UML

Diagramme de classes *(class diagram)* *pour le recueil et l'analyse des besoins*

Emmanuel Pichon

2013

V1.1

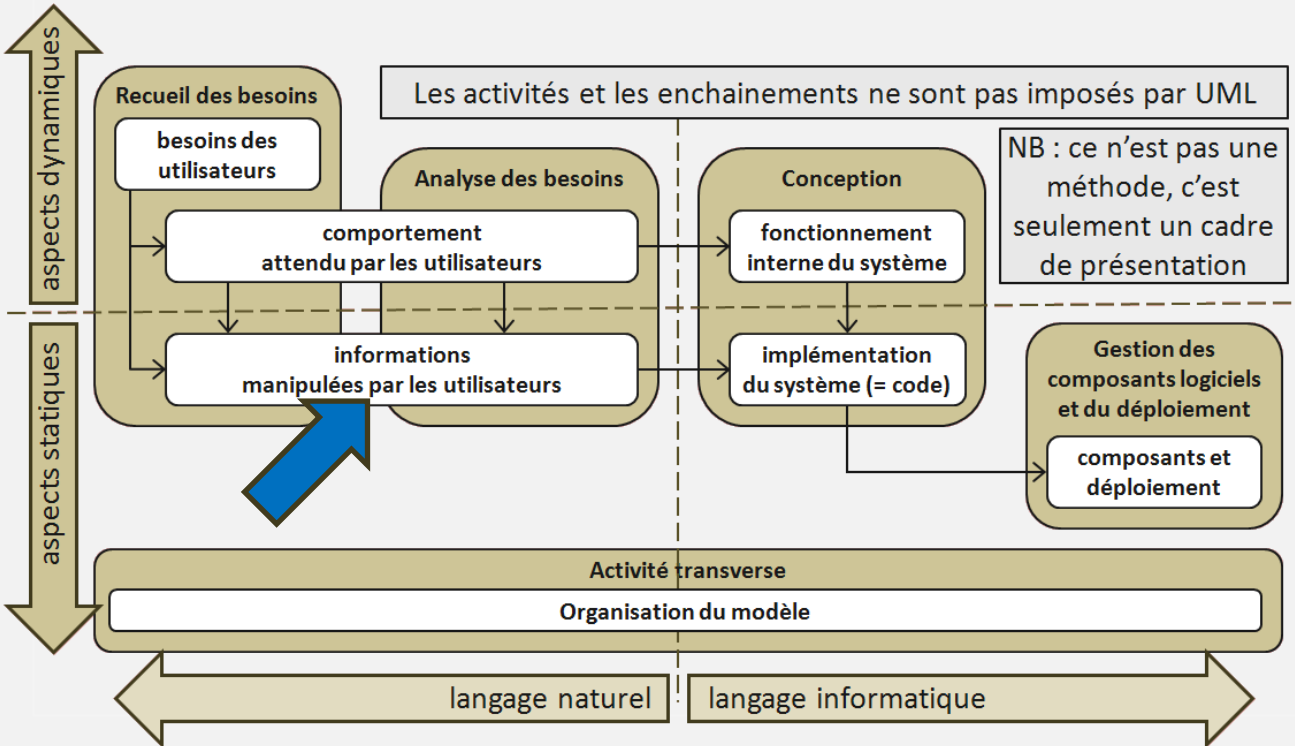
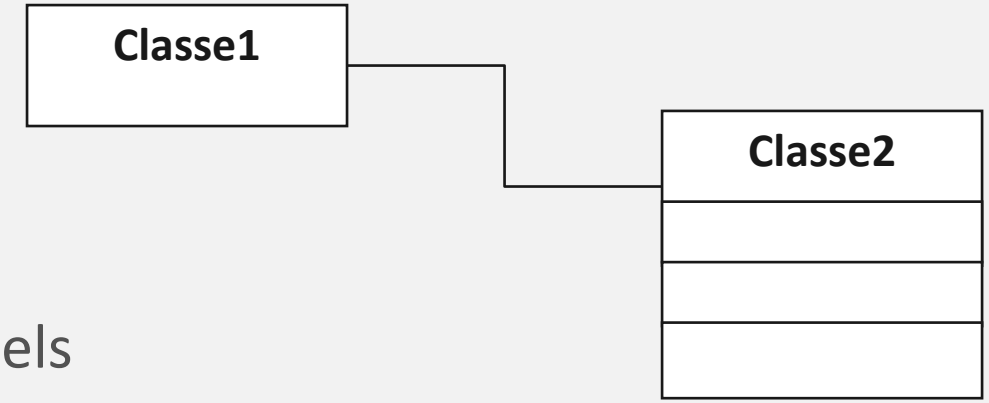


