Analyse et conception

UML

Objectifs

1. Pourquoi modéliser ?
2. Qu'est-ce qu'un modèle ?
3. Vision générale d'UML
4. UML : la syntaxe des diagrammes principaux

Citations

- « Un modèle représente une réalité pour un objectif donné. Le modèle est une abstraction de la réalité, il ne représente pas tous les aspects de la réalité. Cela nous permet une gestion simplifiée, en évitant la complexité, le danger et l'irréversibilité de la réalité. » *Rothenberg, 1989*
- La différence entre la théorie et la pratique, c'est qu'en théorie, il n'y a pas de différence entre la théorie et la pratique, mais qu'en pratique, il y en a une.
- « Avant donc d'écrire, apprenez à penser. Ce que l'on conçoit bien s'énonce clairement, et les mots pour le dire viennent aisément. »
  Art Poétique - Nicolas Boileau, 1815
- *A* est un bon modèle de *B* si *A* permet de répondre de façon satisfaisante à des questions prédéfinies sur *B*
- Un dessin vaut mieux que de longues lignes de code

Pourquoi modéliser ?

- Communiquer avec toutes les personnes dans le périmètre du projet
- Documenter pour les intervenants futurs
- Tracer les modifications
- Produire plus rapidement (*Time to market*)
- Anticiper la complexité

De la même façon qu'il vaut mieux dessiner une maison avant de la construire, il vaut mieux modéliser un système avant de le réaliser.

Les modèles : une vue simplifiée

- Souligner
- Omettre
- Mais aussi une vue complète et précise

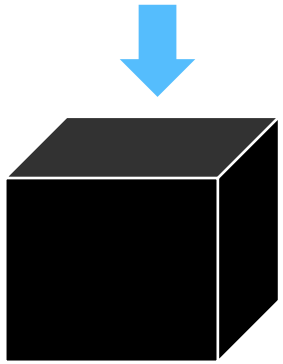
Les modèles : niveaux d'abstractions

- Vue d'ensemble
- Vue détaillée

Les modèles : notion de boîte noire

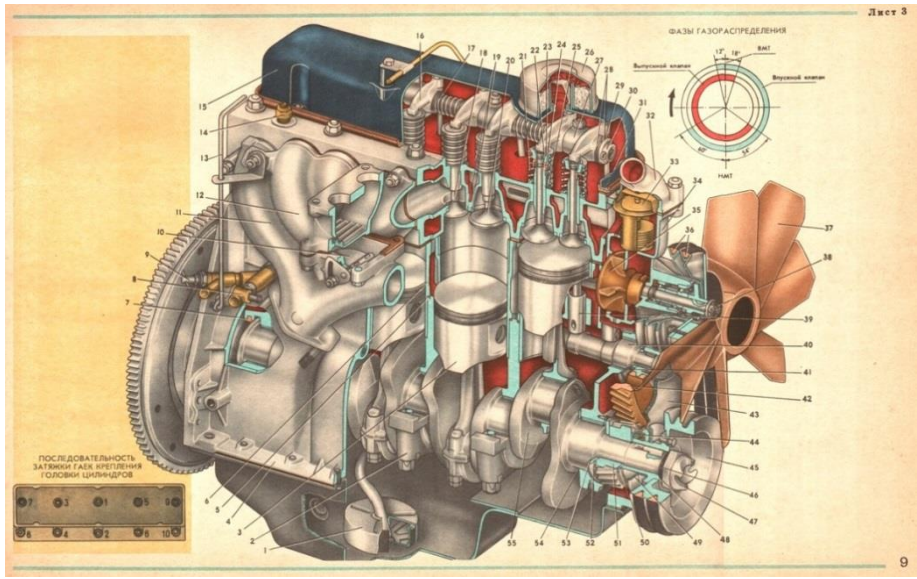
Vue externe

Entrées



Sorties

Vue interne



UML : Unified Modeling Language

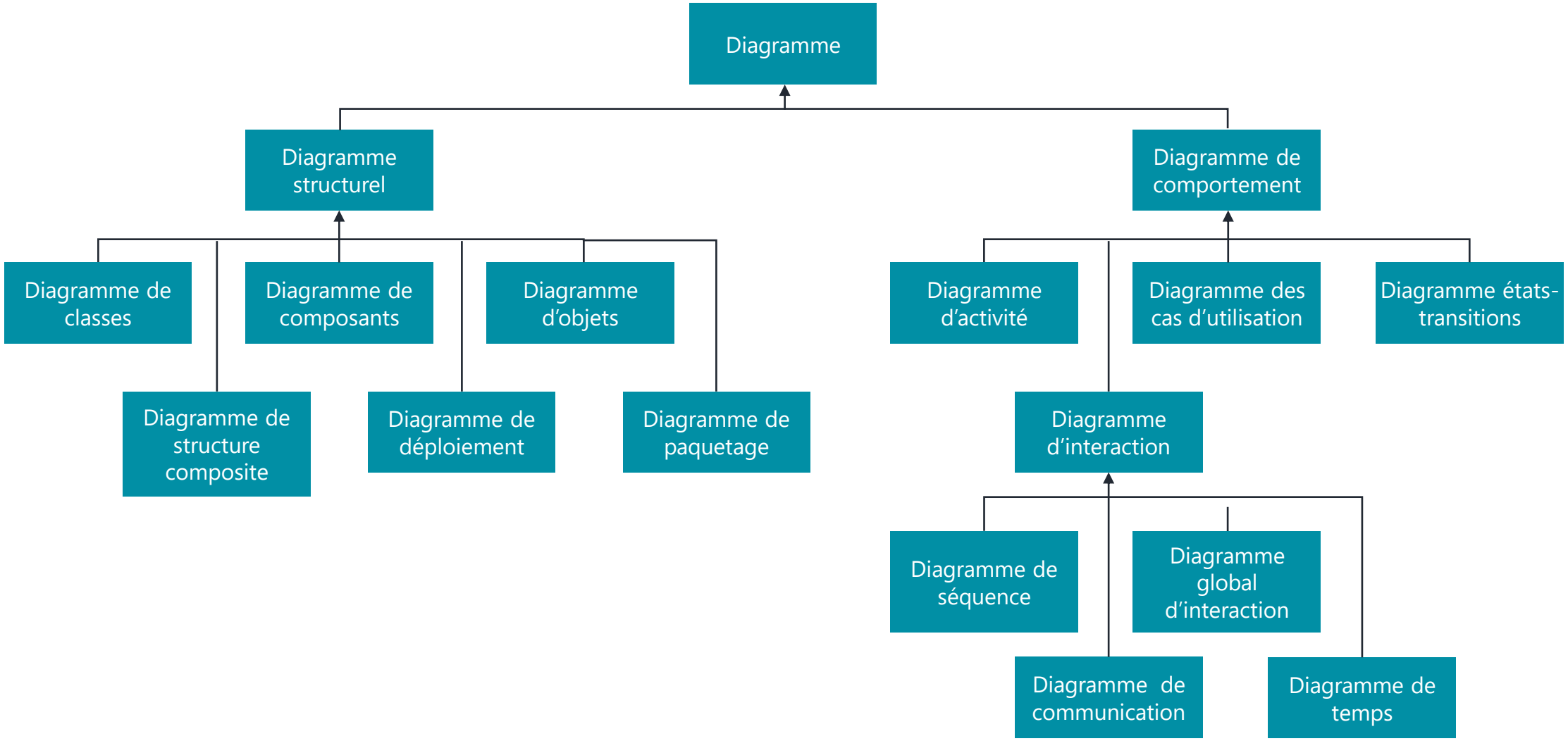
- Langage de modélisation de systèmes
- Langage commun
- Langage visuel
- Notation ≠ Méthode
- En version 2.5 depuis 2015

UML : diagrammes

15 diagrammes

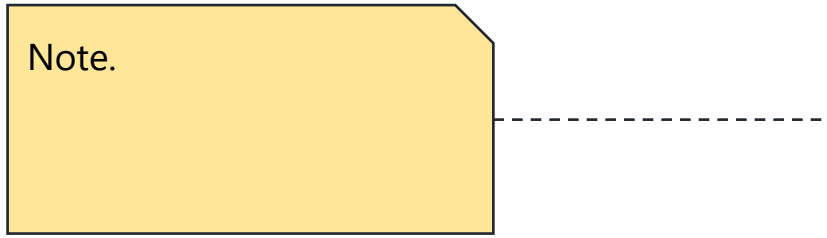
- Diagrammes de structure
- Diagramme de comportement

