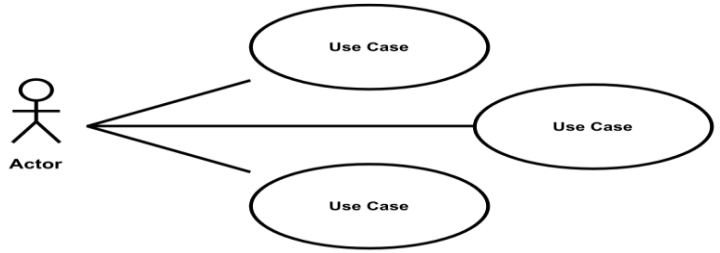
UNIFIED MODELING LANGUAGE™

Le Diagramme de Cas d'Utilisation

Mme GHACHEM Amira

1ère année Ingénieur



Plan

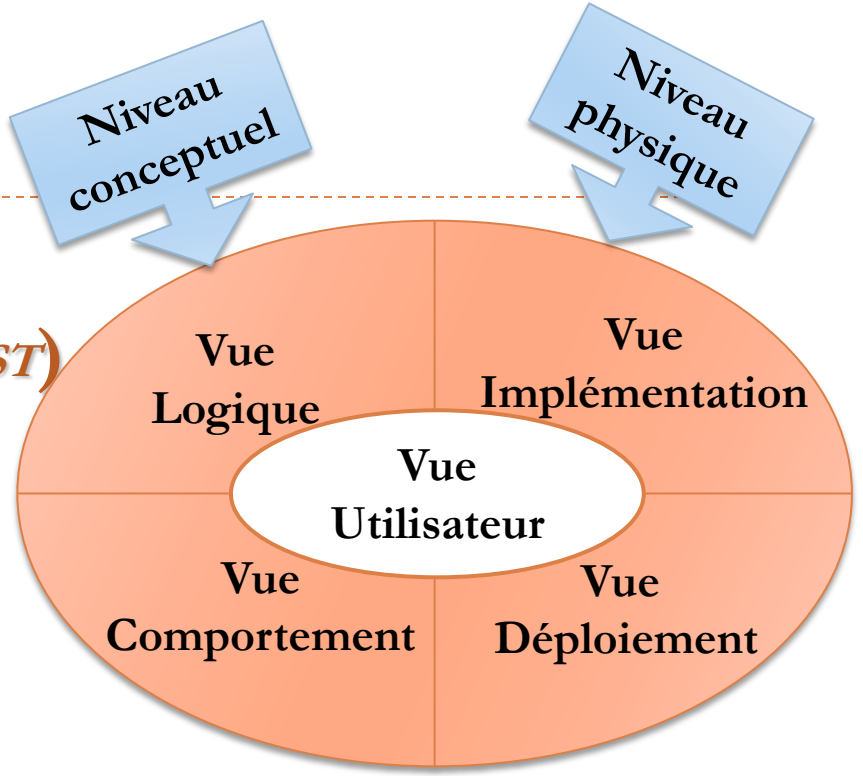
1. Introduction
2. L'importance de bien recueillir les besoins
3. Le diagramme de cas d'utilisation

Introduction

- Les diagramme d'UML peuvent être utilisés pour représenter différents points de vues:
  - **Vue externe** : vue du système par ses utilisateurs finaux
  - **Vue logique statique** : structure des objets et leurs relations
  - **Vue logique dynamique** : comportement du système
  - **Vue d'implémentation** : composants logiciels
  - **Vue de déploiement** : répartition des composants

Introduction

Axes de Modélisation avec UML



Statique (*ce que le système EST*)

- Diagramme de Classes
- Diagramme d'Objets
- Diagramme de Composants
- Diagramme de Déploiement
- Diagramme de paquetages

Fonctionnel (ce que le système FAIT)

- Diagramme de Use Case

Dynamique (*comment le système EVOLUE*)

- Diagramme de Séquence
- Diagramme de collaboration (UML 2.x)
- Diagramme global d'interaction (UML 2.x)
- Diagramme de temps (UML 2.x)
- Diagramme d'Etats-Transitions
- Diagramme d'Activité

Introduction

- Les modèles se complètent et peuvent être rassemblés.
- Ils sont élaborés tout au long du cycle de vie du développement d'un système:
  - depuis le recueil des besoins jusqu'à la phase de conception.

