UML
Dr Khalifa SYLLA
Séquence 2: LES SPECIFICATIONS D'UML

1.1. Introduction

La description par objets a fait ressortir l'étendue du travail conceptuel nécessaire : définition des classes, de leurs relations, des attributs et méthodes, des interfaces, etc. Il convient dans une prochaine étape de faire la modélisation, avec UML.

Comme annoncé dans la précédente section, UML est un langage graphique de modélisation pour **spécifier**, **concevoir**, **construire** et **documenter** des applications informatiques. C'est devenu une norme qui s'impose en technologie à objets et à laquelle se sont rangés tous les grands acteurs du domaine, acteurs qui ont d'ailleurs contribué à son élaboration.

UML 2.0 comporte treize types de diagrammes représentant autant de vues distinctes pour représenter des concepts particuliers du système d'information. Ces diagrammes sont classés en deux groupes :

Les diagrammes statiques

- Diagramme de classe
- Diagramme d'objets
- Diagramme de composants
- Diagramme de déploiement
- Diagramme de paquetages
- Diagramme de structures composites

Les diagrammes dynamiques

- Diagramme de cas d'utilisation
- Diagramme d'activités
- Diagramme d'états-transitions
- Diagramme de séquence
- Diagramme de communication
- Diagramme global d'interaction
- Diagramme de temps

Pour une modélisation, ces diagrammes ne sont pas tous nécessaires. Les plus utiles sont :

- Diagramme de cas d'utilisation
- Diagramme de classe
- Diagramme d'activités
- Diagramme de séquence
- Diagramme d'états-transitions

Dans une approche classique, un projet suivra les étapes suivantes :

1) Fonctionnel
    a. Définition du cahier des charges : dans cette étape c'est le diagramme de cas d'utilisation qui permettra de produire des scénarios écrits
    b. Elaboration des scénarios formels : les diagrammes de séquences permettront d'identifier les objets et classes
2) Statique
    a. Elaboration des diagrammes de classes
3) Dynamique
    a. Dynamique de chaque objet représentée par le diagramme d'états/transitions
    b. Dynamique globale du système est représentée par le diagramme d'activités

1.2. Les diagrammes d'UML

1.2.1. Diagramme de cas d'utilisation

Ce diagramme décrit le comportement général du système vis à vis des utilisateurs du système. Il s'agit de la description des **fonctionnalités** du système d'information. Pour ce faire, on se pose la question suivante : dans quel cas ce système est-il utilisé et par qui ?

Le diagramme de cas d'utilisation est le point d'entrée pour les étapes suivantes du développement. Il permet au client de décrire ses besoins, et pour parvenir à un accord entre clients et développeur.

Qu'est-ce qu'un cas d'utilisation ? C'est l'usage que les acteurs font du système.

1.2.1.1. Eléments des diagrammes de cas d'utilisation

a) Acteur

L'acteur est une entité qui interagit avec les systèmes :

- Il peut consulter ou modifier l'état du système
- Il est celui qui bénéficie de l'utilisation du système
- En réponse à l'action d'un acteur, le système fournit un service qui correspond à son besoin.

Une même personne peut jouer le rôle de différents acteurs ; et un acteur peut être un autre système.

Stéréotype d'un acteur en UML

Exemple d'acteur : client

b) Cas d'utilisation (use case)

C'est un ensemble d'actions réalisées par le système en réponse à une action d'un acteur. L'ensemble des use cases décrit les objectifs (le but) du système. Les use cases permettent de structurer les besoins des utilisateurs. Les use cases centrent l'expression des exigences du système sur ses utilisateurs.

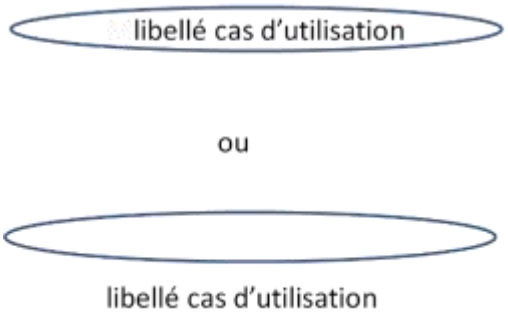


Figure : Stéréotype d'un use case en UML

Il existe des relations de dépendance et de généralisation entre les cas d'utilisation (CU).

1.2.1.2. Les relations entre cas d'utilisation et acteurs

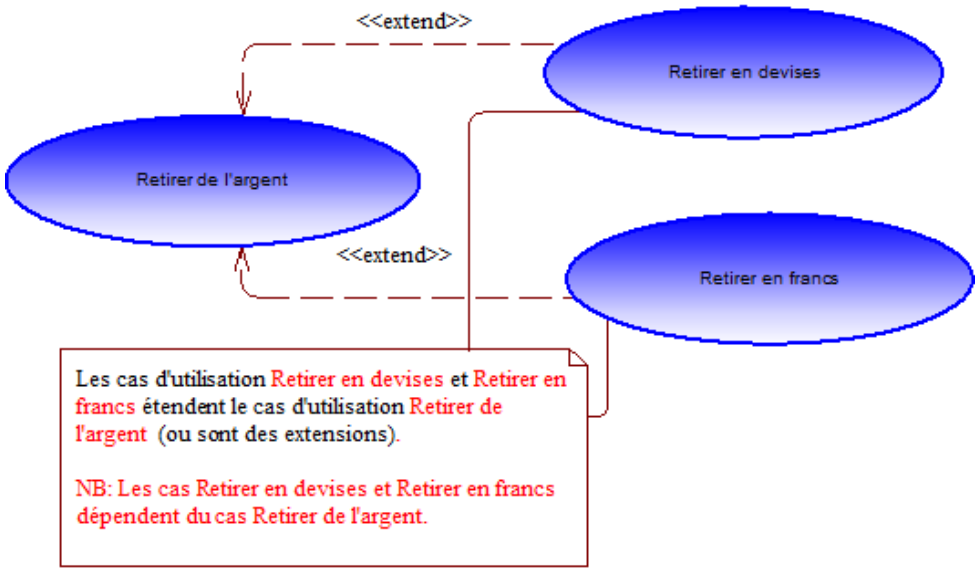
a) Relation d'extension (dépendance)

