Génie Logiciel

Chapitre 3:

Diagramme UML de cas d'utilisation : une fonctionnelle

Niveau: 3ième année Licence informatique

Année: 2025/2026

Modélisation des besoins

- Avant de développer un système, il faut savoir précisément à « *QUOI* » il devra servir,
  - ✓ cad à quels besoins il devra répondre

- Modéliser les besoins permet de :
  - ✓ Faire l'inventaire des fonctionnalités attendues ;
  - ✓ Organiser les besoins entre eux, de manière à faire apparaître des relations (réutilisations possibles, ...)

  - ✓ Avec UML, on modélise les besoins au moyen de diagrammes de cas d'utilisation

Diagramme Cas d'utilisation-*Use Case*

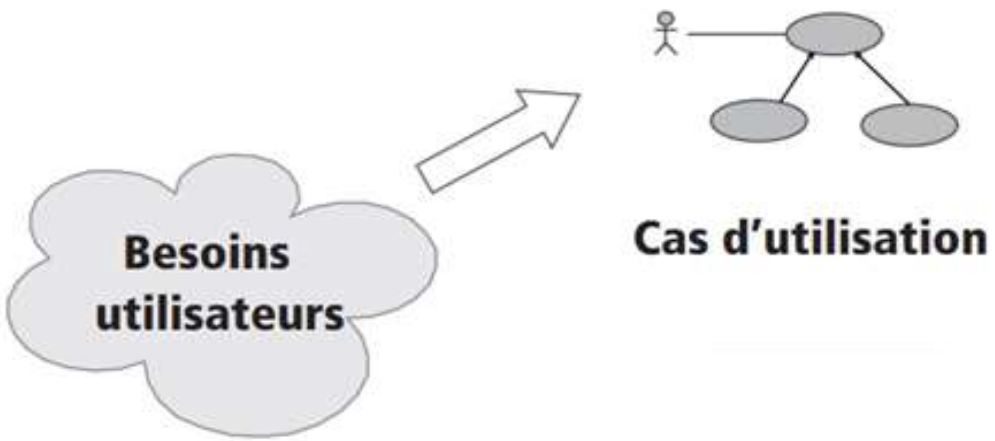
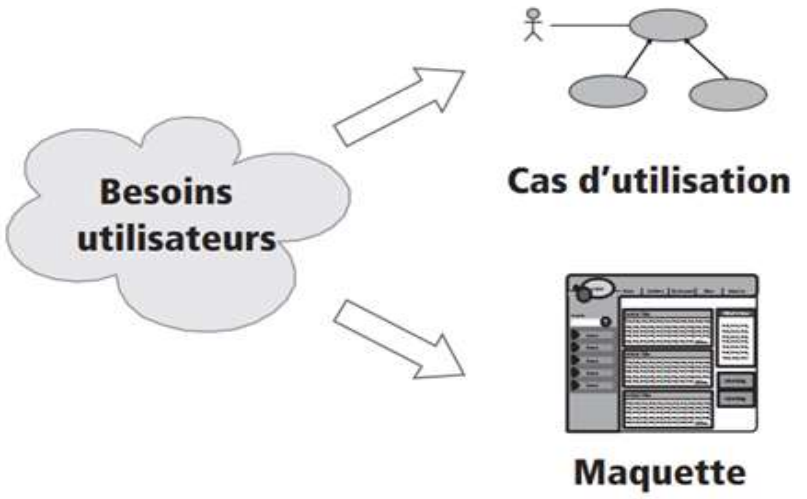


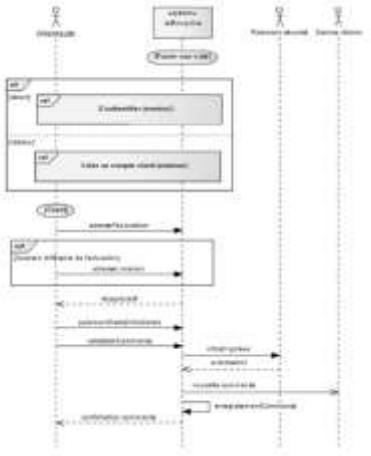
Diagramme Cas d'utilisation

- Utilisé dans l'activité de spécification des besoins
- utilisé pour
  - ✓ Recueillir, analyser et organiser les besoins
  - ✓ Recenser les fonctionnalités d'un système
    - Ce qu'il devra faire (et pas "comment")
    - Description du comportement sous forme d'actions/réactions
      - Représentation des fonctions du système du point de vue des utilisateurs

- Doit permettre de répondre à la question:
  - ✓ Qui fait quoi ?

