Génie Logiciel

UML

Introduction

- Un modèle est une représentation simplifiée d'une réalité.
- Le modèle s'exprime sous une forme simple et pratique pour le travail
- 1 ou plusieurs modèles pour un problème
- UML n'est qu'un langage et il ne sert qu'à formaliser les besoins, c'est-à-dire à les représenter sous une forme graphique suffisamment simple pour être compréhensible par toutes les personnes impliquées dans le projet.

Matériel et logiciel

Systèmes informatiques :

- 80 % de logiciel ;
- 20 % de matériel.

- Depuis quelques années, la fabrication du matériel est assurée par quelques fabricants seulement.
- Le marché est standardisé.

- La crise du logiciel

Étude sur 8 380 projets (Standish Group, 1995) :

Succès : 16 %;

Problématique : 53 % (budget ou délais non respectés, défaut de fonctionnalités) ;

Échec : 31 % (abandonné).

Introduction (2)

- Qualités attendues d'un logiciel:
  - Utilité;
  - Utilisabilité;
  - Fiabilité;
  - Interopérabilité, couplabilité;
  - Performance;
  - Portabilité;
  - Réutilisabilté;
  - Facilite de maintenance
  - Maintenance corrective
  - Maintenance adaptative
  - Maintenance perfective, d'extension
  - Facilite de maintenance

Introduction (3)

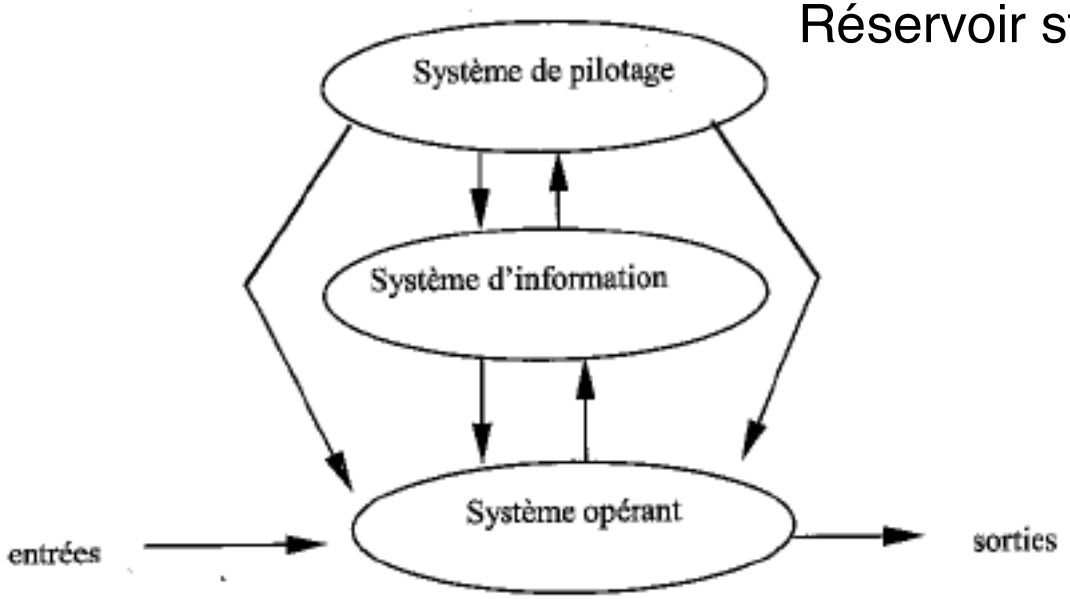
- Facilite de maintenance:

	Répartition effort dév.	Origine des erreurs	Coût de la maintenance
Définition des besoins	6%	56%	82%
Conception	5%	27%	13%
Codage	7%	7%	1%
Intégration Tests	15%	10%	4%
Maintenance	67%		

(Zeltovitz, De Marco)

Système d'information : approche systémique classique

Réservoir structuré d'informations



Toute organisation peut être appréhendée comme l'ensemble de trois sous-systèmes en interaction.

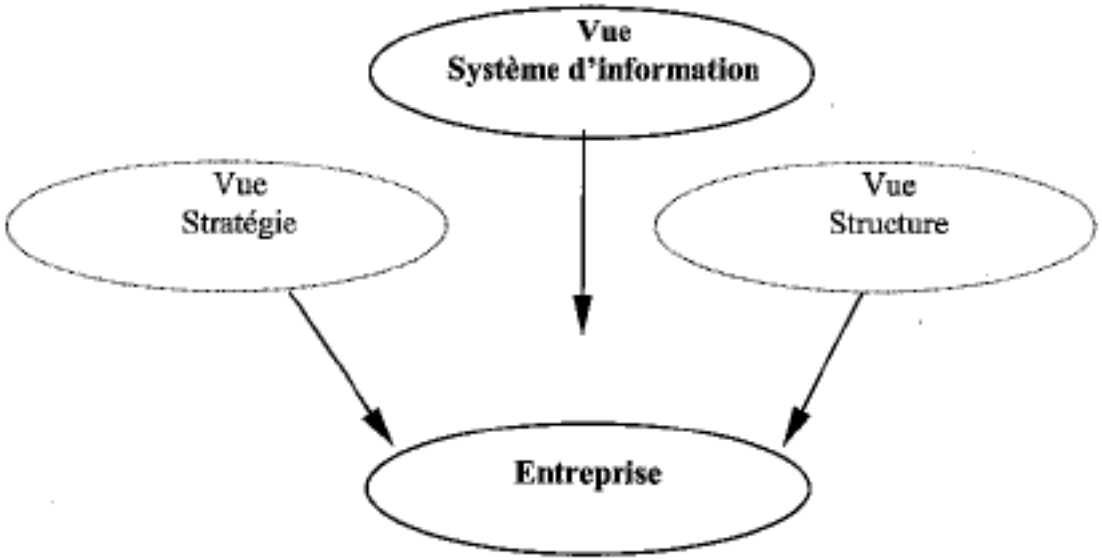
Système opérant - l'activité productive

Système de pilotage - l'activité de l'équipe dirigeante

Système d'information - une mémoire entre les deux systèmes..