Diagrammes UML



Des éléments communs à tous les diagrammes

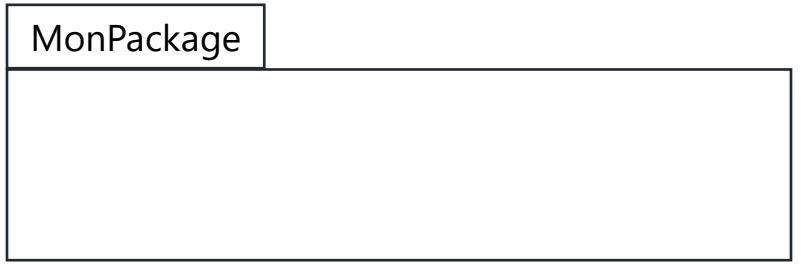
La note
(description textuelle)



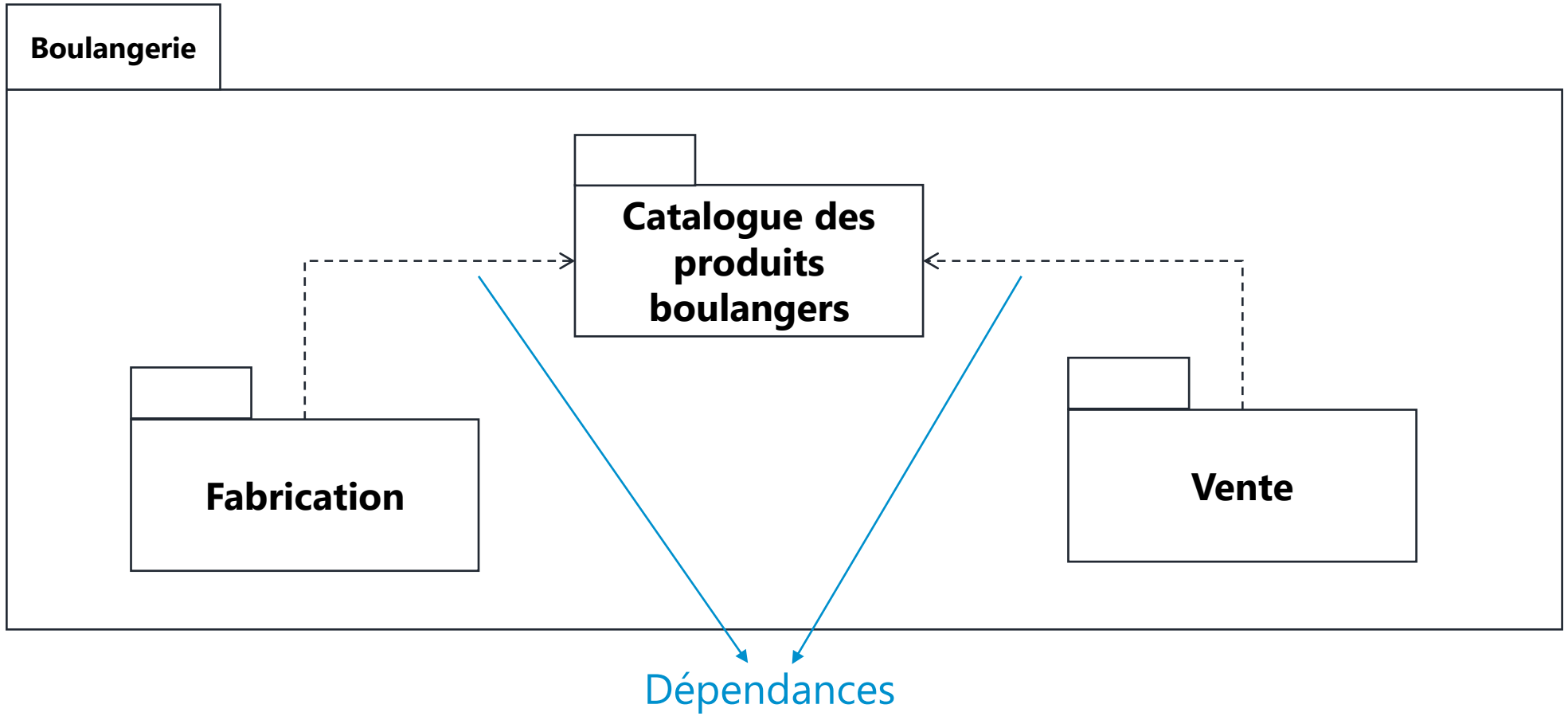
Lien d'héritage
(est une sorte de)

Lien de dépendance
(utilise ou dépend de)

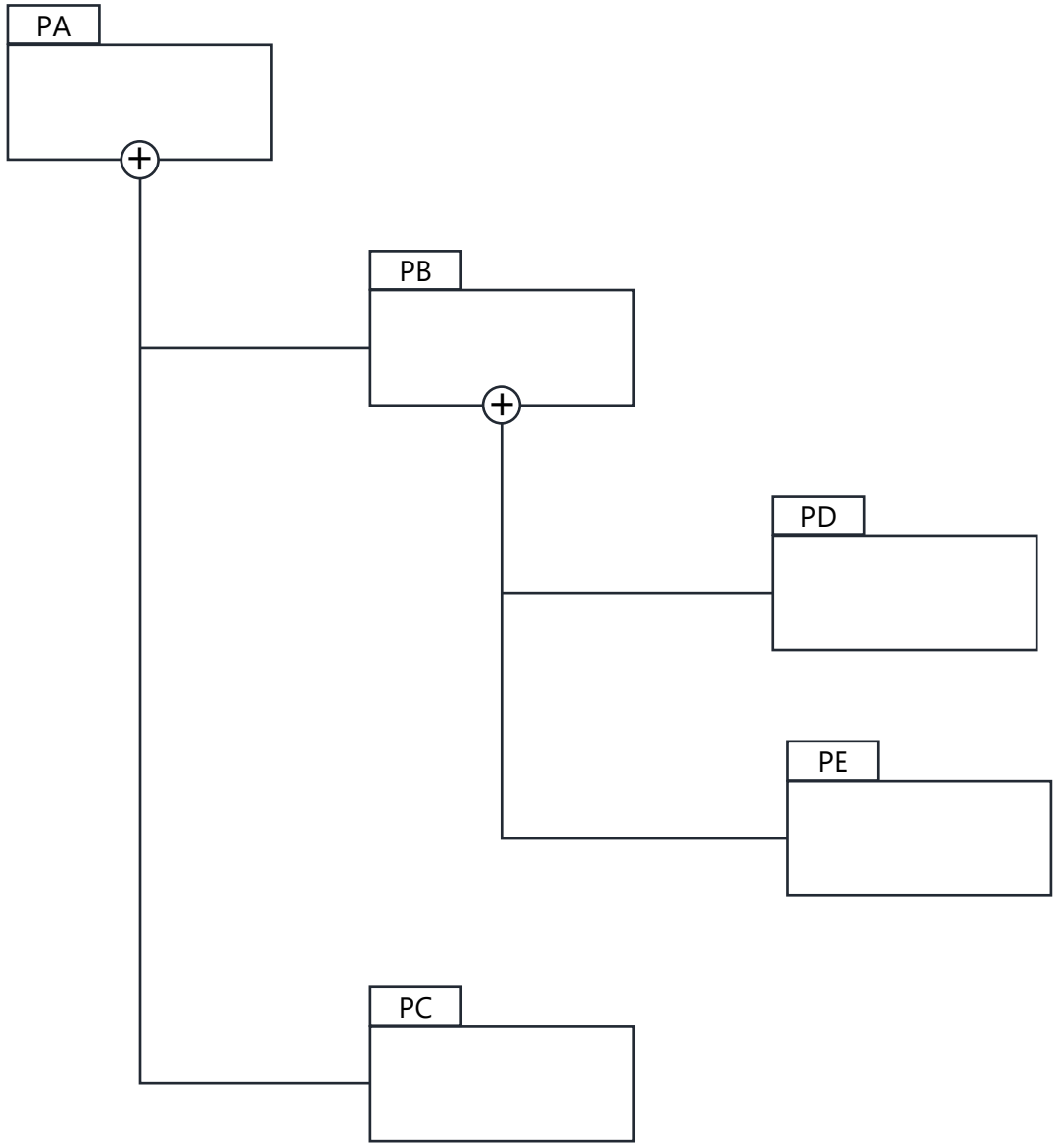
Package
(organise des éléments)