Introduction

- Dans ce chapitre, nous allons étudier:
  - Le diagramme de cas d'utilisation.
- Il permet de:
  - recueillir,
  - Analyser,
  - Organiser les besoins.
- Avec le diagramme de cas d'utilisation, on débute l'étape d'analyse d'un système.

L'importance de bien recueillir les besoins (1/3)

- Le développement d'un nouveau système, ou l'amélioration d'un système existant, doit répondre à un ou à plusieurs besoins.
- Par exemple, une banque a besoin d'un guichet automatique pour que ses clients puissent retirer de l'argent même en dehors des heures d'ouverture de la banque.
  - Celui qui **commande le logiciel** est le **maître d'ouvrage**.
  - Celui qui **réalise le logiciel** est le **maître d'œuvre**.
- 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'œuvre ;
  - valider le produit livré.

L'importance de bien recueillir les besoins

Maître d'œuvre (2/3)

- Le maître d'œuvre est, par exemple, une société de services en informatique.
- Il a été choisi, avant tout, pour ses compétences techniques. Mais son savoir-faire va bien au-delà.
- Au début du projet, il est capable de recueillir les besoins auprès du maître d'ouvrage.
- Le recueil des besoins implique une bonne compréhension des métiers concernés.
  - Réaliser un logiciel pour une banque, par exemple, implique la connaissance du domaine bancaire et l'intégration de toutes les contraintes et exigences de ce métier.
- Cette condition est nécessaire pour bien cerner les cas d'utilisation exprimés par le client afin d'apporter les solutions adéquates.

L'importance de bien recueillir les besoins

Maître d'ouvrage (3/3)

- Chaque cas a ses particularités liées au métier du client.
- Le recueil des besoins peut s'opérer de différentes façons.
- Cela dit, il est recommandé de compléter le cahier des charges par des discussions approfondies avec le maître d'ouvrage et les futurs utilisateurs du système.
- Il convient également d'utiliser tous les documents produits à propos du sujet (rapports techniques, étude de marché…) et d'étudier les procédures administratives des fonctions de l'entreprise qui seront prises en charge par le système.
- La question que doit se poser **le maître d'œuvre** durant le recueil des besoins est la suivante :

" Ai-je toutes les connaissances et les informations pour définir ce que doit faire le système ? "

Le Diagramme de Cas d'Utilisation

- 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.

- N'oublions pas que bien souvent, le maître d'ouvrage et les utilisateurs ne sont pas des informaticiens.
- Il leur faut donc un moyen simple d'exprimer leurs besoins.
- C'est précisément le rôle des diagrammes de cas d'utilisation.
- Donc un diagramme de cas d'utilisation permet de **recenser les grandes fonctionnalités d'un système**.

