UML

1 - Diagramme de cas d'utilisation

Diagramme de comportement statique

Analyse fonctionnelle

Bertrand LIAUDET

SOMMAIRE

Dernière édition : juin 2019 – maj oct 2021

UC - BASES

Il est facile de décrire la méthode encore que son application exige à coup sûr savoir et pratique.
La méthode est dénuée de sens tant qu'elle est déconnectée du rapport au savoir.

1. La notion de cas d'utilisation : UC – Approche par un exemple

Contexte

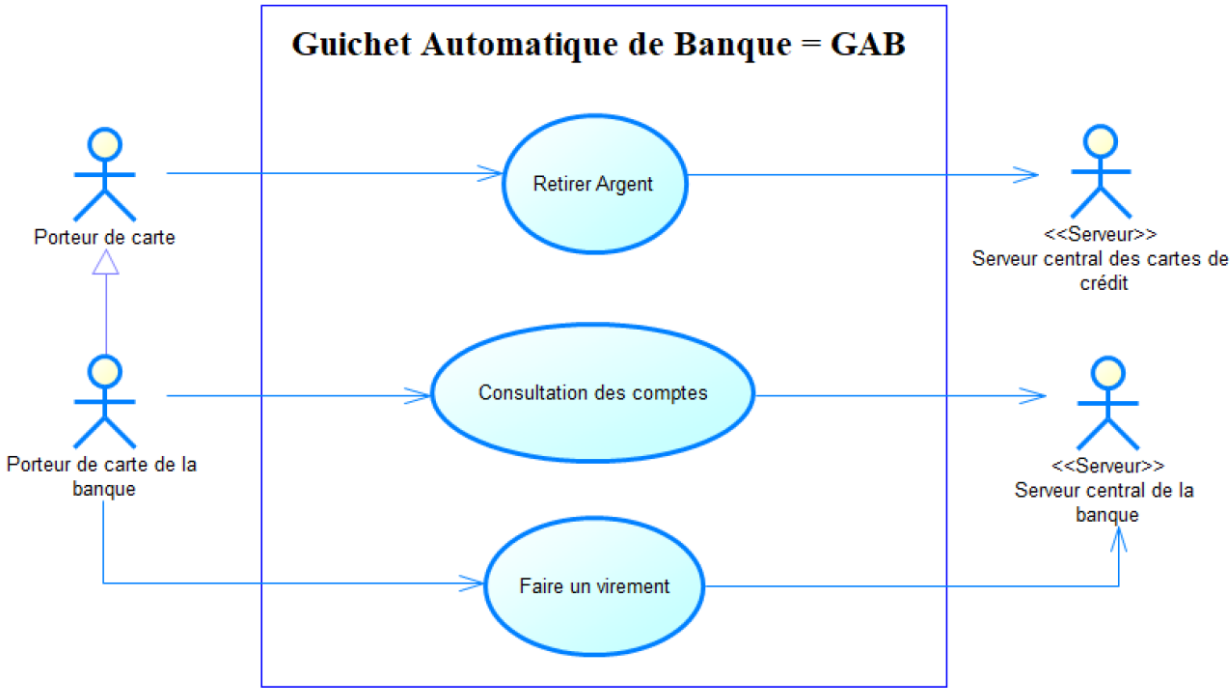
- Les cas d'utilisation relèvent de l'analyse fonctionnelle (par opposition à l'analyse technique).
- On dit : cas d'utilisation ou Use Case ou UC.
- Il y a 4 diagrammes UML qu'on utilise en analyse fonctionnelle :

	Diagramme UML
ANALYSE FONCTIONNELLE	Diagramme de cas d'utilisation - UC
	Activité
	Séquence
	Etats-transitions

- On ne traite pour commencer que du diagramme de cas d'utilisation.

Exemple de diagramme de cas d'utilisation : le GAB

- Un guichet automatique de banque permet aux porteurs de carte de **retirer de l'argent**. Les porteurs de carte de la banque du guichet peuvent en plus **consulter leurs comptes et faire des virements**. Pour retirer de l'argent, le système communique avec le **serveur central des cartes de crédit**. Pour consulter les comptes ou faire un virement, le système communique avec le **serveur central de la banque concernée**.



- Dans ce schéma on trouve un système, des acteurs actifs et passifs, un héritage entre acteurs et 3 cas d'utilisation

Cas d'utilisation = Use Case = UC

Synonyme

- UC = **usage**

Définition théorique

- UC = **fonctionnalité complète** du système (du programme).

Définition du point de vue de l'utilisateur :

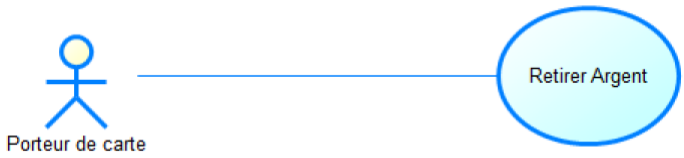
- UC = des actions qui produisent un **résultat intéressant** pour un utilisateur

Définition du point de vue du système :

- UC = des actions qui partent d'un système au repos pour arriver à un système au repos.

Exemple du GAB : retirer de l'argent sur un GAB

- Dans le système « guichet automatique d'une banque », GAB, **« retirer de l'argent » est un UC**.
- C'est une **fonctionnalité complète** du système qui va de l'insertion de la carte de retrait par le client jusqu'à la récupération de la carte de retrait par le client.