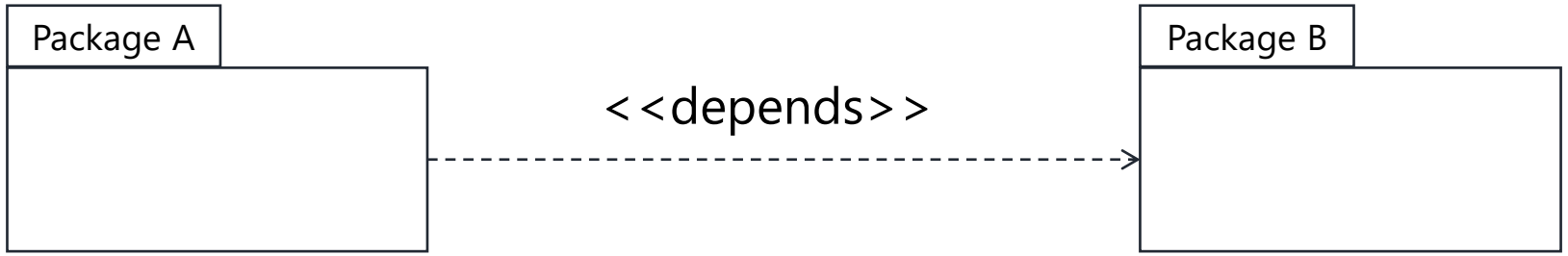
Le diagramme de package



L'imbrication des packages

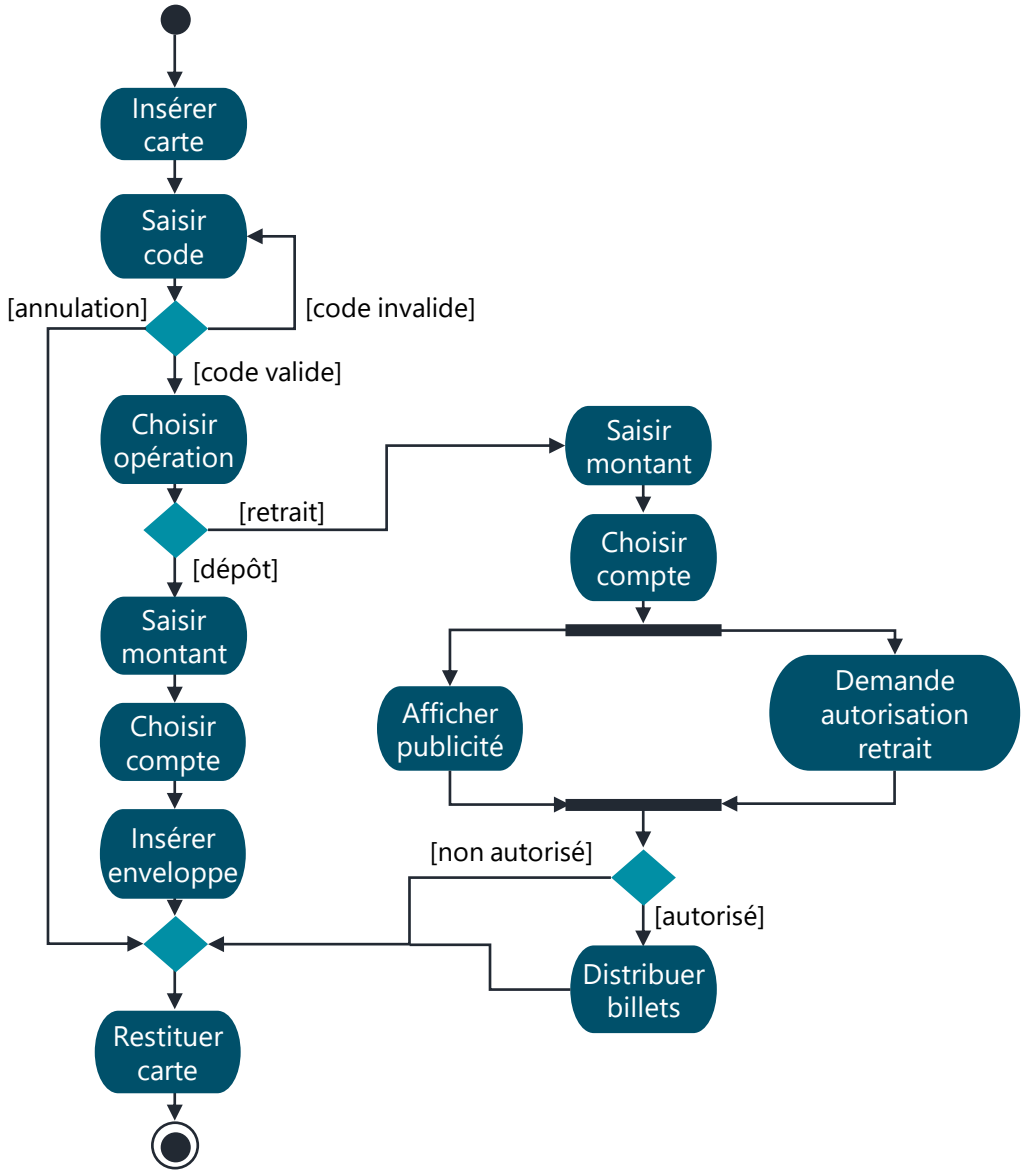


Le lien de dépendances



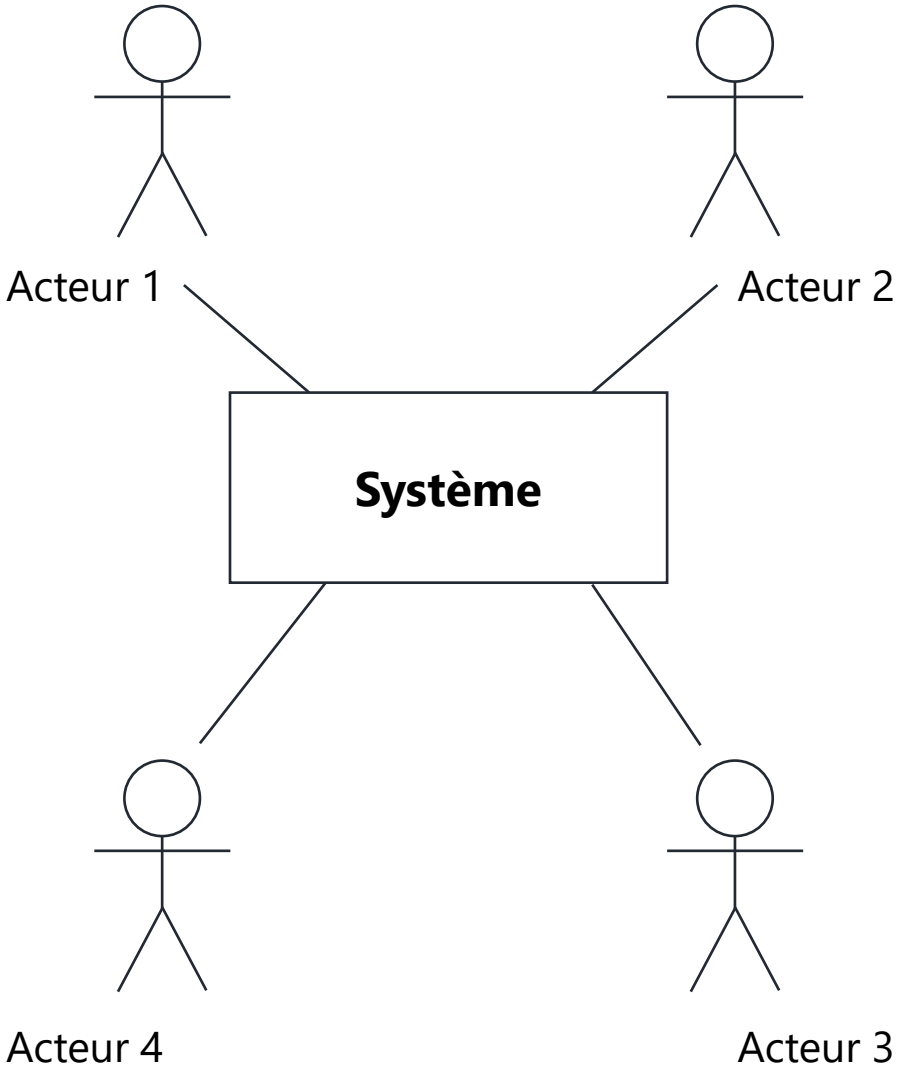
Le diagramme d'activité

- Les concepts manipulés :
  - Les activités
    - Initiale
    - Finale
  - Les enchaînements d'activité
    - Simple
    - Nœud de décision (losange)
    - Fourche
    - Synchronisation
    - Nœud de fusion (losange)
  - Les conditions de garde

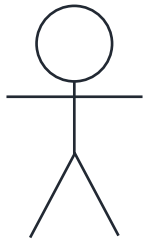


Une activité	
Activité initiale	
Activité finale	
Enchaînement simple	
Nœud de décision	
Fourche	
Synchronisation	
Nœud de fusion	
Activite temporelle	