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

Exemple

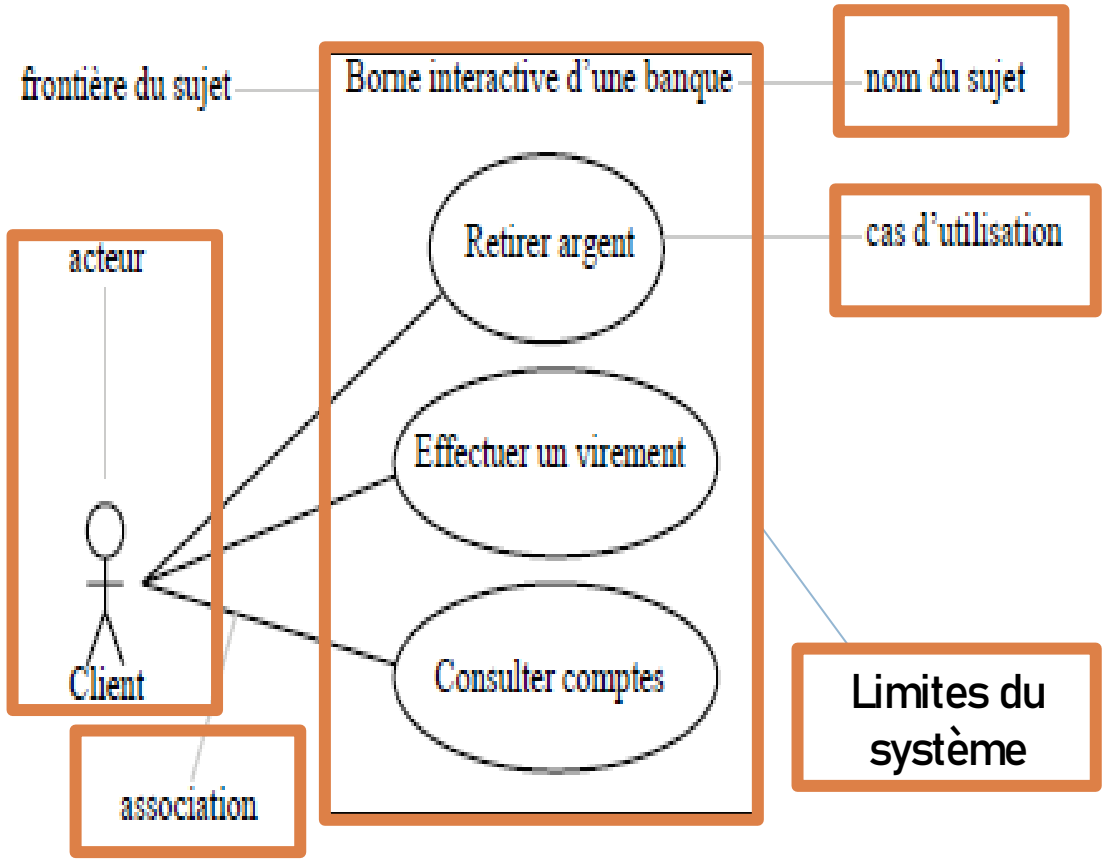


Diagramme de cas d'utilisation modélisant une borne d'accès à une banque.

Cas d'utilisation

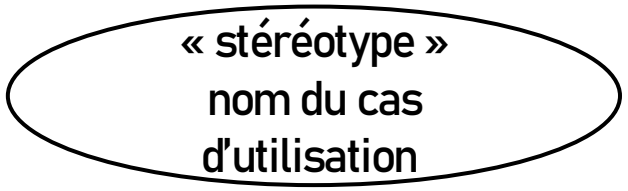
Définition

- 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.

Cas d'utilisation

Représentation

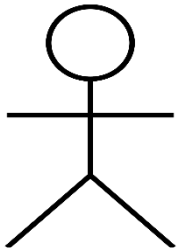
- Un cas d'utilisation se représente par une ellipse.
- Le nom du cas est inclus dans l'ellipse ou bien il figure dessous.



Acteur

Définition et représentation

- Un acteur se représente par un petit bonhomme ayant son rôle inscrit dessous ou par un rectangle contenant le stéréotype *acteurs* avec son rôle juste en dessous.



Acteur
Définition et représentation

- Les acteurs peuvent être de trois types:
  - **Humains**
    - utilisateurs du logiciel à travers son interface graphique, par exemple.
  - **Logiciels**
    - disponibles qui communiquent avec le système grâce à une interface logicielle (API, …)
  - **Matériels**
    - exploitant les données du système ou qui sont pilotés par le système (Imprimante, robots, automates, …)