Diagramme de classes *(class diagram)* pour le recueil des besoins et l'analyse

- Objectif
  - Présenter un ensemble d'entités de gestion et leurs relations
  - Présenter le contenu de chaque entité (en général déconseillé pour la lisibilité des grands diagrammes)

- Contenu
  - Classes
  - Relations
  - Compartiments optionnels



Classe *(Class)*

- Sens
  - Une classe est un élément statique qui représente un ensemble d'objets (éléments dynamiques) qui partagent la même structure et le même comportement

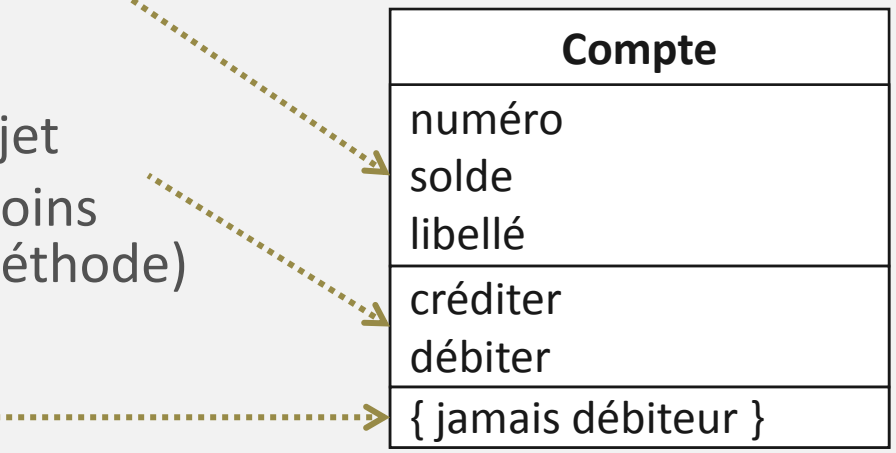
- Utilisation en recueil des besoins et en analyse
  - Elle permet de définir les entités et les informations que les acteurs manipulent en utilisant le système

- Notation UML
  - Un rectangle
  - Un nom en **gras**
  - Parties définies et/ou affichées en option : attributs, opérations, compléments

Nom de la classe
liste des attributs
liste des opérations
liste des compléments

Contenu d'une classe

- Attribut
  - Information contenue dans chaque objet (= instance de la classe)
- Opération
  - Action possible sur chaque objet
  - Peu utilisée en recueil des besoins et en analyse (dépend de la méthode)
- Complément
  - Contraintes et/ou informations
  - Ne remplace pas la documentation