Diagramme de cas d'utilisation système



- Les acteurs
  - Un acteur représente un rôle joué par une entité externe (utilisateur humain, dispositif matériel ou autre système) qui interagit directement avec le système étudié

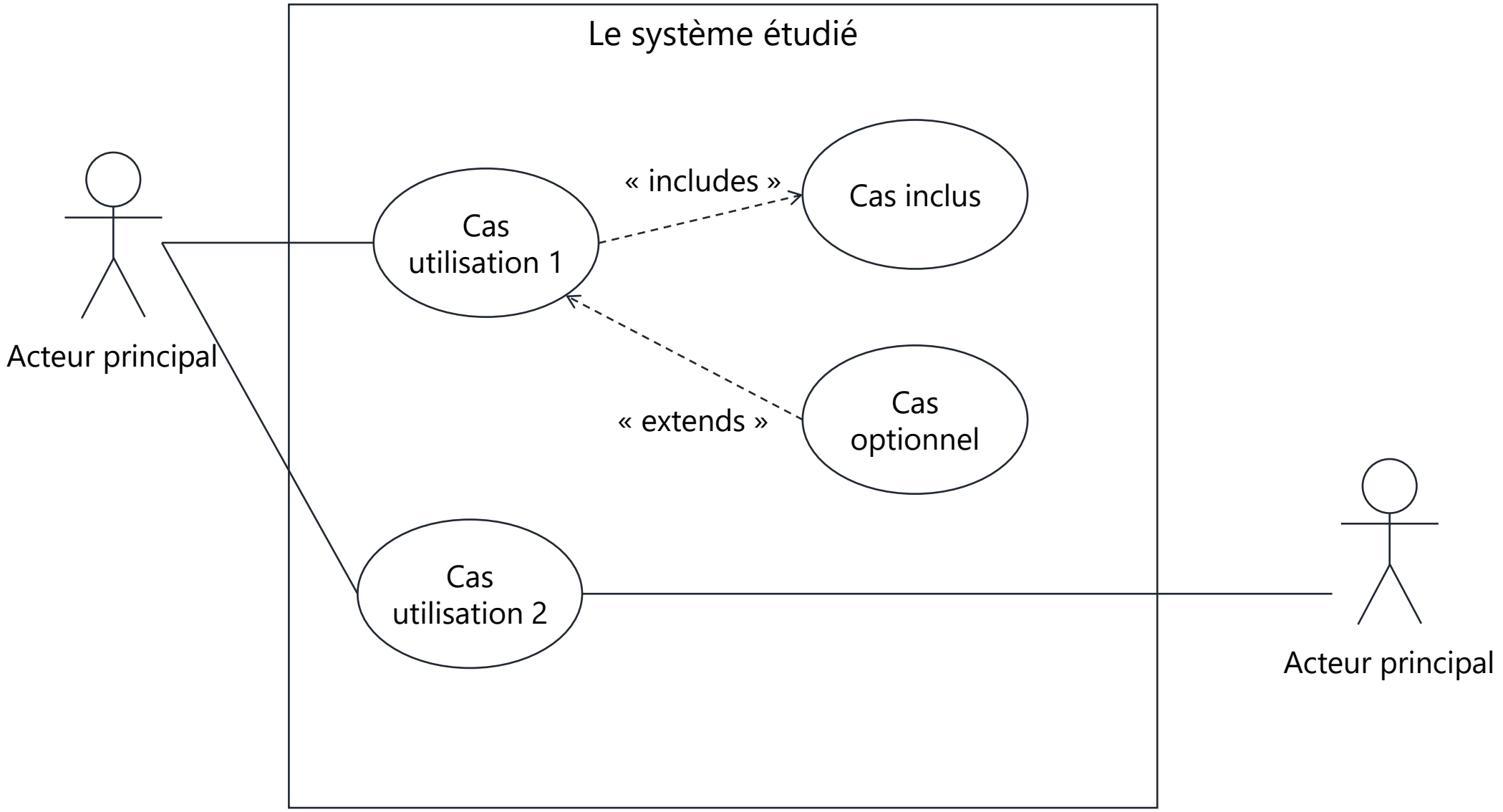


Internaute

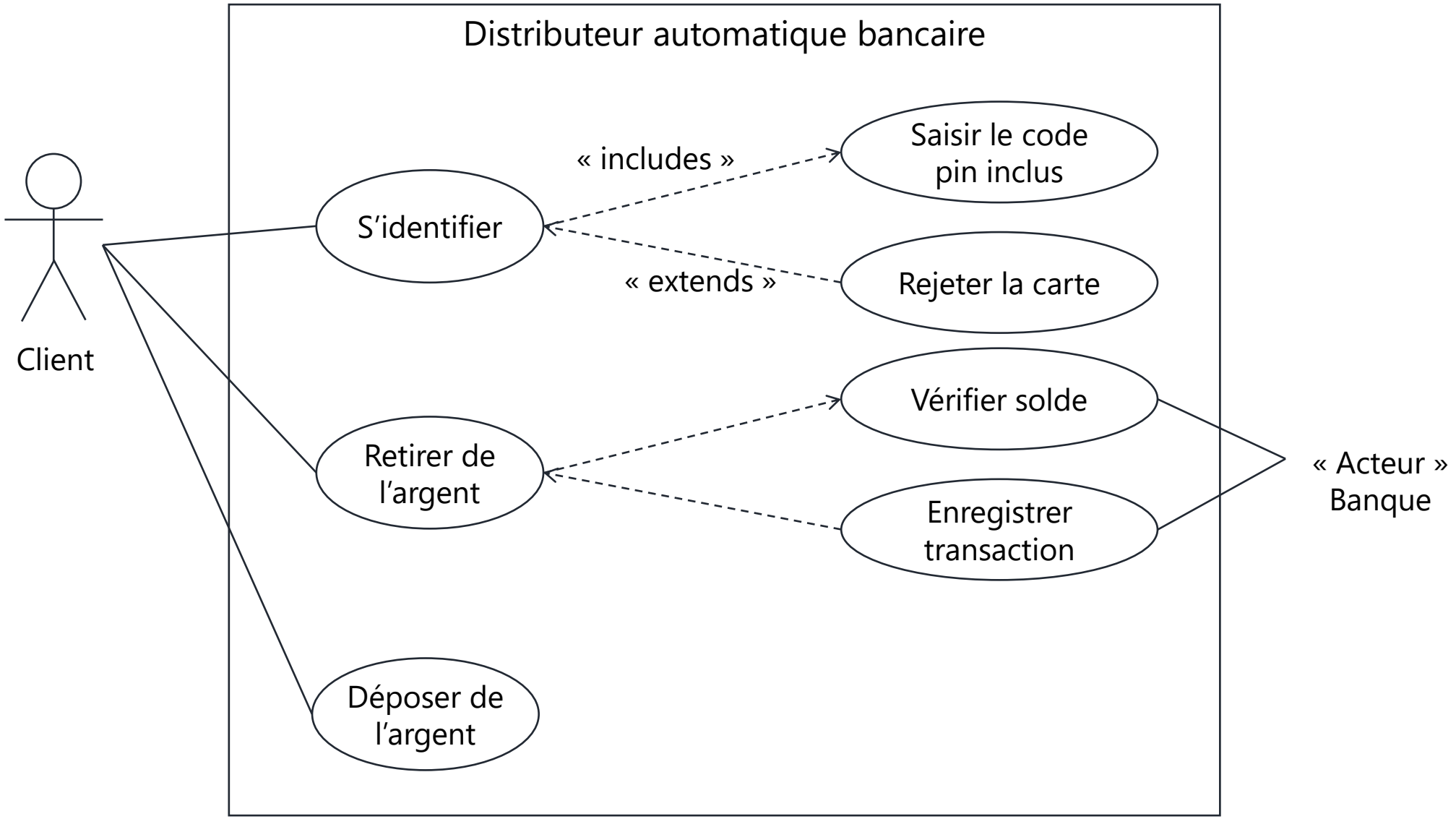
<<Actor>>
Acteur non humain

- Des liens relient le système à chacun des acteurs
- Ne pas lier les acteurs entre eux (sauf lien d'héritage)

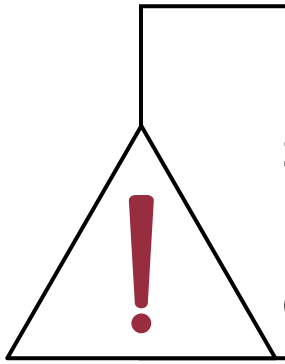
Le diagramme de cas d'utilisation



Un exemple de diagramme de cas d'utilisation



- Un cas d'utilisation représente un ensemble de séquences d'actions qui sont réalisées par le système et qui produisent un résultat observable intéressant pour un acteur particulier
- Un cas d'utilisation modélise un service rendu par le système
- Il exprime les interactions acteurs / système



Un cas d'utilisation est un ensemble d'actions : sûrement pas une seule action.
Les différents enchaînements possibles d'un cas d'utilisation sont appelés *scénarios*.