Soient deux cas d'utilisation A et B. Le cas d'utilisation B étend le cas d'utilisation A (ou est une extension de A) signifie qu'une instance de A peut être étendue par le comportement décrit dans B.

La relation d'extension précise que le CU B dépend du CU A et B est optionnel pour le CU A.

Remarque : Une relation d'extension indique que le CU source précis l'activité, les objectifs (ie le comportement) du CU destination.

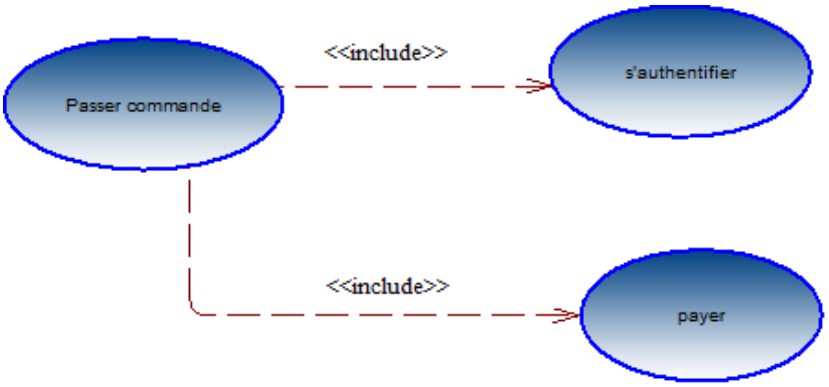
Exemple de relation d'extension



b) Relation d'utilisation (dépendance)

Elle indique que le cas d'utilisation source contient aussi, le cas d'utilisation destination. La destination est une sous-activité, un sous-processus. Un sous-cas d'utilisation de la source. Pour atteindre l'objectif de la source, on passe par la réalisation du ou des objectif(s) de la ou des destination(s).

Exemple de relation d'utilisation

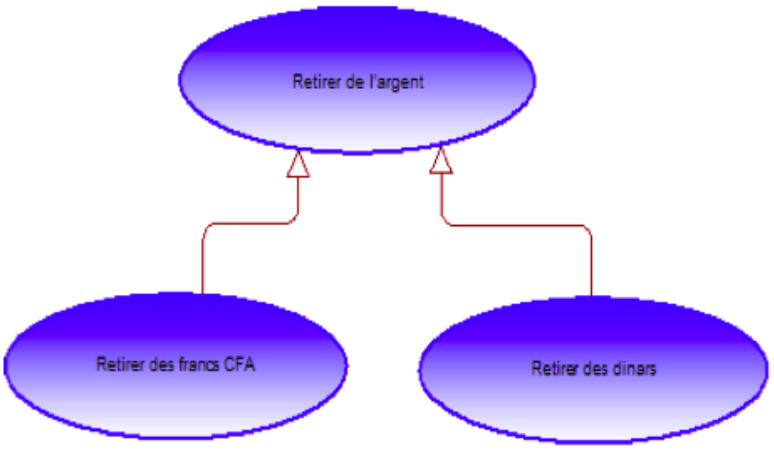


Le cas d'utilisation Passer commande dépend des cas d'utilisation s'authentifier et payer.

c) Relation de généralisation

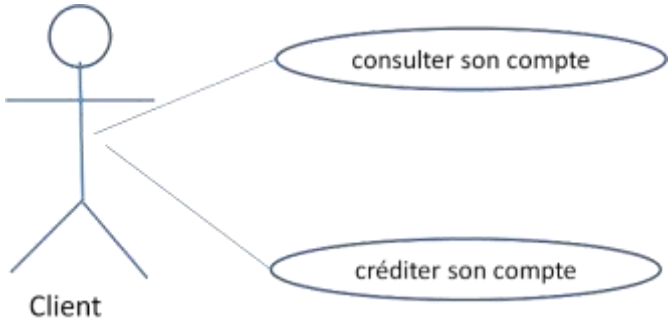
Une relation de généralisation de cas d'utilisation peut être définie conformément au principe de la spécialisation-généralisation déjà présentée pour les classes.

Exemple de généralisation



d) Relation entre acteur et cas d'utilisation

Cette relation est un lien symbolisé par un trait (ou une flèche) partant de l'acteur au cas d'utilisation signifiant que l'acteur est le déclencheur (ou l'initiateur) du cas d'utilisation.