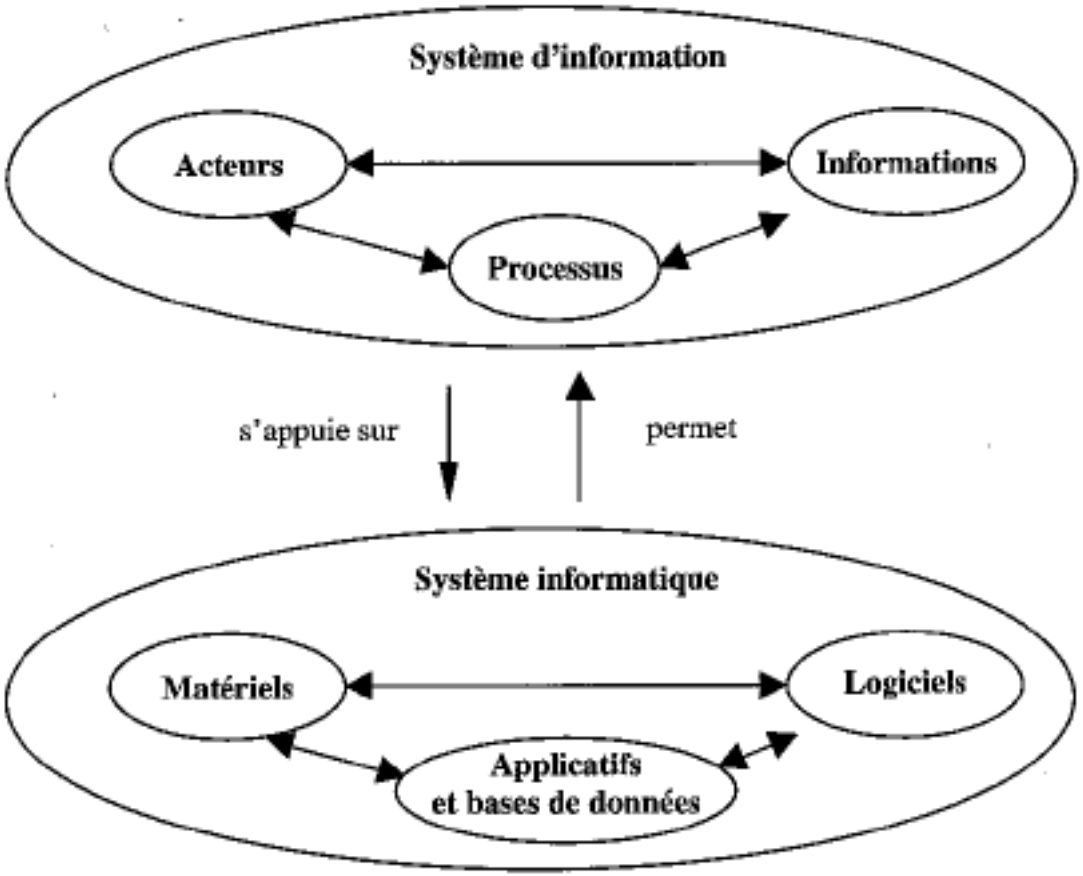
Système d'information : approche par le système de travail

Les vues de l'entreprise



Une façon de regarder une organisation et son fonctionnement à travers ses informations.
La structure : l'entreprise vue à travers son organigramme, la répartition des responsabilités et du pouvoir;
La stratégie : l'entreprise vue sur l'angle de ses décideurs, ses relations avec l'environnement, ses avantages clés.

Systèmes d'informations et système informatique



Principes du Génie Logiciel

- Généralisation
- Structuration
- Abstraction
- Modularité
- Documentation
- Vérification

- Le maître d'ouvrage intervient constamment au cours du projet, notamment pour :
  - définir et exprimer les besoins ;
  - valider les solutions proposées par le maître d'oeuvre ;
  - valider le produit livré.

Documents produits dans le cycle de vie

Document	**Phase de production**
Manuel utilisateur final	Implémentation
Conception architecturale	Conception
Plan d'assurance qualité	Planification
Code source	Implémentation
Cahier des charges	Faisabilité
Plan de test	Spécification
Manuel utilisateur préliminaire	Spécification
Conception détaillée	Conception
Estimation des coûts	Planification
Calendrier du projet	Planification
Rapport des tests	Tests
Documentation	Implémentation

Modèle UML : présentation

- **Présentation stratégique :**

  la motivation de l'élaboration (pourquoi l'entreprise a voulu se doter de l'outil considéré), les buts qu'elle cherche à atteindre, le calendrier de réalisation prévu, les coûts,...
- **Présentation des processus de travail par lesquels la stratégie entend se réaliser :**

  présenter comment l'application va fonctionner en pratique;

  illustration par une esquisse des écrans qui seront affichés devant les utilisateurs de terrain.
- **Explication des choix qui ont guidé la modélisation formelle :**

  il s'agit de synthétiser les discussions qui ont présidé à ces choix.
- **Modèle formel :**

  Il est préférable de le présenter sur l'Intranet de l'entreprise. En effet, les diagrammes peuvent être alors équipés de liens hypertextes permettant l'ouverture de diagrammes plus détaillés ou de commentaires.

Principe

- Le développement d'un nouveau système, ou l'amélioration d'un système existant, doit répondre à un ou à plusieurs besoins.

- Celui qui commande le logiciel est le maître d'ouvrage. Celui qui réalise le logiciel est le maître d'oeuvre.
- Le maître d'ouvrage intervient constamment au cours du projet, notamment pour :
  - définir et exprimer les besoins ;
  - valider les solutions proposées par le maître d'oeuvre ;
  - valider le produit livré.

Le maître d'oeuvre est, par exemple, une société de services en informatique (SSII).

Les connaissances du domaine par le maître d'ouvrage est nécessaire pour bien cerner les cas d'utilisation exprimés par le client afin d'apporter les solutions adéquates.

Question importante : ai-je toutes les connaissances et les informations pour définir ce que doit faire le système ?

UML : diagrammes (1)

- **Diagrammes structurels ou diagrammes statiques (*UML Structure*)**
  - diagramme de classes (*Class diagram*)
  - diagramme d'objets (*Object diagram*)
  - diagramme de composants (*Component diagram*)
  - diagramme de déploiement (*Deployment diagram*)
  - diagramme de paquetages (*Package diagram*)
  - diagramme de structures composites (*Composite structure diagram*)

UML : diagrammes (2)

- **Diagrammes comportementaux ou diagrammes dynamiques (*UML Behavior*)**
  - diagramme de cas d'utilisation (*Use case diagram*)
  - diagramme d'activités (*Activity diagram*)
  - diagramme d'états-transitions (*State machine diagram*)
  - **Diagrammes d'interaction (*Interaction diagram*)**
    - diagramme de séquence (*Sequence diagram*)
    - diagramme de communication (*Communication diagram*)
    - diagramme global d'interaction (*Interaction overview diagram*)
    - diagramme de temps (*Timing diagram*)

UML : diagrammes (3)

- **Diagramme de cas d'utilisation**

Le diagramme de cas d'utilisation représente la structure des grandes fonctionnalités nécessaires aux utilisateurs du système. C'est le premier diagramme du modèle UML, celui où s'assure la relation entre l'utilisateur et les objets que le système met en œuvre.

- **Diagramme de classes**

Le diagramme de classes est généralement considéré comme le plus important dans un développement orienté objet. Il représente l'architecture conceptuelle du système : il décrit les classes que le système utilise, ainsi que leurs liens, que ceux-ci représentent un emboîtage conceptuel (héritage) ou une relation organique (agrégation).