Diagramme d’états - Transitions

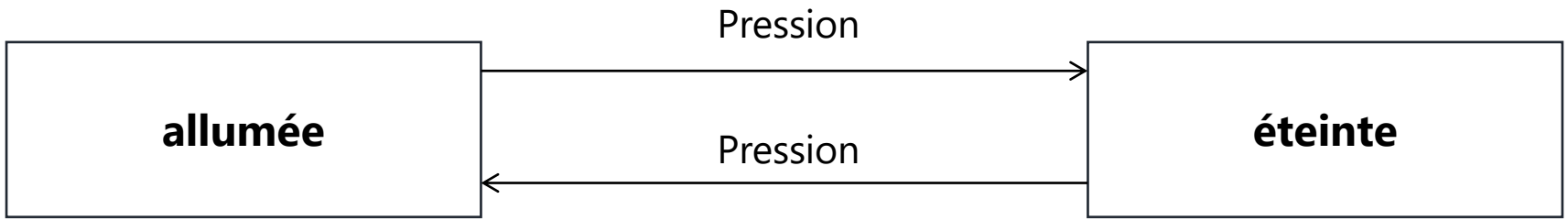


Diagramme d’état d’une lampe de chevet

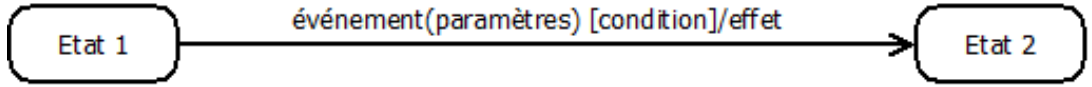
Diagramme d'états - Éléments

Pseudo-états

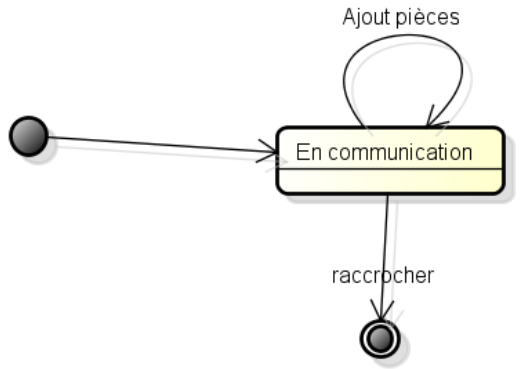
État initial

État final

Transition externe

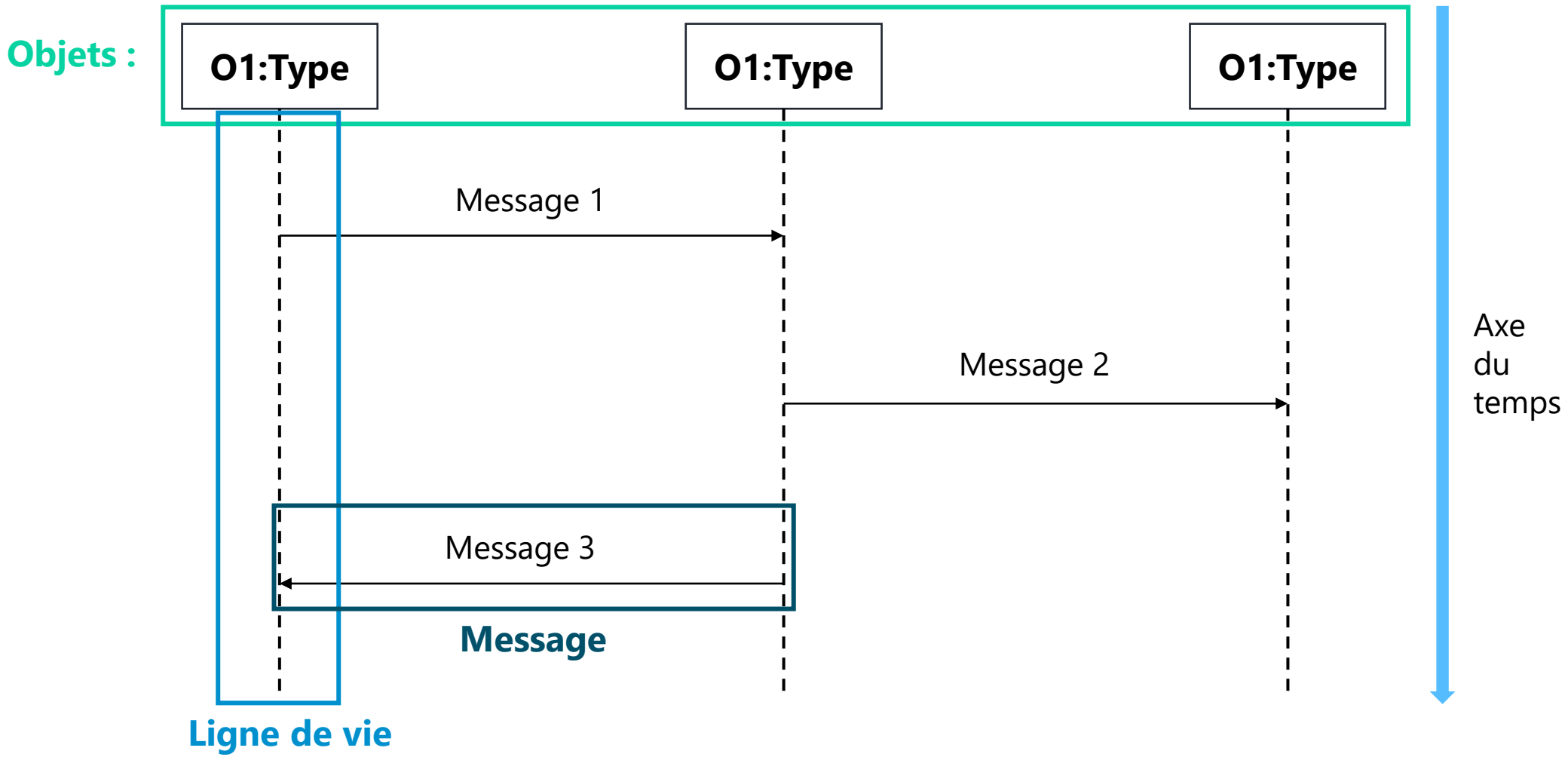


Transition réflexive

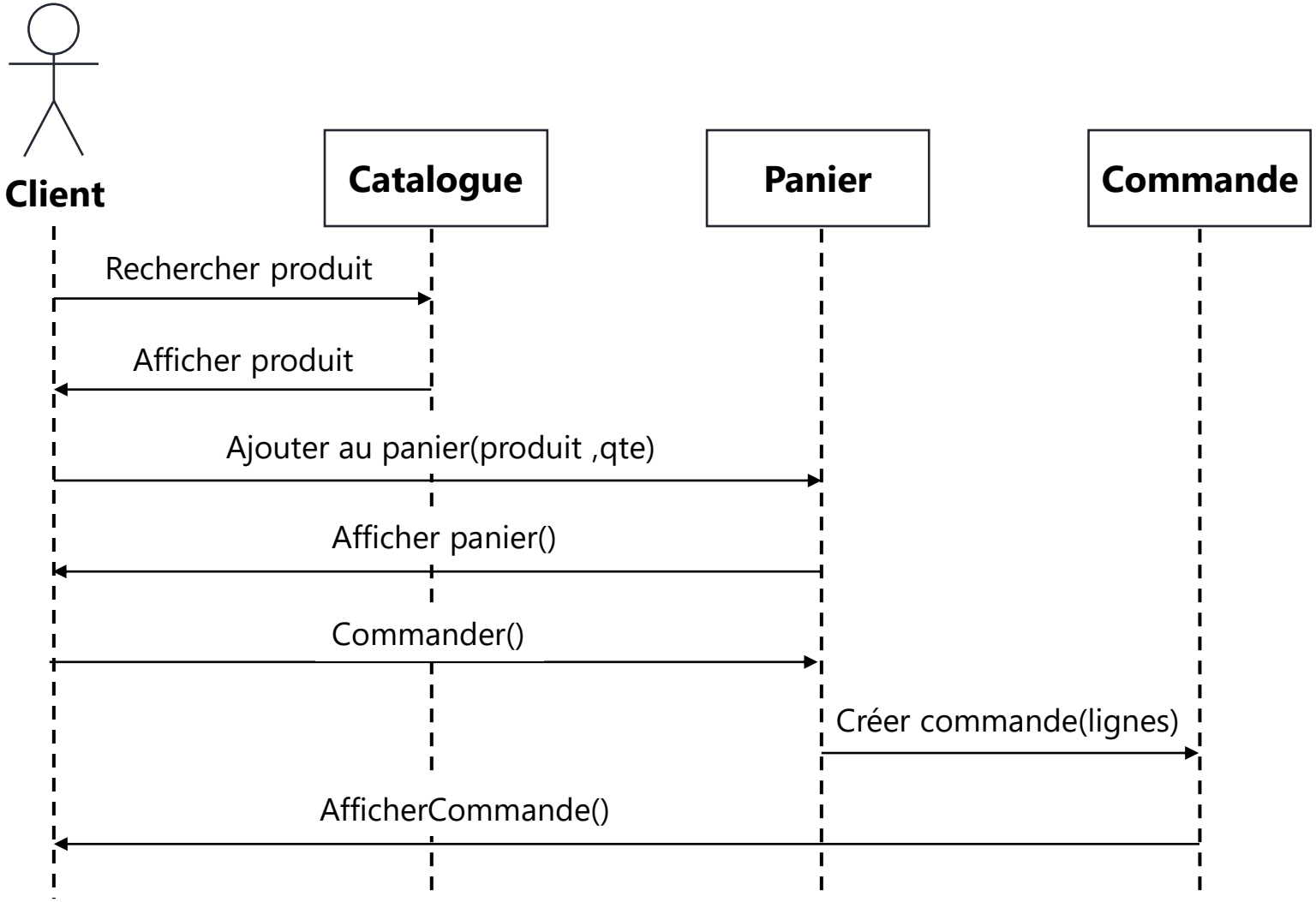


Le diagramme de séquence : les bases