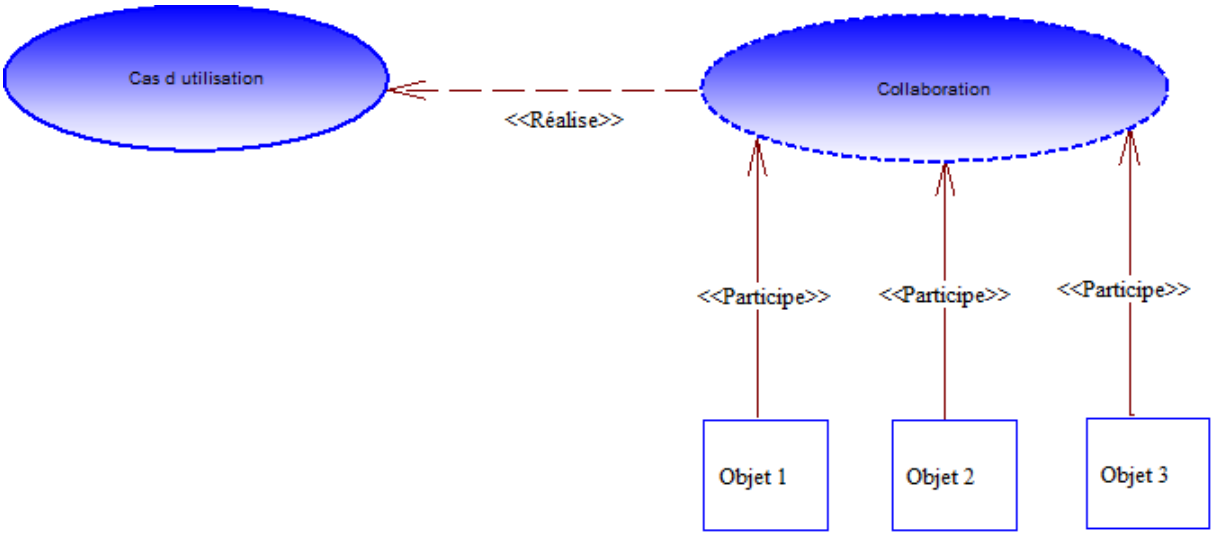


Exemple : Système bancaire

1.2.1.3. Les collaborations

Une collaboration dérive d'un use case. Elle correspond à un objectif du système.

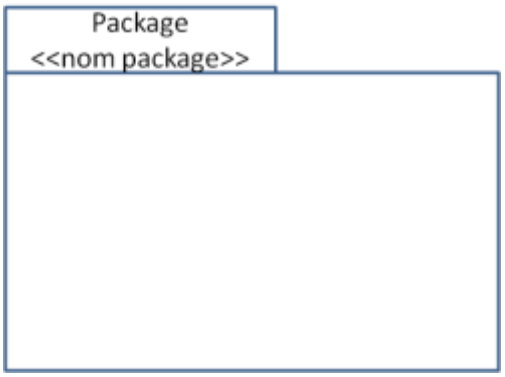
Une collaboration fait intervenir toutes les classes (les objets) dont la collaboration est requise pour l'atteinte de l'objectif visé à travers le use case en question.



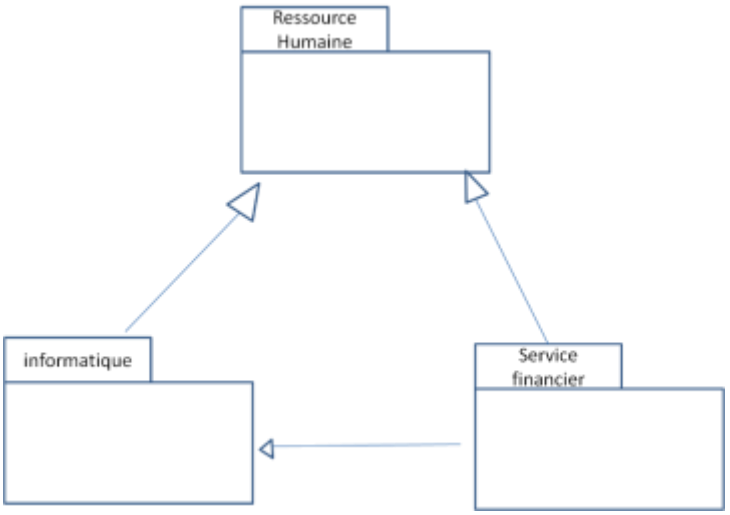
1.2.1.4. Les packages (ou paquetages)

Les packages sont des éléments d'organisation des modèles. Ils regroupent des éléments de modélisation, selon des critères purement logiques.

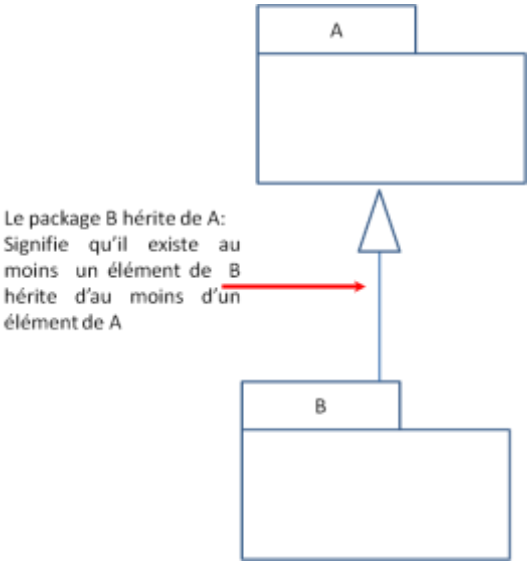
Les paquetages permettent d'encapsuler des éléments de modélisation (ils possèdent interface, son nom). Ils servent de "briques" de base dans la construction d'une architecture.