- **Diagramme d'objets**

Le diagramme d'objets permet d'éclairer un diagramme de classes en l'illustrant par des exemples. Il est, par exemple, utilisé pour vérifier l'adéquation d'un diagramme de classes à différents cas possibles.

UML : diagrammes (4)

- **Diagramme d'états-transitions**

Le diagramme d'états-transitions représente la façon dont évoluent (*i.e.* cycle de vie) les objets appartenant à une même classe. La modélisation du cycle de vie est essentielle pour représenter et mettre en forme la dynamique du système.

- **Diagramme d'activités**

Le diagramme d'activités n'est autre que la transcription dans UML de la représentation du processus telle qu'elle a été élaborée lors du travail qui a préparé la modélisation : il montre l'enchaînement des activités qui concourent au processus.

- **Diagramme de séquence et de communication**

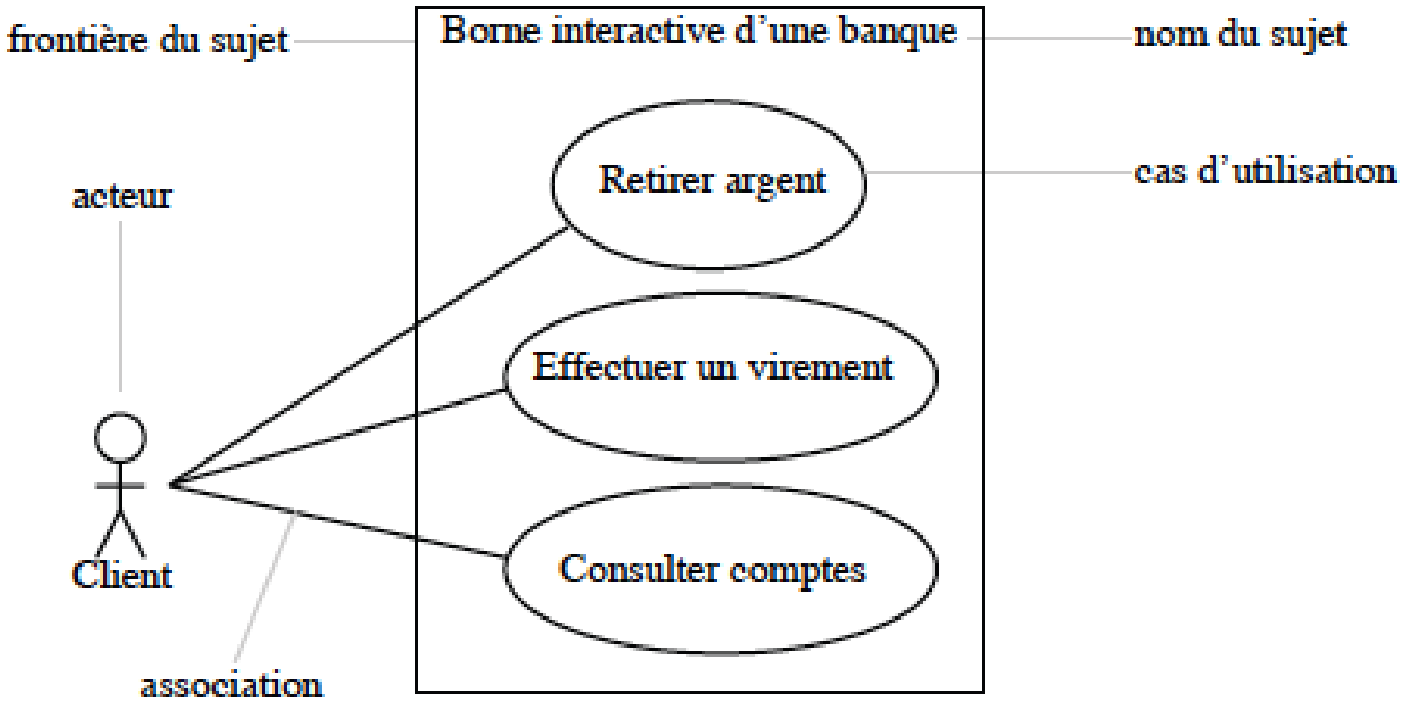
Le diagramme de séquence représente la succession chronologique des opérations réalisées par un acteur. Il indique les objets que l'acteur va manipuler et les opérations qui font passer d'un objet à l'autre.

Le diagramme de cas d’utilisation

Définition

- Un cas d'utilisation est une manière spécifique d'utiliser un système. Les acteurs sont à l'extérieur du système ; ils modélisent tout ce qui interagit avec lui. Un cas d'utilisation réalise un service de bout en bout, avec un déclenchement, un déroulement et une fin, pour l'acteur qui l'initie.

Exemple



borne interactive qui permet d'accéder à une banque

Définition et notations (1)

- Le système à modéliser apparaît dans un cadre (cela permet de séparer le système à modéliser du monde extérieur). Les utilisateurs sont représentés par des petits bons hommes, et les grandes fonctionnalités (les cas d'utilisation) par des ellipses.

- L'ensemble des cas d'utilisation contenus dans le cadre constitue « un sujet ». Les petits bonshommes sont appelés « acteurs ». Ils sont connectés par de simples traits (appelés « associations ») aux cas d'utilisation et mettent en évidence les interactions possibles entre le système et le monde extérieur.

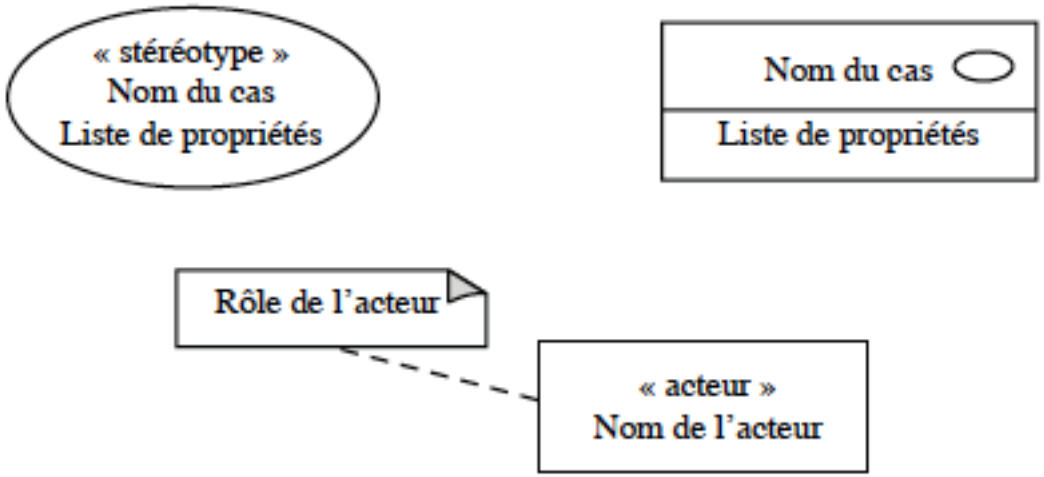
- Chaque cas modélise une façon particulière et cohérente d'utiliser un système pour un acteur donné.

