Modélisation avec UML

Karim Benouaret

Université Claude Bernard Lyon 1
karim.benouaret@liris.cnrs.fr

Plan

Plan

Pourquoi modéliser ?

- Modèle : une simplification de la réalité qui permet de mieux comprendre le système à développer
- Il permet aussi de :
  - Disposer d'un support de communication clair et précis entre les clients, collègues ou partenaires.
  - Limiter le nombre d'anomalies ou de cas non prévus

UML, un langage

- UML : Unified Modeling Language
  - Langage de modélisation unifié
  - UML est un langage (pas une méthode) de modélisation graphique et textuel qui permet de représenter et de communiquer les divers aspects d'un système
- Langage vs méthode
  - Langage : notations, grammaire, sémantique
  - Méthode : comment utiliser un langage, ensemble d'étapes

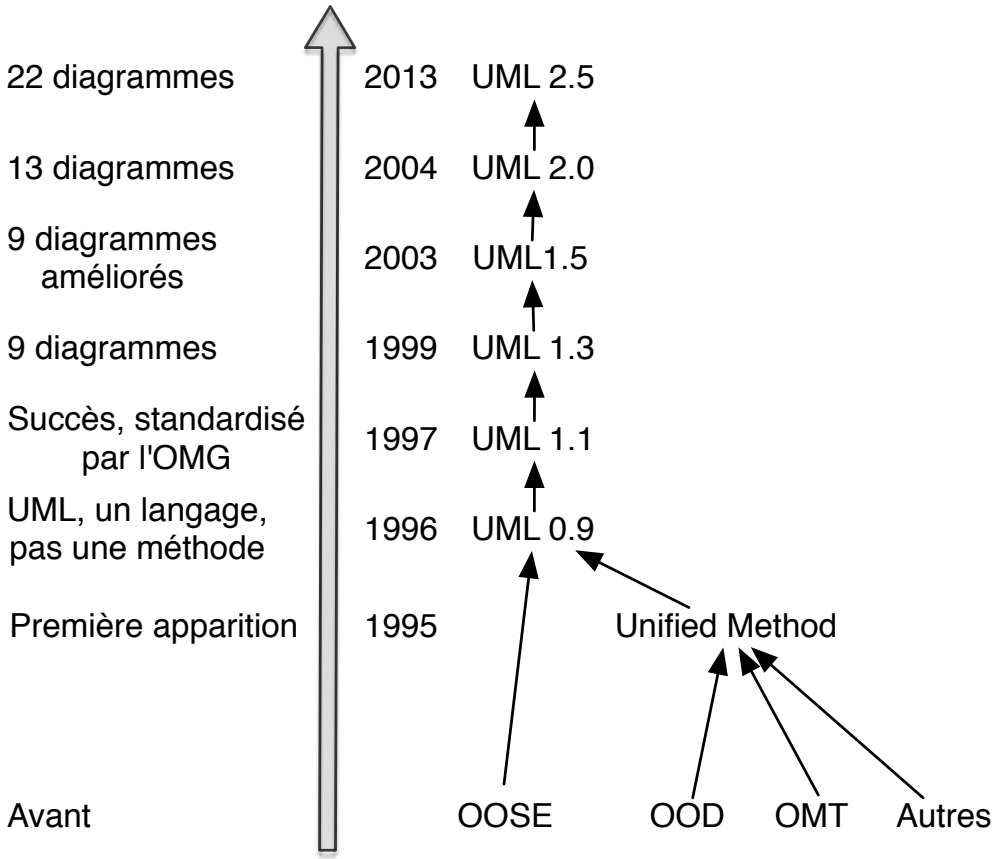
Avant UML

- La guerre des méthodes
  - Les méthodes utilisées dans les années 80 ne répondaient pas au besoins des utilisateurs
  - $\Longrightarrow$ plus de 50 méthodes sont apparues entre 1990 et 1995
  - Mais, aucune ne parvenait à s'imposer
- Le consensus se fait autour de trois méthode :
  - OMT de James Rumbaugh : fournit une représentation graphique des aspects statiques et dynamiques des systèmes
  - OOD de Grady Booch : définie pour le département de la défense ; introduit le concept de paquetage (package)
  - OOSE d'Ivar Jacobson : fonde l'analyse sur la description des besoins des utilisateurs (cas d'utilisation)

L'unification

- But Initial
  - Définir une méthode complète, i.e., de l'analyse à l'implémentation
- Obstacle
  - La méthode dépend du système
- Constat
  - Un langage, pas une méthode
  - Un ensemble de notations pour que chacun ait à sa disposition les éléments nécessaires
- $\Longrightarrow$ UML, un langage indépendant de la méthode

Apparition et évolution



Les diagrammes courants

- Diagrammes structurels
  - Diagramme de classes
  - Diagramme d'objets
  - Diagramme de composants
  - Diagramme de déploiement
- Diagrammes de comportement
  - Diagramme de cas d'utilisation
  - Diagramme d'interaction
    - Diagramme de séquence
    - Diagramme de communication
  - Diagramme d'états-transitions
  - Diagramme d'activités

D'un point de vue modélisation

- Spécifier le système
  - Diagramme de cas d'utilisation
- Modéliser la structure de l'application
  - Diagramme de classes
  - Diagramme d'objets
- Modéliser les objets communicants
  - Diagramme de séquence
  - Diagramme de communication
- Modéliser le comportement des objets
  - Diagramme d'états-transitions
- Modéliser les traitements
  - Diagramme d'activités
- Modéliser l'instanciation de l'application
  - Diagramme de composants
  - Diagramme de déploiement

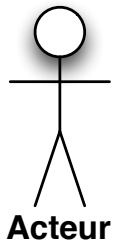
Plan

Rôle

- Décrit le comportement d'un système d'un point de vue utilisateur
- Permet de définir les limites du système et les relations entre le système et l'environnement

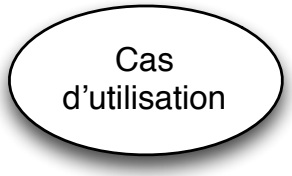
Les acteurs

- Acteur : entités qui interagit avec le système



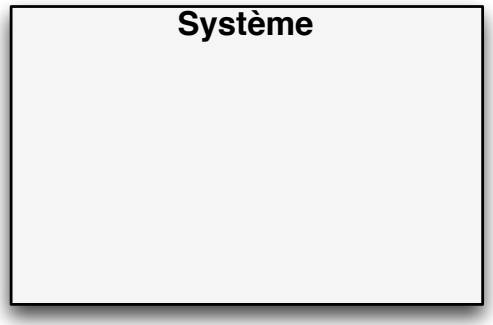
Les cas d'utilisation

- Cas d'utilisation : ensemble d'actions réalisées par le système en réponse à une action d'un acteur



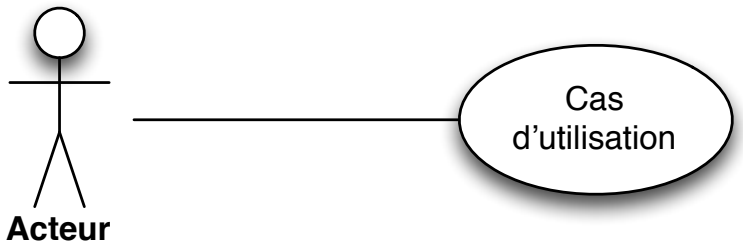
Le système

- Système : définit l'application informatique



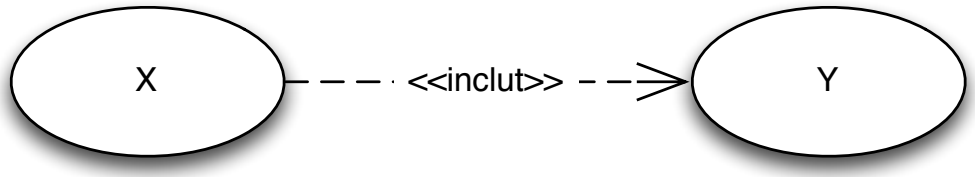
Les associations

- Association : relation entre acteurs et cas d'utilisation qui représente la possibilité pour l'acteur de déclencher le cas

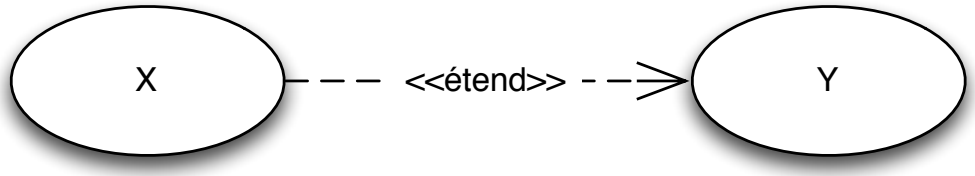


Relations entre cas d'utilisation

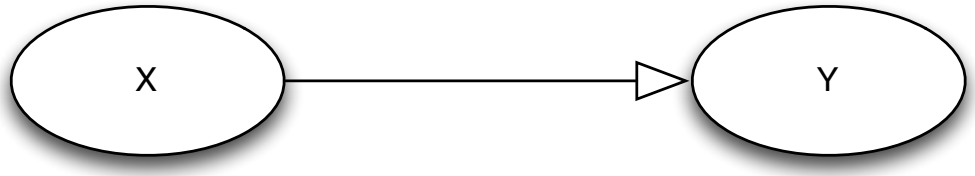
- Inclusion : X inclut Y $\Longleftrightarrow$ X implique Y