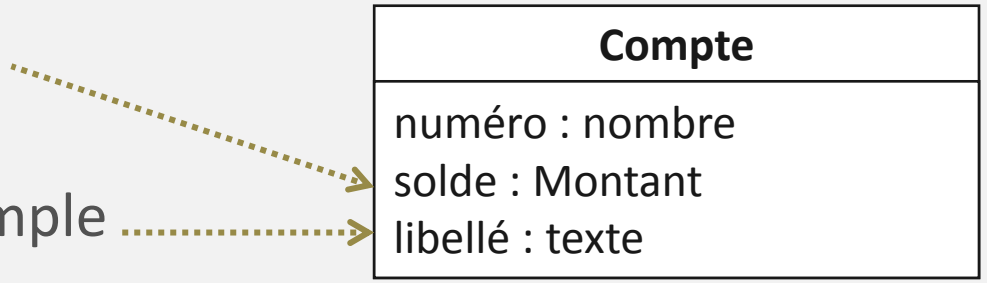


- Rappel : recueil et analyse des besoins = langage naturel

Typage des attributs

- Notation UML
  - : nom d'une classe

  ou
  - : nom d'un type simple



- Dans l'illustration
  - La classe Montant contient un nombre et une monnaie

- Usage fréquent en recueil des besoins et en analyse
  - Types simples indépendant des langages informatiques
  - Par exemple : texte, nombre, date, indicateur (booléen), …

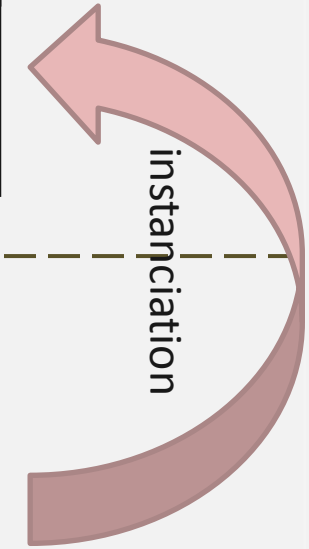
Aspects statiques / aspects dynamiques
Une classe permet de créer des objets

aspects dynamiques

- Exemple d'objets instance de la classe Compte dans un diagramme dynamique

un compte courant:Compte
numéro solde libellé

un compte épargne:Compte
numéro solde libellé



aspects statiques

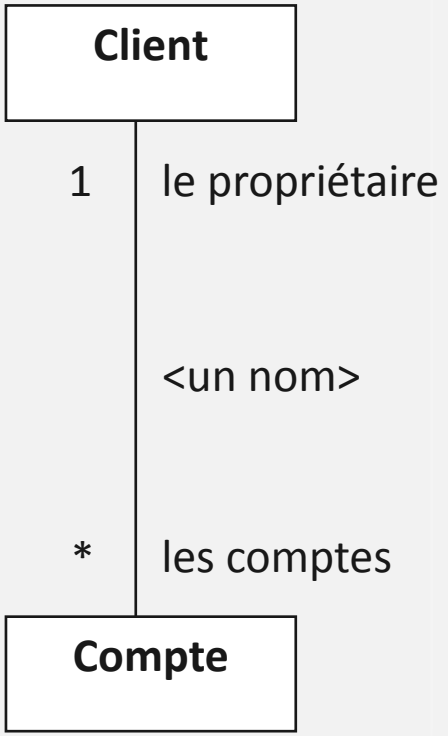
- Exemple de classe dans un diagramme de classes

Compte
numéro solde libellé

Association *(Association)*

- Sens
  - Relation indiquant un lien entre les instances issues des deux classes (illustration pages suivantes)

- Notation UML
  - Un trait simple avec
    - Un nom = sémantique de l ’association
      - Peu utilisée car difficile à lire dans les 2 sens
      - Parfois une flèche indique le sens de lecture
    - Deux extrémités avec chacune
      - Le rôle relatif d’une classe par rapport à l’autre
      - Une cardinalité : nombre d’occurrences (dans l’illustration : plusieurs comptes pour 1 client)



Cardinalités pour les associations

- Valeurs possibles
  - Tout entier positif et * (parfois noté n dans certains outils UML)