Cas d'utilisation et Maquette

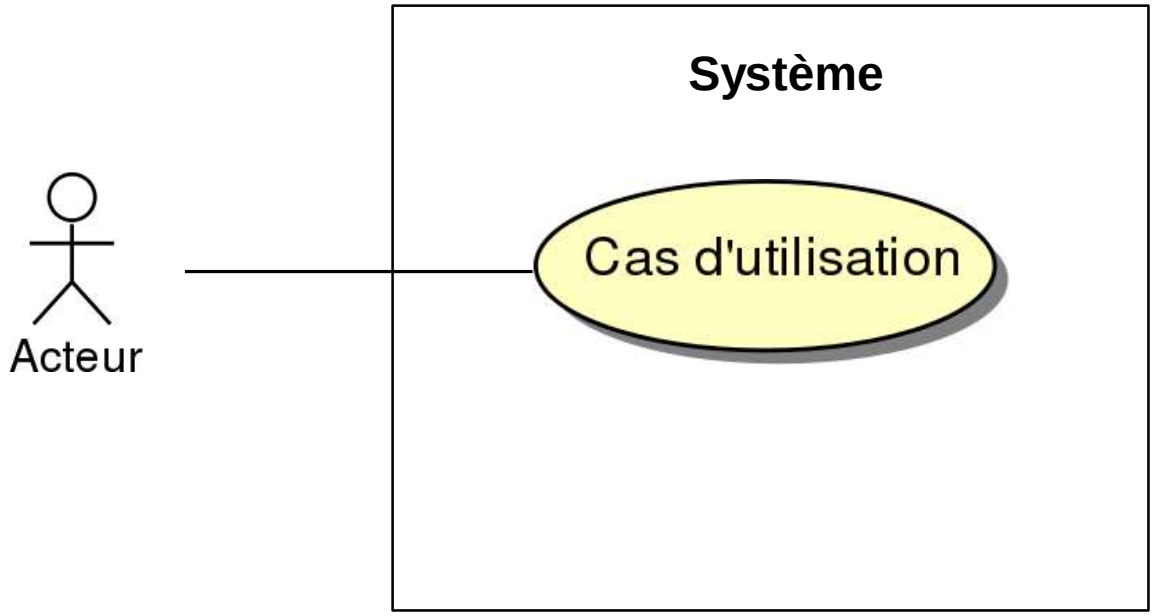




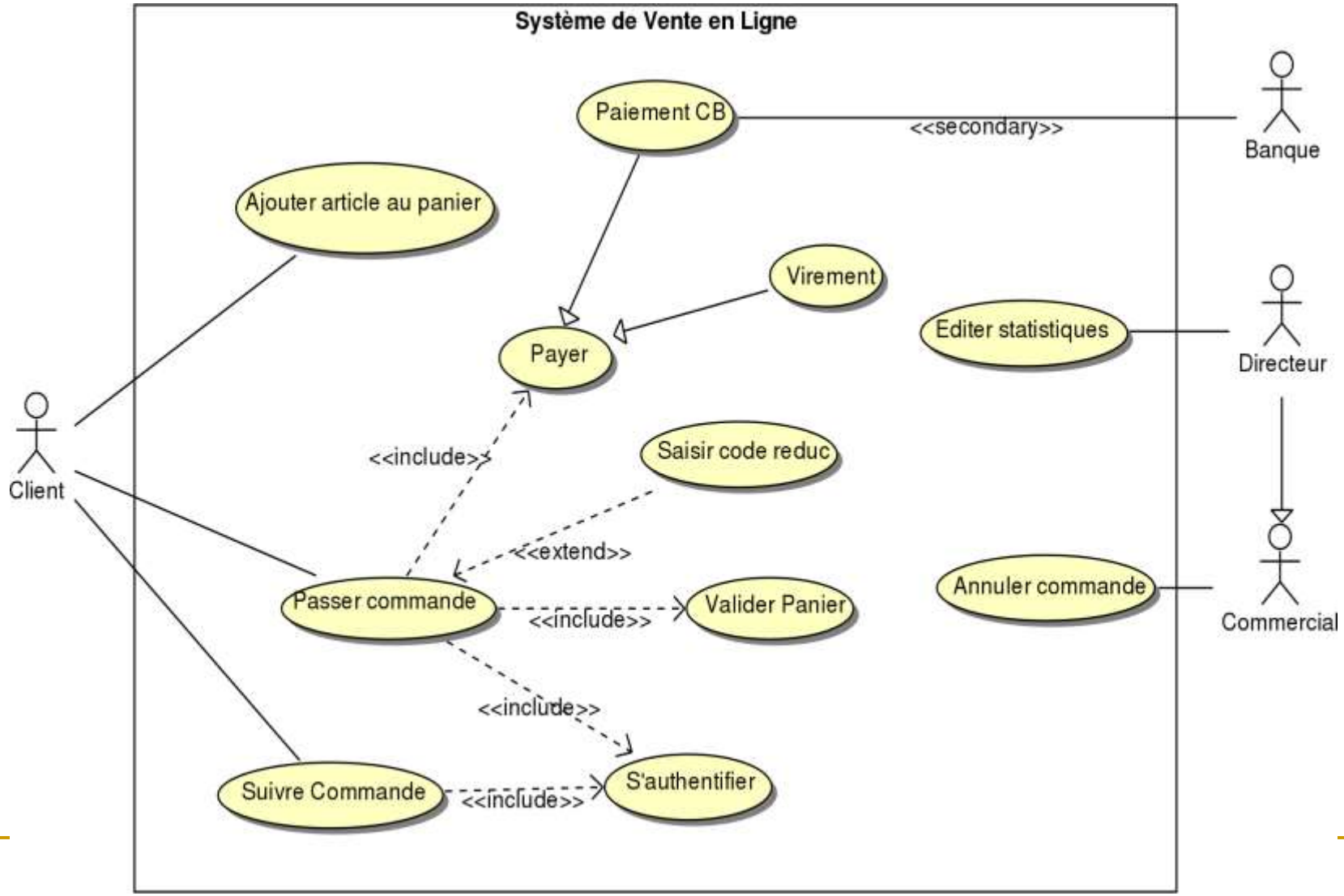
PPT Balsamiq

Les éléments d'un Diagramme Cas d'utilisation

- Le diagramme est constitué de
  - ✓ Système
  - ✓ Acteurs
  - ✓ cas d'utilisation



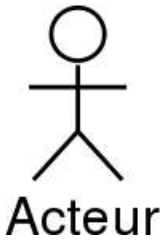
Exemple de diagramme de cas d'utilisation



Acteur

- Un rôle joué par une entité externe qui interagit directement avec le système
  - ✓ Utilisateur humain
  - ✓ Dispositif matériel
  - ✓ Un autre système

stick man



Un acteur peut être représenté par un rectangle doté du **stéréotype*** "actor"

<<actor>>

Gestion des stockes

- Un acteur peut consulter et/ou modifier directement l'état du système, en émettant et/ou en recevant des messages

Acteur

- Les principaux acteurs sont les utilisateurs du système.
- Acteur correspond à un rôle, pas à une personne physique
  - ✓ Une même personne physique peut être représentée par plusieurs acteurs si elle a plusieurs rôles
  - ✓ Si plusieurs personnes jouent le même rôle vis-à-vis du système, elles seront représentées par un seul acteur
  - ✓ un acteur n’est pas forcément "humain"

Acteur

- En plus des utilisateurs, les acteurs peuvent être :
  - ✓ Des périphériques manipulés par le système (imprimantes...) ;
  - ✓ Des logiciels déjà disponibles à intégrer dans le projet ;
  - ✓ Des systèmes informatiques externes au système mais qui interagissent avec lui, (SGBD, horloge, etc.)

- Les acteurs se trouvent obligatoirement à l'extérieur du système

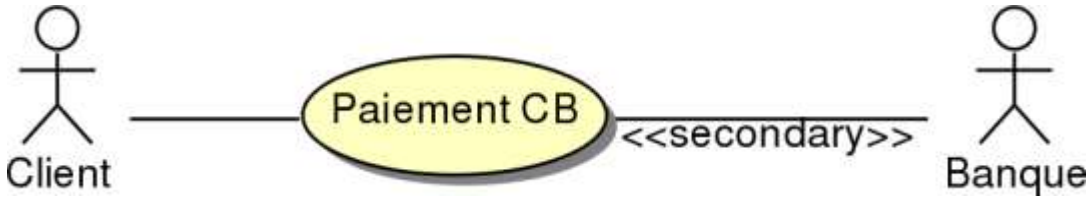
- Pour faciliter l'identification des acteurs, on se fonde sur les frontières du système.

Acteurs principaux et secondaires

- Un acteur est qualifié de principal lorsqu'il a l'initiative des interactions nécessaires à la réalisation du cas d'utilisation
  - ✓ C'est lui qui déclenche l' exécution du cas d'utilisation.

- Les acteurs secondaires sont souvent sollicités pour des informations complémentaires ; ils peuvent uniquement consulter ou informer le système lors de l'exécution du cas d'utilisation.