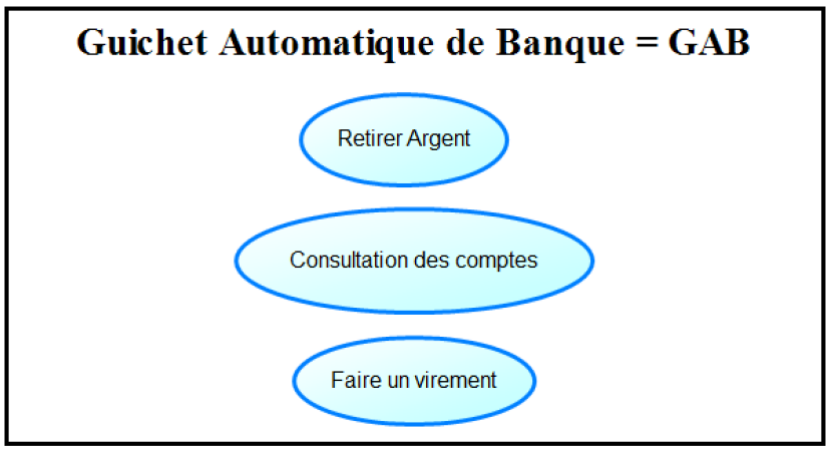
Le système

Définition – représentation UML

- **Le système** c'est **le programme** qu'on réalise. Il est constitué par **la totalité des UC.**
- On le représente par un rectangle qui regroupe les cas d'utilisation.
- Le rectangle matérialise les **frontières du système**.
- **Le système et les UC répondent à la question QUOI ?**

Exemple du GAB :

Le guichet automatique de la banque propose 3 services qui correspondent à 3 UC :



Précisions UML : notion de « classeur » : un rectangle de regroupement

- Un **classeur** est un élément de modélisation qui décrit une **unité comportementale ou structurelle**.
- C'est la **forme la plus simple du regroupement**. Un classeur est représenté par un rectangle.
- Le système complet est un classeur.

Les interfaces du système : interactions avec le monde extérieur

Notion générale d'interface : port de communication

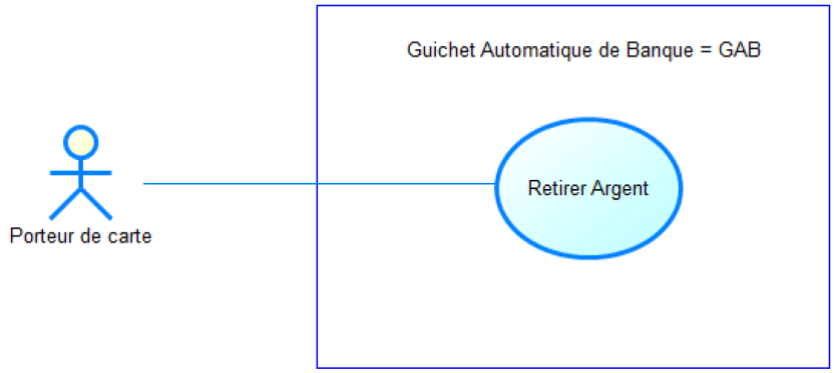
- **Le système** est relié avec le monde extérieur : il **communique avec le monde extérieur**. Il échange des informations, des données, des images, des fichiers, du papier, etc. avec le monde extérieur.
- Il interagit principalement avec les **utilisateurs** = **acteurs**.
- Une interface représente un canal (ou un port) de communication du système avec l'extérieur.
- Une interface est caractérisée par son **protocole de communication**.

Interface proposée : principalement l'IHM

- L'**IHM**, Interface Homme Machine, c'est l'interface avec les utilisateurs. **Le système fournit cette interface** pour communiquer avec les utilisateurs.
- L'interface proposée, ce sont les cas d'utilisation qui la fournisse.
- Les **utilisateurs** sont appelés « **acteurs** ». Les acteurs sont **à l'extérieur** du système. Ils communiquent avec l'interface.
- **Les utilisateurs répondent à la question QUI ?**

➢ ***Exemple du GAB :***

- Le porteur de carte **communique** avec système. Il peut accéder au cas d'utilisation « retirer de l'argent »

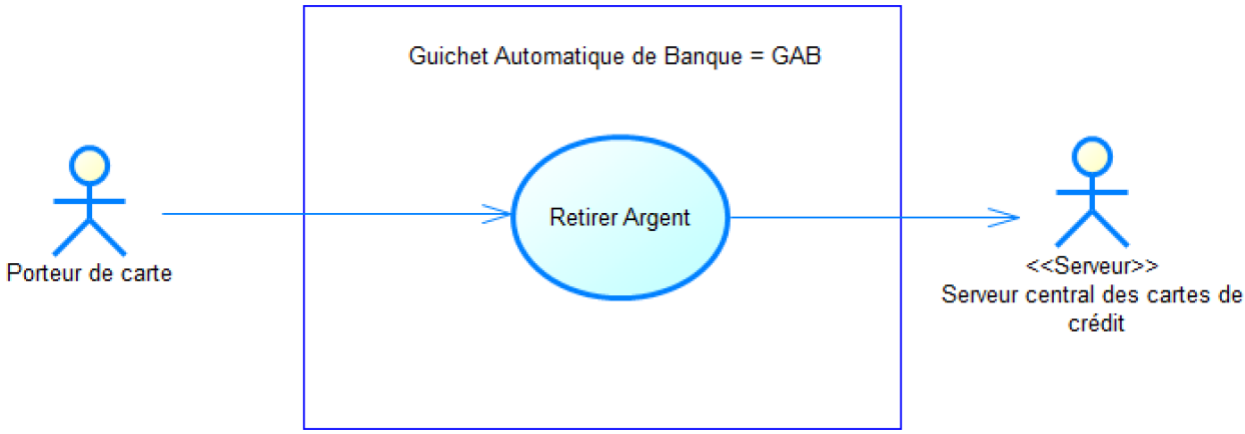


Interface utilisée : acteur passif

- Le système peut utiliser des machines, des logiciels, des services externes à lui pour réaliser ses UC.
- Ces **machines**, **logiciels**, **services externes** sont appelés « **acteur passif** ».
- Le système utilise les interfaces des acteurs passifs qui eux proposent ces interfaces.
- Un acteur passif est sollicité par un UC et communique avec le système mais un acteur passif n'est pas à l'origine du déclenchement du l'UC.
- **L'acteur passif répond à la question : AVEC QUI ? AVEC QUOI ?**

➢ ***Exemple du GAB :***

- Pour retirer de l'argent le système GAB doit communiquer avec le serveur central des cartes de crédit pour vérifier les autorisations.



