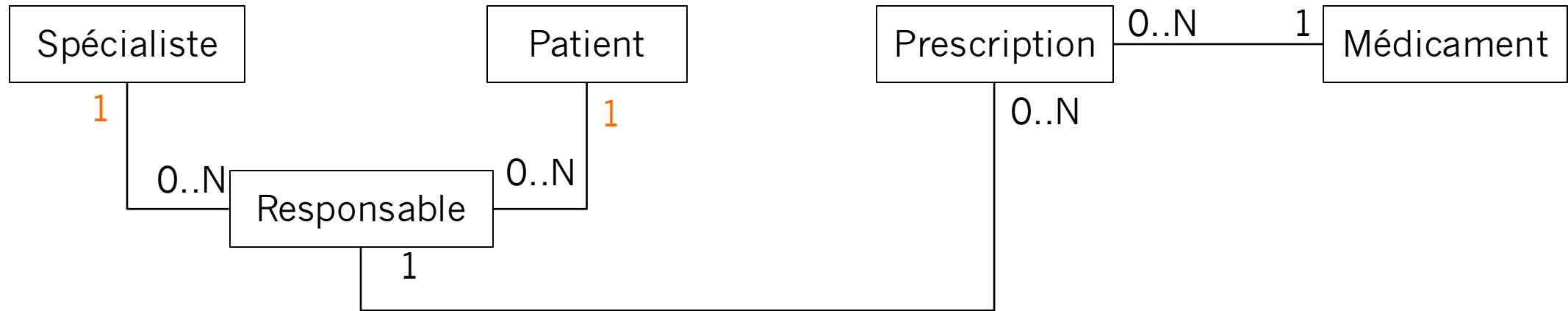


On introduit la notion de spécialiste responsable qui peut faire des prescriptions.
Un patient peut maintenant changer de spécialiste ou même avoir plusieurs spécialistes.

Mais quel spécialiste est responsable de quelle prescription ?

P9

Transformation d'une association 0..1



Ce modèle permet de faire l'historique des prescriptions associées à un patient.
On sait maintenant quel spécialiste a fait quelle prescription.

Inclusion sémantique des associations

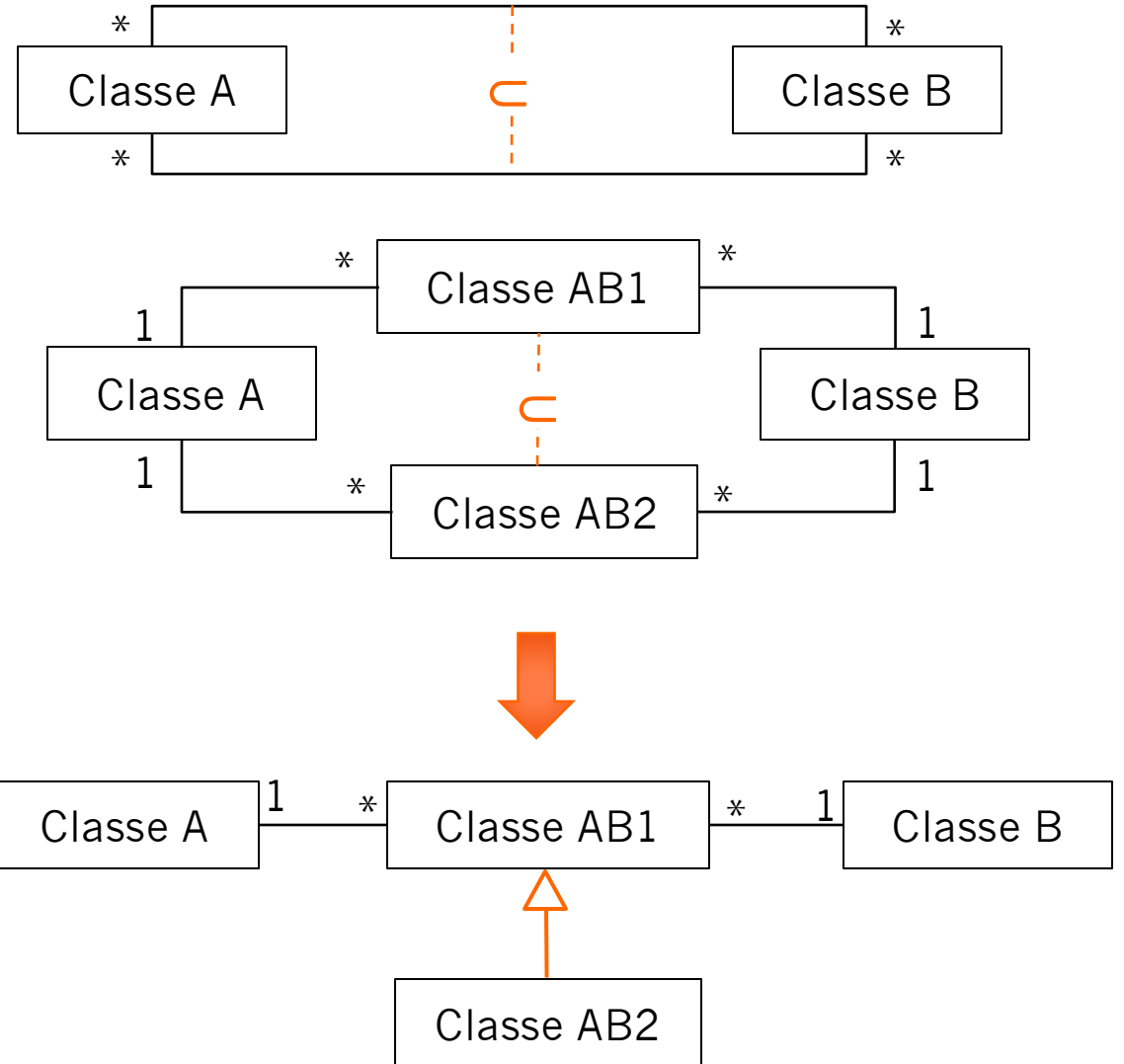
Situation :