<actor>
Nom-système

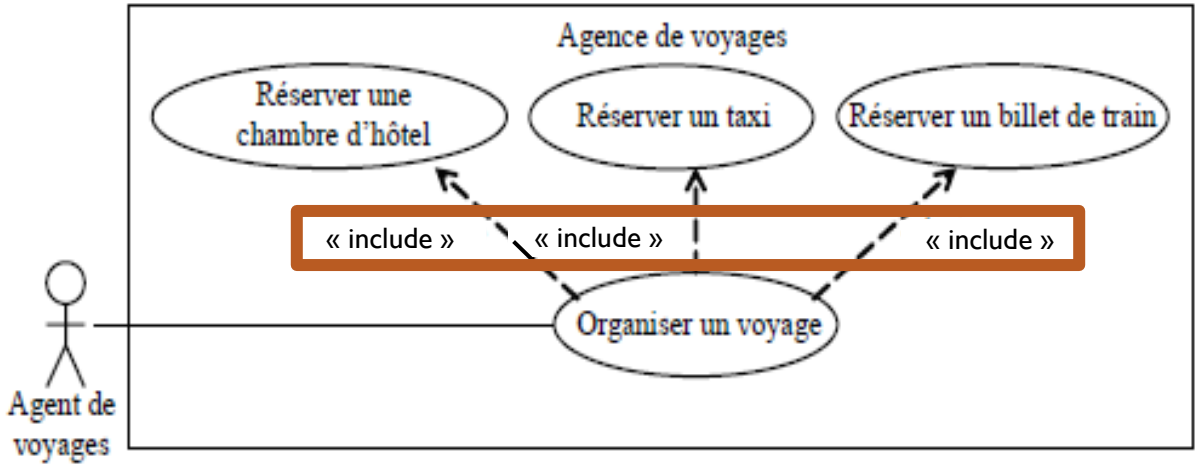
Relations entre cas d'utilisation

- Pour clarifier un diagramme, UML permet d'établir des relations entre les cas d'utilisation.
- Il existe principalement deux types de relations :
  - les dépendances stéréotypées
  - et 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

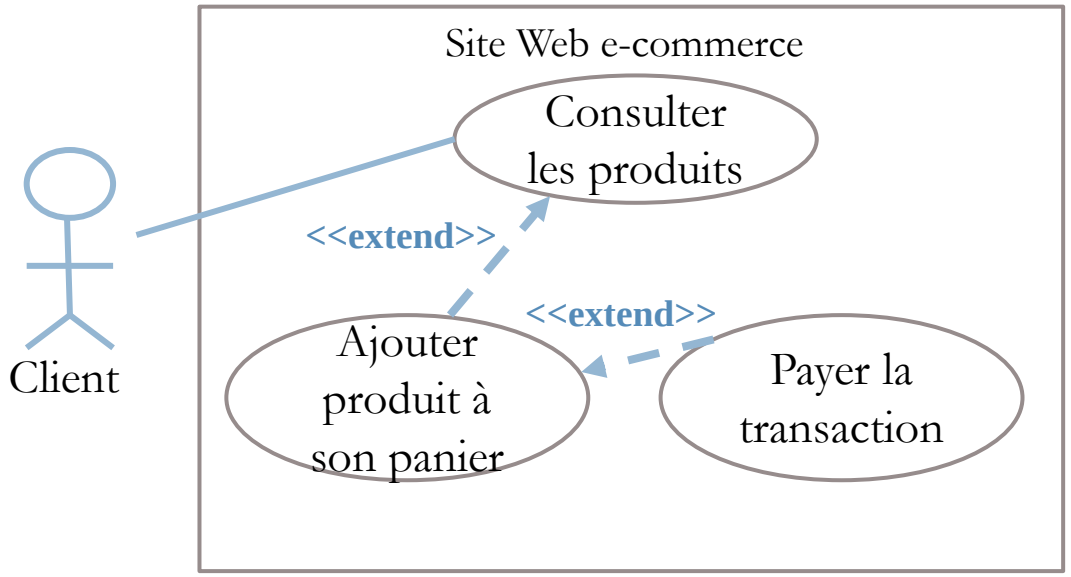
La relation d'inclusion

- 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 *include*.

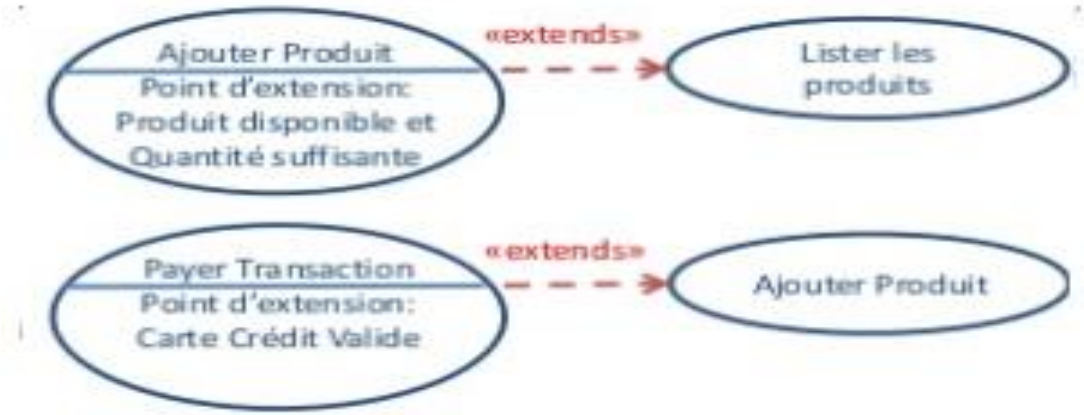


Relations entre cas d'utilisation
la relation d'extension

- La 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.
  - Elle est symbolisée par le stéréotype *extend*.