Stéréotype d'un paquetage en UML



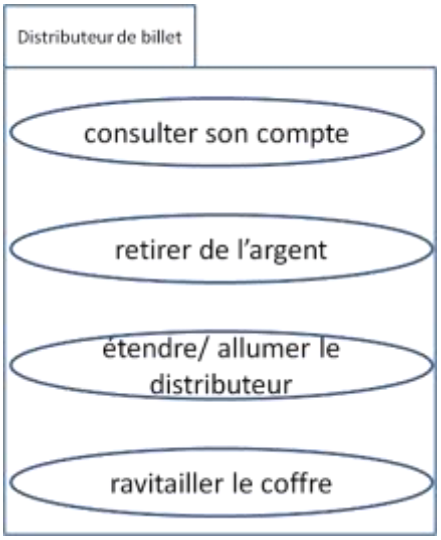
Exemple de relation de dépendance entre packages



Exemple de relation d'héritage entre packages

- **Package de cas d'utilisation**

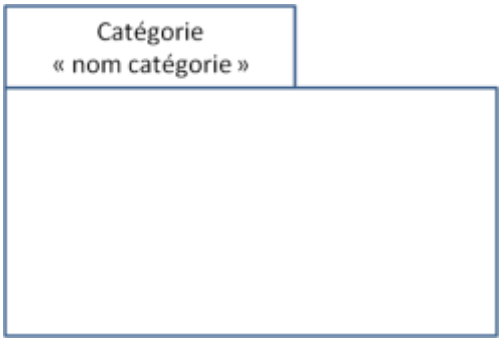
Il contient des cas d'utilisation qui concerne un même domaine ou sous-domaine d'un système.



Exemple de package de cas d'utilisation : système de distributeur automatique

- **Catégories de classes**

C'est un package regroupant un ensemble de classes fortement liées, en ce sens qu'elles concourent étroitement, à la réalisation d'au moins un objectif, du système. C'est le cas d'un ensemble de classes participant à au moins une collaboration.

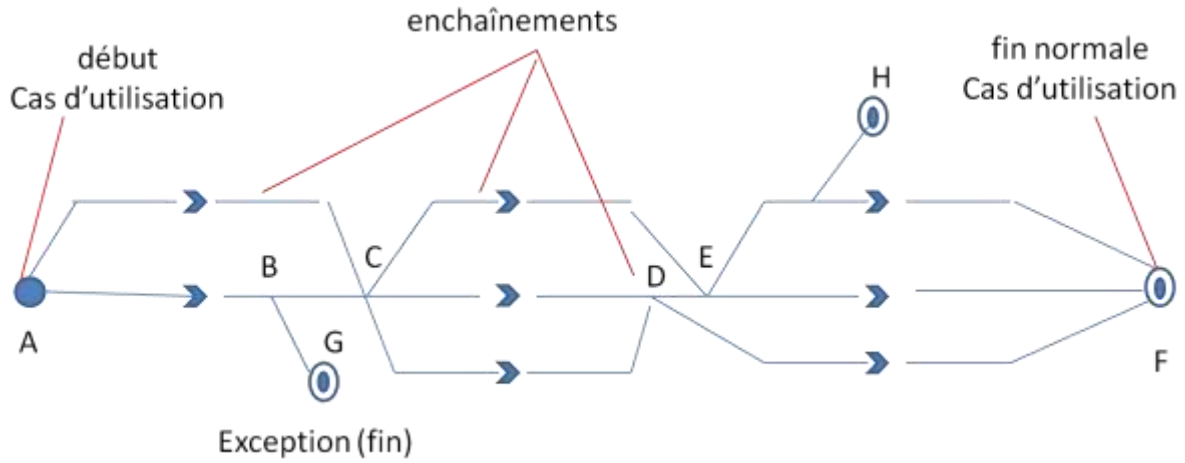


Stéréotype d'une catégorie en UML

Remarques : Certains modélisateurs utilisent indifféremment **Package** et **Catégorie**. D'autres réservent le concept de package pour les use cases et de Catégorie pour les classes.

1.2.1.5. Enchaînement de séquence

- Une séquence est une suite d'interactions entre le système et un ou plusieurs de ses acteurs. La détermination d'une séquence est relative, en ce sens que le nombre d'interactions simplifiées ainsi que leurs durées ne sont pas figés.
- Un enchainement de séquence va représenter une unité de description de séquences cohérentes permettant d'atteindre un objectif opérationnel simple.
- Un scénario représente une succession particulière d'enchaînements, qui s'exécute du début à la fin du cas d'utilisation. Les scénarii sont aux cas d'utilisation ce que les objets sont aux classes. En d'autres termes un scénario est une instance d'un cas d'utilisation.



Enchaînements: AC, CD, CE, BG, DE,

Scénarios nominaux: ABCDEF, ACEF, ACDF, ACDEF

Scénarios d'exception: ABG, ACEH, ABCDEH

Exemple : Use case - Acteur

Use Case - Acteur

Nom use case: Achat ticket

Acteur: Passager

Condition d'entrée:

- Passager devant le distributeur de ticket.
- Passager a suffisamment d'argent pour acheter le ticket.

Condition de sortie: Passager a le ticket.

Voici le scénario correspond :

1) Passager choisit le nombre de zones traversées.
2) Le distributeur affiche la somme due.
3) Passager insère l'argent correspondant à au moins la somme due.
4) Distributeur rend la monnaie.
5) Distributeur donne le ticket.