- Il existe deux (ou plus) associations ou classes d'association entre deux classes.
- La sémantique d'une association est incluse dans la sémantique d'une autre association.

Problème : validation des règles d'intégrité – validation de la condition d'inclusion.

Tout objet de la classe AB2 doit correspondre à au moins un objet de la classe AB1.

Solution : spécialisation de la classe d'association.



Inclusion sémantique des associations

Exemple : Gestion des diplômes

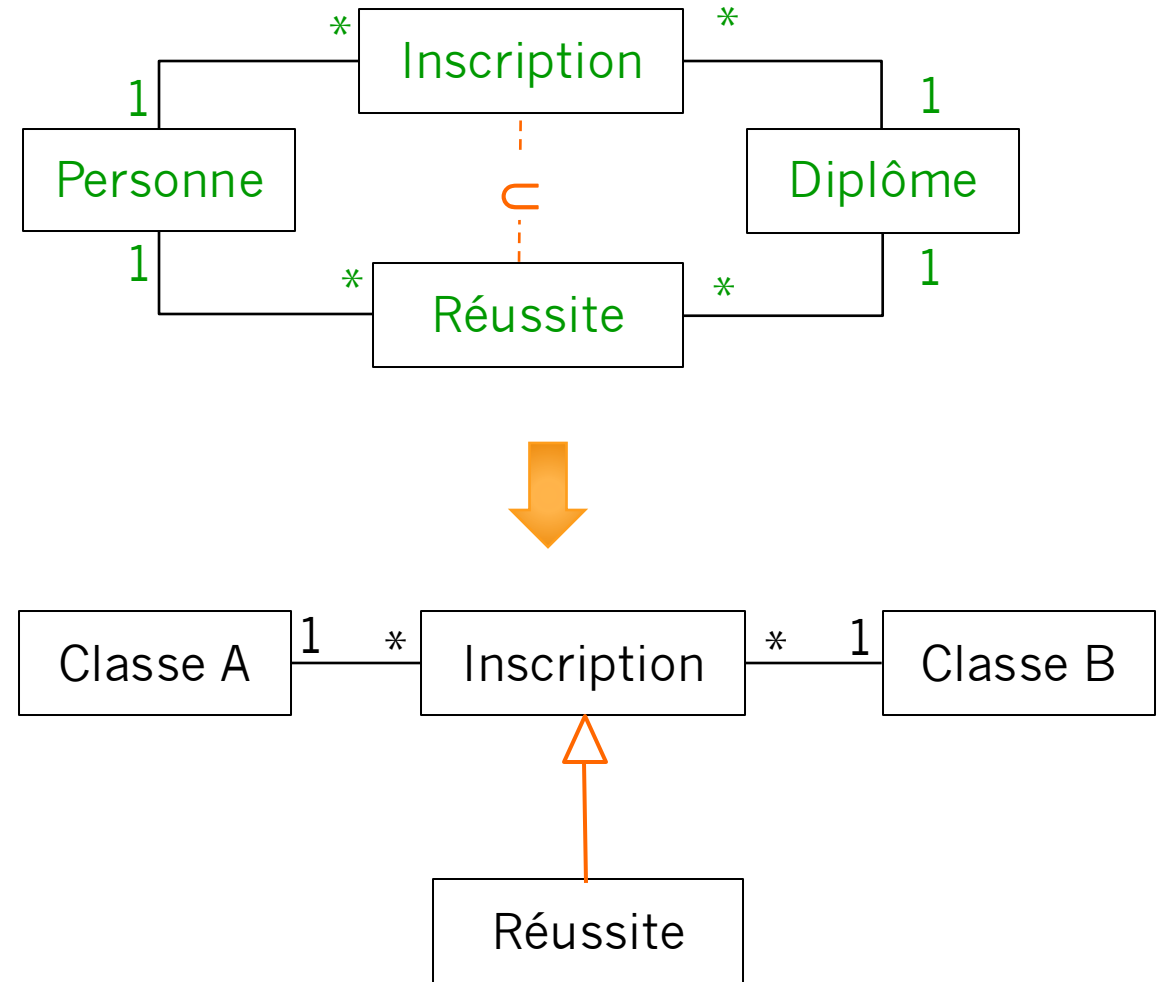
Une personne s'inscrit à un diplôme. Elle réussit le diplôme si elle satisfait aux exigences de celui-ci.