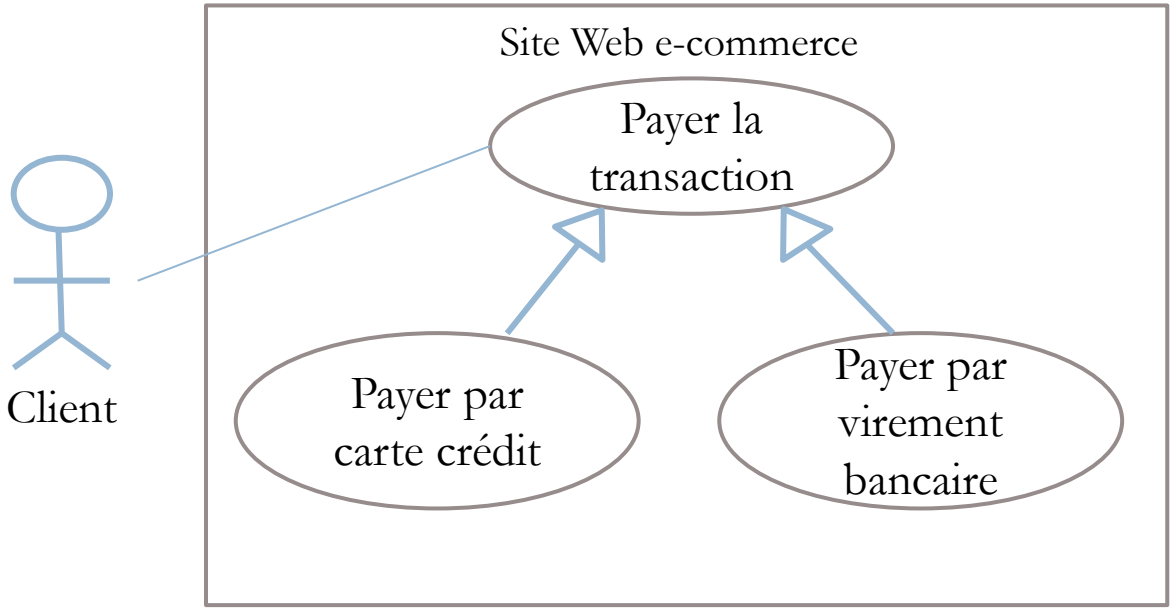


- L'extension peut intervenir à un point précis du cas étendu ; ce point s'appelle le point d'extension ;
- Il est éventuellement associé à une contrainte indiquant le moment où l'extension intervient.



Relations entre cas d'utilisation
la relation de généralisation

- Un cas A est une généralisation d'un cas B si B est un cas particulier de A.
  - À la figure ci-contre, 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.