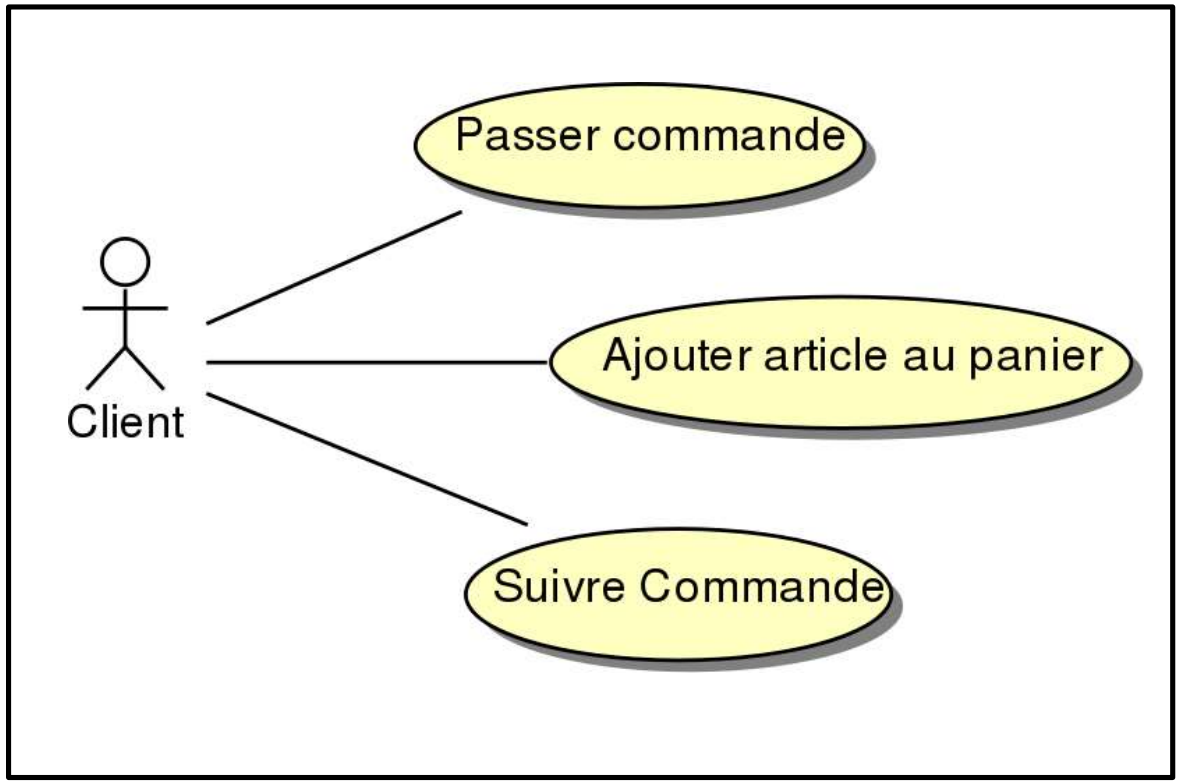
- Pour une meilleure lisibilité du diagramme, disposez les acteurs principaux à gauche et les acteurs secondaires à droite.



Cas d'utilisation (*use case*)

- Est un service rendu à l'utilisateur, il implique des séries d'actions plus élémentaires.
- Permet de décrire ce que le futur système devra faire, sans spécifier comment il le fera
- Est un l'expression d'un service réalisé de bout en bout, avec un déclenchement, un déroulement et une fin, pour l'acteur qui l'initie.
- Représente un ensemble de séquences d'actions qui sont réalisées par le système et qui produisent un résultat observable intéressant pour un acteur particulier

Acteurs et cas d'utilisation



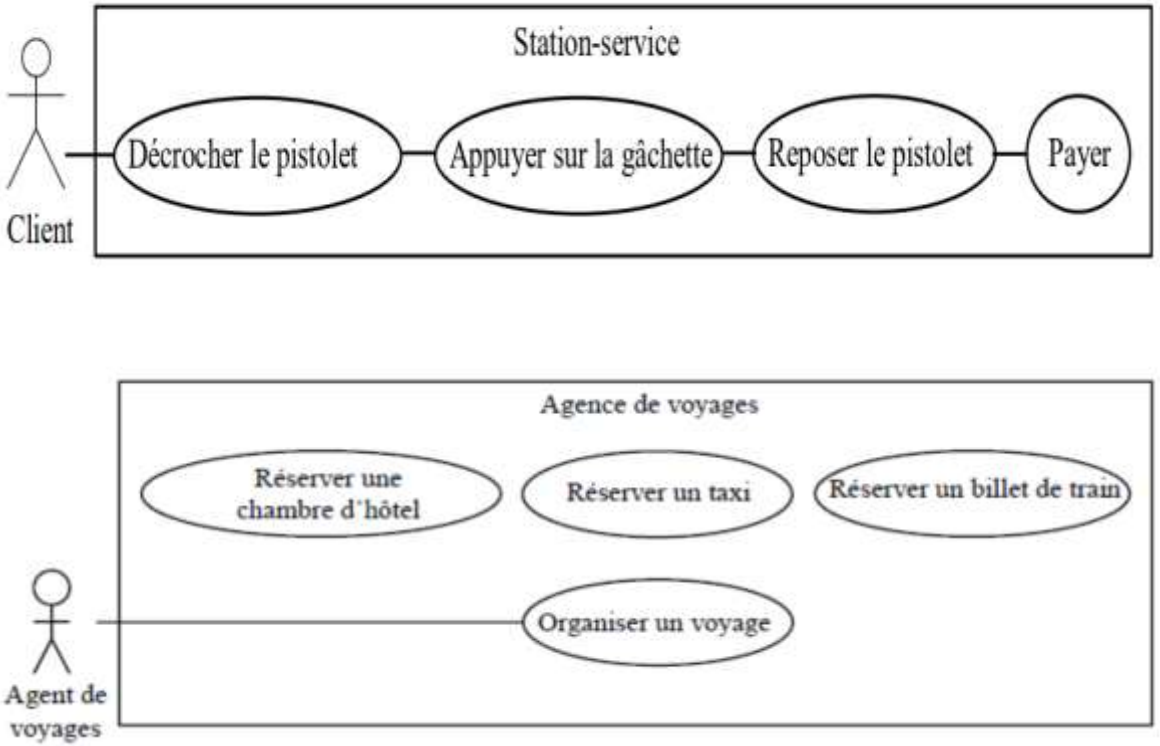
Recenser les cas d'utilisation

- Il n'y a pas une manière mécanique et totalement objective de repérer les cas d'utilisation.
  - ✓ Il faut se placer du point de vue de chaque acteur et déterminer comment il se sert du système, dans quels cas il l'utilise, et à quelles fonctionnalités il doit avoir accès.
  - ✓ Il faut éviter les redondances et limiter le nombre de cas en se situant au bon niveau d'abstraction (par exemple, ne pas réduire un cas à une seule action).
  - ✓ Il ne faut pas faire apparaître les détails des cas d'utilisation, mais il faut rester au niveau des grandes fonctions du système.

Trouver le bon niveau de détail pour les cas d'utilisation est un problème difficile qui nécessite de l'expérience.