Une personne ne peut réussir un diplôme que si elle y est inscrite.

Au moment de la création d'un objet de Réussite ou suppression d'un objet d'Inscription il faut vérifier cette règle d'intégrité.

Spécialisation de la classe Inscription en une nouvelle classe Réussite.

Maintenant on ne peut créer un objet de Réussite seulement s'il existe un objet d'Inscription correspondant.



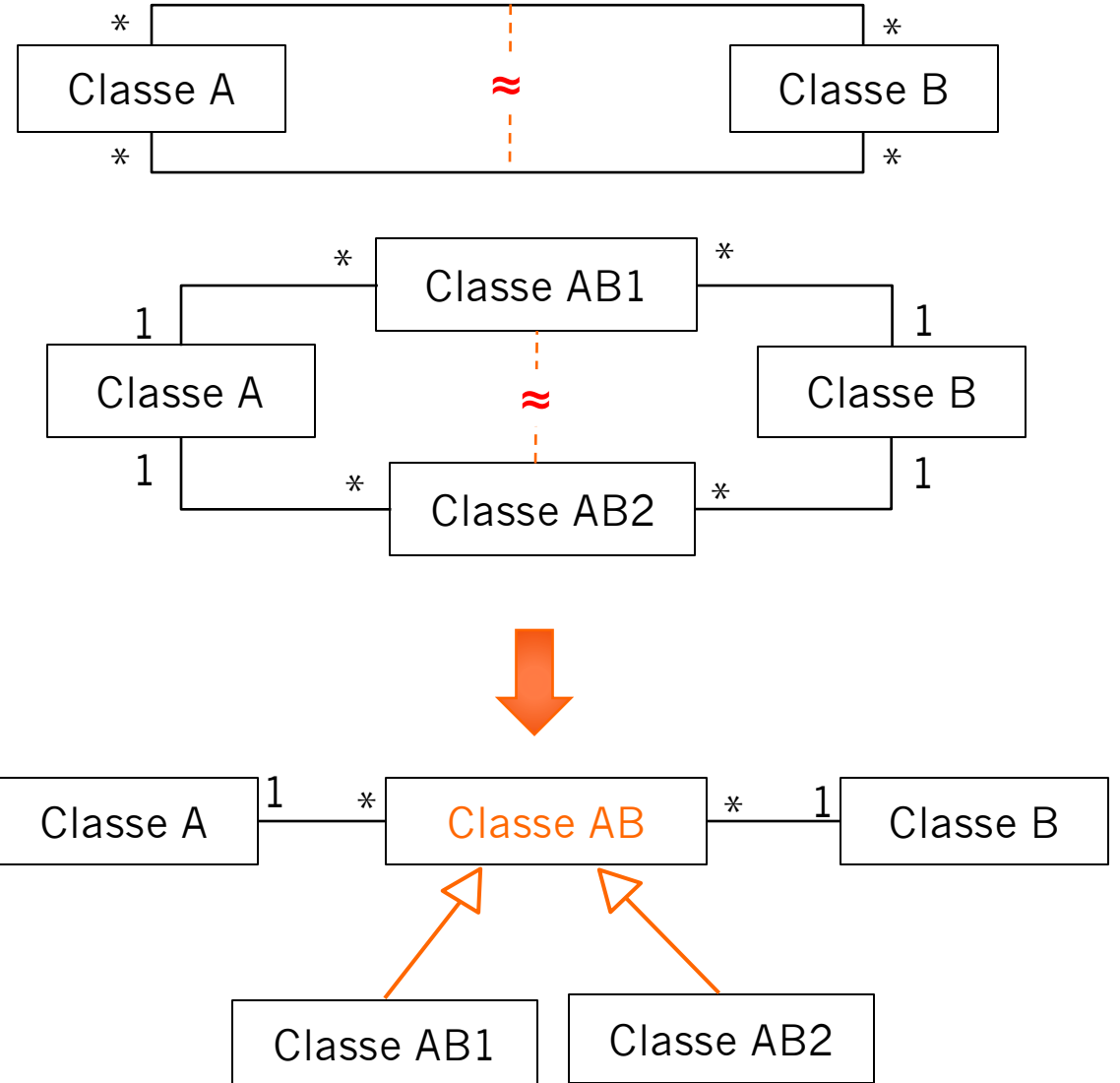
Interdépendance sémantique des associations