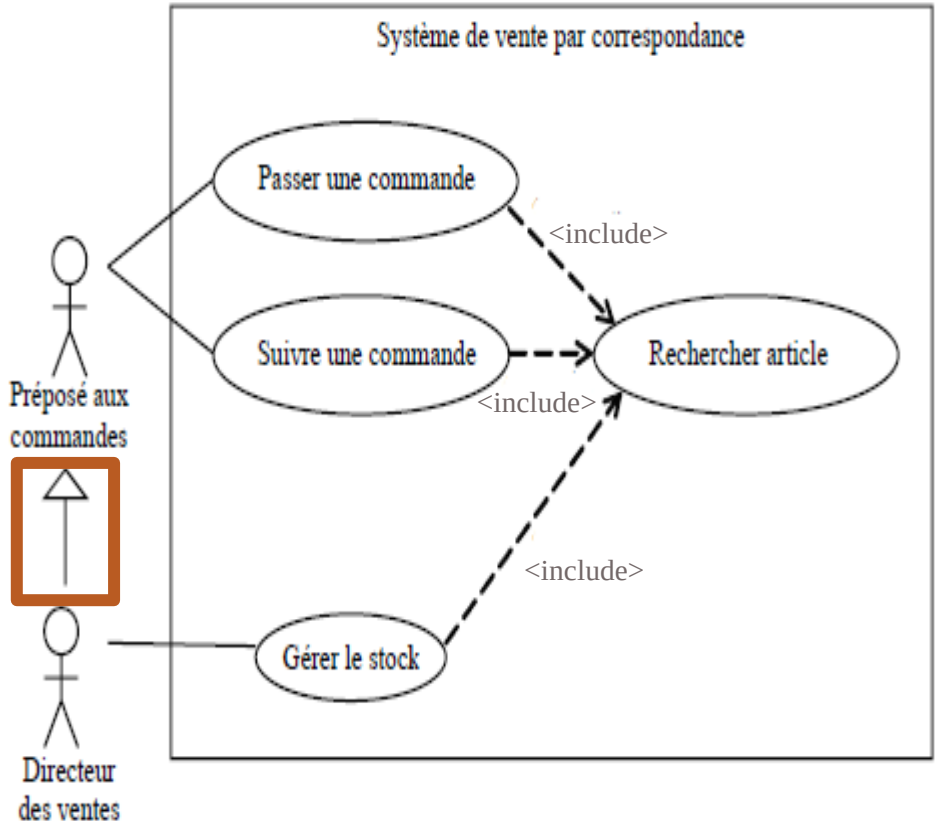


- Le symbole utilisé pour la généralisation est une flèche en traits pleins dont la pointe est un triangle fermé.
- La flèche pointe vers le cas le plus général.

Relations entre acteurs

- 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.
- La figure, ci-contre, montre que le directeur des ventes est un préposé aux commandes avec un pouvoir supplémentaire:
  - en plus de pouvoir passer et suivre une commande, il peut gérer le stock.
- Le préposé aux commandes ne peut pas gérer le stock.

→ Le symbole utilisé pour la généralisation entre acteurs est une flèche en traits pleins dont la pointe est un triangle fermé.



Modélisation des besoins avec UML

Qui sont les acteurs? Comment peut-on les identifier?

- Les principaux acteurs sont les utilisateurs du système. Ils sont en général faciles à repérer.
- C'est le maître d'ouvrage qui les désigne. Chaque acteur doit être nommé, mais attention, pour trouver son nom, il faut penser à son rôle : un acteur représente un ensemble cohérent de rôles joués vis-à-vis d'un système.
- Ainsi, pour un logiciel de gestion de paie, le nom correct d'un acteur est Comptable plutôt que Mr Ben Mohamed.
- Plusieurs personnes peuvent avoir le même rôle. C'est le cas des clients d'une banque par exemple. Il n'y aura alors qu'un seul acteur.
- Réciproquement, une même personne physique peut jouer des rôles différents vis-à-vis du système et donc correspondre à plusieurs acteurs.

Modélisation des besoins avec UML

Qui sont les acteurs? Comment peut-on les identifier?

- En général, les utilisateurs ne sont pas difficiles à trouver, mais il faut veiller à ne pas oublier les personnes responsables de l'exploitation et de la maintenance du système.
- Par exemple, un logiciel de surveillance qui limite les accès à un bâtiment doit avoir un administrateur chargé de créer des groupes de personnes et leur donner des droits d'accès. Il ne s'agit pas ici des personnes qui installent et paramètrent le logiciel avant sa mise en production, mais des utilisateurs du logiciel dans son fonctionnement nominal.

Modélisation des besoins avec UML

Qui sont les acteurs? Comment peut-on les identifier?

- En plus des utilisateurs, les acteurs peuvent être :
  - des périphériques manipulés par le système (imprimantes, robots…)
  - des logiciels déjà disponibles à intégrer dans le projet
  - des systèmes informatiques externes au système mais qui interagissent avec lui, etc.
- Pour faciliter la recherche des acteurs, on peut imaginer les frontières du système. Tout ce qui est à l'extérieur et qui interagit avec le système est un acteur ; tout ce qui est à l'intérieur est une fonctionnalité du système que le maître d'oeuvre doit réaliser.

Modélisation des besoins avec UML

Qui sont les acteurs? Comment peut-on les identifier?

- Un cas d'utilisation a toujours au moins un acteur principal pour qui le système produit un résultat observable, et éventuellement d'autres acteurs ayant un rôle secondaire.
- Par exemple, l'acteur principal d'un cas de retrait d'argent dans un distributeur automatique de billets est la personne qui fait le retrait, tandis que la banque qui vérifie le solde du compte est un acteur secondaire.
- En général, l'acteur principal initie le cas d'utilisation par ses sollicitations.

Comment recenser les 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 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 action.

Comment recenser les cas d'utilisation?