- Notation UML
  - Cardinalité minimale .. cardinalité maximale — par exemple : 1..*

  ou
  - Cardinalité unique — par exemple : 1

- Valeurs les plus utilisées

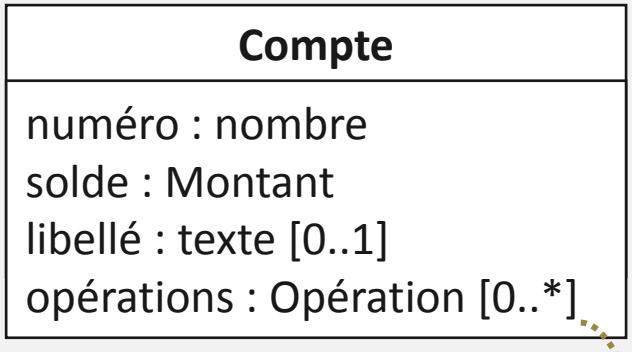
	Présence	
	Facultative	Obligatoire
Unicité	0..1	1 (= 1..1)
Multiplicité	* (= 0..*)	1..*

- Autres valeurs sont possibles, par exemple 2..5 mais usage déconseillé car moins évolutif qu'une règle de gestion (aspect dynamique)

Cardinalités pour les attributs

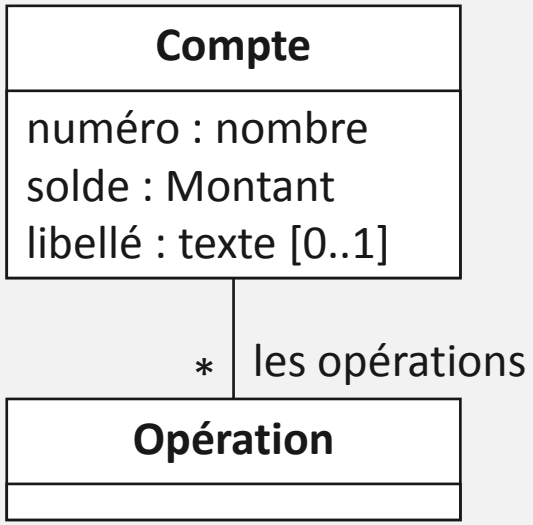
- Notation UML
  - [Cardinalité min .. cardinalité max]

  ou
  - [Cardinalité unique]