Suite : Précisions UML : notion de « classeur » : un rectangle de regroupement

➢ ***Liaison fléchée et acteur passif :***

- La relation entre l'UC et un acteur passif est fléchée vers l'acteur passif.
- On peut flécher ou pas la relation entre un acteur actif et un cas d'utilisation.

➢ ***Notion de Stéréotype UML :***

- L'acteur « Serveur central des cartes de crédit » est **stéréotypé « Serveur »**.
- Un stéréotype est un type qu'on peut donner à tous les éléments graphiques UML pour les distinguer les uns des autres.
- Les stéréotypes sont toujours présentés entre guillemets : « Serveur ».

Responsabilités – frontières du système

Principes

- Le responsable du système est responsable du fonctionnement du système et des interfaces avec ceux avec qui ils communiquent : utilisateurs et acteurs passifs.

  ⇒ **Il fournit une interface pour les acteurs actifs** = les utilisateurs. C'est l'IHM.

  ⇒ **Il utilise les interfaces des acteurs passifs** et doit donc connaître le protocole de communication avec ces interfaces.

- Le responsable du système doit prendre en compte tous les cas, dont les problèmes de communication avec les acteurs passifs. Si un acteur passif ne répond pas (un serveur par exemple), il doit le prendre en compte.

Relativité des frontières du système : sous-système et acteur passif

- A l'intérieur d'un système, **un sous-système peut devenir acteur passif** et sortir de la responsabilité du système.

➢ ***Exemple du GAB :***

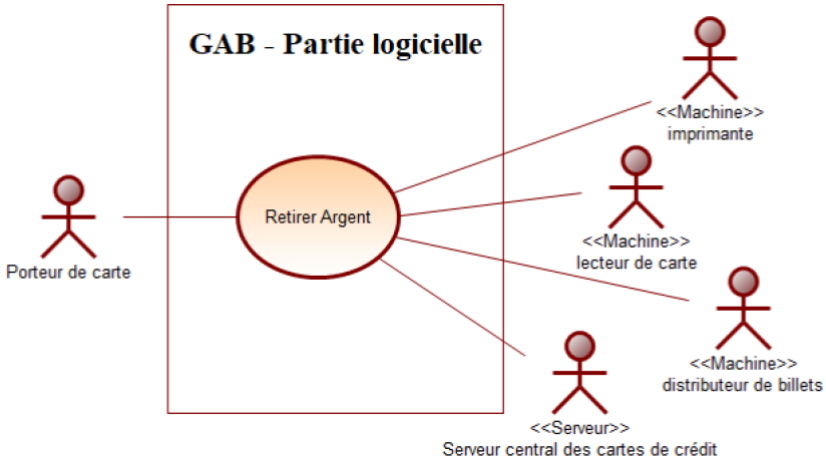
- Pour le développeur de la partie logiciel du GAB (l'interface utilisateur), l'automate de lecture de carte, le distributeur de billets et l'imprimante sont des sous-systèmes.

  ⇒ On peut sortir ces sous-systèmes du GAB et en faire des acteurs passifs.

  ⇒ Dans ce cas, ils ne font plus partie de la responsabilité du développeur de la partie logiciel du GAB.

  ⇒ Le développeur devra connaître les protocoles de communication avec ces acteurs passifs.

➢ ***Exemple : le GAB avec une responsabilité limitée au logiciel et pas aux automates (machines)***



2. Les acteurs

L'acteur est un type

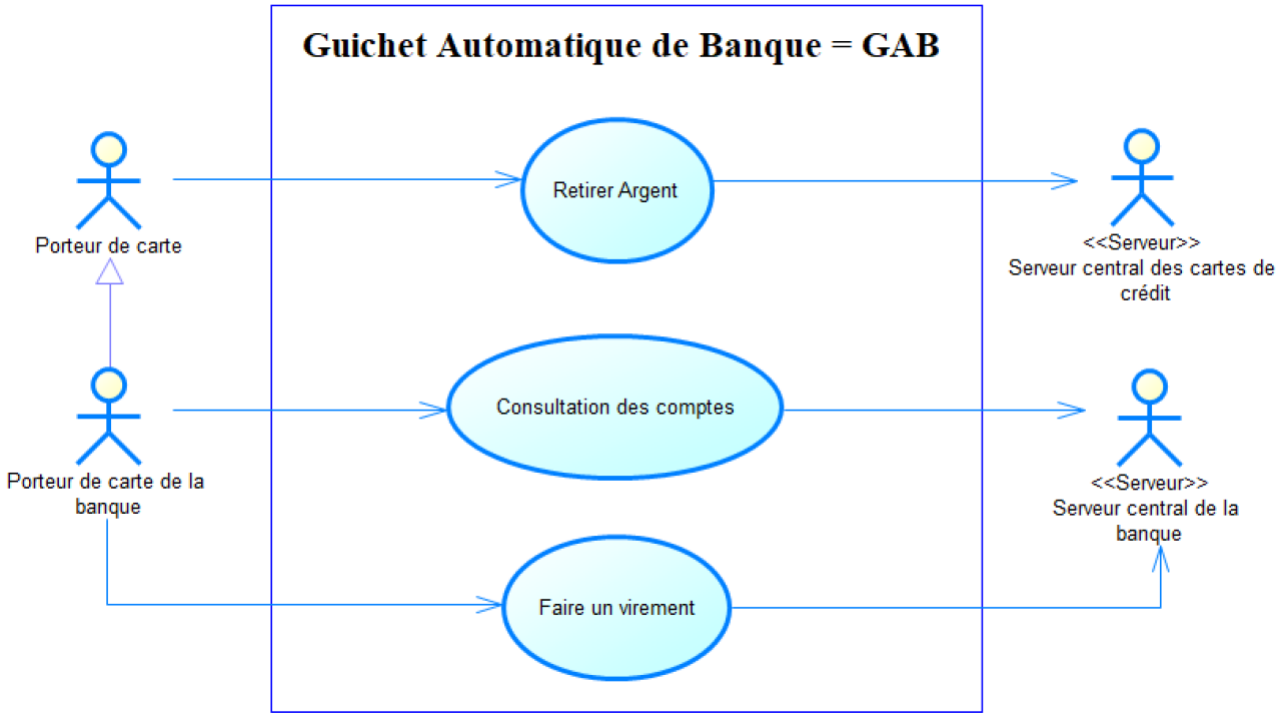
- La notion d'acteur est une abstraction : un type.
- L'acteur « porteur de carte » correspond à l'ensemble de toutes les personnes ayant une carte.
- L'acteur « imprimante » correspond à l'ensemble des machines de type « imprimante ».