- Extension : X étend Y $\Longleftrightarrow$ X peut être provoqué par Y

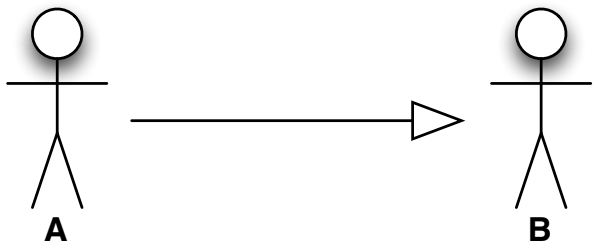


- Généralisation : X est une généralisation de Y $\Longleftrightarrow$ X est un cas particulier de Y

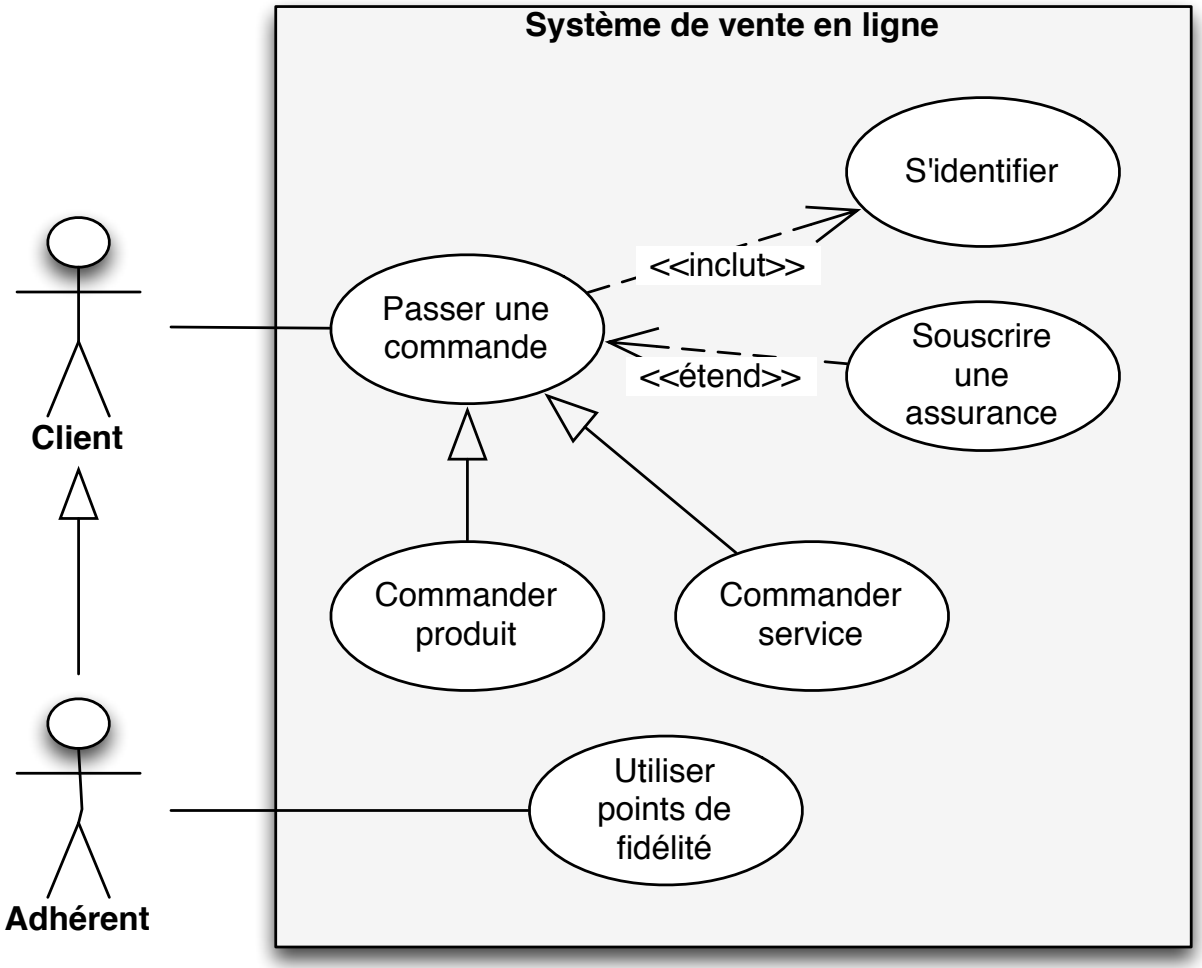


Relations acteurs

- Généralisation : A est une généralisation de B $\Longleftrightarrow$ A peut faire tout ce que fait B



Exemple


Système de vente en ligne
S'identifier
<<inclut>>
Passer une commande
Souscrire une assurance
<<étend>>
Commander produit
Commander service
Utiliser points de fidélité
Client
Adhérent

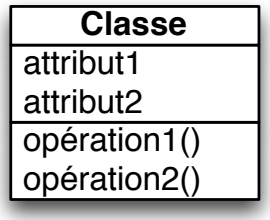
Plan

Rôle

- Montre la structure statique d'un système
- Explique ce qu'il faut réaliser, pas comment le réaliser

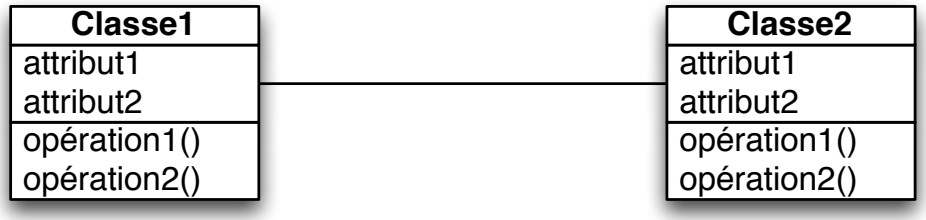
Les classes

- Classe : regroupement d'objets de même nature



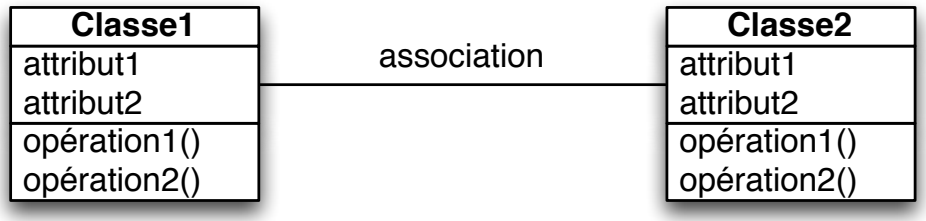
Les associations

- Association : relation entre classes



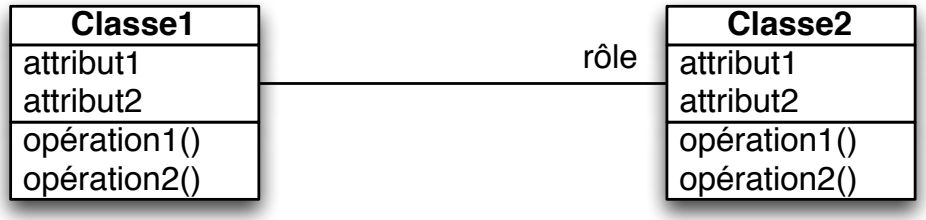
Nommage des associations

- Les association peuvent être nommées



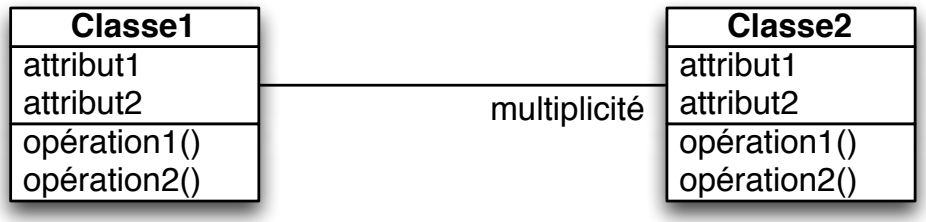
Les rôles

- Rôle : indique comment une classe voit une autre classe au travers de l'association