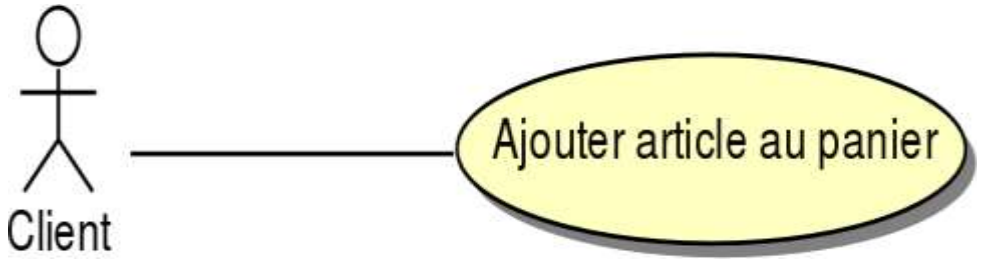
Astuce pratique

- Un bon moyen de savoir si un ensemble d'actions forme un use case :
  - ✓ Si l'utilisateur entre dans le système uniquement pour exécuter ces actions et en sort ensuite, alors il s'agit bien d'un cas d'utilisation.



Relations entre cas d'utilisation et acteurs

- Les acteurs impliqués dans un cas d'utilisation lui sont liés par une association*.
- Un acteur peut utiliser plusieurs fois le même cas d'utilisation.

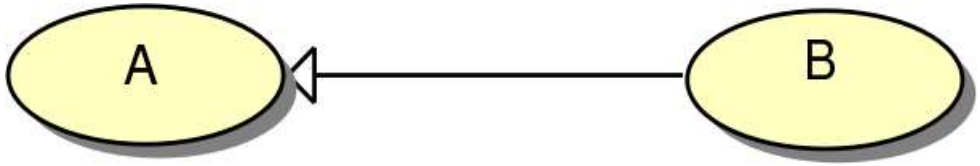


Relations entre cas d'utilisation

- Inclusion : le cas A inclut le cas B (B est une partie obligatoire de A).

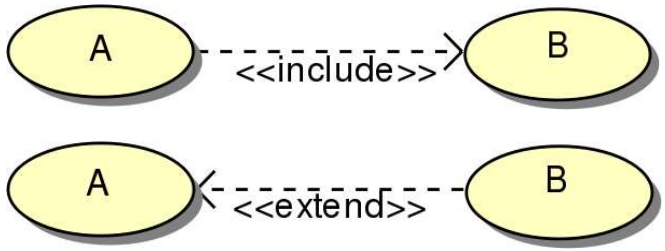
- Extension : le cas B étend le cas A (B est une partie optionnelle de A).

- Généralisation : le cas A est une généralisation du cas B (B est une sorte de A).



Dépendances d'inclusion et d'extension

- Les inclusions et les extensions sont représentées par des dépendances.
  - ✓ Lorsqu'un cas B inclut un cas A, B dépend de A.
  - ✓ Lorsqu'un cas B étend un cas A, B dépend aussi de A.
  - ✓ On note toujours la dépendance par une flèche pointillée B ---> A qui se lit « B dépend de A ».
- Lorsqu'un élément A dépend d'un élément B, toute modification de B sera susceptible d'avoir un impact sur A.
- Les **« include »** et les **« extend »** sont des stéréotypes* (entre guillemets) des relations de dépendance.

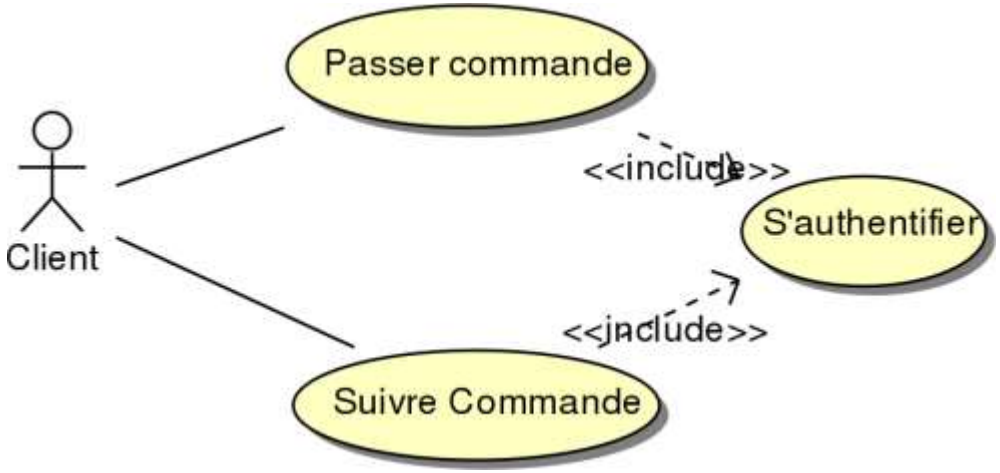


Attention!!!

- Le sens des flèches indique la dépendance, pas le sens de la relation d'inclusion.

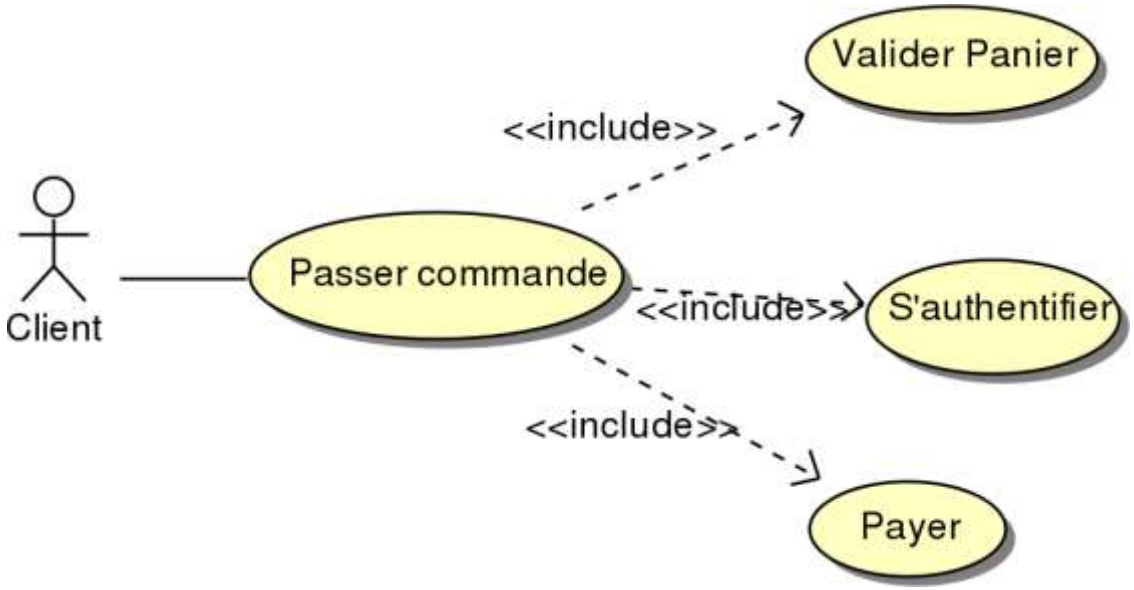
Réutilisabilité avec les inclusions et les extensions