Situation :

- Il existe deux (ou plus) associations ou classes d'association entre deux classes.
- La sémantique des associations est proche – une partie peut être commune (*attributs communs dans les classes AB1 et AB2*).

Problème : Création des liens communs avec d'autres classes.

Solution : Généralisation des deux classes d'association en une nouvelle classe plus générique.



Interdépendance sémantique des associations

Exemple : Gestion des projets

Une personne travaille sur un projet en tant que concepteur et une autre en tant que développeur.

La même personne peut-elle être à la fois concepteur et développeur sur le même projet ?

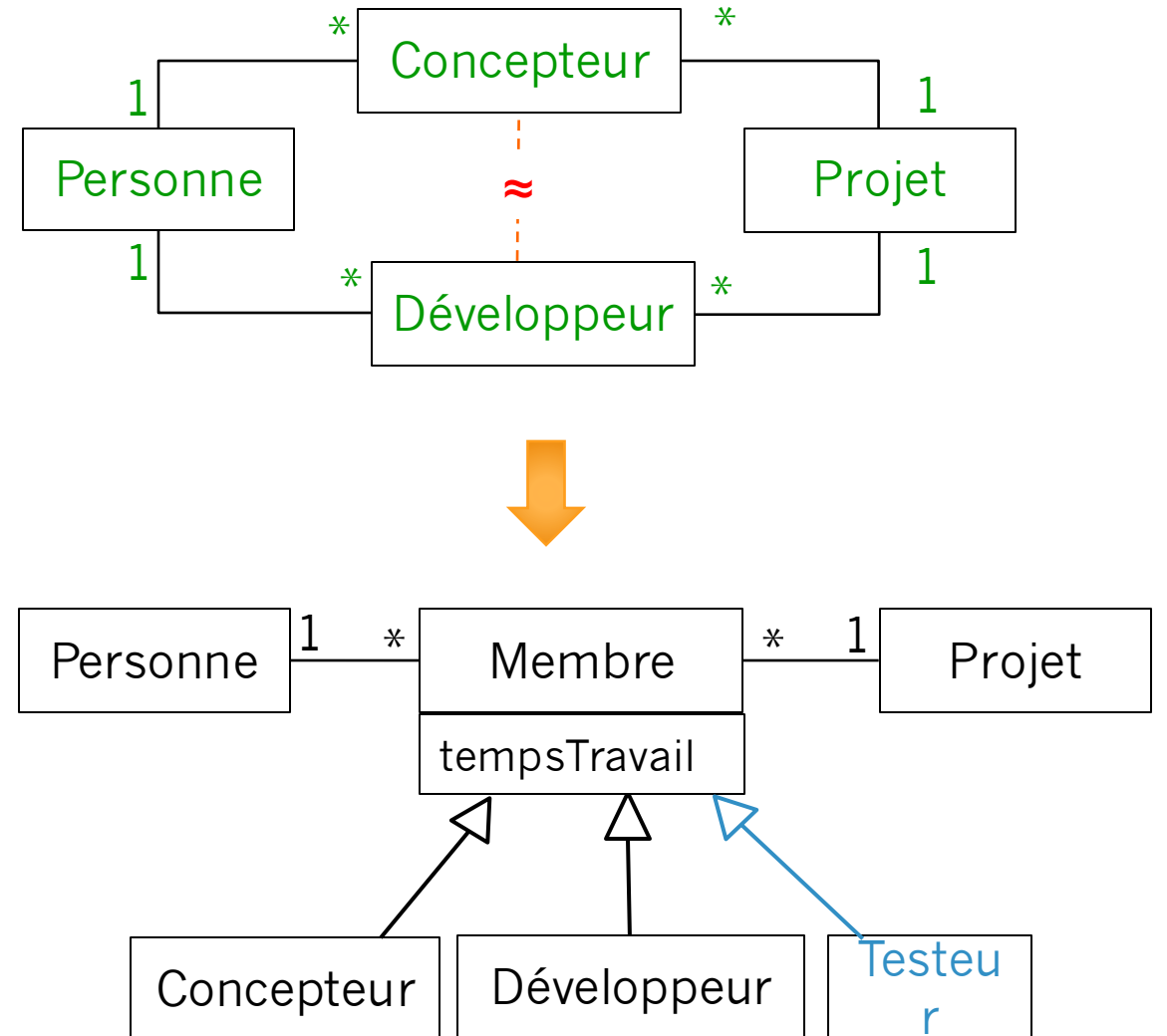
On veut connaître le temps de travail par personne et par projet.