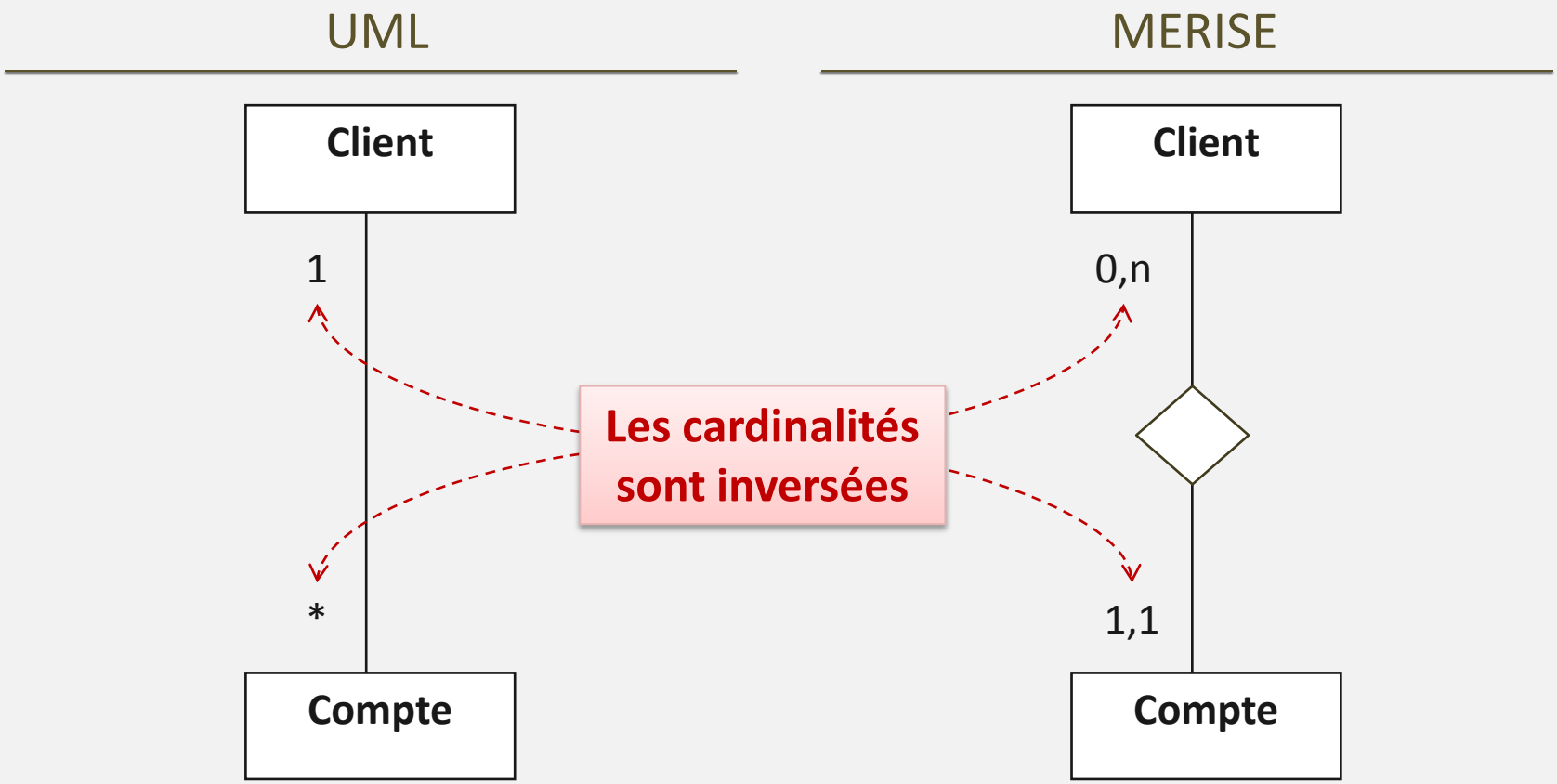


- NB : un attribut est équivalent à une relation

  Comment choisir entre les deux ?
  - Toujours un attribut pour un type simple
  - En général, une entité créée pour votre projet apparait sous forme de classe
    - Exemple : les opérations bancaires



Cardinalités
Comparaison UML et Merise

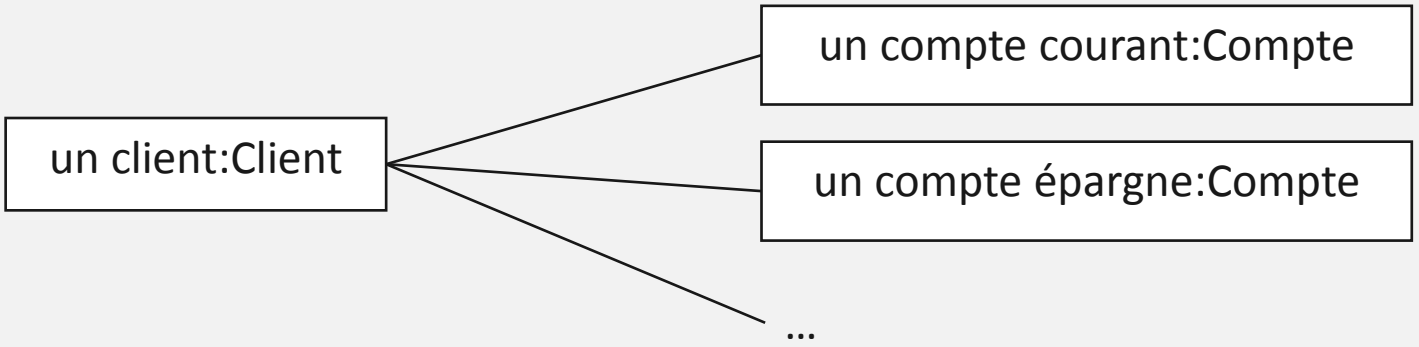


Les deux notations signifient « Un client peut avoir plusieurs comptes »