Relation entre les acteurs : l'héritage

- Il n'y a qu'une seule relation possible entre les acteurs : **l'héritage.**
- Comme pour tout héritage, c'est une **relation « est un ».**
- L'héritage est une **relation ensembliste d'inclusion** : un ensemble d'acteur peut être inclus dans un autre. L'ensemble inclus est un **acteur-espèce**. L'ensemble incluant est un **acteur-genre**.
- Comme pour tout héritage, l'acteur-espèce hérite des caractéristiques de l'acteur-genre : **l'acteur-espèce hérite de tous les UC de l'acteur-genre**.
- Comme pour tout héritage, ce mécanisme permet de factoriser l'écriture.

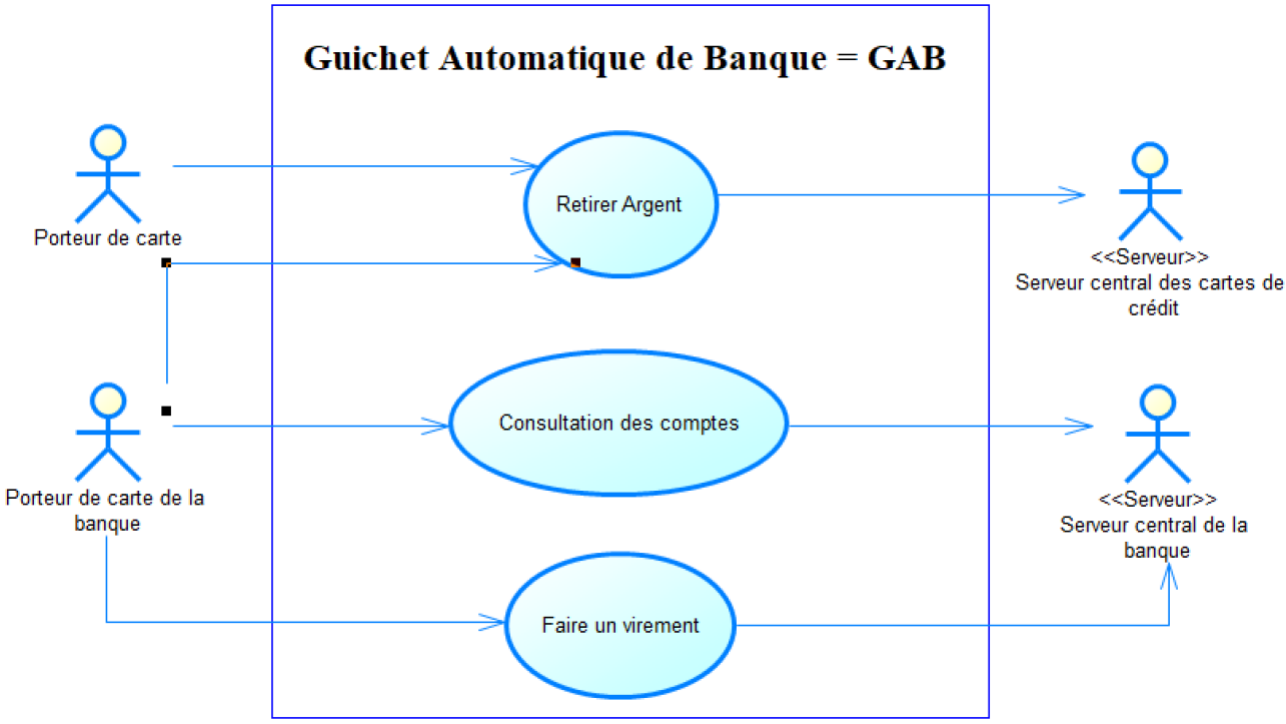
➢ ***Exemple du GAB :***

- Seuls les porteurs de carte de la banque peuvent consulter les comptes et faire un virement. Ils utilisent pour cela le serveur central de la banque.
- Les porteurs de carte de la banque peuvent faire tout ce que fait un simple porteur de carte.



➢ ***Version sans héritage***

- On peut représenter les UC sans héritage.
- Le schéma est moins explicite car on ne voit pas qu'un acteur est une espèce d'un autre.
- Le schéma est plus complexe car l'héritage permet de « factoriser » plusieurs relations entre acteur et UC : ici uniquement la relation du « Porteur de la carte de la banque » vers « Retirer Argent »



4 catégories d'acteurs : principal/secondaire – humain/mécanique – actif/passif – externe/interne

1er distinction : Acteur principal *vs* Acteur secondaire

- **L'acteur principal** : l'utilisateur. Celui pour qui est fait le système.
- **L'acteur secondaire** : l'administrateur du système, etc.

  ⇒ On utilise le **stéréotype UML « Secondaire»** pour faire apparaître les acteurs secondaires.

2ème distinction : Acteur actif *vs* Acteur passif

- **L'acteur actif** est à l'origine du UC. Il utilise le système.
- **L'acteur passif** n'est pas à l'origine du UC. Il est utilisé par le système.