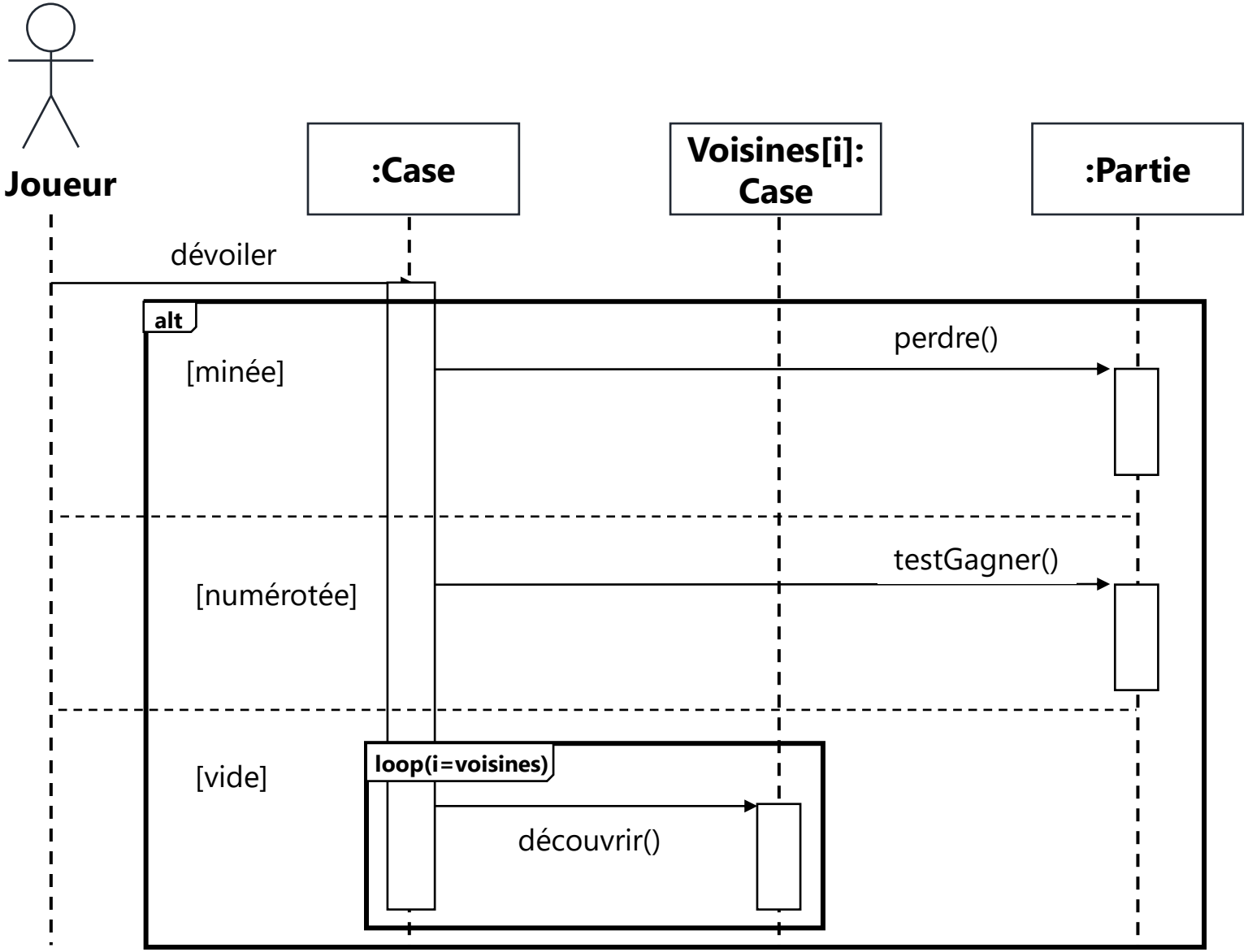
Diagramme de séquence
Identifiant du diagramme



Le diagramme de séquence : exemple



Le diagramme de séquence – Cadres d'interactions

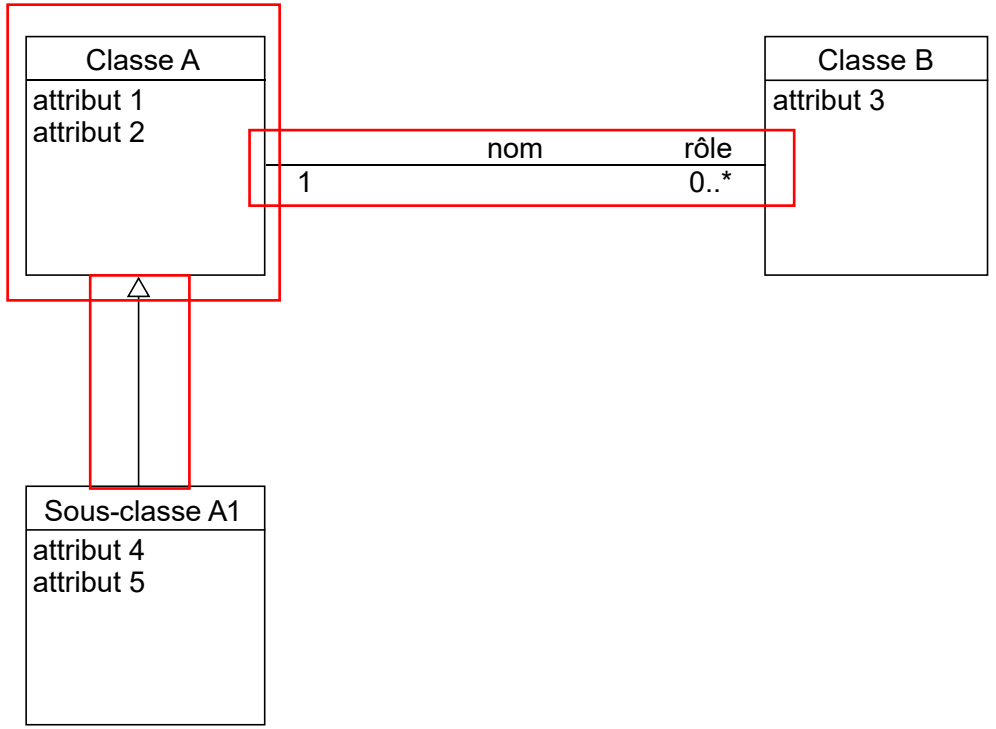


Le diagramme de séquence – Cadres d'interactions

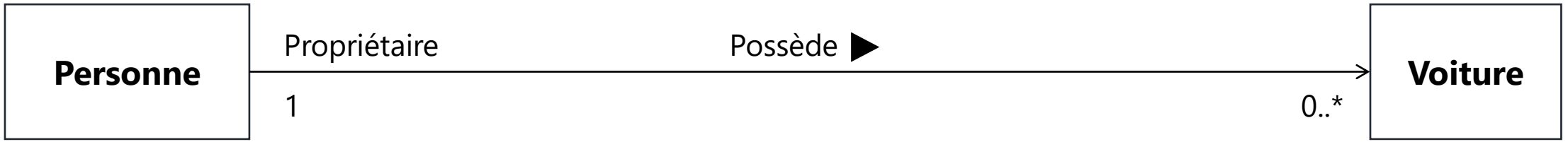
alt	Opérateur alternatives – équivalent au switch en java
Opt	Option
Loop	Boucle
Par	Exécution parallèle
Ref	Référence à un autre diagramme d'interaction