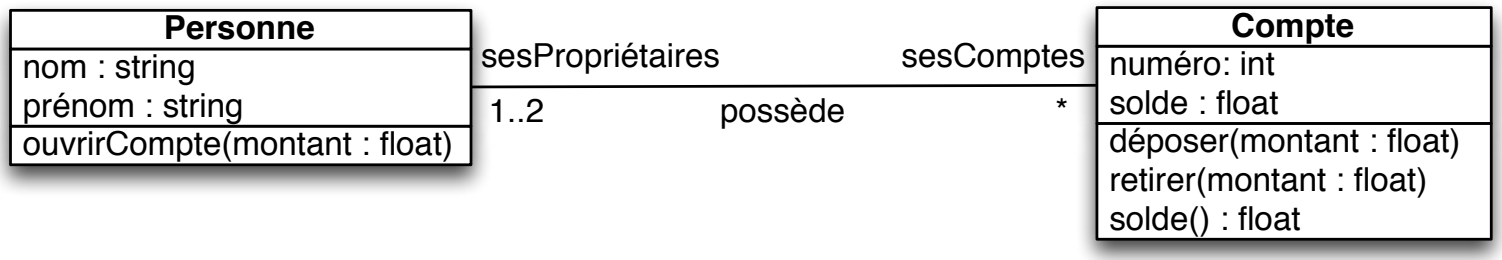


La multiplicité

- Multiplicité : montre le nombre d'objets liés par une association
  - n : exactement n
  - n..m : de n à m
  - * : de zéro à plusieurs
  - n..* : de n à plusieurs

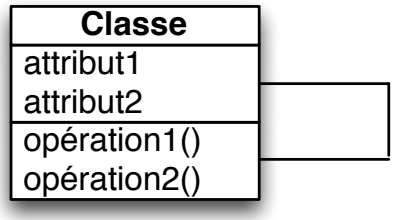


Exemple

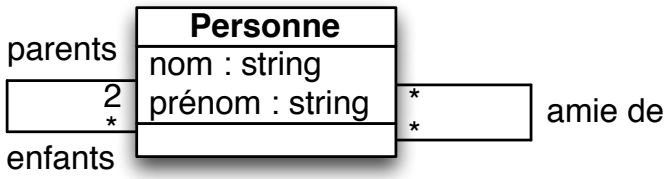

Personne
nom : string
prénom : string
ouvrirCompte(montant : float)
sesPropriétaires
1..2
possède
sesComptes
*
Compte
numéro: int
solde : float
déposer(montant : float)
retirer(montant : float)
solde() : float

Les associations réflexives

- Association réflexive : association d'une classe vers elle même

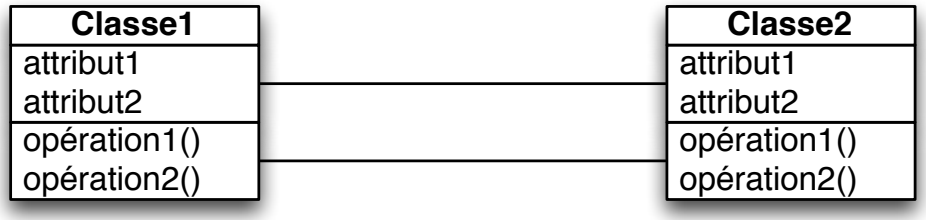


Exemple

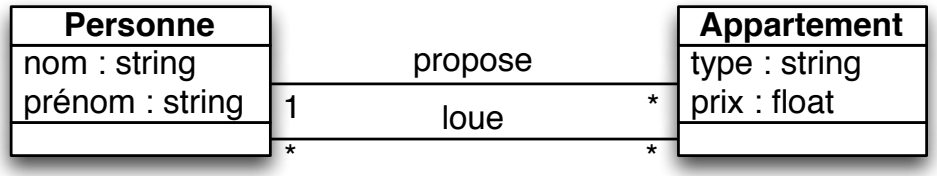

Personne
nom : string
prénom : string
parents
2
*
enfants
*
*
amie de

Les associations multiples

- Associations multiple : plusieurs associations entre deux classes

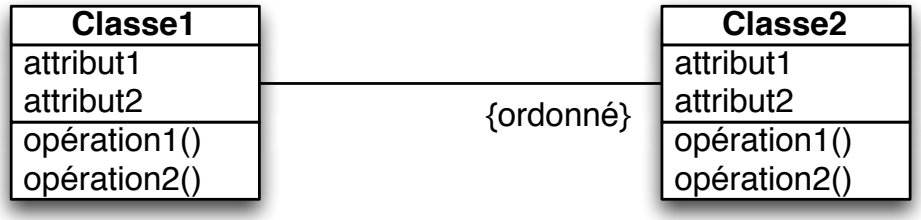


Exemple

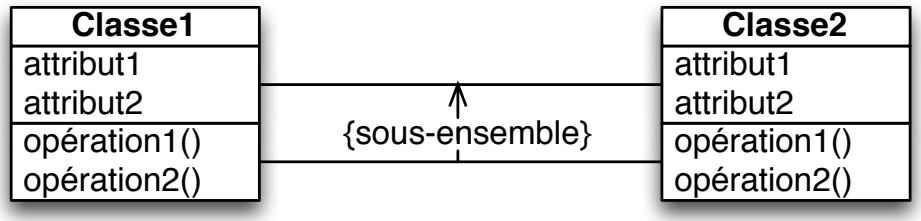

Personne
nom : string
prénom : string
propose
1
*
loue
*
*
Appartement
type : string
prix : float

Les contraintes sur les associations

- {ordonné} : l'ordre doit être maintenu

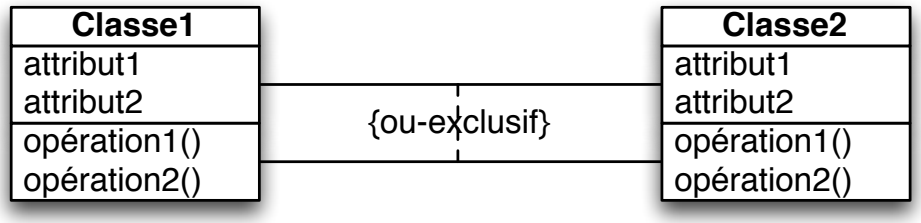


- {sous-ensemble} : une collection est incluse dans une autre

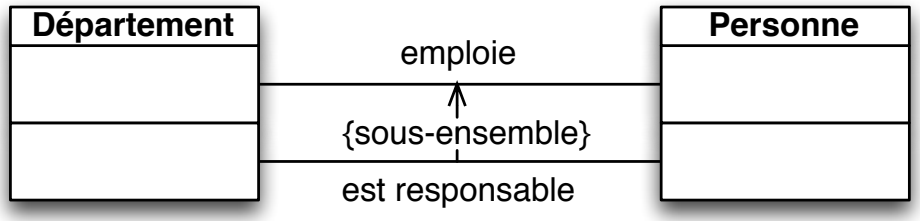


Les contraintes sur les associations

- {ou-exclusif} : une seule association est valide

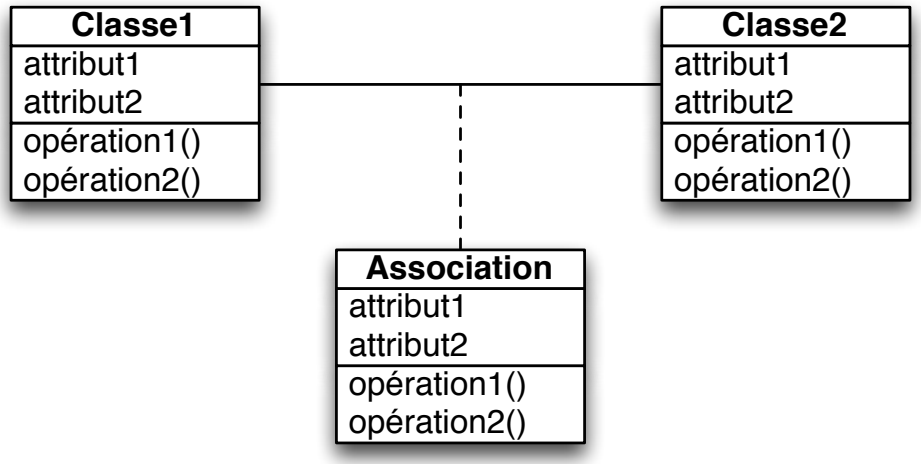


Exemple

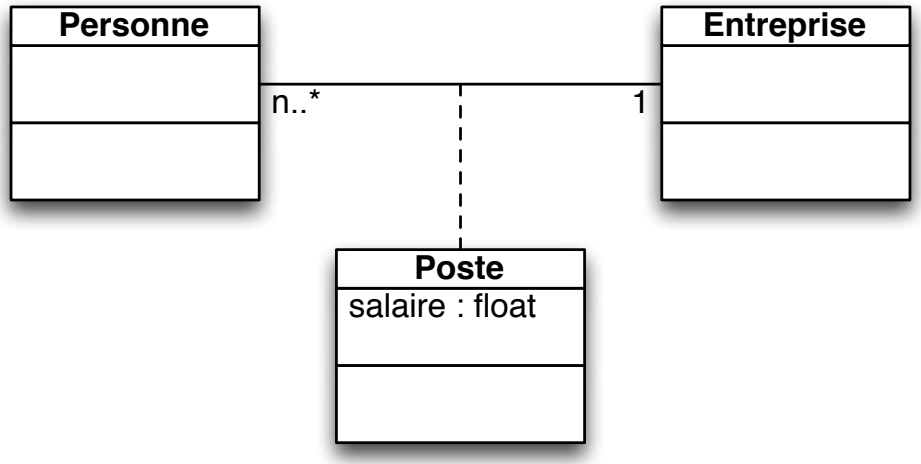

Département
Personne
emploie
{sous-ensemble}
est responsable

Les classes-associations

- Classe-association : une classe permettant de paramétrer une association entre deux classes