1.2.1.6. Elaboration des diagrammes de cas d'utilisation

Pourquoi faire d'un cas d'utilisation?

C'est pour permettre au client de décrire ses besoins :

- Parvenir à un accord entre clients et développeurs
- Point d'entrée pour les étapes suivantes du développement

Qu'est-ce qu'un cas d'utilisation ?

C'est l'usage que les acteurs font du système.

L'acteur est l'entité extérieure qui interagit avec le système. Une même personne peut jouer le rôle de différents acteurs. Un acteur peut être un autre système.

L'usage va représenter une séquence d'interactions entre système et les acteurs.

Un cas d'utilisation est généralement composé de plusieurs scénarios (instances). Il s'agit de :

- Scénario de base et ses variantes
- Description des scénarios à l'aide de diagrammes de séquence

Comment découvrir les cas d'utilisation ?

- Délimiter le périmètre du système
- Identifier les acteurs interagissant avec le système:
    - Ceux qui utilisent le système
    - Ceux qui fournissent un service au système
- Identifier les acteurs principaux
    - Ceux qui utilisent système pour atteindre un but
- Définir les cas d'utilisation correspondant à ces buts
    - Nom - verbe à l'infinitif - groupe nominal

Comment décrire les cas d'utilisation ?

La description d'un cas d'utilisation se fait à l'aide des :

- Diagramme de cas d'utilisations qui fait un récapitulatif graphique des interactions entre acteurs et cas d'utilisations
- Diagramme de séquence
- Description de chaque scénario
- Séquence d'interactions entre les acteurs et le système

Relation entre cas d'utilisations

A << include >> B: le use case A inclut obligatoirement le use case B (permet de décomposer et de factoriser)

A << extend >> B: le use case A est extension du B à un certain point de son exécution