Aspects statiques / aspects dynamiques
Une association permet de créer des liens

aspects dynamiques

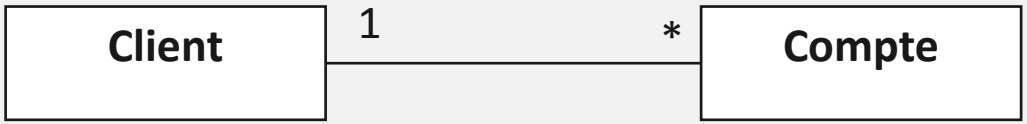
- Exemple d'objets et de liens issus de l'instanciation



instanciation

aspects statiques

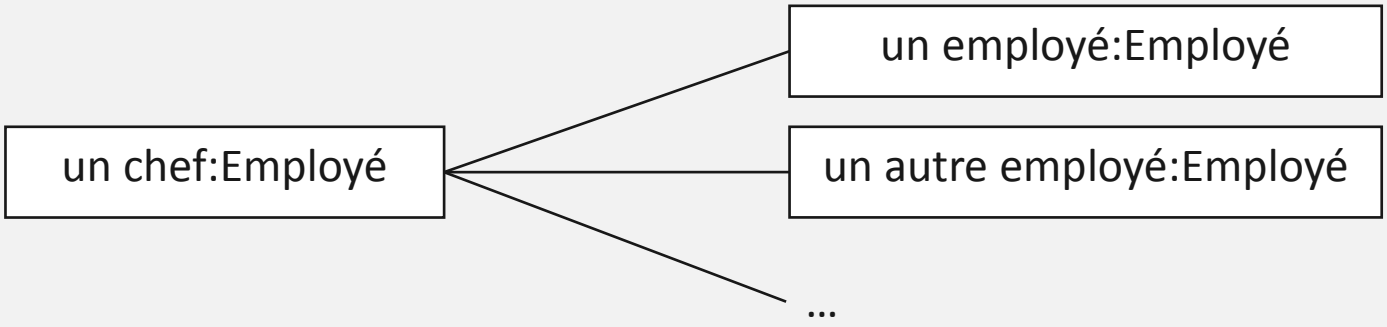
- Exemple : diagramme de classes pour représenter les comptes bancaires d'un client



Aspects statiques / aspects dynamiques
Association réflexive

aspects dynamiques

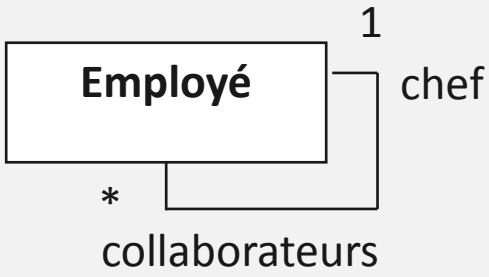
- Exemple d'objets et de liens issus de l'instanciation



instanciation

aspects statiques

- Exemple : diagramme de classes pour représenter la relation entre 2 employés d'une société

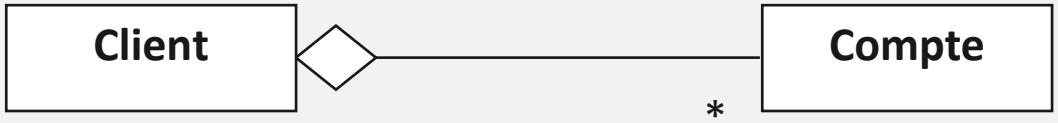


Agrégation *(Aggregation)*

- Sens
  - Une entité est constituée de plusieurs entités
  - NB : sémantique variable selon les domaines et les modélisateurs

    (OMG : Precise semantics of shared aggregation varies by application area and modeler)