  ⇒ **Les acteurs passifs sont souvent des machines ou des logiciels** (des serveurs).

3ème distinction : Acteur humain *vs* Acteur mécanique

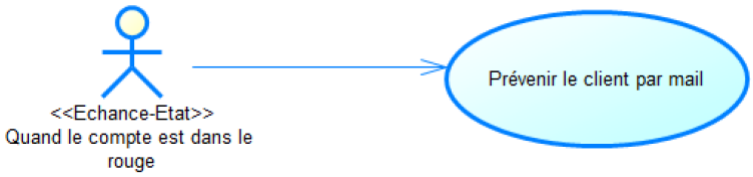
- **Les acteurs humains** sont les fonctions des personnes (le bibliothécaire, le client, l'administrateur) ou les services (la comptabilité).
- **Les acteurs mécaniques** peuvent être **matériels** (des périphériques), **logiciels** (un serveur, un autre système), ou **temporels** (une échéance).

  ⇒ On utilise les stéréotype UML pour faire apparaître les acteurs mécanique : « Serveur» ou « Machine », etc.

4ème distinction : Acteur externe *vs* Acteur interne

- **Les acteurs externes** sont les acteurs « normaux » **:** à l'extérieur du système.
- **Les acteurs internes sont internes au système.** Ils permettent de montrer une situation technique particulière :
  1. soit une **échéance interne** gérée par le système (tous les jours à minuit, le système fait telle action),
  2. soit à un **état particulier du système** gérée par le système (dans telle situation particulière, le système fait telle action).

⇒ On utilise les stéréotype UML « Echéance » ou « Etat » pour faire apparaître les acteurs interne.



➢ ***Remarque***

- Les acteurs internes n'existent pas vraiment. Ils sont utiles pour clarifier les usages. Réfléchir sur les échéances ou des actions déclencheuses d'actions est une bonne pratique.

3. Premiers série d'exercices : bases

Les premiers exercices peuvent être très simples ou complexes.

Leur principe est qu'ils n'utilisent que les notions abordées dans le chapitre sur les bases et n'utilisent pas les relations entre UC (héritage, include et extend) ni la notion de sous-système.

L'objectif est de bien comprendre les bases.

Les sujets d'exercices sont dans le document :

➔ UML-03_TP_UC_SeqSys_Acti_Etats.pdf

On traitera les exercices suivants en se contentant de faire une analyse sans héritage, ni include, ni extends.

1.1 Compteur
1.2 TV
1.4 Enchère
2.2 Médiathèque
2.3 Le portail

UC – RELATIONS ENTRE UC

4. Relations entre UC : héritage et composition

Présentation

- On va présenter maintenant les **relations entre les UC** : **héritage** et **composition**.
- **Héritage** et **composition** sont **2 grands types de relation** entre des éléments de même nature.

  ⇒ **L'héritage**, déjà vu avec les acteurs, traduit une **relation « est un »** et une inclusion ensembliste. L'ensemble des chats est inclus dans l'ensemble des animaux. Félix le chat est un chat, Félix est un animal, tous les chats sont des animaux.

  ⇒ **La composition** consiste à diviser un objet en morceaux (à couper un saucisson en tranches). La relation de composition traduit une relation **« est un composant de »** dans un sens et **« est composé de »** dans l'autre sens.

Distinction

- **Les UC traduisent objectivement le cahier des charges**.
- **Les relations entre UC sont subjectives**.
  - **L'héritage organisent des regroupement d'UC** qui permettent d'avoir une lecture synthétique. On peut choisir plusieurs formes d'organisation.
  - **La décomposition permet de zoomer** sur un UC et présentant certain de ses composants. On peut choisir de zoomer sur ce qu'on veut.