Définition et notations (2)

- Un cas d'utilisation se représente par une ellipse. Le nom du cas est inclus dans l'ellipse ou bien il figure dessous. Un stéréotype, peut être ajouté optionnellement au-dessus du nom, et une liste de propriétés placée au dessous.
- Un cas d'utilisation se représente aussi sous la forme d'un rectangle à deux compartiments :
  - celui du haut contient le nom du cas ainsi qu'une ellipse (symbole d'un cas d'utilisation);
  - celui du bas est optionnel et peut contenir une liste de propriétés.
- Un acteur se représente par un petit bonhomme ayant son nom inscrit dessous ou par un rectangle contenant le stéréotype acteur avec son nom juste en dessous.
- Il est recommandé d'ajouter un commentaire sur l'acteur pour préciser son rôle.

Un *stéréotype* représente une variation d'un élément de modèle existant.

Définition et notations (3)



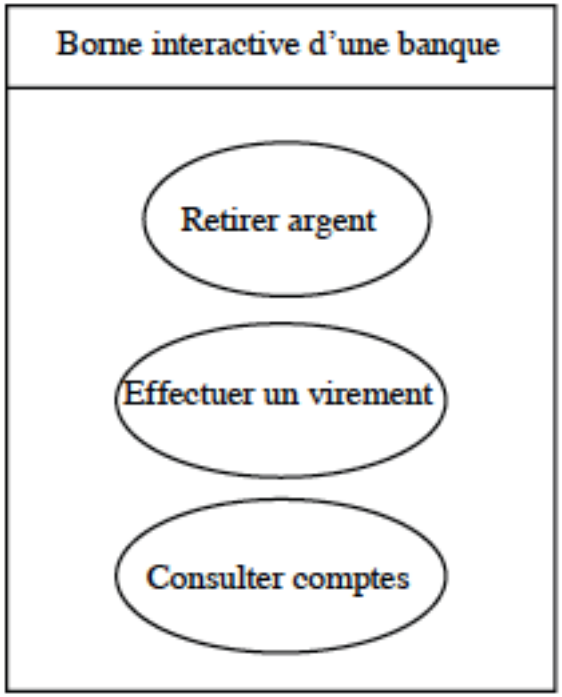
Représentations d'un cas d'utilisation.

À un niveau d'abstraction plus élevé, UML permet de représenter tous les cas d'utilisation d'un système par un simple rectangle (appelé ***classeur***).

Un classeur est un élément de modélisation qui décrit une unité comportementale ou structurelle. Les acteurs et les cas d'utilisation sont des classeurs.

Définition et notations (4)

Un classeur se représente par un rectangle contenant éventuellement des compartiments

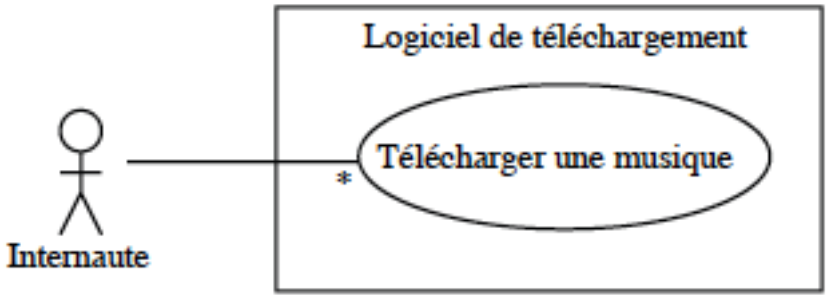


Les cas d'utilisation dans un classeur

Relations entre acteurs et cas d'utilisation (1)

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.

Ex: un internaute qui télécharge plusieurs morceaux de musique sur Internet

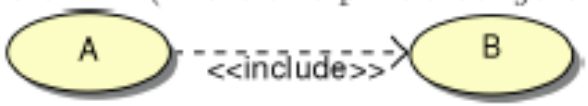


Le symbole * qui signifie « plusieurs » est ajouté à l'extrémité du cas et s'appelle une multiplicité.
Plusieurs valeurs sont possibles pour la multiplicité : exactement n s'écrit tout simplement
n, m..n signifie entre m et n, etc.

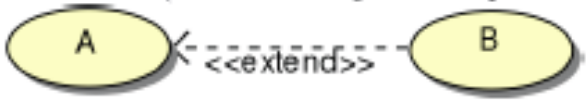
Relations entre acteurs et cas d'utilisation (2)

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 (A est une partie *optionelle* de A).



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

Relations entre cas d'utilisation (1)

- Il existe principalement deux types de relations :
  les dépendances stéréotypées
  la généralisation/spécialisation

Les dépendances stéréotypées sont des dépendances dont la portée est explicitée par le nom du stéréotype.

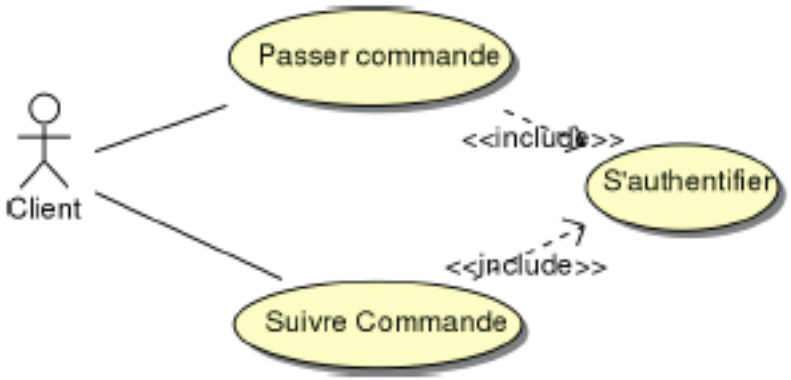
Les stéréotypes les plus utilisés sont l'inclusion et l'extension.