- Les relations entre cas permettent la réutilisabilité du cas (s'authentifier) : il sera inutile de développer plusieurs fois un module d'authentification. (factoriser une partie d'un cas d'utilisation commune à d'autres cas d'utilisation)



Décomposition grâce aux inclusions et aux extensions

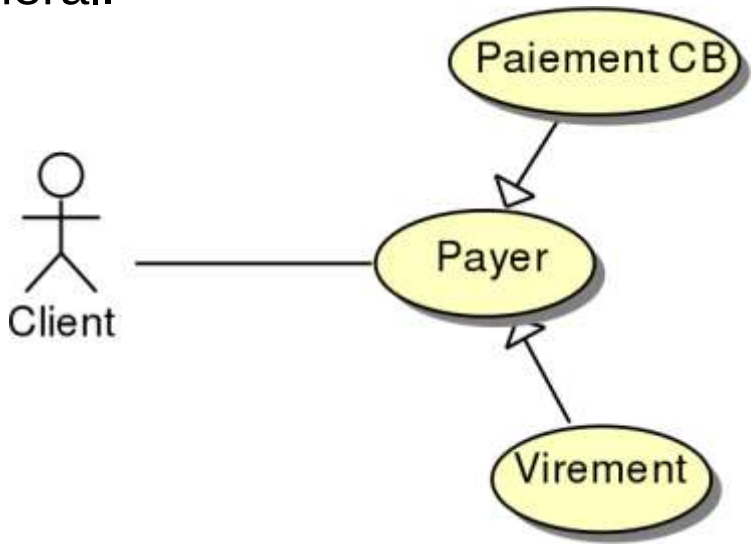
- Quand un cas est trop complexe (faisant intervenir un trop grand nombre d'actions), on peut procéder à sa décomposition en cas plus simples.



Généralisation

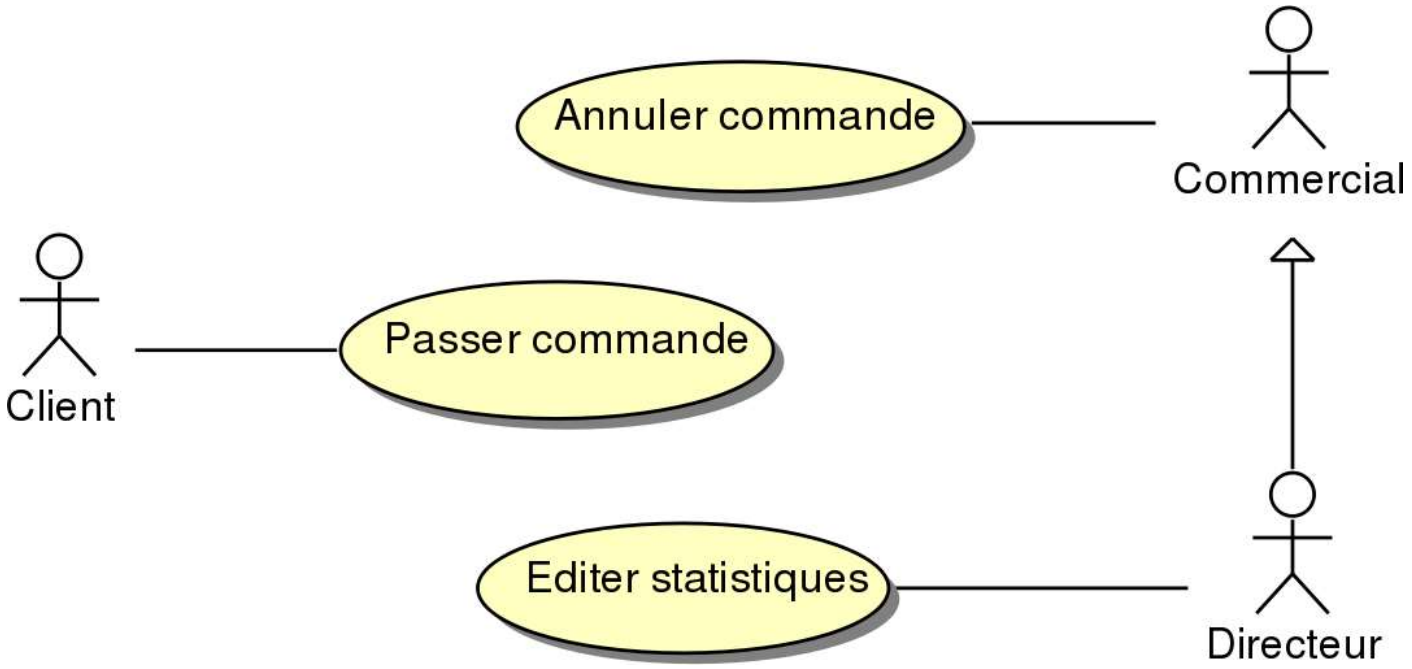
- Cette relation de généralisation/spécialisation est présente dans la plupart des diagrammes UML et se traduit par le concept d'héritage dans les langages orientés objet.

- Un virement est un cas particulier de paiement.
- Un virement est une sorte de paiement.
- La flèche pointe vers l'élément général.