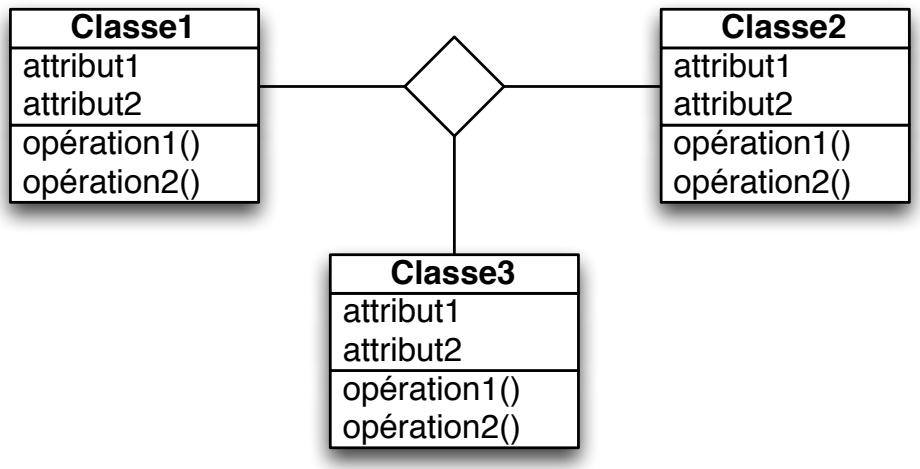


Exemple

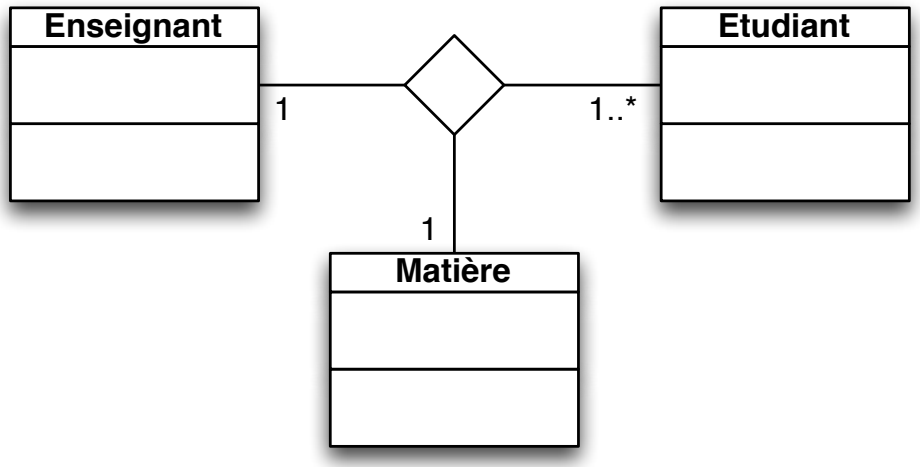


Les associations n-aires

- Association n-aire : une association reliant plus de deux classes

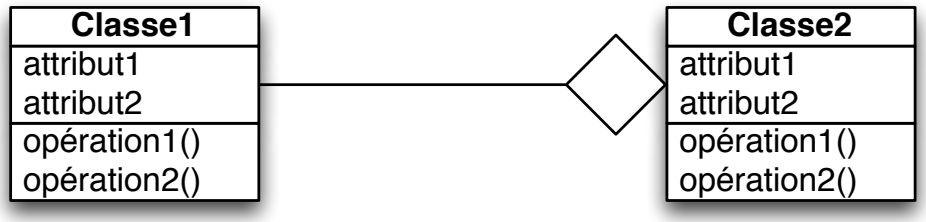


Exemple

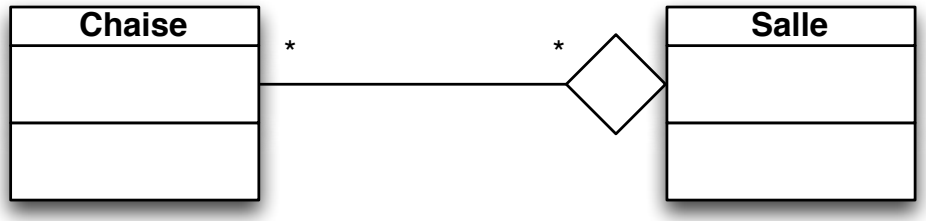


Les agrégations

- Agrégation : montre qu'une classe fait partie d'une autre classe

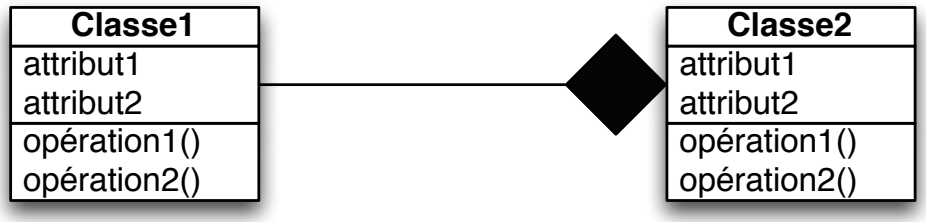


Exemple

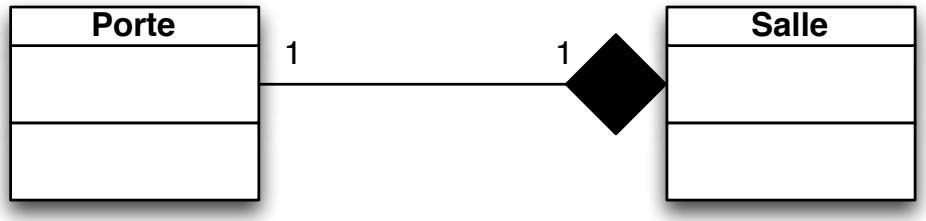


Les compositions

- Composition : cas particulier de l'agrégation

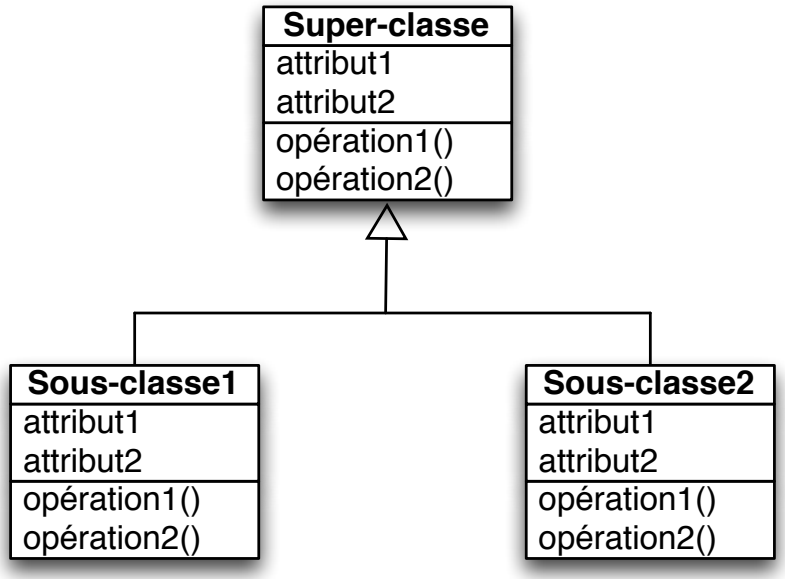


Exemple

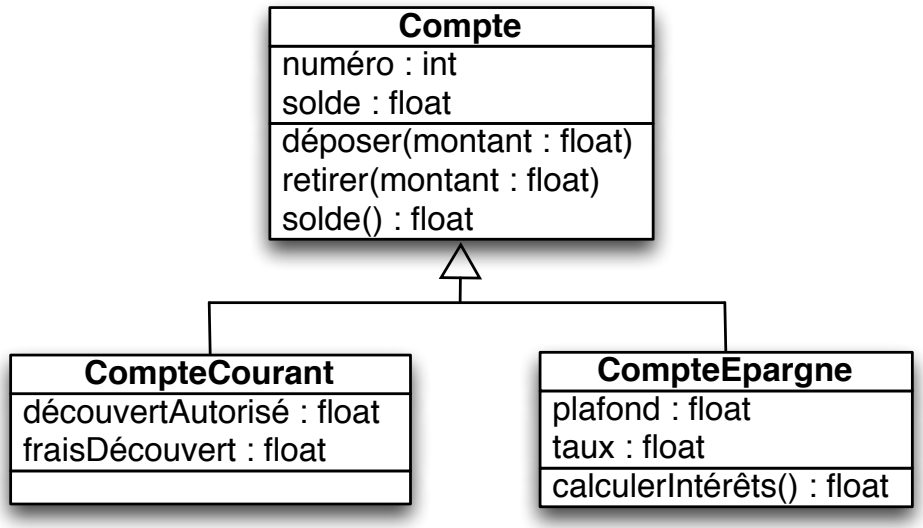


L'héritage

- Héritage : désigne la relation de la classification entre un élément général et un élément plus spécifique