Relations entre cas d'utilisation (2)

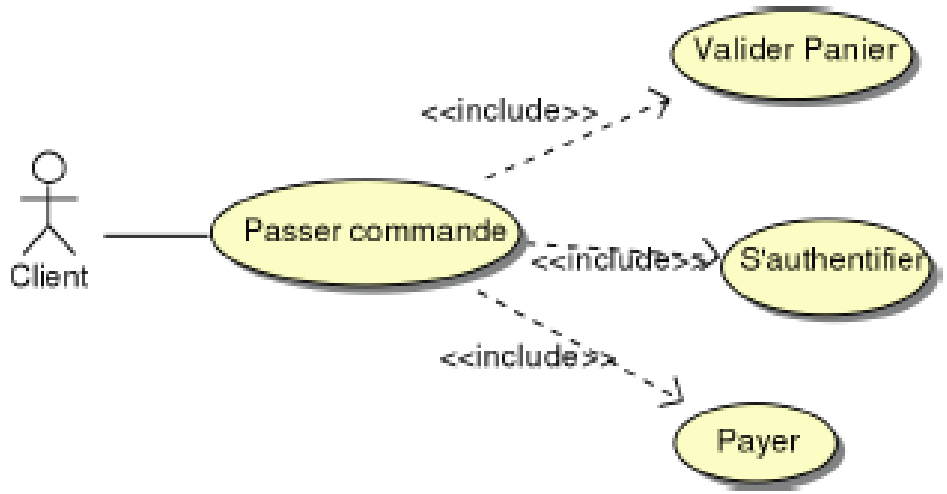
- Les inclusions et les extensions sont représentées par des **dépendances.**
  - Lorsqu'un cas B inclut un cas A, B dépend de A.
  - Lorsqu'un cas B étend un cas A, B dépend aussi de A.
  - On note toujours la dépendance par une èche pointillée B --> A qui se lit B dépend de A .
- Lorsqu'un élément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

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

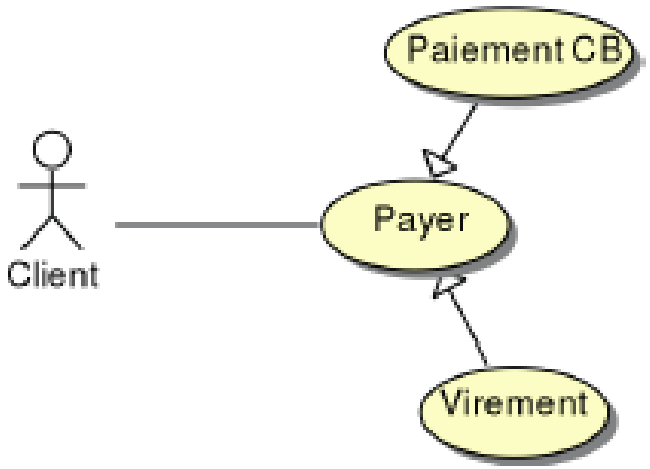
Réutilisabilité avec les inclusions et les extensions