5. L'héritage entre UC : technique de regroupement

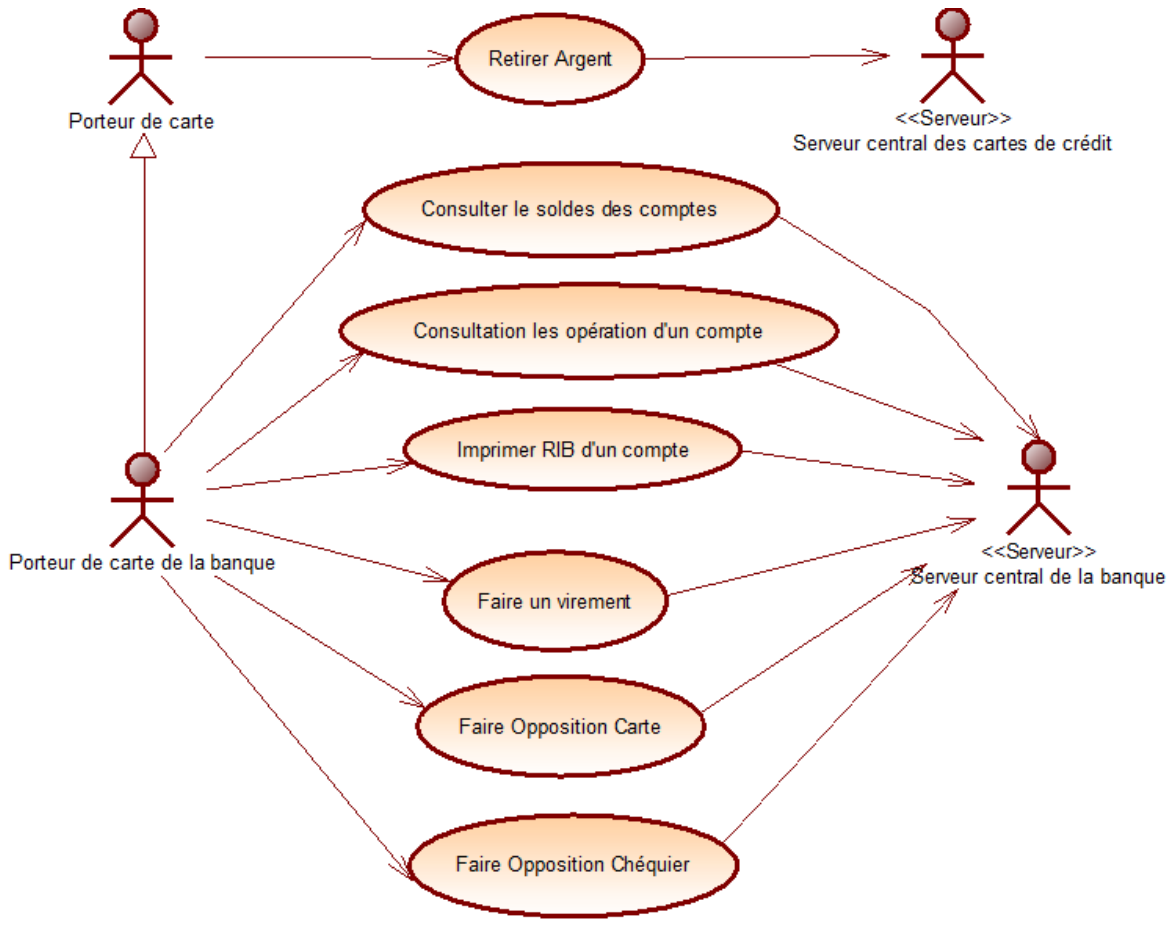
Principes

- Dès qu'on a plus de 3 ou 4 UC, on a intérêt à **réfléchir à des regroupements** pour clarifier la présentation.
- C'est une **logique de « menu déroulant »** : on met les UC qui vont ensemble dans un UC qui les regroupe et qui correspond à un menu déroulant.
- **Ca rend le modèle plus clair**, d'autant que ça permet aussi de regrouper de lien « acteur – cas d'utilisation » et aussi des liens de composition entre cas d'utilisation (include et extend).
- Attention, **les UC sont « objectifs »** : ils traduisent le cahier des charges. **Les regroupements sont toujours « subjectifs »** : ils traduisent une façon de voir les choses.
- Les regroupements doivent essayer d'être **proches de ce que sera l'interface utilisateur** et/ou le manuel utilisateur.

Exemple : le GAB

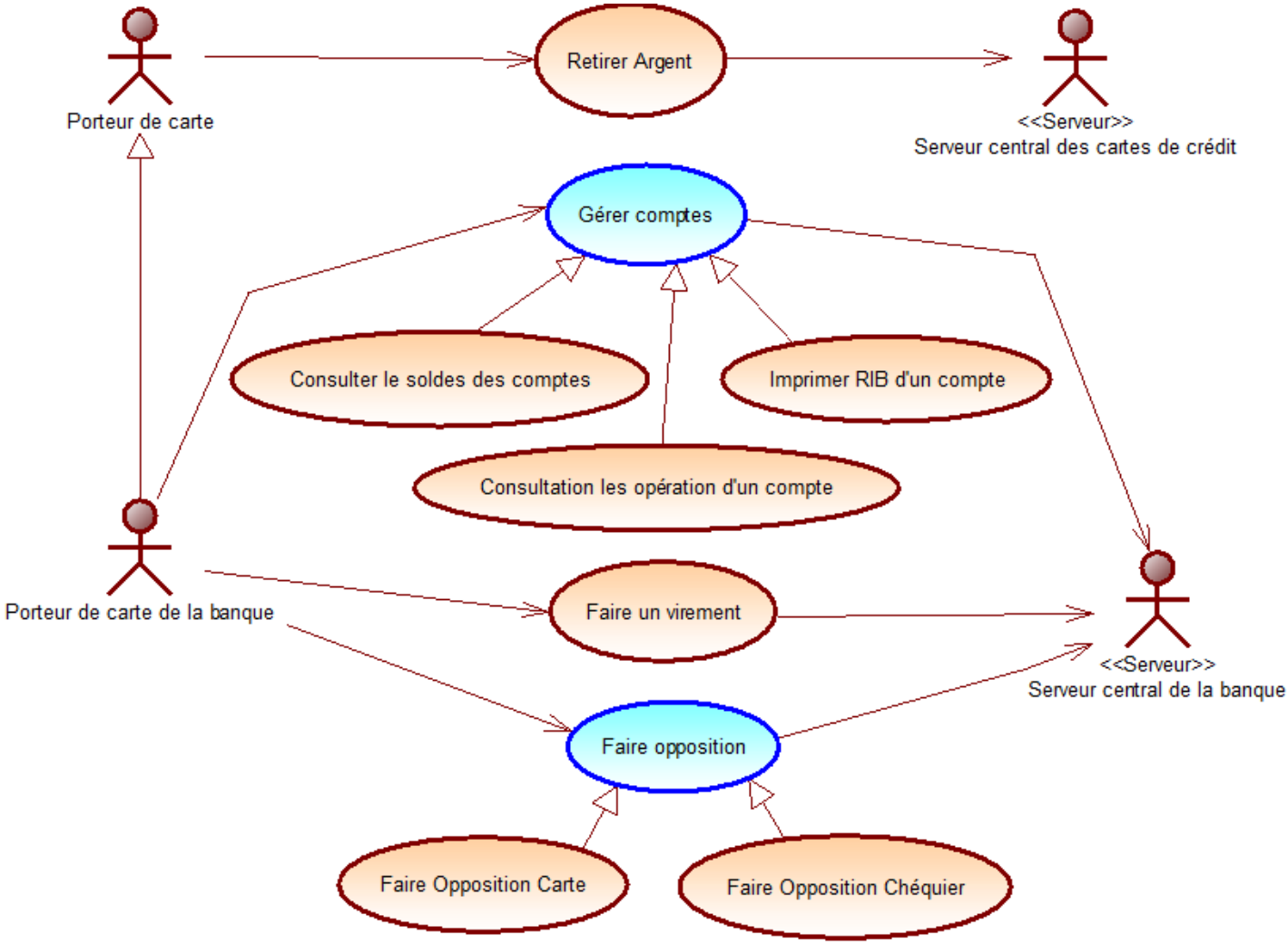
- On repart du GAB déjà analysé.
- A la place de « consulter les comptes », on précise :
  - consulter le soldes des comptes,
  - consulter les opérations pour chaque compte,
  - imprimer un RIB pour chaque compte.
- On ajoute :
  - Faire opposition sur une carte.
  - Faire opposition sur un chéquier.
- Le diagramme des cas d'utilisation est le suivant, avec 7 cas d'utilisation

Version sans généralisation : 7 UC accédés directement par les utilisateurs



Version avec généralisation : 4 UC accédés directement par les utilisateurs

Les UC « Gérer comptes » et « Faire opposition » permettent de faire des regroupements.