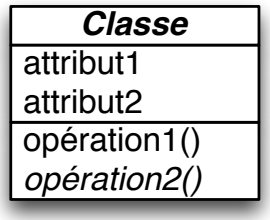


Exemple


Compte
numéro : int
solde : float
déposer(montant : float)
retirer(montant : float)
solde() : float
CompteCourant
découvertAutorisé : float
fraisDécouvert : float
CompteEpargne
plafond : float
taux : float
calculerIntérêts() : float

Les classes abstraites

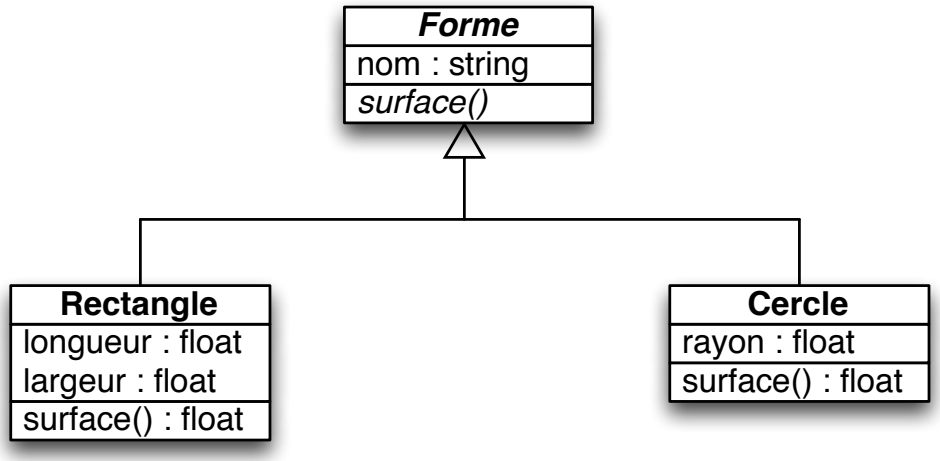
- Classe abstraite : classe non instantiable



Le polymorphisme

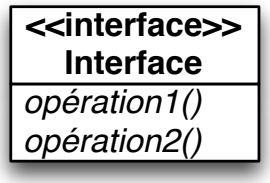
- Polymorphisme : représente la faculté d'une méthode à pouvoir s'appliquer à des objets de classes différentes

Exemple

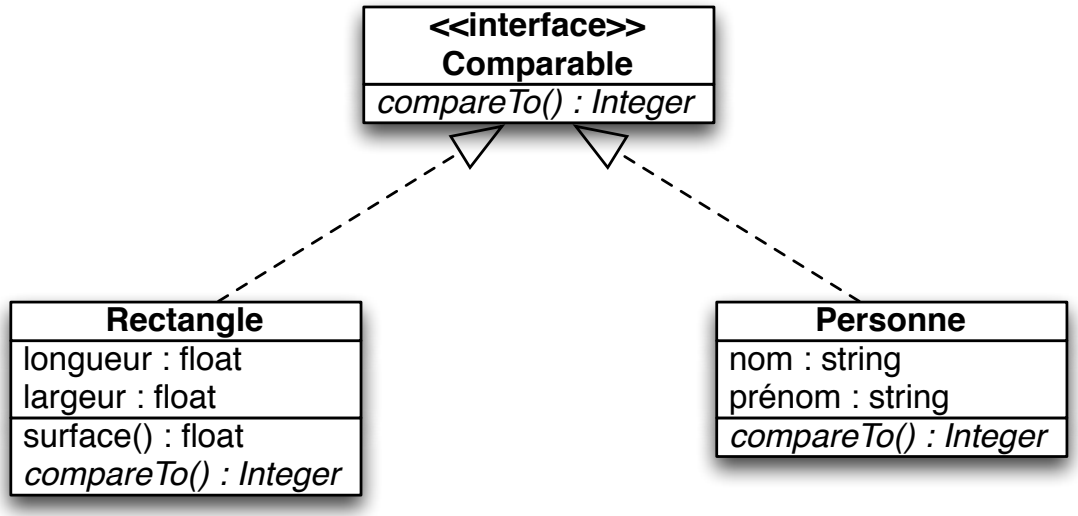

Forme
nom : string
surface()
Rectangle
longueur : float
largeur : float
surface() : float
Cercle
rayon : float
surface() : float

Les interfaces

- Interface : regroupe un ensemble d'opération à respecter par un certain nombre de classes

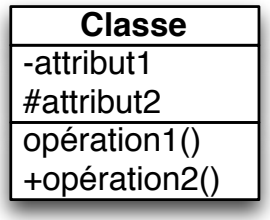


Exemple


<<interface>>
Comparable
compareTo() : Integer
Rectangle
longueur : float
largeur : float
surface() : float
compareTo() : Integer
Personne
nom : string
prénom : string
compareTo() : Integer

Visibilité

- Visibilité
  - Public (+) : visible partout
  - Protégé (#) : visible dans la classe et par tous ses descendants
  - Privé (-) : visible uniquement dans la classe
  - Aucun : visible uniquement dans le paquetage où la classe est définie

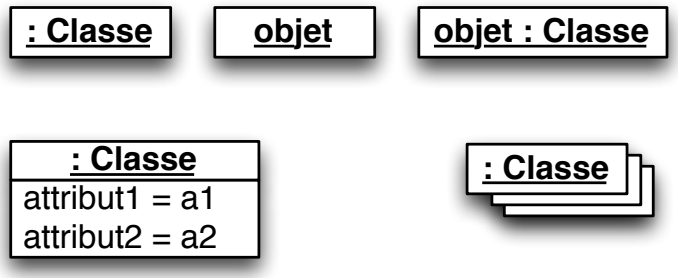


Rôle

- Une instance d'un diagramme de classe
- Montre l'état du système à un instant donné

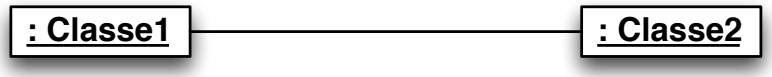
Les objets

- Objet : instanciation d'un classe



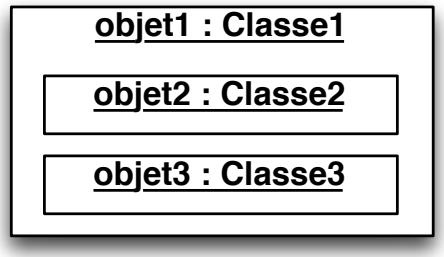
Les liens

- Lien : instance d'une association

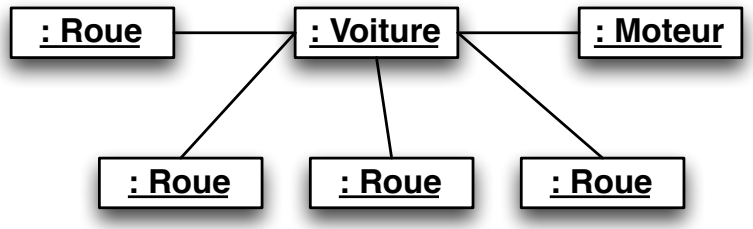


Les objets composites

- Objet composite : objet classique, sauf que les attributs sont remplacés par des objets



Exemple



Plan

Rôle

- Montre les interactions entre objets/acteurs selon un point de vue temporel
- Illustre les cas d'utilisation

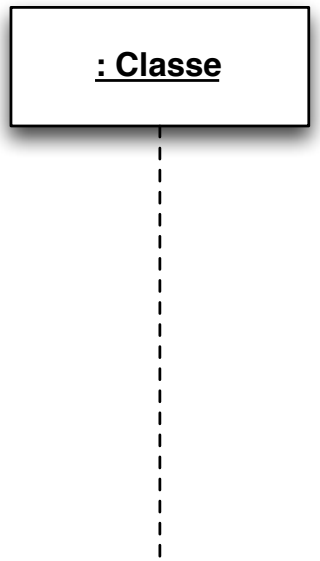
Les acteurs et les objets

- Acteurs : acteurs d'un cas d'utilisation
- Objets : objets qui communiquent entre eux ou avec les acteurs