Exemple

- 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.

Comment recenser les cas d'utilisation?

Exemple

- L'ensemble des cas d'utilisation doit couvrir exhaustivement tous les besoins fonctionnels du système.
- L'étape de recueil des besoins est souvent longue car les utilisateurs connaissent l'existant et n'ont qu'une vague idée de ce que leur apportera un futur système ;
- en outre, le cahier des charges contient des imprécisions, des oublis, voire des informations contradictoires difficiles à extraire.
- L'élaboration du diagramme de cas d'utilisation permet de pallier ces problèmes en recentrant le système sur les besoins fonctionnels et ce, dès le début du projet.

Comment recenser les cas d'utilisation?

Exemple

- On peut se demander pourquoi adopter un point de vue fonctionnel, et non technique ?
- Trop souvent, par le passé, les logiciels étaient techniquement très élaborés sans pour autant satisfaire les utilisateurs.
- Avec les diagrammes de cas d'utilisation, on se place clairement du côté des utilisateurs.
- Le côté technique n'est pas oublié mais abordé différemment :
  - Les besoins sont affinés lors de l'écriture des spécifications
  - On parle de spécifications techniques des besoins (c'est la « Description textuelle des cas d'utilisation »).

Remarques

- Il ne faut pas faire apparaître les détails des cas d'utilisation, mais il faut rester au niveau des grandes fonctions du système.
  - La question qui se pose alors est de savoir jusqu'à quel niveau de détails descendre ?
- Si le nombre de cas est trop important, il faut se demander si on a fait preuve de suffisamment d'abstraction. Le nombre de cas d'utilisation est un bon indicateur de la faisabilité d'un logiciel.
- Il ne doit pas y avoir de notion temporelle dans un diagramme de cas d'utilisation.
- Il ne faut pas se dire que l'acteur fait ceci, puis le système lui répond cela, ce qui implique une réaction de l'acteur, et ainsi de suite.
- Le séquencement temporel sera pris en compte plus tard, notamment dans la description détaillée des cas.
- La description des cas d'utilisation peut servir de base de travail pour établir les tests de vérification du bon fonctionnement du système, et orienter les travaux de rédaction de la documentation à l'usage des utilisateurs.

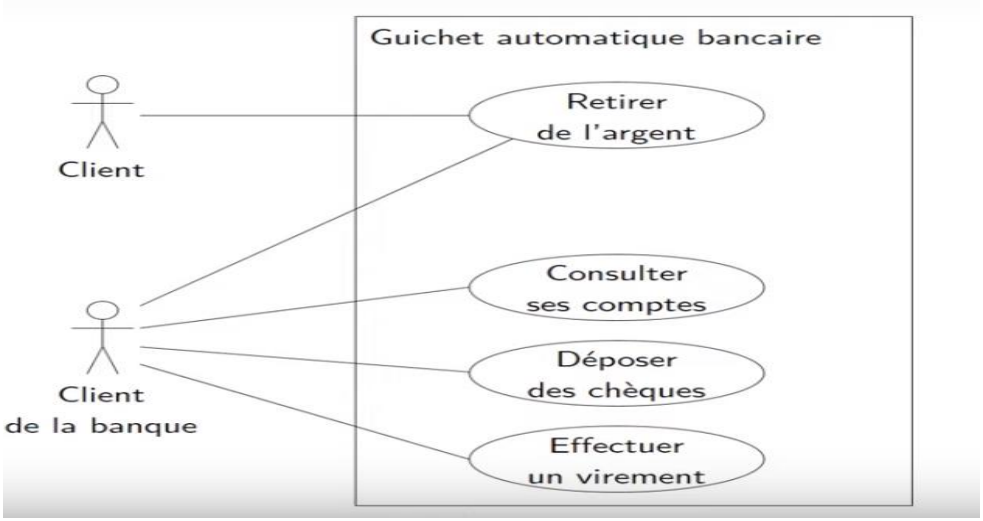
Description des cas d'utilisation

- 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.
- L'exemple qui suit, 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 exemple,
  - Les spécifications techniques permettent de préciser le contexte d'exécution du système.
- Par exemple:
  - le logiciel qui gère la distribution des billets doit être compatible avec tel ou tel système d'exploitation,
  - ou encore, un retrait d'argent doit se faire en moins de 5 secondes.