Décomposition grâce aux inclusions et aux extensions

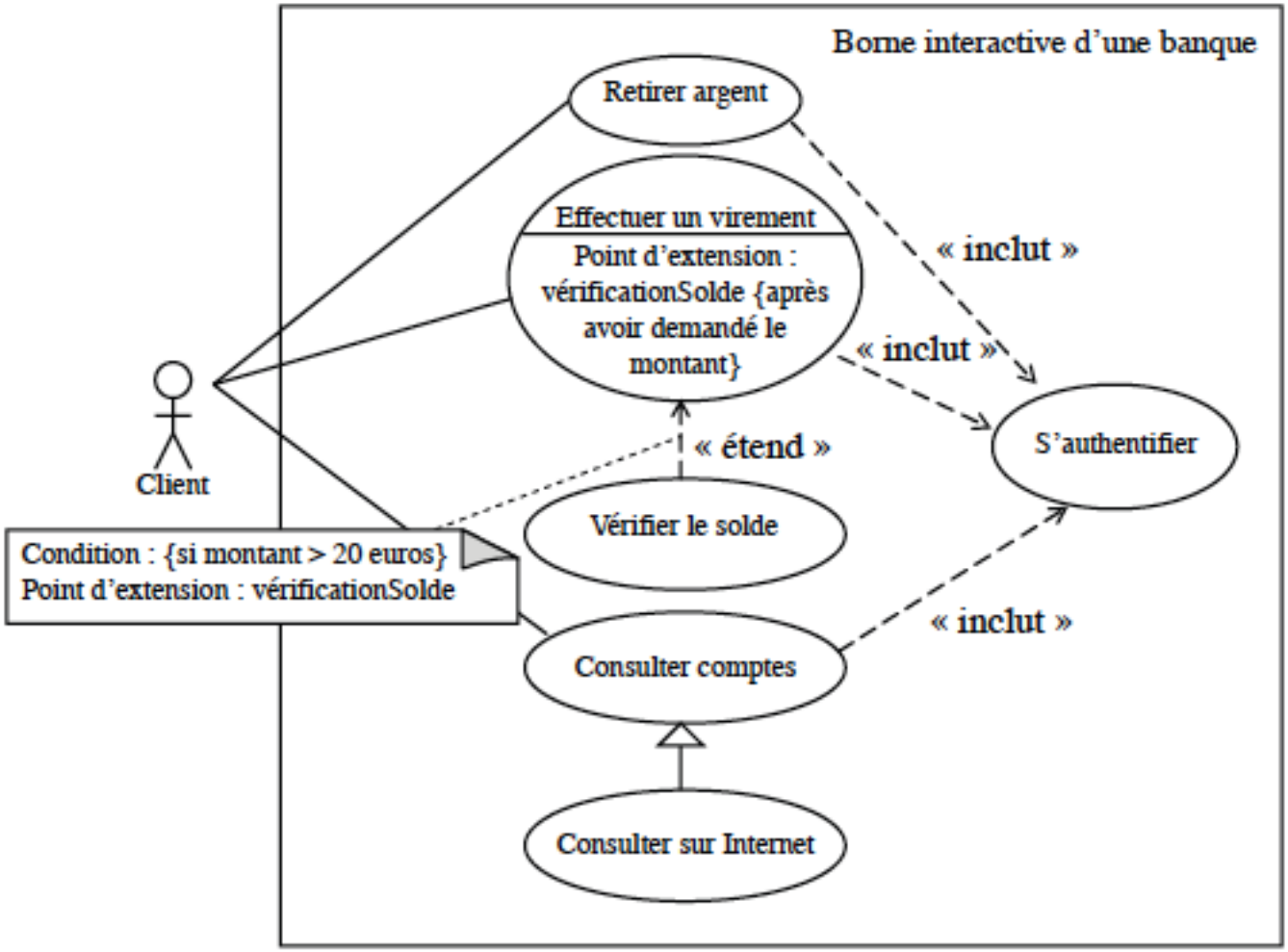


Généralisation



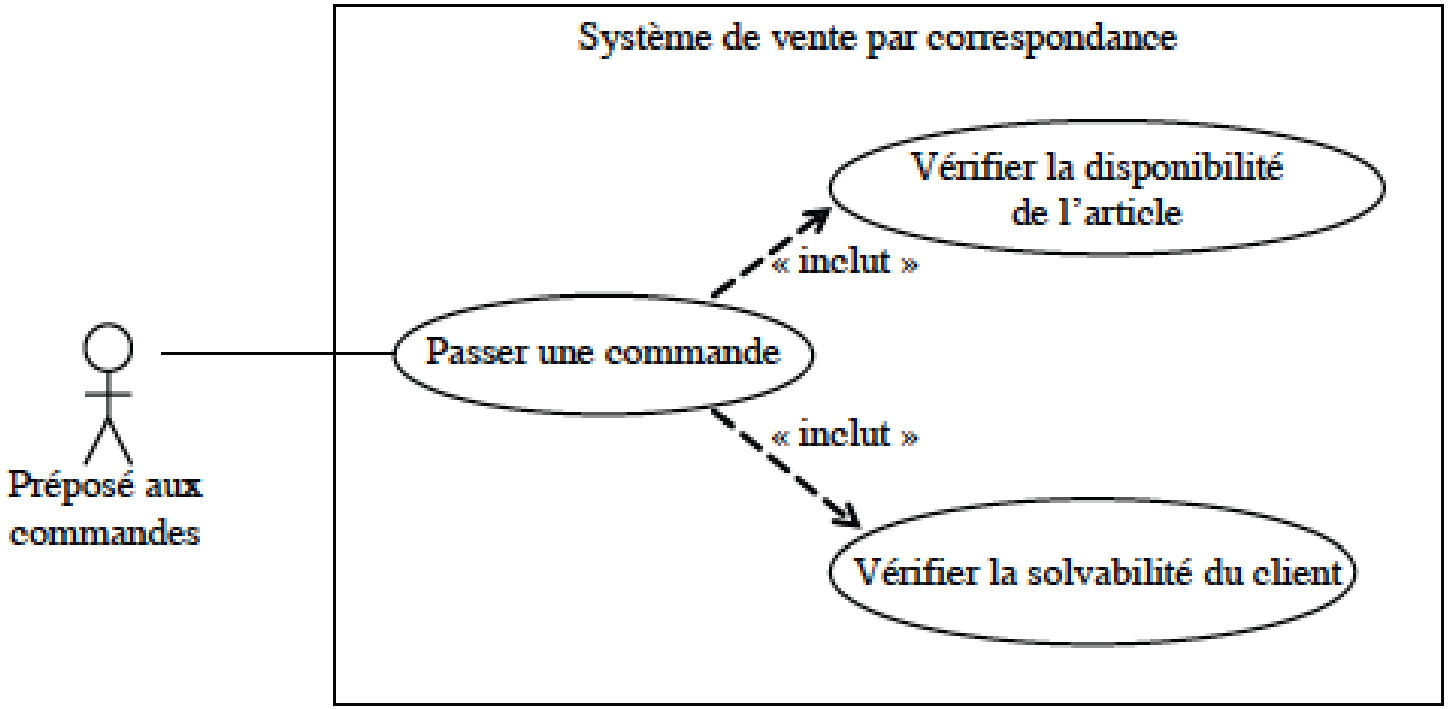
Un virement est est un cas particulier de paiement.
Un virement est une sorte de paiement. La flèche pointe vers l'élément général.
Cette relation de généralisation/spécialisation est présente dans la plupart des diagrammes UML et se traduit par le concept d'héritage dans les langages orientés objet.

Relations entre cas d'utilisation (3)



Relations entre cas dans un diagramme de cas d'utilisation.

Un cas d'utilisation est dit « interne » s'il n'est pas relié directement à un acteur.



Un cas relié à un autre cas peut ne pas être directement accessible à un acteur = « cas interne ».

Relations entre cas d'utilisation (4)

- **La relation d'inclusion.**

Un cas A est inclus dans un cas B si le comportement décrit par le cas A est inclus dans le comportement du cas B : on dit alors que le cas B dépend de A. Cette dépendance est symbolisée par le stéréotype inclut. Par exemple, l'accès aux informations d'un compte bancaire inclut nécessairement une phase d'authentification avec un mot de passe.

- **La relation d'extension.**

Si le comportement de B peut être étendu par le comportement de A, on dit alors que A étend B. Une extension est souvent soumise à condition. Graphiquement, la condition est exprimée sous la forme d'une note. Par exemple, la vérification du solde du compte n'intervient que si la demande de retrait d'argent dépasse 20 euros.

- **La relation de généralisation.**

Un cas A est une généralisation d'un cas B si B est un cas particulier de A. Par exemple, la consultation d'un compte bancaire via Internet est un cas particulier de la consultation. 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.

Les inclusions permettent aussi de décomposer un cas complexe en sous-cas plus simples.

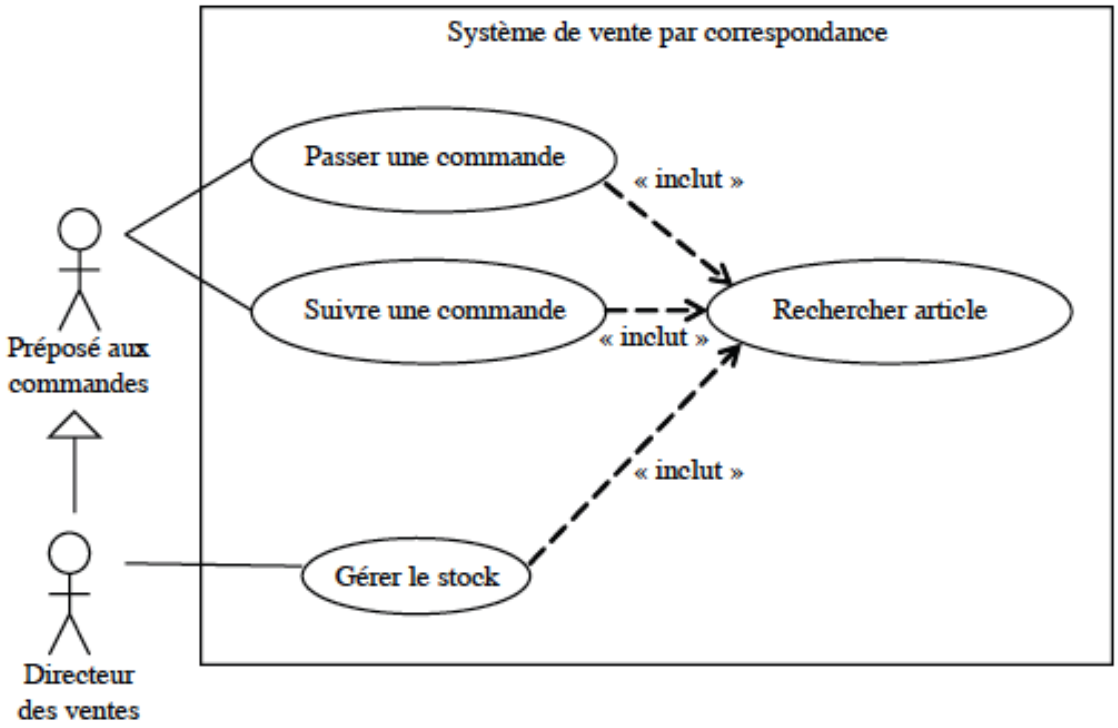
Relations entre acteurs (1)

La seule relation possible entre deux acteurs est la généralisation : un acteur A est une généralisation d'un acteur B si l'acteur A peut être substitué par l'acteur B (tous les cas d'utilisation accessibles à A le sont aussi à B, mais l'inverse n'est pas vrai).