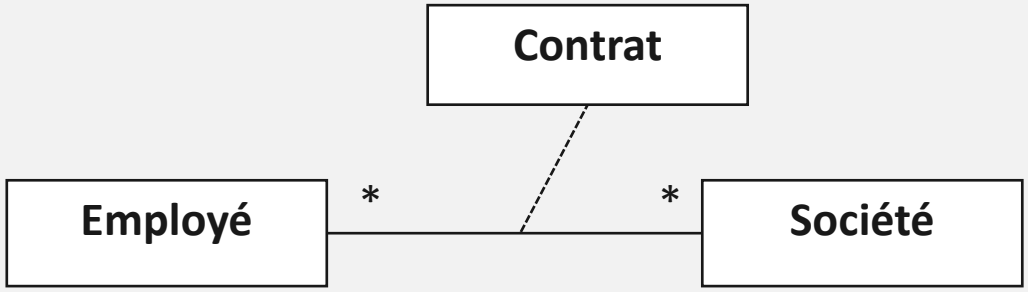
- Notation UML
  - Un diamant vide du coté de l'agrégat (souvent sans cardinalité)



  - La version forte de l'agrégation s'appelle la composition. Elle est utilisée plus souvent pour la conception

Classe association *(Association Class)*

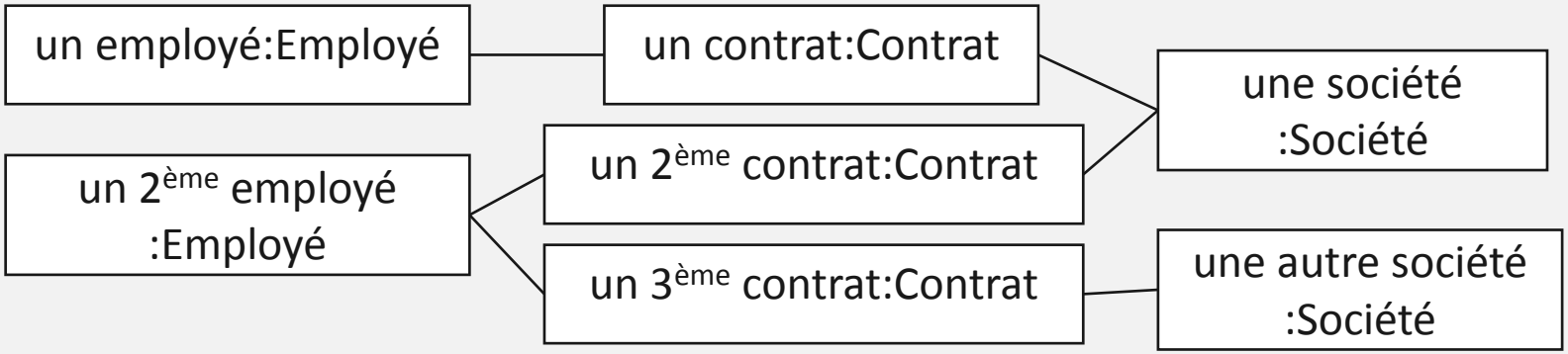
- Usage
  - Caractérisation d'une association
- Sens
  - Chaque instance de la classe association correspond à un couple d'instances des classes associées (cf. illustration page suivante)
- Notation UML
  - Une classe reliée à une association par un trait en pointillé



Aspects statiques / aspects dynamiques
Classe association *(Association Class)*

aspects dynamiques

- Exemple d'objets et de liens issus de l'instanciation



instanciation

aspects statiques

- Diagramme équivalent à la page précédente sans classe association

- Les cardinalités de l'association se retrouvent sur la classe association
- Les autres cardinalités sont 1
- (un contrat n'existe qu'avec un employé et une société)

Synthèse

- Usage en recueil des besoins et en analyse
  - Présenter un ensemble d'entités de gestion, leurs relations et les informations contenues dans ces entités