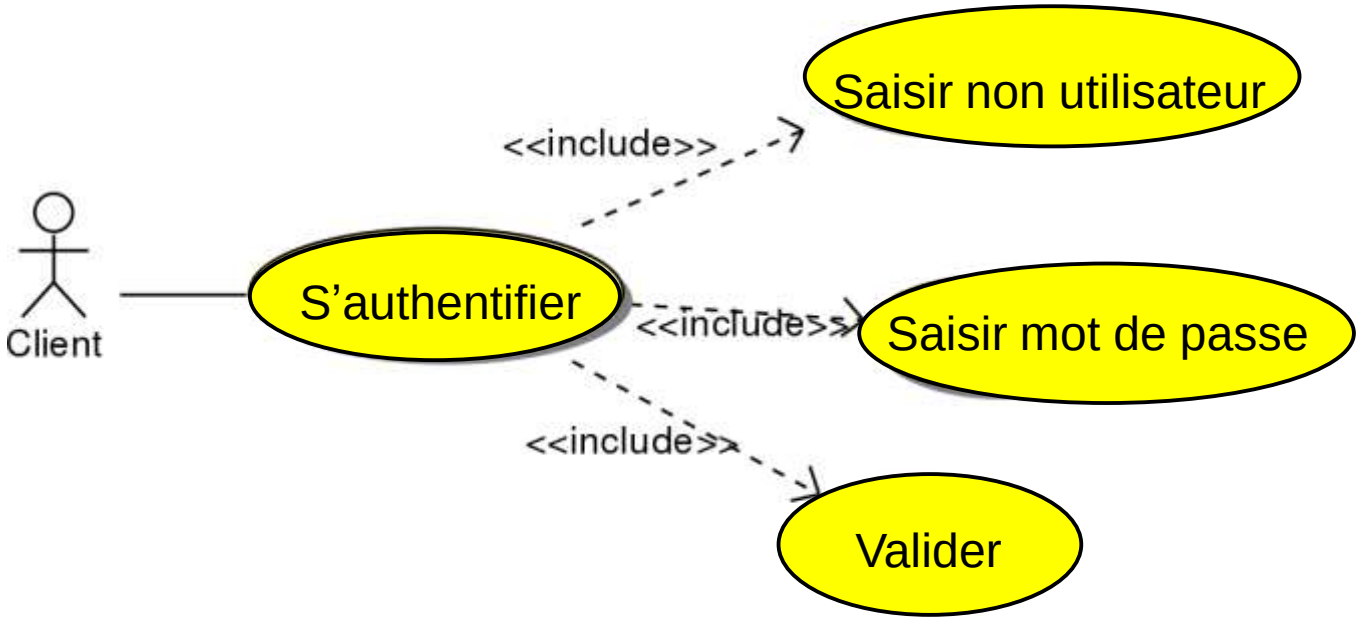
Relations entre acteurs

- Une seule relation possible : la généralisation.