Description textuelle

Exemple



- Les spécifications fonctionnelles découlent directement du diagramme de cas d'utilisation.
- Il s'agit de reprendre chaque cas et de le décrire très précisément.
- En d'autres termes, on doit décrire comment les acteurs interagissent avec le système.

Les différentes façons de décrire les cas d'utilisation

Description textuelle

- Bien que de nombreux diagrammes d'UML permettent de décrire un cas, il est recommandé de rédiger une description textuelle car c'est une forme souple qui convient bien dans toutes les situations.
- Une description textuelle couramment utilisée se compose de trois parties:
  1. La première partie permet d'**identifier le cas** qui contient:
     - 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.

Les différentes façons de décrire les cas d'utilisation

Description textuelle

2. La deuxième partie contient la **description du fonctionnement du cas** sous la forme d'une séquence de messages échangés entre les acteurs et le système.

- Elle contient toujours une séquence nominale qui correspond au fonctionnement nominal du cas
  - par exemple, un retrait d'argent qui se termine par l'obtention des billets demandés par l'utilisateur.
  - 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**: Elles 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**. Elles 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).

Les différentes façons de décrire les cas d'utilisation

Description textuelle - Remarque

- Les acteurs n'étant pas sous le contrôle du système, ils peuvent avoir des comportements imprévisibles.
- La séquence nominale ne suffit donc pas pour décrire tous les comportements possibles.
- 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 croisement 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).

Les différentes façons de décrire les cas d'utilisation

Description textuelle - Remarque

- 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.
- La survenue des erreurs dans les séquences doit être signalée de la façon suivante : « appel de l'exception Y » où Y est le nom de l'exception.

Les différentes façons de décrire les cas d'utilisation

Description textuelle

3. La troisième partie de la description textuelle d'un cas d'utilisation est **une rubrique optionnelle.**

- Elle contient généralement des spécifications non fonctionnelles;
- ce sont le plus souvent des spécifications techniques,
  - par exemple pour préciser que l'accès aux informations bancaires doit être sécurisé.
- Cette rubrique contient aussi 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,
  - ...

Exemple

Description d'un retrait d'argent

Identification
Nom du cas : retrait d'espèces en euros.
But : détaille les étapes permettant à un guichetier d'effectuer l'opération de retrait d'euros demandé par un client.
Acteur principal : Guichetier.
Acteur secondaire : Système central.
Date : le 18/02/2005.
Responsable : M. Dupont.
Version : 1.0.

Séquencement
Le cas d'utilisation commence lorsqu'un client demande le retrait d'espèces en euros.
Pré-conditions
Le client possède un compte (donne son numéro de compte).
Enchaînement nominal

1. Le guichetier saisit le numéro de compte client.
2. L'application valide le compte auprès du système central.
3. L'application demande le type d'opération au guichetier.
4. Le guichetier sélectionne un retrait d'espèces de 200 euros.
5. L'application demande au système central de débiter le compte.
6. Le système notifie au guichetier qu'il peut délivrer le montant demandé.

Post-conditions
Le guichetier ferme le compte.
Le client récupère l'argent.

Rubriques optionnelles
Contraintes non fonctionnelles
Fiabilité : les accès doivent être extrêmement sûrs et sécurisés.
Confidentialité : les informations concernant le client ne doivent pas être divulguées.
Contraintes liées à l'interface homme-machine
Donner la possibilité d'accéder aux autres comptes du client.
Toujours demander la validation des opérations de retrait.

TD 1 – Le Diagramme de Cas d'Utilisation

TD1 – Exercice 1

Considérons le système informatique qui gère une station-service de distribution d'essence.

On s'intéresse à la modélisation de la prise d'essence par un client.

Le client se sert de l'essence de la façon suivante. Il prend un pistolet accroché à une pompe et appuie sur la gâchette pour prendre de l'essence.

1. Qui est l'acteur du système ? Est-ce le client, le pistolet ou la gâchette ?

Le pompiste peut se servir de l'essence pour sa voiture.

2. Est-ce un nouvel acteur ?

La station a un gérant qui utilise le système informatique pour des opérations de gestion.

3. Est-ce un nouvel acteur ?

La station-service a un petit atelier d'entretien de véhicules dont s'occupe un mécanicien.

Le gérant est remplacé par un chef d'atelier qui, en plus d'assurer la gestion, est aussi mécanicien.

4. Comment modéliser cela ?