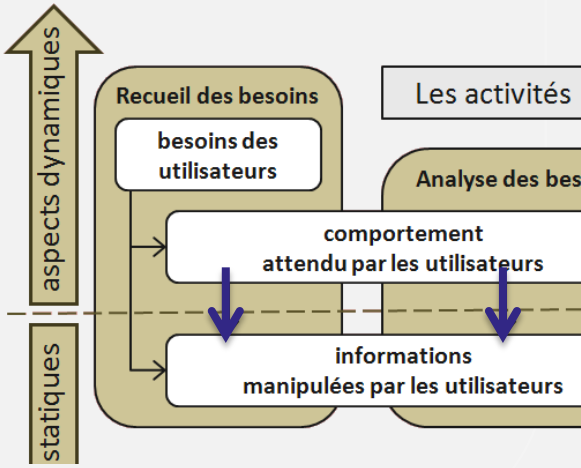
- Pratique courante pour démarrer
  - Identifier les classes, les attributs et les relations à partir d'un texte (énoncé de TD, cahier des charges, expression des besoins, …)

  ⚠ Attention : cette technique ne garantit pas la cohérence de la modélisation

- Comment garantir la cohérence de la modélisation ?
  - Comparaison des diagrammes de classes avec les informations manipulées dans les diagrammes dynamiques (diagramme de séquence, …)

    ↓

    Chaque information manipulée justifie une classe, un attribut ou une relation



Bonnes pratiques

- Souvent, on n'utilise pas les termes UML « classe » et « attributs » quand on communique avec les utilisateurs
  - On peut utiliser des termes génériques, tels que « entité », « information »
  - Ceci permet de rappeler que l'on ne conçoit pas encore le système

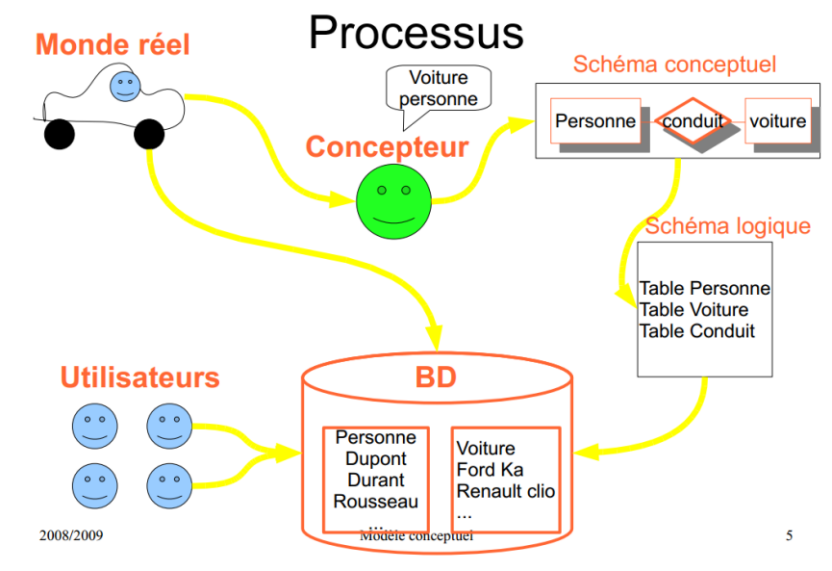
- Nommage et documentation des classes
  - Utiliser les termes du métier des utilisateurs
  - Chaque classe doit être définie par rapport au périmètre du système et non pas en général ou par nature

    le « monde réel » n'existe pas en modélisation

- Passage de l'analyse à la conception
  - Aucune méthode (fiable) ne permet de passer (avec succès) directement des classes d'analyse aux classes de conception
  - Dans l'approche objet, le comportement du système justifie les choix de conception (cf. suite du cours, application du principe d'encapsulation)

De mon point de vue
un exemple classique à ne pas suivre

- Exemple de démarche d'analyse et de conception des données indépendamment des traitements et des besoins des utilisateurs
  Séparation classique données et traitements (non objet)



- Rien ne garantit que la modélisation des données correspondra à l'usage que souhaitera en faire les utilisateurs via le système
- Source : http://etienne.baudrier.free.fr/L3_BI/L3_BI_ModelesEA.pdf