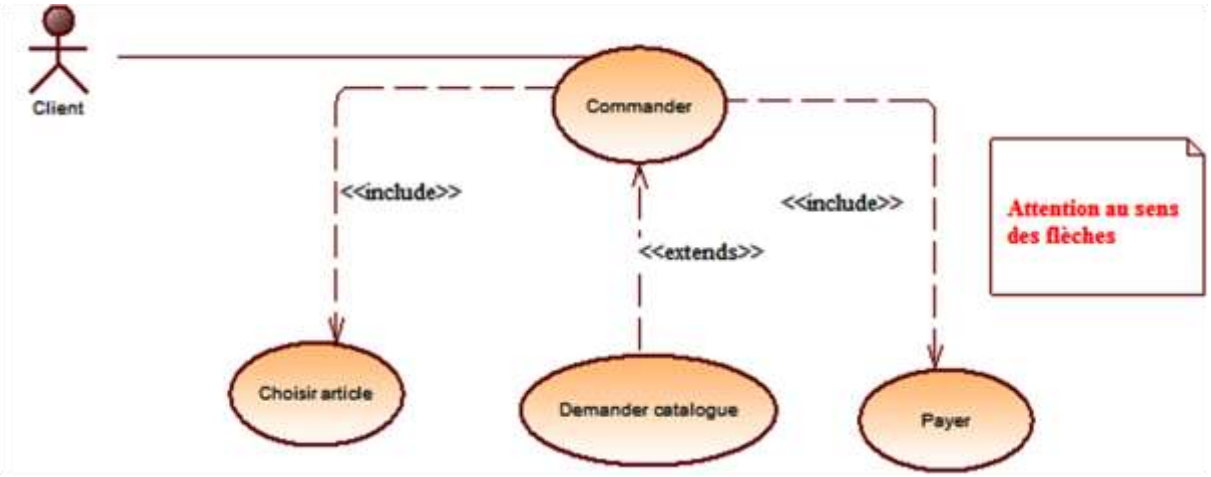
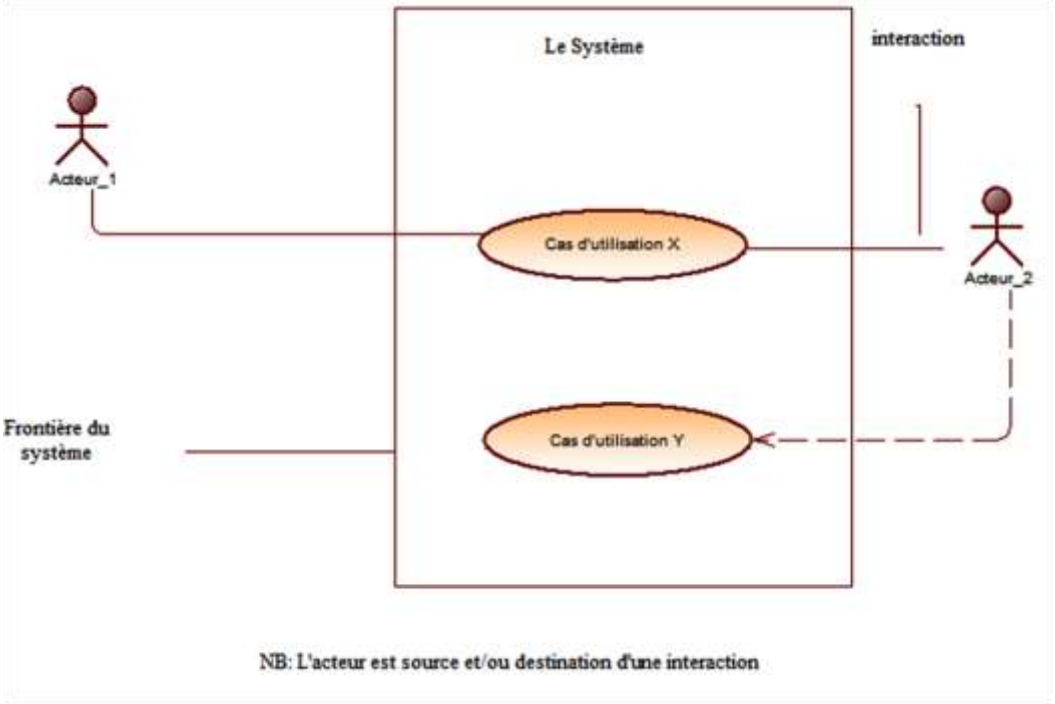
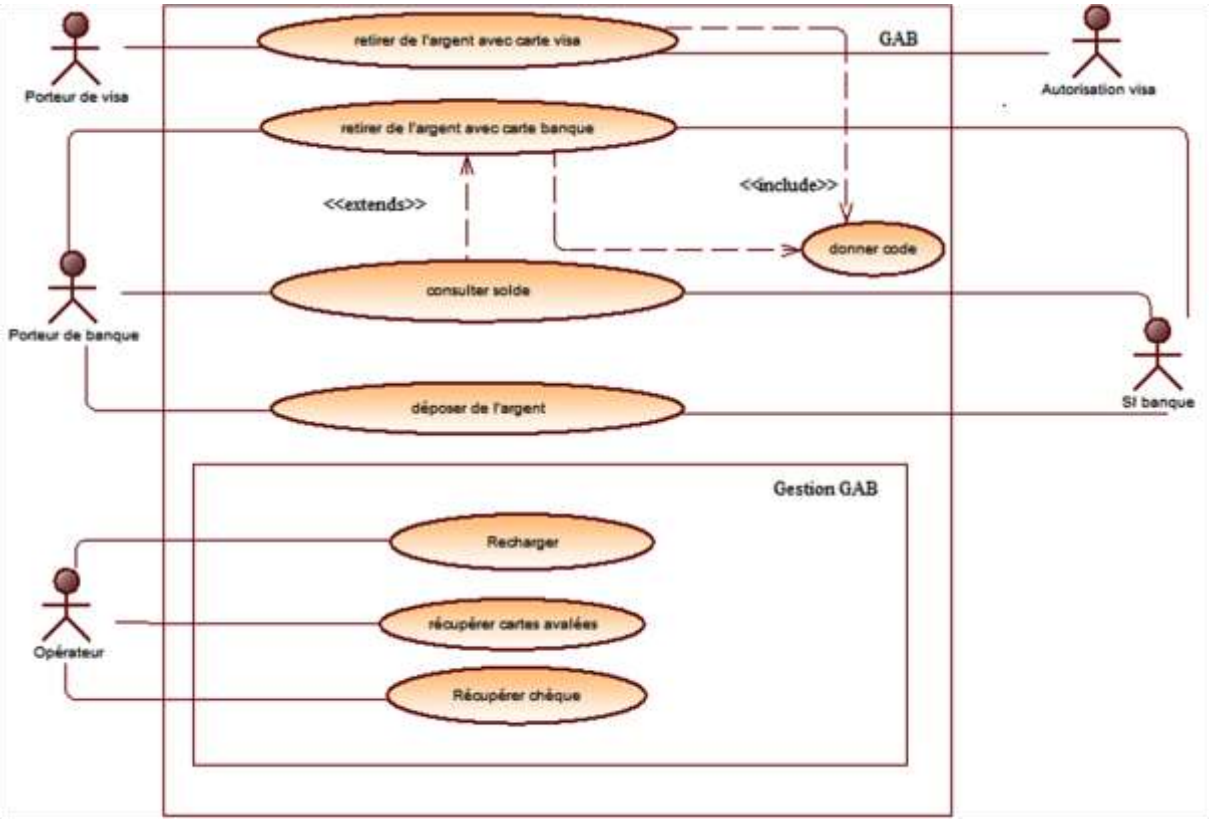