Le symbole utilisé pour la généralisation entre acteurs est une flèche en traits pleins dont la pointe est un triangle fermé. La flèche pointe vers l'acteur le plus général.

-Les principaux acteurs sont les utilisateurs du système.
-Un 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.
Les acteurs peuvent être :
- Des périphériques manipulés par le système (imprimantes, scanners, cameras,...) ;
- Des logiciels déjà disponibles à intégrer dans le projet ;
- Des systèmes informatiques externes au système mais qui interagissent avec lui, etc.

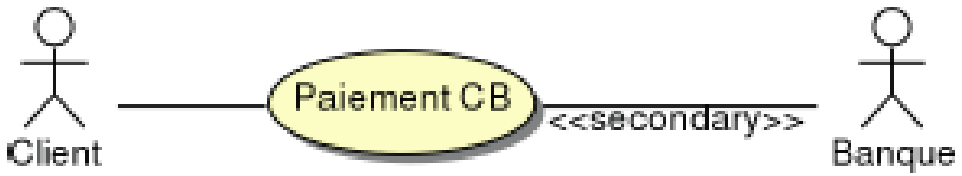
Relations entre acteurs (2)



Acteurs principaux et secondaires

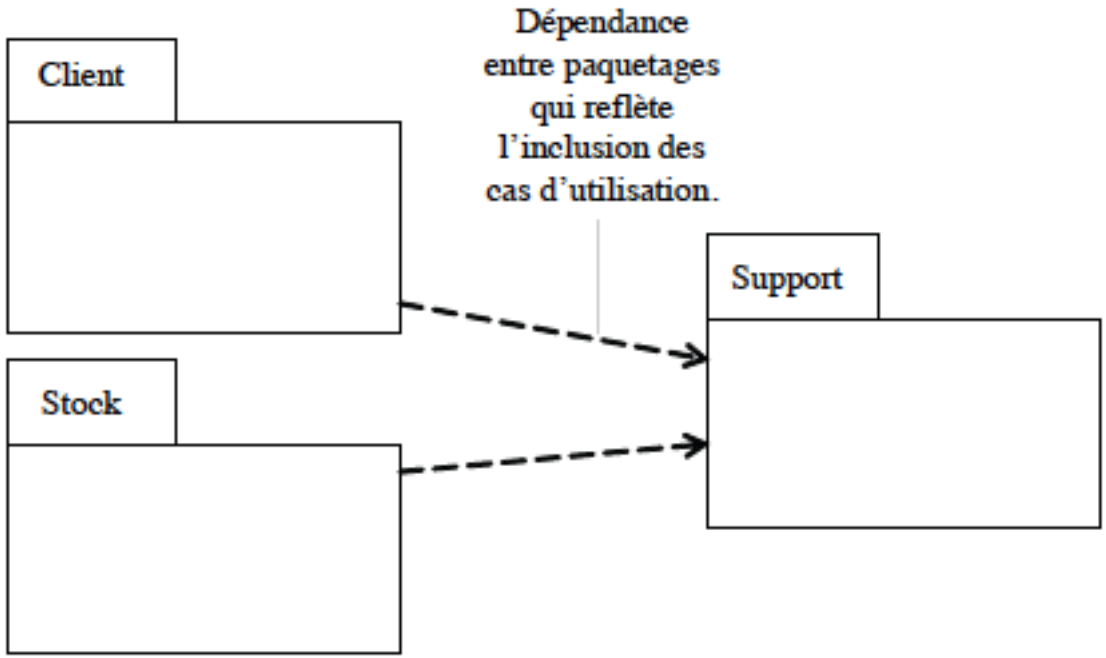
L'acteur est dit *principal* pour un cas d'utilisation lorsque l'acteur est à l'initiative des échanges nécessaires pour réaliser le cas d'utilisation.
Les acteurs secondaires sont sollicités par le système alors que le plus souvent, les acteurs principaux ont l'initiative des interactions.

Le plus souvent, les acteurs secondaires sont d'autres systèmes informatiques avec lesquels le système développé est inter-connecté.



REGROUPEMENT DES CAS D'UTILISATION EN PAQUETAGES

Un paquetage permet d'organiser des éléments de modélisation en groupe.
Un paquetage peut contenir des classes, des cas d'utilisations, des interfaces, etc.



Recensement d'un cas d'utilisation

- Il n'y a pas une façon unique de repérer les cas d'utilisation. Il faut se placer du point de vue de chaque acteur et déterminez 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 action).

Considérons un système de réservation et d'impression de billets de train via des bornes interactives situées dans des gares. En prenant pour acteur une personne qui souhaite obtenir un billet, on peut obtenir la liste suivante des cas d'utilisation :

- rechercher un voyage ;
- réserver une place dans un train ;
- acheter son billet.

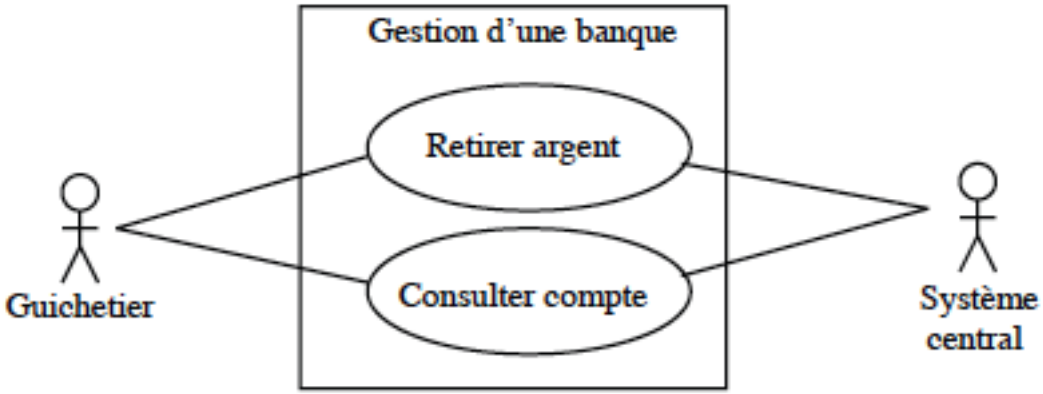
Description des cas d’utilisation (1)

- 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 des cas d'utilisation (2)