Même représentation que dans les diagrammes de cas d'utilisation et le diagramme d'objets

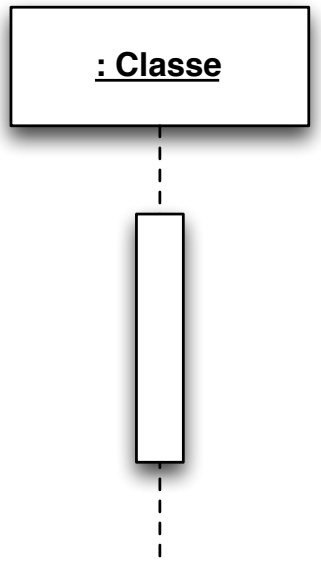
Lignes de vie

- Ligne de vie : précise l'existence d'un l'objet/acteur



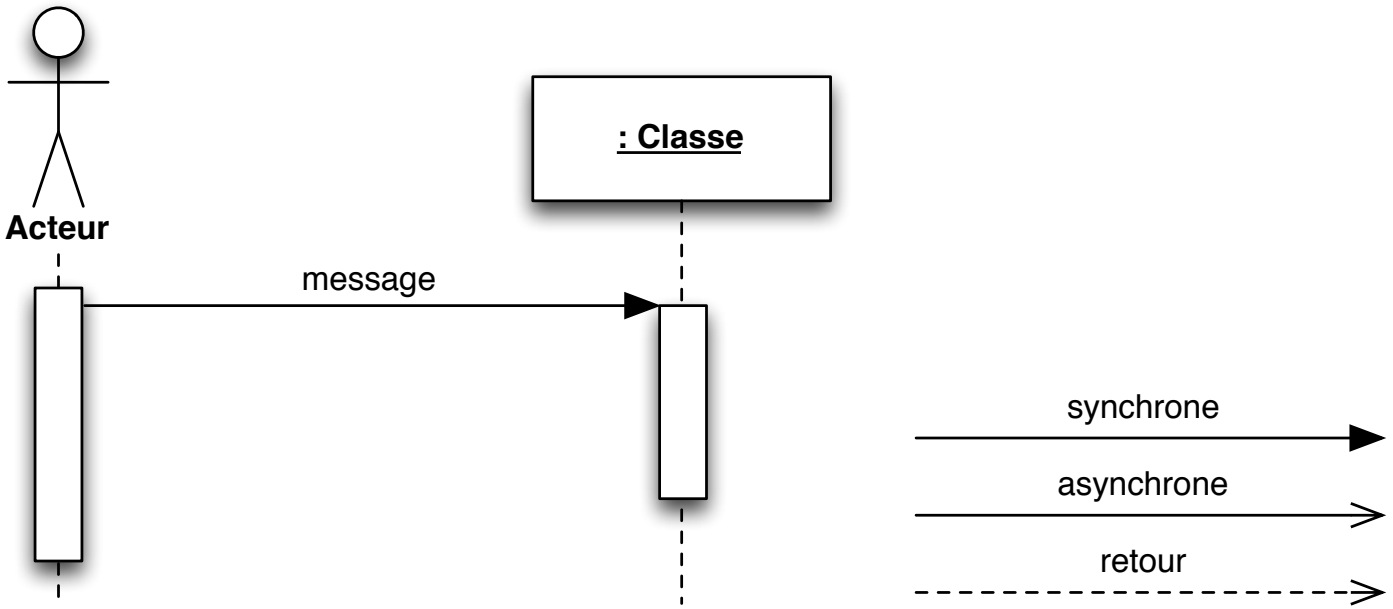
Les périodes d'activités

- Période d'activité : temps pendant lequel un objet/acteur est en activité



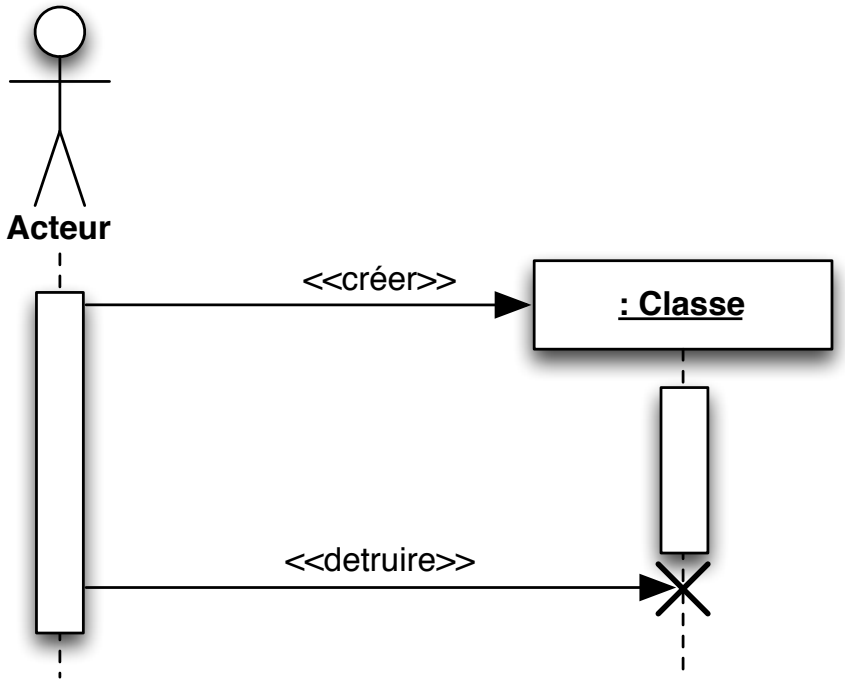
Les messages

- Message : émission d'un événement d'un objet/acteur vers un autre objet/acteur

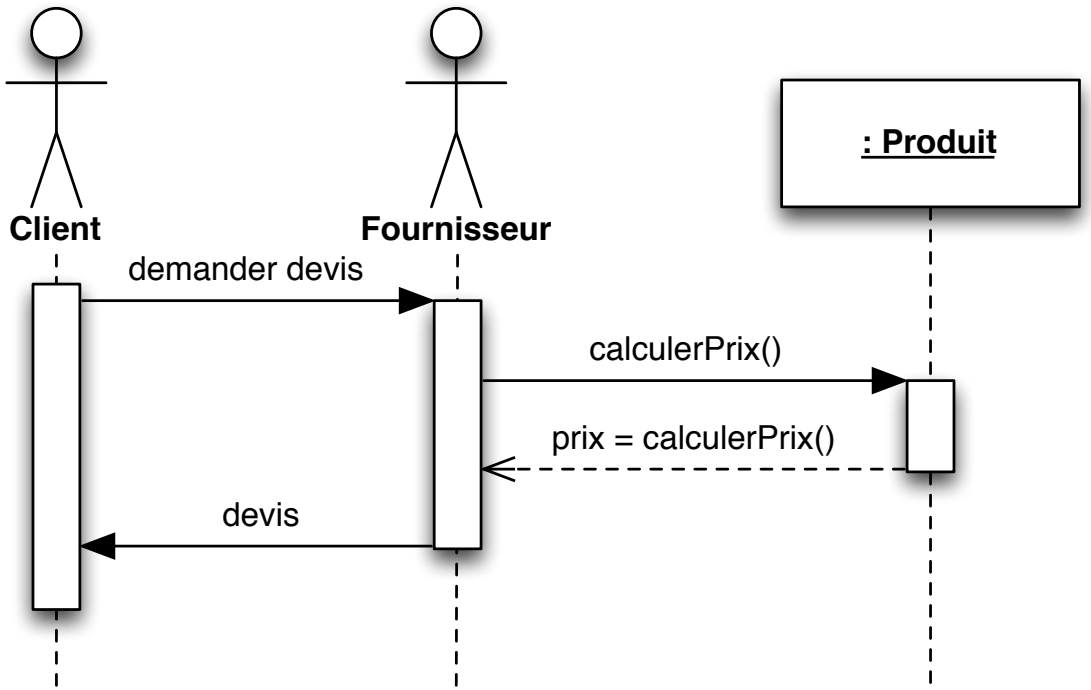


Création et destruction d'instances

- Création : création d'un objet qui n'existe pas
- Destruction : destruction d'un objet qui n'existera plus



Exemple


Client
Fournisseur
: Produit
demander devis
calculerPrix()
prix = calculerPrix()
devis

Rôle

- Montre le comportement collectif des objets/acteurs en vue de réaliser une opération
- Met en évidence les interactions entre les objets/acteurs

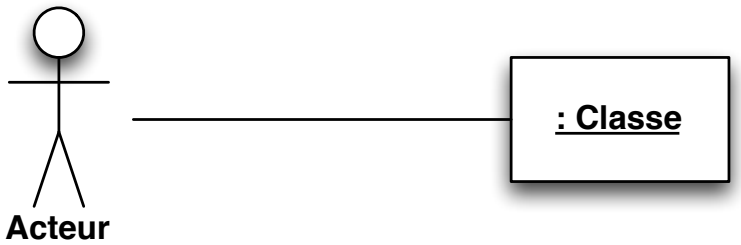
Les acteurs et les objets

- Acteurs : acteurs d'un cas d'utilisation
- Objets : objets qui communiquent entre eux ou avec les acteurs