Introduction de la classe Membre (de projet), générique aux classes Concepteur et Développeur.

Maintenant chaque personne peut participer au projet en tant que concepteur et/ou développeur.

On peut enregistrer le temps de travail de chaque personne par projet.

On peut aussi compléter la spécialisation avec d'autres rôles de membres de projet.



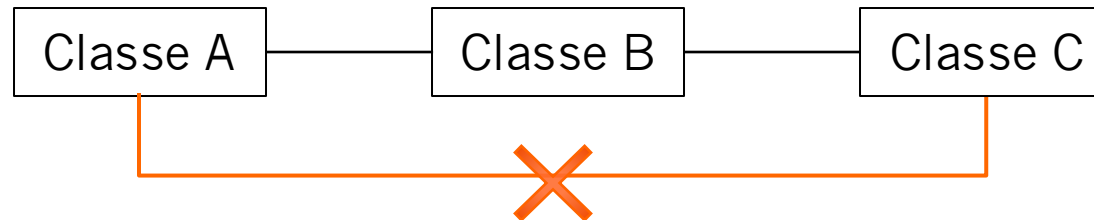
Association redondante

Situation : une association est sémantiquement redondante avec d'autres associations, notamment par transitivité

Problème : redondance d'information

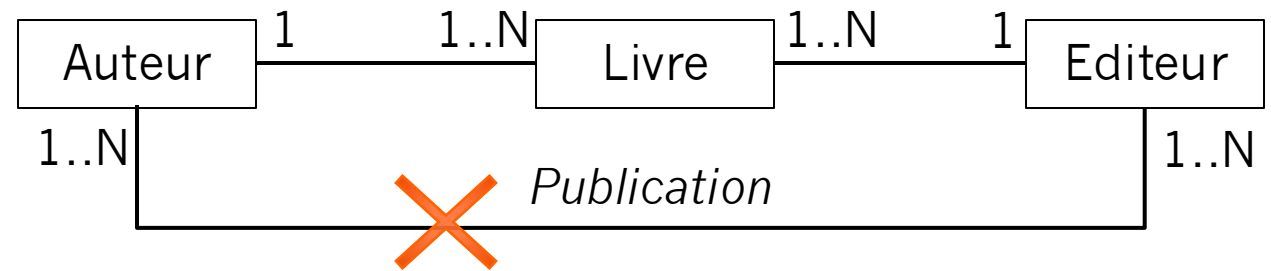
Solution : éliminer l'association redondante

Attention : avant d'éliminer il faut vérifier la sémantique et les cardinalités de l'association à éliminer



Association redondante

Exemple : on retrouve les éditeurs chez qui un auteur a publié par transitivité via la classe Livre.
L'association *Publication* est redondante.



Contre-exemple : L'association *Habitation* n'est pas redondante, elle ne peut pas être supprimée.