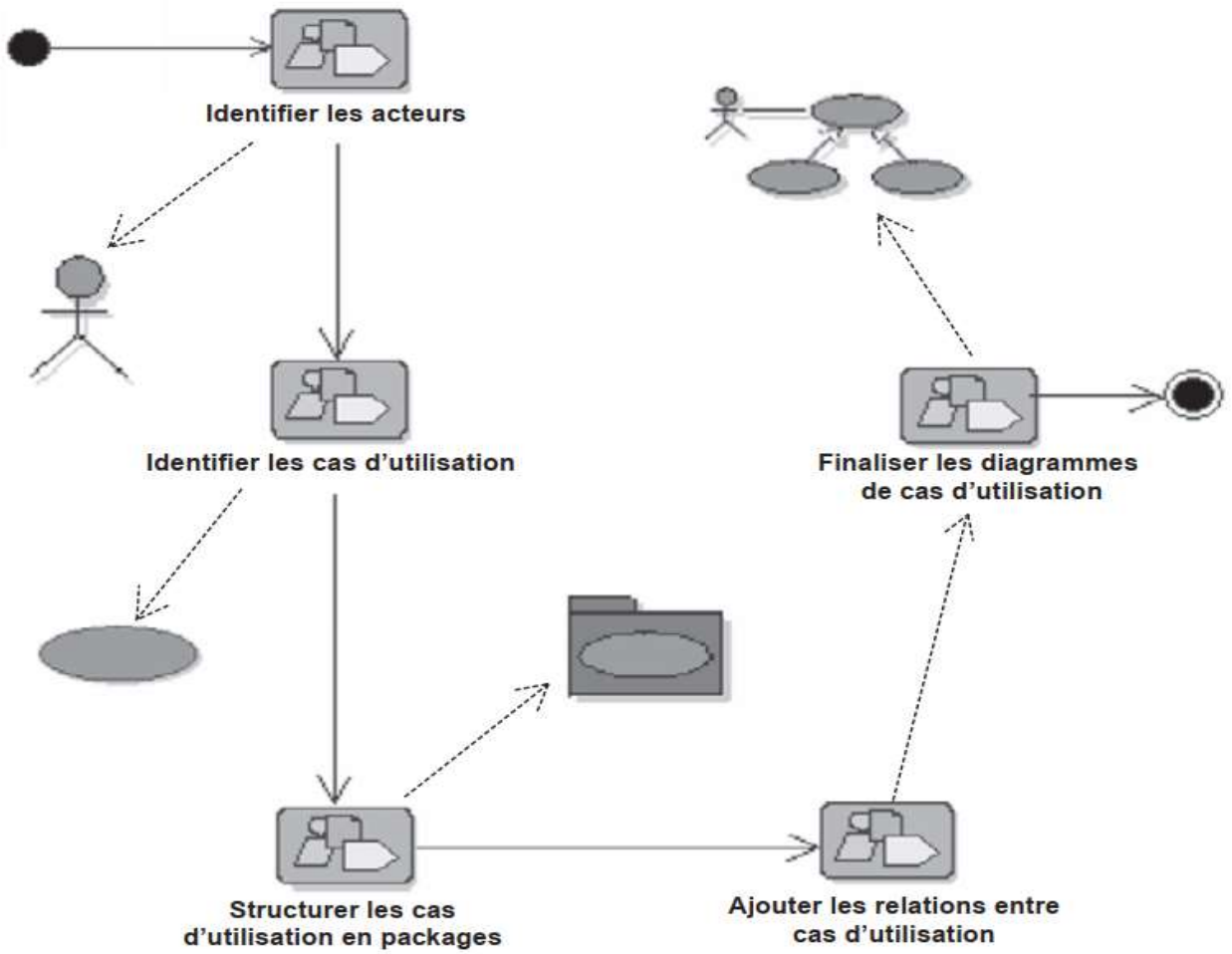


Exercice

- Quel est le défaut de ce diagramme

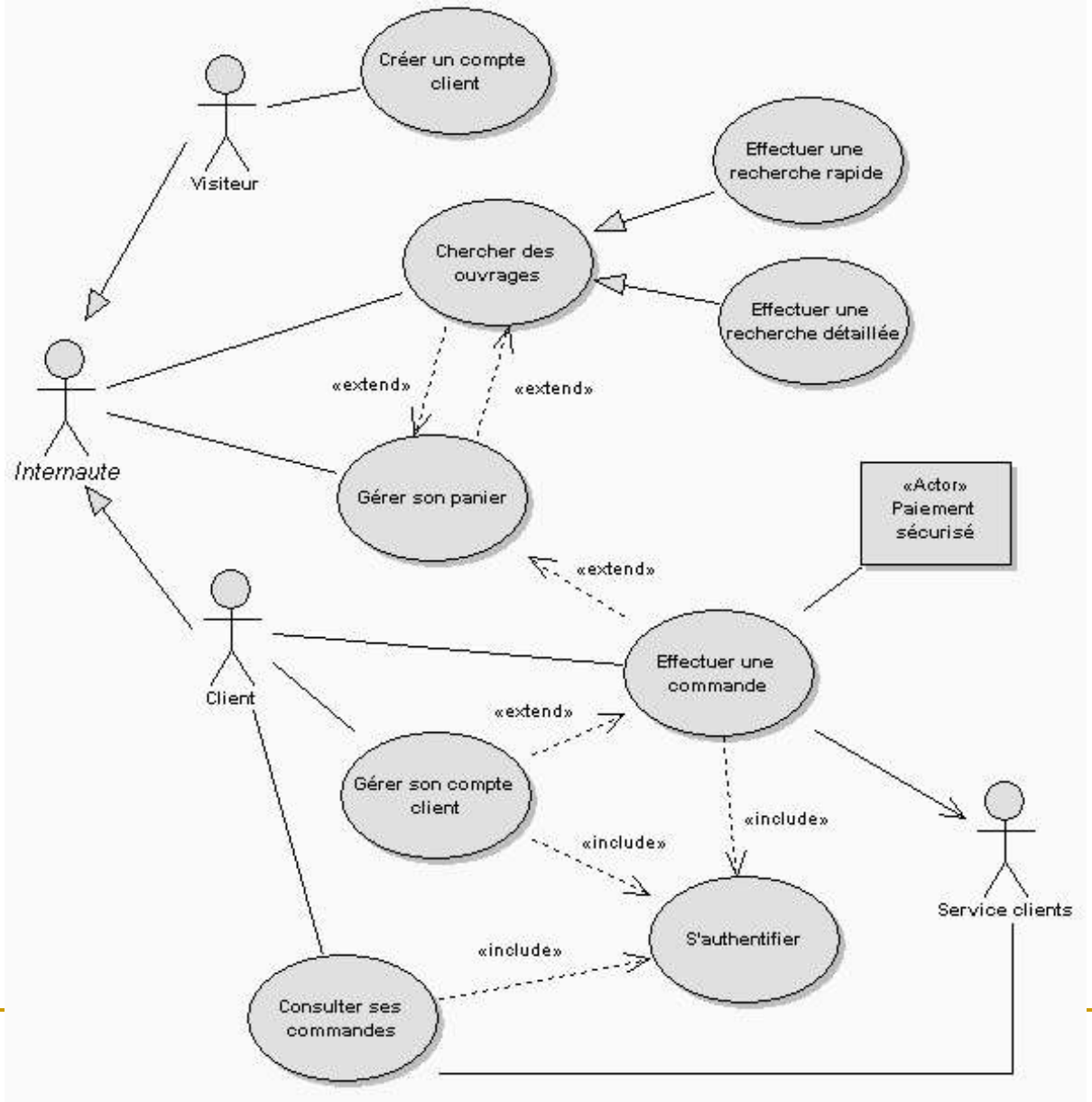


Une démarche de construction du modèle des cas d'utilisation



Exemple: Librairie en ligne

Exemple: Librairie en ligne



Description des cas d'utilisation

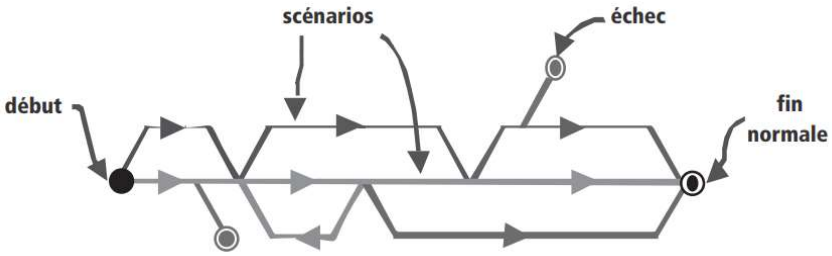
- Le diagramme de cas d'utilisation décrit les grandes fonctions d'un système du point de vue des acteurs, mais n'expose pas de façon détaillée le dialogue entre les acteurs et les cas d'utilisation.
- *Un simple nom est tout à fait insuffisant pour décrire un cas d'utilisation.*

Chaque cas d'utilisation doit être documenté pour qu'il n'y ait aucune ambiguïté concernant son déroulement et ce qu'il recouvre précisément.

Description textuelle

- Pour détailler la dynamique d'un diagramme de cas d'utilisation, la procédure la plus évidente consiste à recenser de façon textuelle toutes les interactions entre les acteurs et le système
- Fiche de description textuelle
  - ✓ Pas normalisée par UML,
  - ✓ mais fortement recommandé

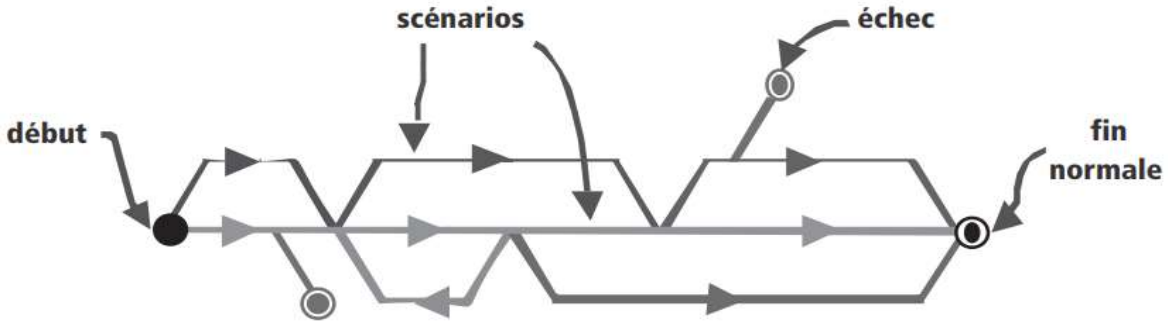
- Un scénario



Scénario

- Une suite spécifique d'interactions entre les acteurs et le système à l'étude
- Décrit une exécution particulière d'un cas d'utilisation du début à la fin
- On peut dire que c'est une « instance » du cas d'utilisation

- Le cas d'utilisation doit avoir un début et une fin clairement identifiés



Scénario

- Chaque scénario est composé d'étapes qui peuvent être de trois sortes
  - ✓ un message d'un acteur vers le système,
  - ✓ une validation ou un changement d'état du système,
  - ✓ un message du système vers un acteur.

- Les étapes sont numérotées séquentiellement afin de pouvoir facilement indiquer par la suite les alternatives possibles

EXEMPLE **DAB : scénario nominal**

Cas d'utilisation : Retirer de l'argent au distributeur (DAB) avec une carte bancaire.

1. Le porteur de carte introduit sa carte dans le DAB.
2. Le DAB vérifie que la carte introduite est bien une carte bancaire.
3. Le DAB demande au porteur de carte de fournir son code d'identification.
4. Le porteur de carte entre son code d'identification.
5. Le DAB valide le code d'identification (par rapport à celui qui est codé sur la puce de la carte).
6. Le DAB demande une autorisation au système d'autorisation externe.
7. Le système d'autorisation externe donne son accord et indique le solde hebdomadaire.
8. Le DAB demande au porteur de carte de saisir le montant désiré du retrait.
9. ...

Scénario nominal/Alternatives

- Un cas d'utilisation contient en général :
  - ✓ Un scénario « nominal », celui qui satisfait les objectifs des acteurs par le chemin le plus direct de succès,
  - ✓ Plusieurs alternatives, qui comprennent tous les autres scénarios, de succès (fin normale) ou d'échec (erreur)

EXEMPLE **DAB : alternatives**

2a. La carte introduite n'est pas reconnue par le DAB.

1. Le DAB éjecte la carte et le cas d'utilisation se termine en échec.

5a. Le DAB détecte que le code saisi est erroné, pour la première ou deuxième fois.

1. Le DAB indique au porteur de carte que le code est erroné.
2. Le DAB enregistre l'échec sur la carte et le cas d'utilisation reprend à l'étape 5 du scénario nominal.