Même représentation que dans le diagramme de cas d'utilisation et le diagramme d'objets

Les liens d'interaction

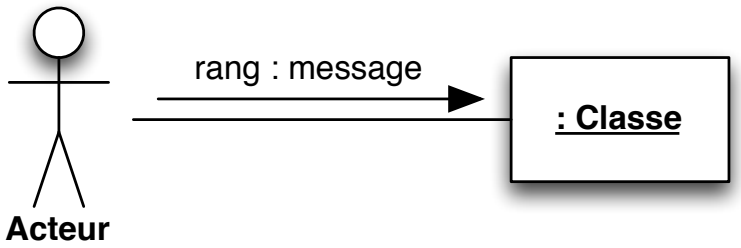
- Lien d'interaction : lien entre un objet/acteur et un autre objet/acteur
- On dit qu'ils communiquent entre deux



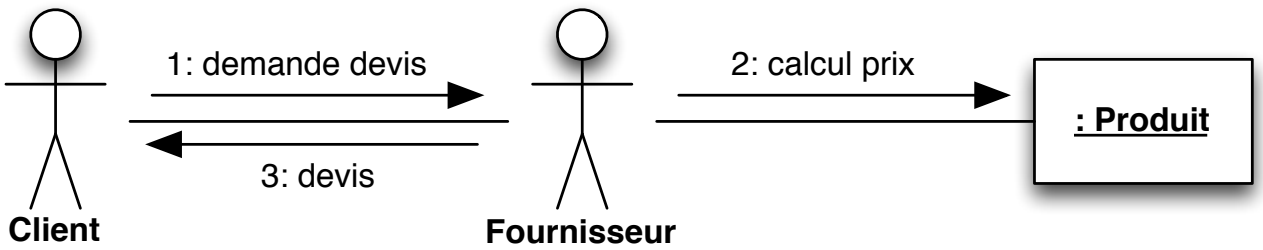
Les messages

- Message : moyen de communication entre les objets/acteurs

Même représentation que dans le diagramme de séquence ; il est possible d'indiquer son rang



Exemple


1: demande devis
2: calcul prix
3: devis
: Produit
Client
Fournisseur

Plan

Rôle

- Décrit les changements d'état d'un objet, en réponse aux interactions avec d'autres objets/acteurs
- Décrit le cycle de vie d'un objet

Les états

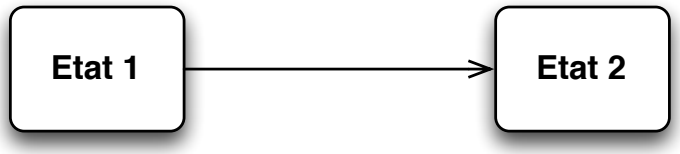
- Etat : situation durable dans laquelle peut se trouver des objet d'une classe donnée

L'état initial & l'état final

- Il faut toujours un état initial
- En revanche, il peut y avoir zéro ou plusieurs états finaux

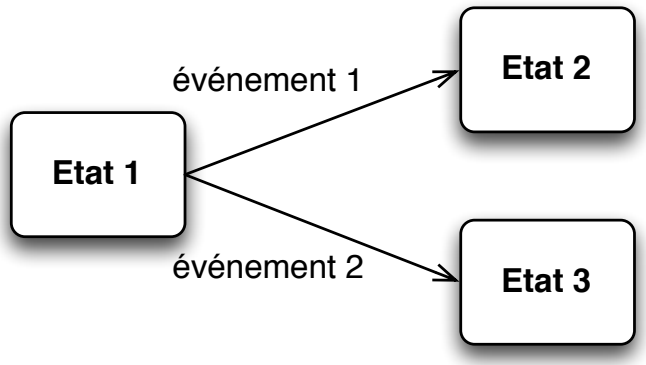
Les transitions

- Transition : représente le passage d'un état vers un autre état



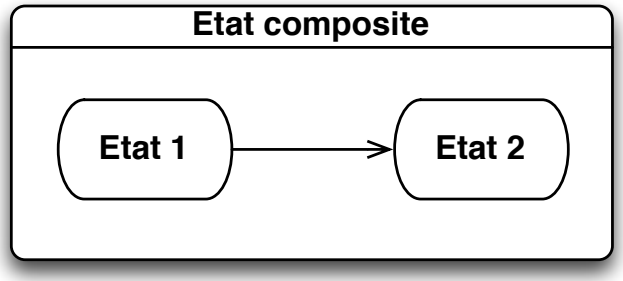
Les événement

- Evénement : indique quel chemin suivre dans le graphe.
- Un événement est caractérisé par son nom et ou bien par :
  - La liste des paramètres
  - L'objet expéditeur
  - L'objet destinataire
  - La description de l'événement



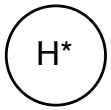
Les états composites

- Etat composite (super-état) : état qui englobe d'autres états et transitions



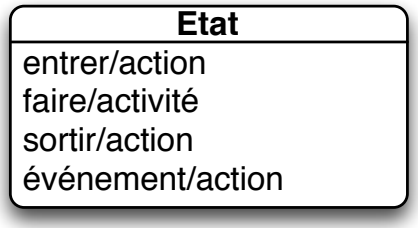
L'historique

- Historique : mémorise le dernier sous-état actif d'un super-état pour y revenir directement ultérieurement



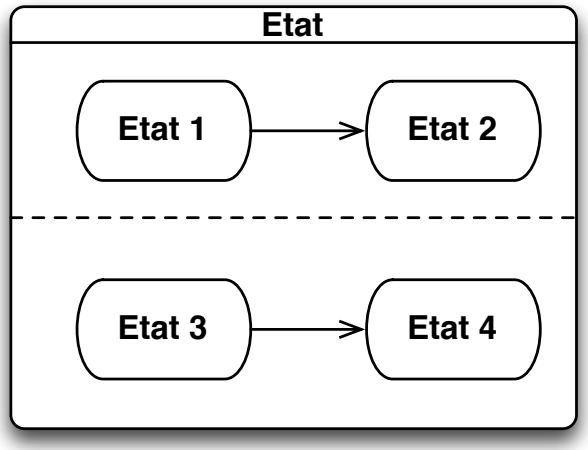
Les actions dans un état

- Action : réaction à un événement

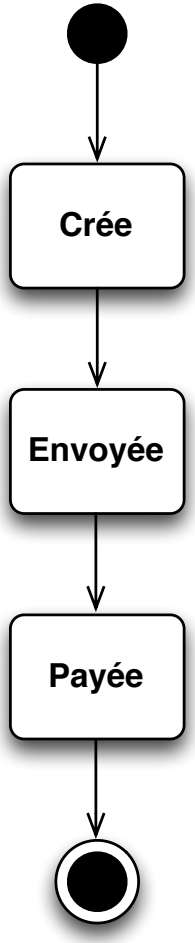


Les états concurrents

- Etat concurrent : état composite dans lequel plusieurs états sont actifs simultanément



Exemple



Plan

Rôle

- Montre le déroulement des activités liées aux cas d'utilisation
- Détermine le flux de données traversant plusieurs cas d'utilisation

Les activités

- Activité (état-action) : action réalisée dans un cas d'utilisation

Le début et la fin des activités ont la même représentation que dans le diagramme d'états-transitions

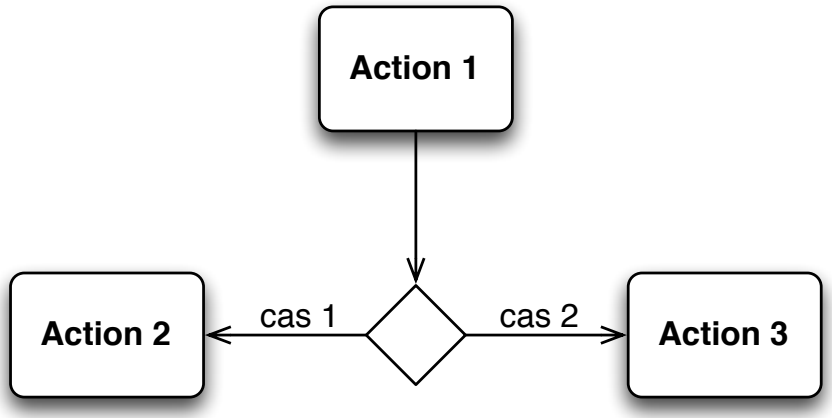
Les transitions

- Transition : représente le passage d'une activité à une autre activité
- Une transition est déclenchée par la fin d'une activité et provoque le début immédiat d'une autre activité ; jusqu'à la fin des activités