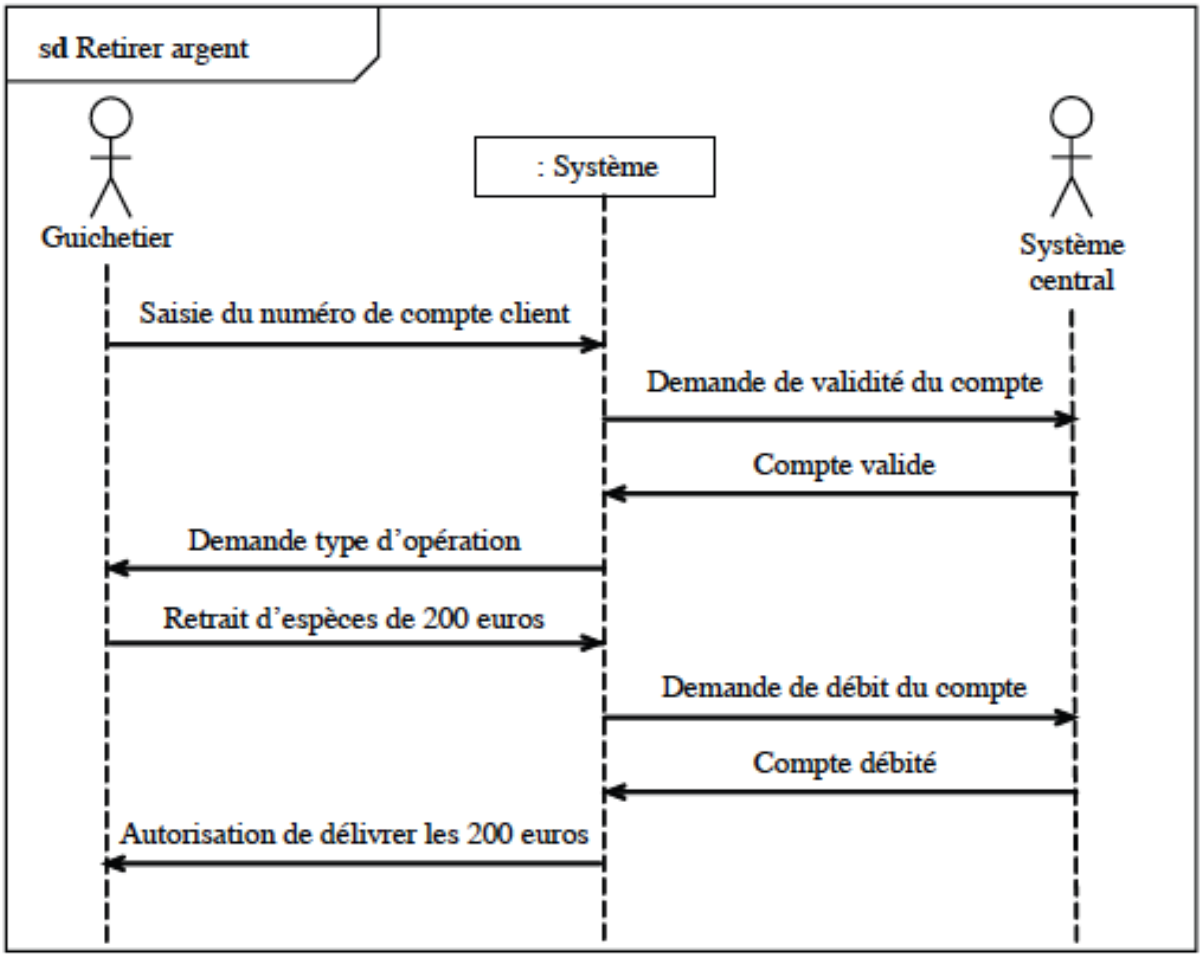
Le diagramme ne permet pas de savoir ce qui entre et ce qui sort du logiciel bancaire :
- *le retrait d'argent se fait-il en euros ou en dollars ?*
- *dans quel ordre les opérations sont-elles effectuées ?*
- *faut-il choisir le montant du retrait avant de choisir le compte à débiter, ou bien l'inverse ?*

Tous ces détails sont des éléments de spécification. Spécifier un produit, c'est le décrire de la façon la plus précise possible.

- Les spécifications peuvent être divisées en deux catégories selon qu'elles sont fonctionnelles ou techniques.
- Les spécifications fonctionnelles concernent les fonctions du système (la fonction de retrait d'argent par ex.), tandis que les spécifications techniques permettent de préciser le contexte d'exécution du système.
- Les spécifications fonctionnelles découlent directement du diagramme de cas d'utilisation.

Il s'agit de reprendre chaque cas et de le décrire très précisément.

Description d’un cas d’utilisation par une séquence de messages.



Description textuelle des cas d'utilisation (1)

- Une description textuelle couramment utilisée se compose de trois parties:
- Identification du cas :
  - le nom du cas ;
  - un résumé de son objectif ;
  - les acteurs impliqués (principaux et secondaires) ;
  - les dates de création et de mise à jour de la description courante ;
  - le nom des responsables ;
  - un numéro de version.

Description textuelle des cas d'utilisation (2)

La description du fonctionnement du cas contient toujours une séquence nominale qui correspond au fonctionnement nominal du cas

Cette séquence nominale commence par préciser l'événement qui déclenche le cas (l'utilisateur introduit sa carte bancaire par exemple) et se développe en trois points :

- **Les pré-conditions** - indiquent dans quel état est le système avant que se déroule la séquence (le distributeur est alimenté en billets par exemple).
- **L'enchaînement des messages**.
- **Les post-conditions** - indiquent dans quel état se trouve le système après le déroulement de la séquence nominale (une transaction a été enregistrée par la banque par exemple).

Description textuelle des cas d'utilisation (3)

- À la séquence nominale s'ajoutent fréquemment des séquences alternatives et des séquences d'exceptions.
- Une séquence alternative diverge de la séquence nominale (c'est un embranchement dans une séquence nominale) mais y revient toujours, alors qu'une séquence d'exception intervient quand une erreur se produit (le séquencement nominal s'interrompt, sans retour à la séquence nominale).

Dans le cas d'un retrait d'argent, des séquences alternatives se produisent par exemple dans les situations suivantes :

- Le client choisit d'effectuer un retrait en euros ou en dollars.
- Le client a la possibilité d'obtenir un reçu.
- Une exception se produit si la connexion avec le système central de la banque qui doit vérifier la validité du compte est interrompue.

Description textuelle des cas d'utilisation (4)

- La dernière partie de la description d'un cas d'utilisation est une rubrique optionnelle. Elle contient généralement des spécifications non fonctionnelles

- des spécifications techniques, par exemple pour préciser que l'accès aux informations bancaires doit être sécurisé);
- d'éventuelles contraintes liées aux interfaces homme-machine (par exemple, pour donner la possibilité d'accéder à tous les comptes d'un utilisateur à partir de l'écran principal

Description textuelle des cas d'utilisation (5)

- Description d'un retrait d'argent

Conclusions

- Le diagramme de cas d'utilisation est un premier modèle d'un système.

- Que savons-nous sur le système après avoir créé ce diagramme ?
  - nous ne savons toujours pas comment réaliser le système
  - son interface avec le monde qui l'entoure est partiellement connue

- **L'objectif de cette phase de la modélisation est d'identifier clairement les frontières du système et les interfaces qu'il doit offrir à l'utilisateur**.