Le diagramme de classe d'analyse : les bases

- Classes
  - Attribut
- Association
  - Nom
  - Rôle
  - Multiplicité
- Héritage
  - Sous-classe
  - Super classe



Le diagramme de classe d'analyse : Notion d'association

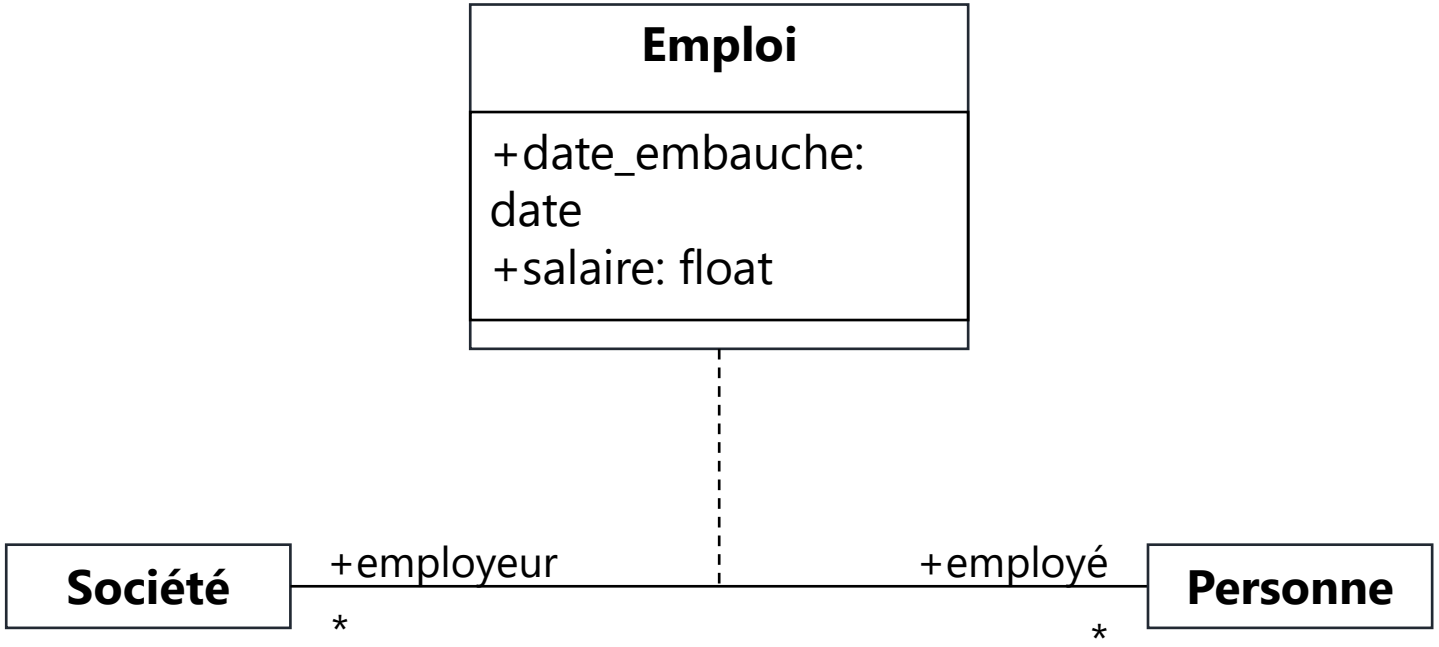


Le diagramme de classe d'analyse : Multiplicité

- Syntaxe : min .. max
- La multiplicité minimale peut être :
  - 0 : optionnelle
  - 1 : obligatoire
- La multiplicité maximale peut être 1 ou *

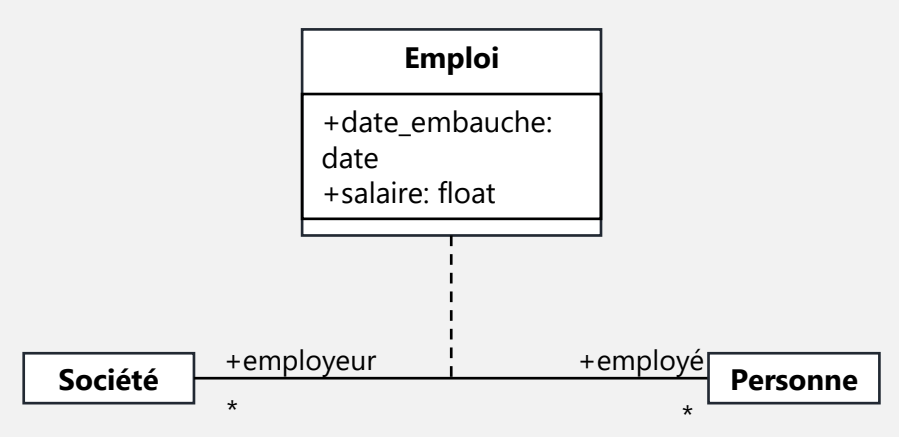
0..1	Au plus un
1..1 ou 1	Un seul
0..* ou *	Un nombre indéterminé
1..*	Au moins un

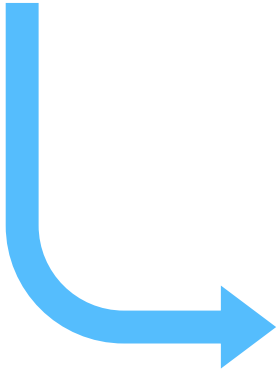
Le diagramme de classe d'analyse : Classe d'association