Même représentation que dans le diagramme d'états-transitions

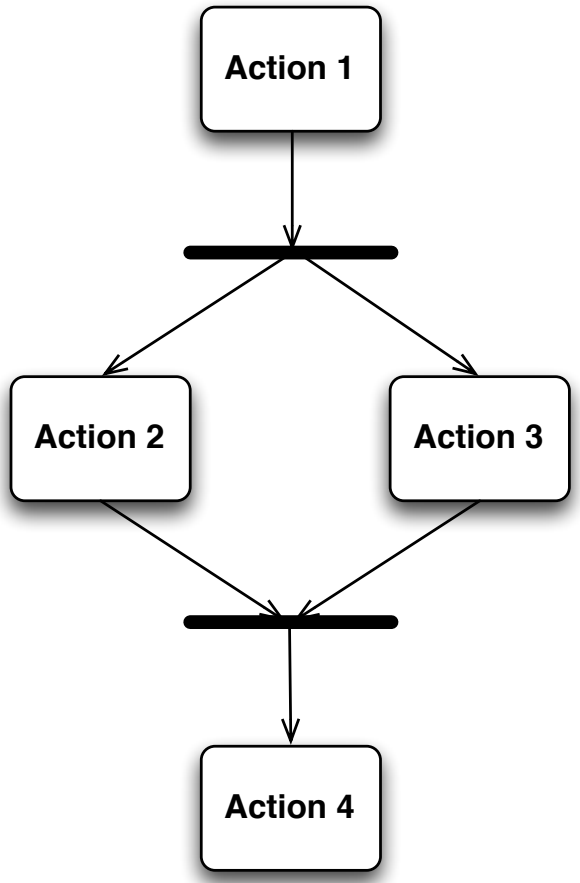
Les alternatives

- Alternative : permet d'indiquer les différents scénarios dans un même diagramme



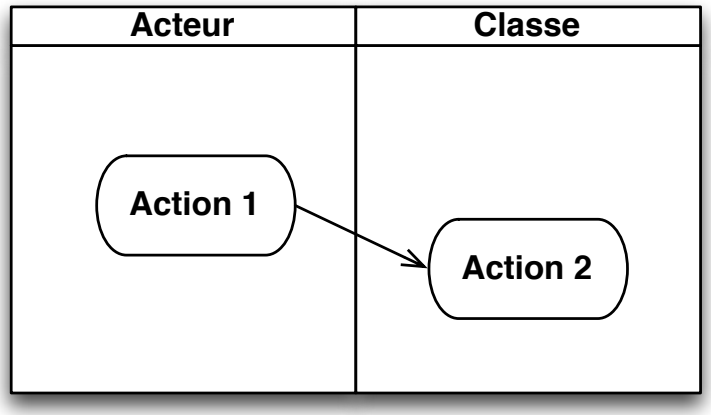
Les barres de synchronisation

- Barre de synchronisation : permet de synchroniser les transitions (fork et join)

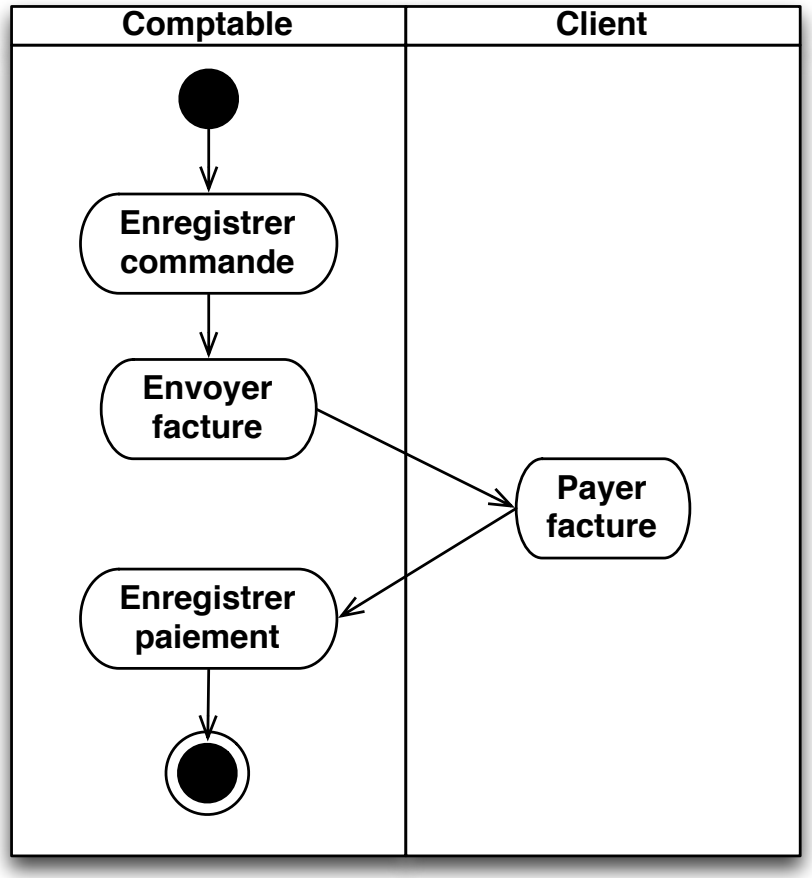


Les couloirs d'activités

- Couloir d'activités : permet d'organiser un diagramme d'activités selon les différents responsables des action représentées



Exemple


Comptable
Client
Enregistrer commande
Envoyer facture
Payer facture
Enregistrer paiement

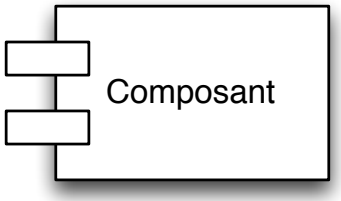
Plan

Rôle

- Décrit l'architecture physique et statique d'une application
- Montre la mise en œuvre physique des modèles de la vue logique avec l'environnement de développement

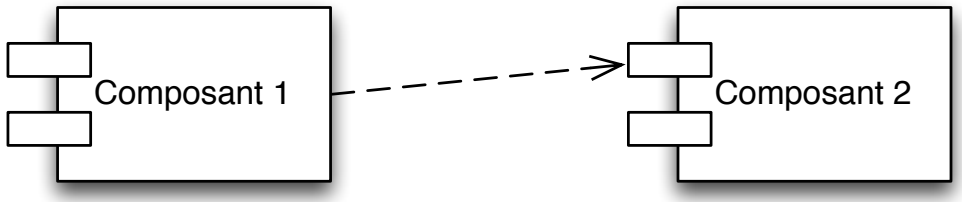
Les composants

- Composant : élément physique qui représente une partie implémentée ou à implémenter d'un système

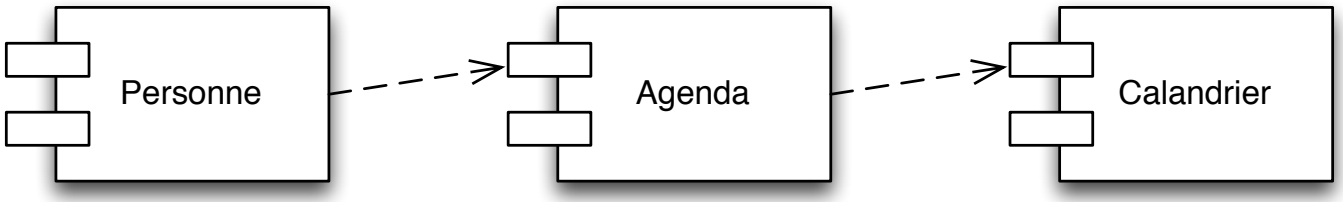


Les dépendances

- Dépendance : indique qu'un élément d'implémentation d'un composant fait appel aux services offerts par les éléments d'implémentation d'un autre composant



Exemple

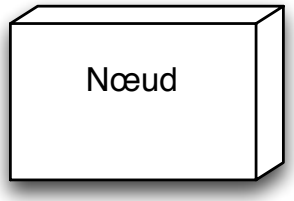


Rôle

- Montre la disposition physique des matériels qui composent le système
- Décrit la répartition des composants sur ces matériels

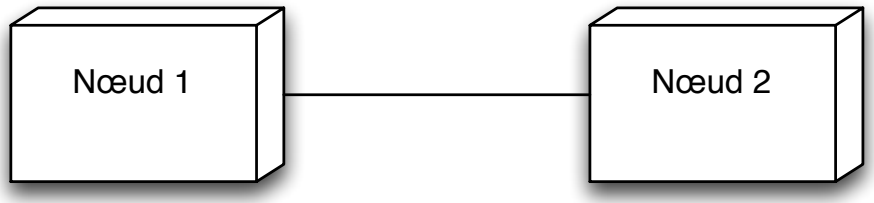
Les nœuds

- Nœud : représente une ressource physique



Les supports de communication

- Support de communication : sert à connecter des nœuds
- Peuvent avoir des cardinalités



Exemple