5b. Le DAB détecte que le code saisi est erroné, pour la troisième fois.

1. Le DAB indique au porteur de carte que le code est erroné pour la troisième fois.
2. Le DAB confisque la carte.
3. Le DAB informe le système d'autorisation externe et le cas d'utilisation se termine en échec.

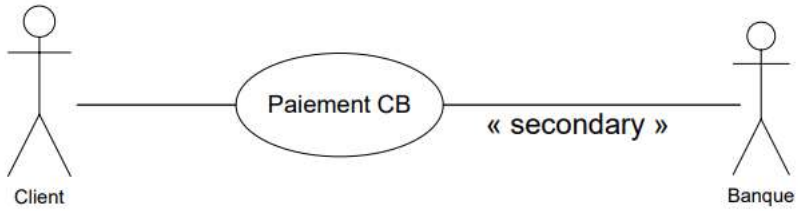
EXEMPLE **DAB : scénario nominal**

Cas d'utilisation : Retirer de l'argent au distributeur (DAB) avec une carte bancaire.

1. Le porteur de carte introduit sa carte dans le DAB.
2. Le DAB vérifie que la carte introduite est bien une carte bancaire.
3. Le DAB demande au porteur de carte de fournir son code d'identification.
4. Le porteur de carte entre son code d'identification.
5. Le DAB valide le code d'identification (par rapport à celui qui est codé sur la puce de la carte).
6. Le DAB demande une autorisation au système d'autorisation externe.
7. Le système d'autorisation externe donne son accord et indique le solde hebdomadaire.
8. Le DAB demande au porteur de carte de saisir le montant désiré du retrait.
9. ...

Fiche de description textuelle

- Rappel: Pas normalisée par UML
- Exemple de description textuelle d'un cas d'utilisation



- **Identification** :
  - ✓ Nom du cas : Payer CB
  - ✓ Objectif : Détailler les étapes permettant à client de payer par carte bancaire
  - ✓ Acteurs : Client, Banque (secondaire)
  - ✓ Date : 21/11/2010
  - ✓ Responsables : Toto Version : 1.0

Fiche de description textuelle (2)

- **Séquencements** :
  - ✓ Le cas d'utilisation commence lorsqu'un client demande le paiement par carte bancaire

  - ✓ **Pré-conditions** :
    - Le client a validé sa commande

  - ✓ **Enchaînement nominal** :
    1. Le client saisit les informations de sa carte bancaire
    2. Le système vérifie que le numéro de CB est correct
    3. Le système vérifie la carte auprès du système bancaire
    4. Le système demande au système bancaire de débiter le client
    5. Le système notifie le client du bon déroulement de la transaction

  - ✓ **Enchaînements alternatifs** :
    1. En (2) : si le numéro est incorrect, le client est averti de l'erreur, et invité à recommencer
    2. En (3) : si les informations sont erronées, elles sont re-demandées au client

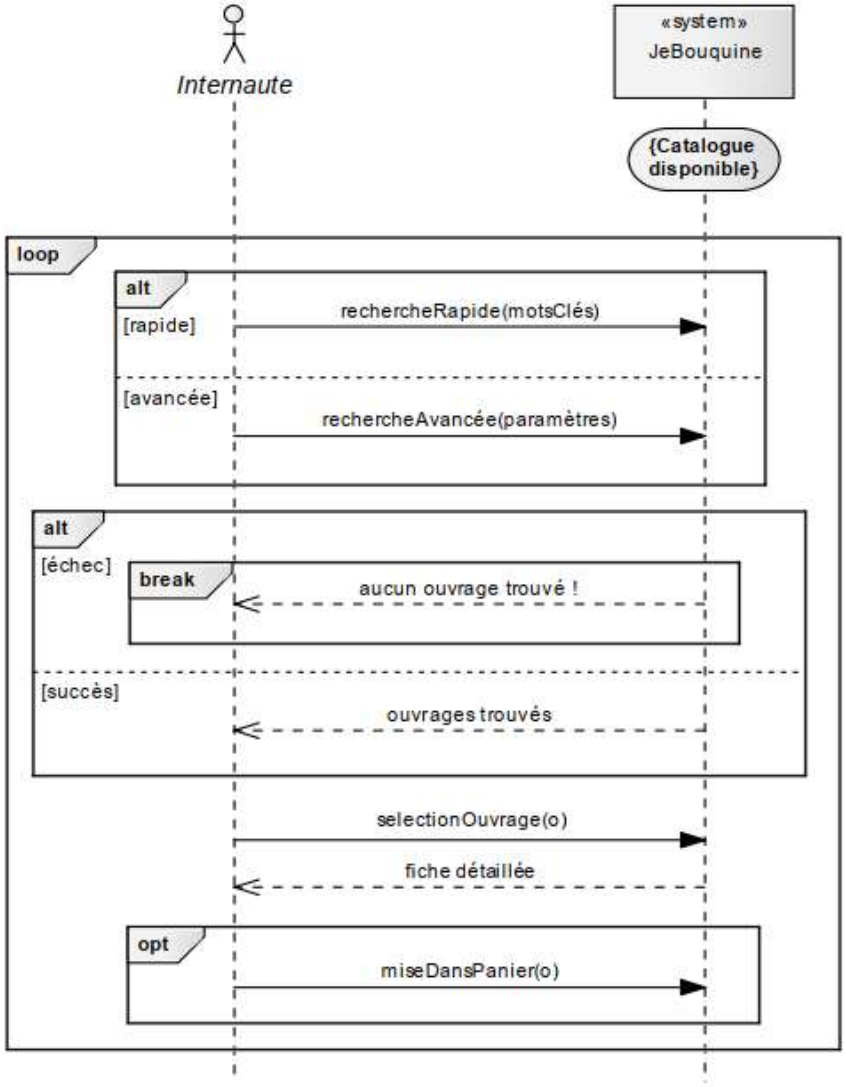
Fiche de description textuelle (2)

- ✓ **Post-conditions** :
  - La commande est validée
  - Le compte de l'entreprise est crédité

- **Rubriques optionnelles** :
  - ✓ **Contraintes non fonctionnelles** :
    - Fiabilité : les accès doivent être sécurisés
    - Confidentialité : les informations concernant le client ne doivent pas être divulgués

  - ✓ Contraintes liées à l'interface homme-machine :
    - Toujours demander la validation des opérations bancaires

Et les diagrammes de séquences !



DS Chercher des ouvrages

Références

- ✓ Pierre Gérard *Cours : Introduction à UML 2,* Université de Paris 13 - IUT Villetaneuse
- ✓ Pascal Roques : *UML2 par la pratique,* Edition Eyrolles
- ✓ Pascal Roques : *UML2 par la pratique,* Edition Eyrolles
- ✓ Pascal Roques : *UML2 en Action,* Edition Eyrolles
- ✓ Ilhem Boussaïd: UML - Diagramme de cas d'utilisation (Use case diagram), Université des Sciences et de la Technologie Houari Boumediene