Diagramme de cas d'utilisations

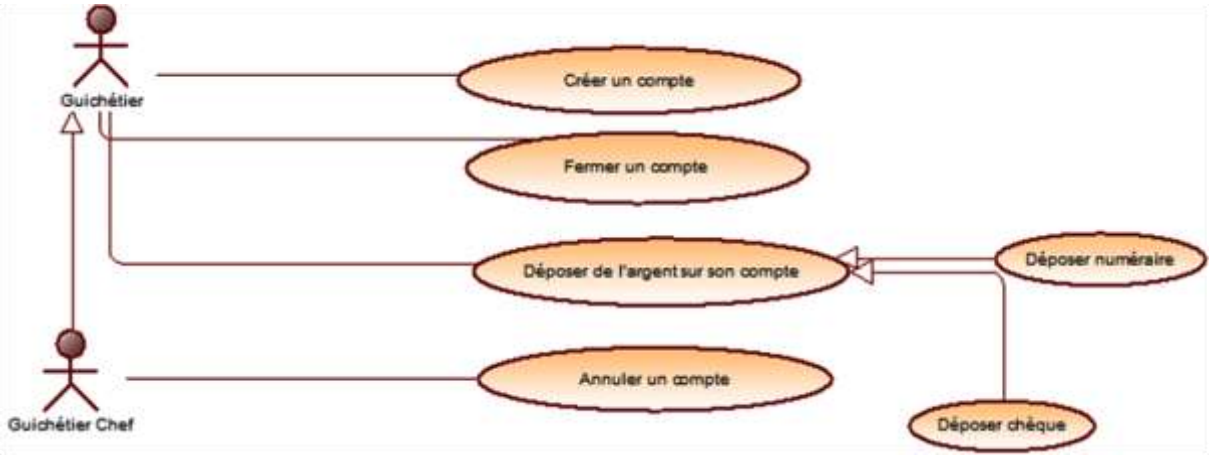


Exemple de diagramme de cas d'utilisations



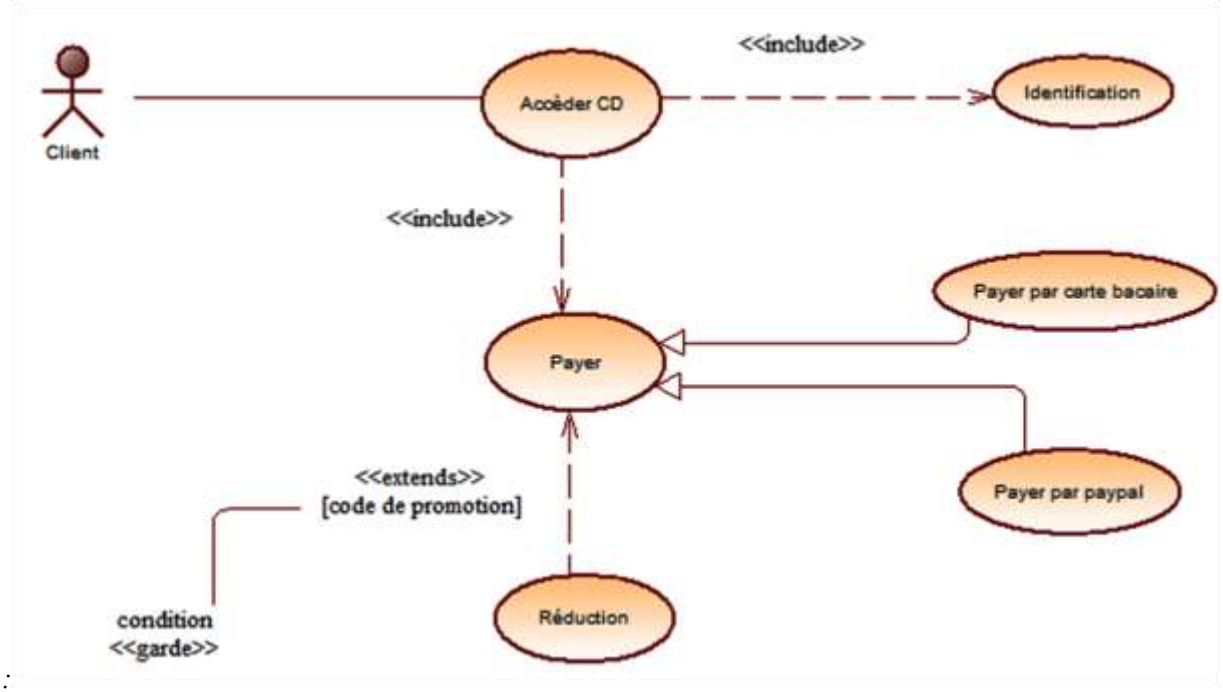
1.2.1.7. Généralisation et Spécialisation

Il est possible d'avoir de l'héritage entre acteurs et entre uses case.



Explication: Un "Guichétier Chef" est un "Guichétier" spécialisé qui peut faire tout ce que peut faire un "Guichétier" et, en plus, il peut annuler un compte. "Déposer chèque" et "Déposer numéraire" sont deux spécialisations de "Déposer de l'argent sur un compte"

Exemple :

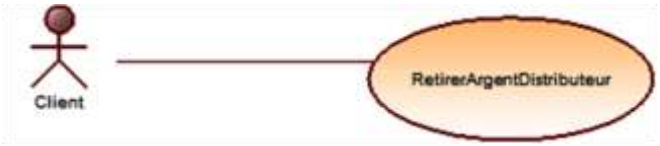


1.2.1.8. Spécification des cas d'utilisations

Chaque cas d'utilisation doit être spécifié par une description textuelle qui peut être structurée en plusieurs sections:

- conditions au démarrage (pré-conditions)
- conditions à la terminaison (post-condition)
- étapes du déroulement normal (”nominal”)
- variantes possibles et les ca d'erreurs
- contraintes non fonctionnelles (performance, sécurité, disponibilité, confidentialité,...

Exemple : le cas RetirerArgentDistributeur



Précondition: le distributeur contient des billets; il est en attente d'une opération: il n'est en panne ni en maintenance

Postcondition: si l'argent a pu être retiré, la somme de l'argent sur le compte est égale à la somme de l'argent qu'il avait moins le retrait. Sinon la somme d'argent sur le compte reste inchangée.

Déroulement normal:(1)le client introduit sa carte bancaire,(2)le système lit la carte et vérifie si la carte est valide, (3)le système demande au client de taper son code,(4)le client tape son code confidentiel,(5)le système vérifie que code correspond à la carte,(6)le client choisit une opération de retrait,(7)le système demande le montant à retirer etc.

Variantes:

(A) Carte invalide: au cours de l'étape (2), si la carte est jugée invalide, le système affiche un message d'erreur, rejette la carte et le cas d'utilisation se termine.