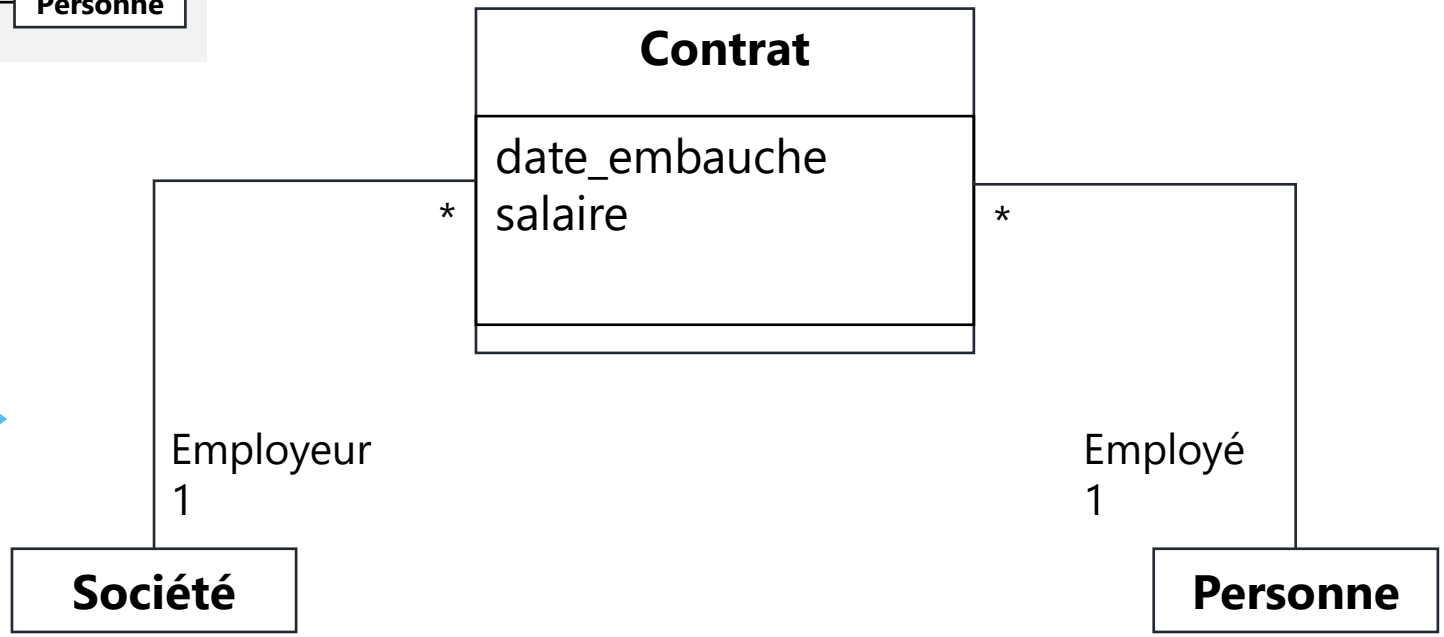


Objectif : faire porter des attributs sur l'association

Le diagramme de classe d'analyse : Classe d'association







Notion équivalente sans classe d'association

Le diagramme de classe d'analyse : Agrégation et composition

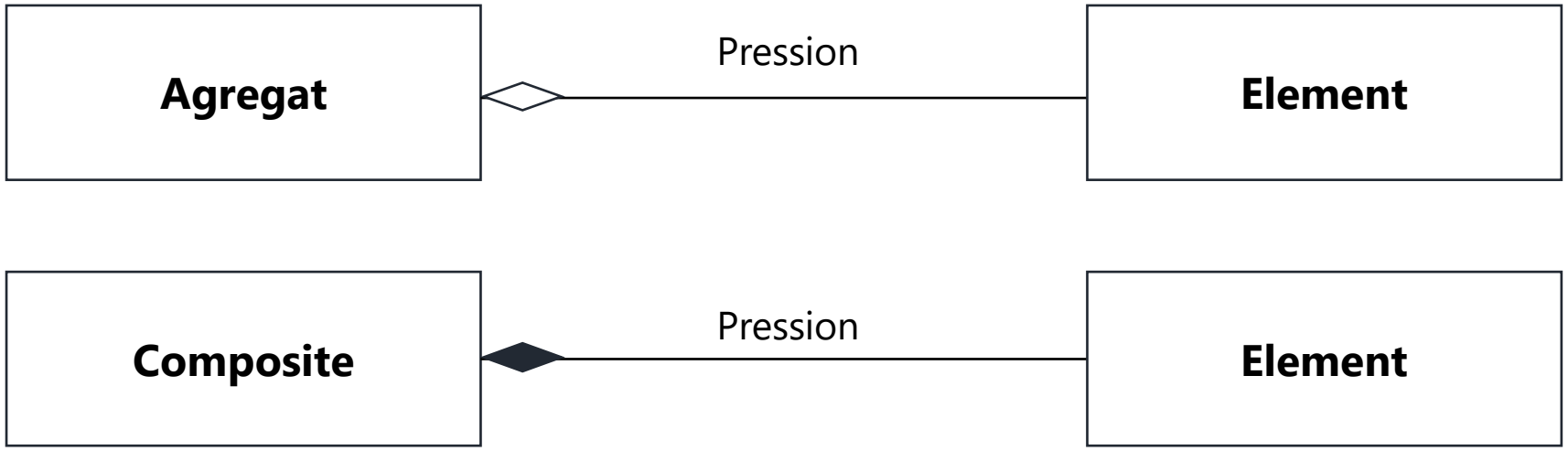


Diagramme de classe de conception

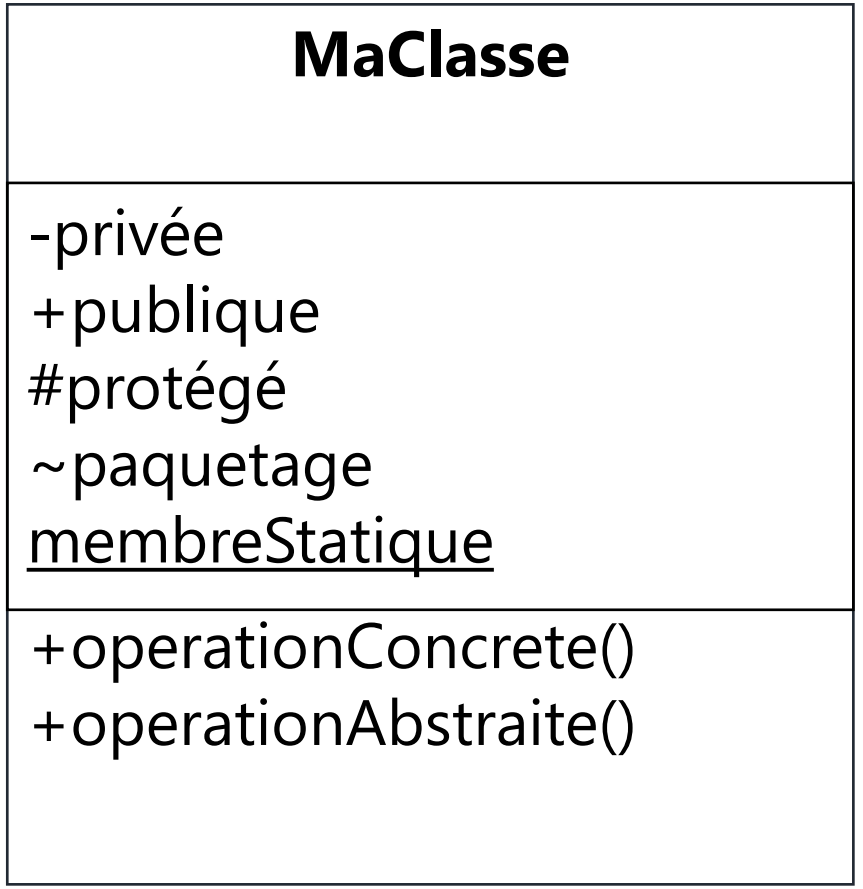
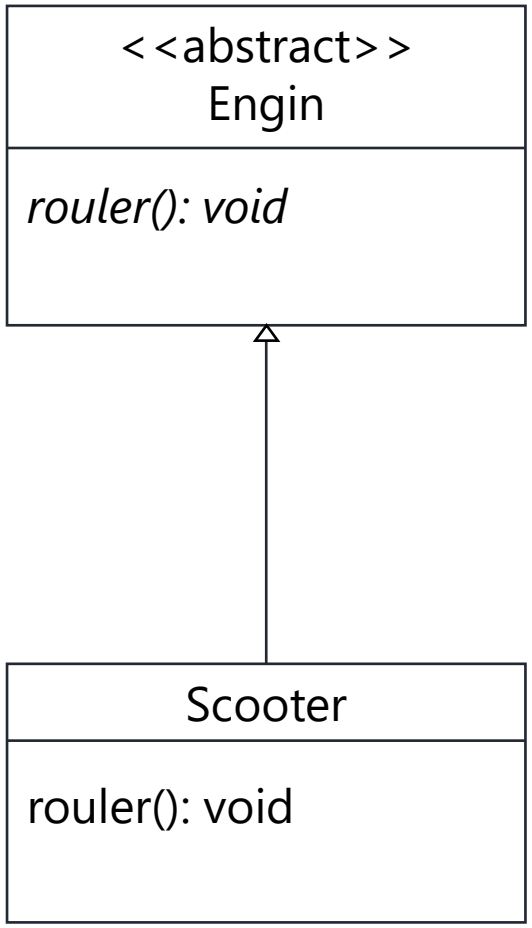


Diagramme de classe de conception : La navigabilité

Diagramme de classe de conception : L'héritage



Le nom d'une classe abstraite est noté en italique.

Diagramme de classe de conception : Le lien de réalisation

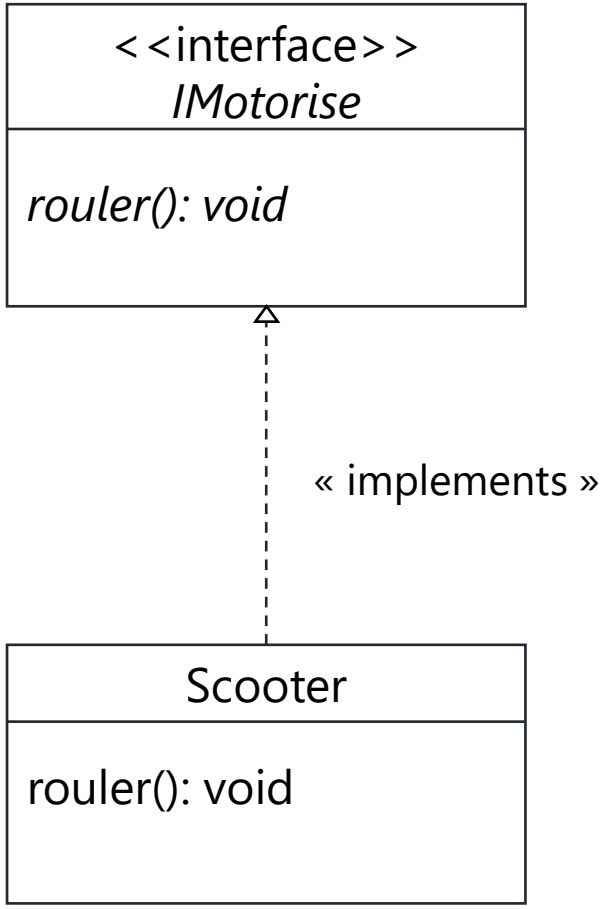


Diagramme de classe de conception : Qualificatif