Distinction entre UC abstraits et UC concrets

Définition

- On peut distinguer entre UC abstrait et UC concret.
- Un **UC abstrait**, c'est un **UC qui regroupe** des UC concrets et/ou des UC abstraits. C'est **l'équivalent d'un dossier** dans une arborescence de fichiers.
- Un **UC concret**, c'est un **« vrai UC »** : celui que l'utilisateur vient exécuter.

Exemple

- Il y **a 7 UC concrets** dans notre exemple : les UC avant le regroupement. Ils correspondent à un usage concret du logiciel.
- Il y a **2 UC abstraits** dans notre exemple : les 2 UC de regroupement. En bleu sur le schéma.

Remarque

- Cette distinction n'est pas « standard ».

Distinction entre UC général (ou direct) et UC dérivé

Définition

- On peut distinguer entre UC direct et UC dérivé.
- Un **UC généraux (de premier niveau)**, c'est un **UC accessible directement par l'utilisateur**. Ça peut être un UC concret ou un UC abstrait. On peut aussi dire : UC « direct ».
- Un **UC dérivé (de niveau inférieur)**, c'est un **UC qui hérite** d'un UC général ou d'un UC dérivé.

Exemple

- Il y **a 4 UC généraux** dans notre exemple : la généralisation des UC a permis de limiter le nombre d'UC généraux et rend ainsi le schéma plus synthétique.
- Il y a **5 UC dérivés** dans notre exemple.

Méthodologie

- L'utilisation de l'héritage entre UC permet de réduire le nombre d'UC généraux et donc de rendre le diagramme plus lisible.

Remarque

- Cette distinction n'est pas « standard ».

6. Composant d'un UC : include et extend, techniques pour préciser

Principes

- Un UC, c'est une succession d'activités élémentaires qui se déroulent dans le temps.
- Le porteur de carte vient pour retirer de l'argent :

  ⇒ il rentre sa carte,

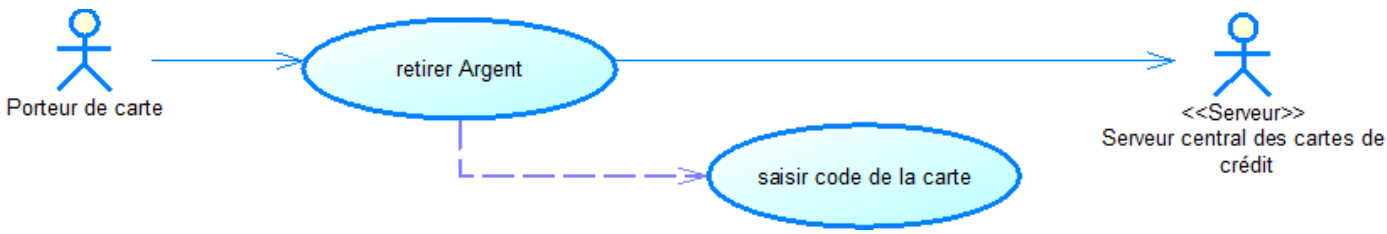
  ⇒ saisit son code,

  ⇒ saisit un montant,

  ⇒ récupère sa carte et

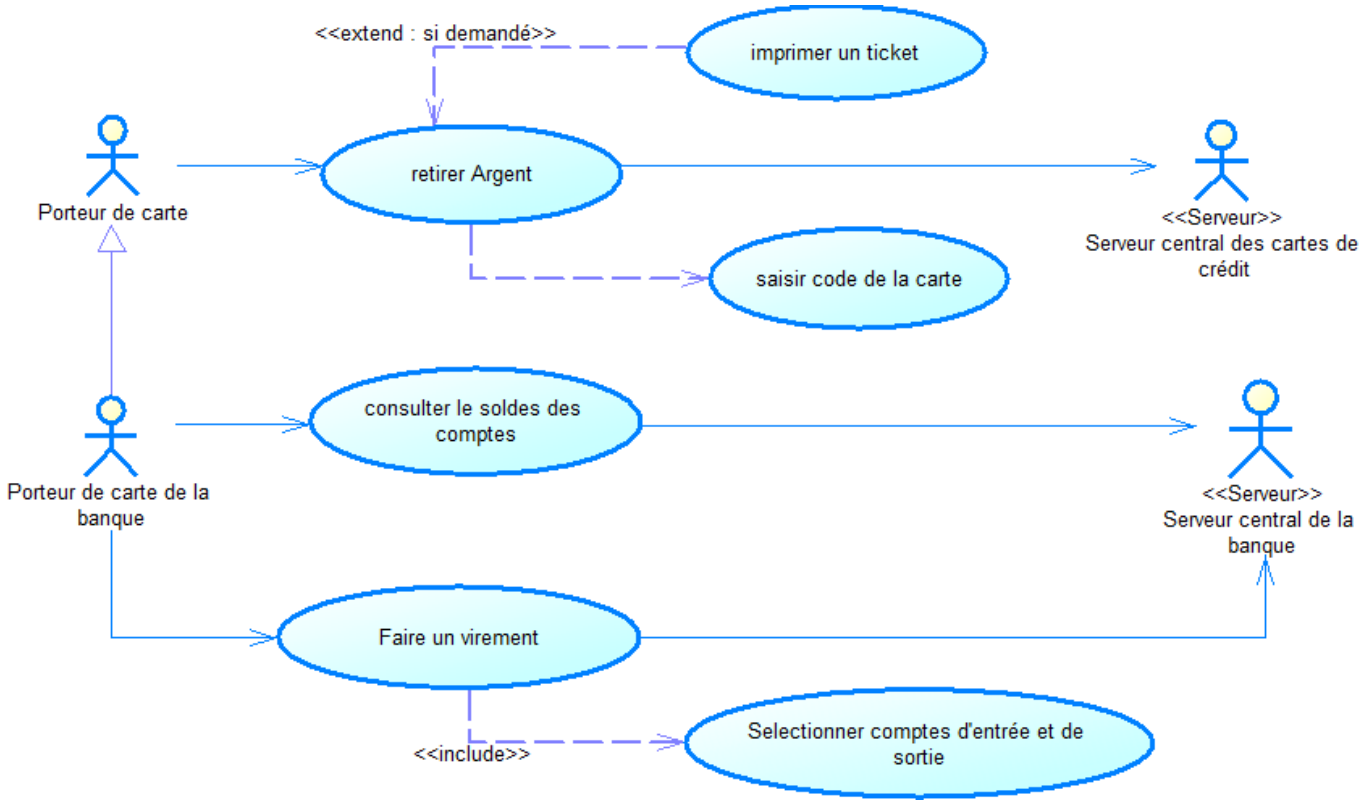
  ⇒ récupère son argent.
- **Les étapes décomposent l'UC : le coupe en tranches (= partie = composant)**
- On peut présenter le ou les composants qu'on souhaite.
- Le but est de clarifier le diagramme.