(B) ⋯

Contraintes non fonctionnelles:

(A) Performance: le système doit réagir un délai inférieur à 4 secondes quelque soit l'action de l'utilisateur.

(B) Sécurité···

Les cas d'utilisations peuvent être vue comme une classe de scénarios. Chaque scénario correspond à une utilisation particulière, par un acteur donné, dans des circonstances données. On peut décrire les principaux (préparation des tests finaux de l'application (ou du système))

SCENARIO 1

Pierre insère sa carte dans le distributeur x352

le système accepte la carte et lit le numéro de compte

le système demande le code confidentiel

Pierre tape "xsdsds"

le système détermine que ce n'est pas le bon code.

le système affiche un message et propose à l'utilisateur de recommencer

Pierre tape "adfste"

le système affiche que le code est correct

le système demande le montant du retrait

Pierre tape 45000

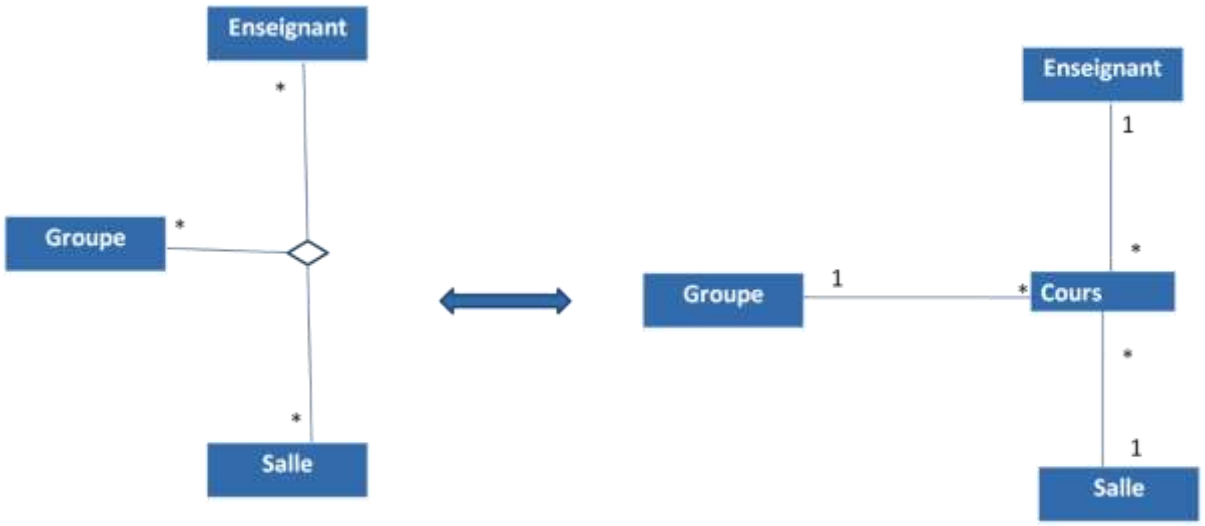
le système vérifie s'il y a assez d'argent sur le compte.

1.2.2. Diagramme de classe

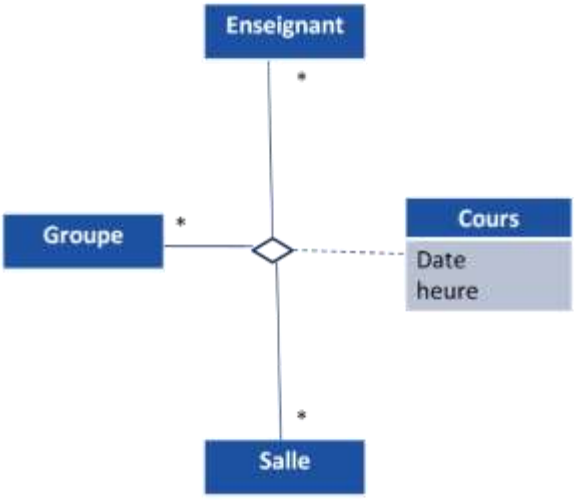
C'est une représentation statique du système modélisé. Il montre la structure d'un modèle.

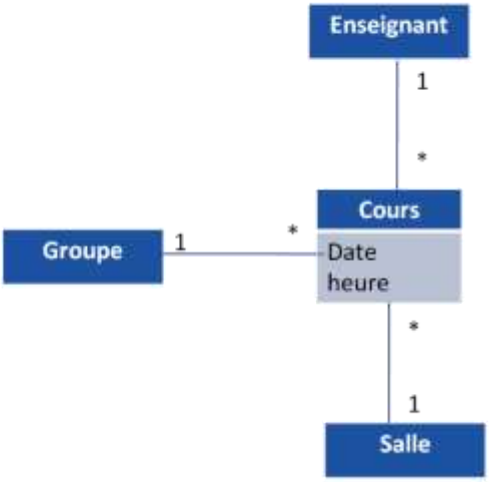
Le diagramme est composé de classes, d'associations et de cardinalites. Il est intégré dans chaque classe du diagramme la partie dédiée aux données et celle consacrée aux traitements. C'est le diagramme pivot de la modélisation d'un système.

Association n-aire(utilisation)



Association n-aire (utilisation)





Remarque :

Pour un modèle complexe, plusieurs diagrammes de classes complémentaires peuvent être construits. On peut par exemple avoir des diagrammes de classes par catégorie.