  Chaque « tranche » (ou partie ou composant) d'un UC est représentée comme un UC et relié à son UC complet (le composé) par une relation d'include ou d'extend. Les include et les extend apportent des précisions sur le contenu d'un UC.
- Exemple : l'UC « retirer Argent » inclut l'UC « saisir code de la carte ».



Exemple du GAB

Diagramme de Cas d'utilisation :



2 include

- Retirer de l'argent inclus forcément : saisir le code de la carte.
- Faire un virement inclus la sélection des comptes d'entrée et de sortie.

1 extend

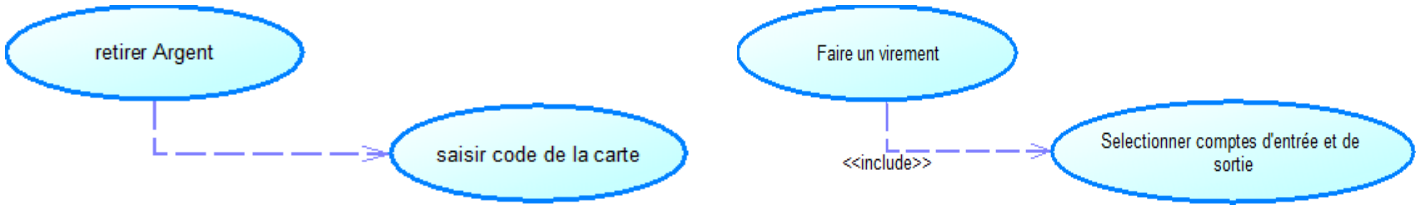
- Retirer de l'argent inclus l'impression du ticket **à condition qu'on l'ait demandé**.

Distinction « include » et « extend »

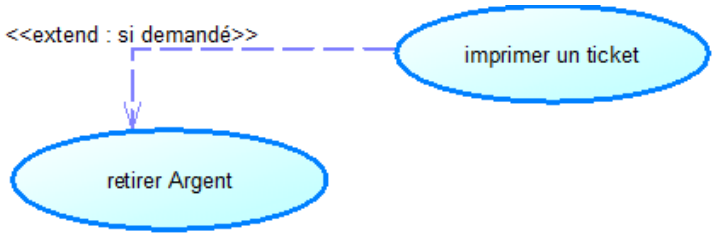
- Quand on réalise l'UC complet (par exemple quand on retire de l'argent) :
  - Un **include** est un composant qui **se réalise forcément** (par exemple : saisir le code de la carte).
  - Un **extend** est un composant qui **se réalise si une condition est vérifiée** (par exemple : imprimer un ticket). En général, la condition est : « si demandé ».

Notation UML :

- Le lien de composition entre 2 UC, lien en pointillé, est fléché.
- **Include** : L'UC de départ est le composé, l'UC d'arrivé est le composant. S'il n'y a pas de stéréotype sur un lien de composition entre 2 UC, c'est forcément un include. On peut préciser le stéréotype sur l'include



- **Extend** : L'UC de départ est le composant, l'UC d'arrivé est le composé. Attention ! C'est l'inverse de l'include. On est obligé d'afficher le stéréotype « extend » en précisant la condition (souvent simplement « si demandé »).



Précisions méthodologiques

Niveau de l'analyse

- La recherche des composants relève de la 4ème distinction parmi les 5 distinctions capitales.

  ⇒ Il s'agit d'une analyse détaillée.

Quand mettre des include et des extend

- Il faut éviter de mettre trop d'include ou d'extend pour éviter d'alourdir inutilement le diagramme des UC.
- On met des « include » ou des « extend » dans 3 cas :
  1. Quand on pense que cela apporte quelque chose à la compréhension du diagramme
  2. Quand l'UC inclus est partagé par plusieurs UC.
  3. Quand l'UC inclus est aussi un UC pour un acteur passif

7. Deuxième série d'exercices : héritage, include et extend

La deuxième série d'exercices met en œuvre les notions d'héritage, d'include et d'extend.
Elle reprend les exercices déjà abordés en première partie et en ajoute d'autres.
L'objectif est de bien comprendre les notions d'héritage, d'include et d'extend.

Les sujets d'exercices sont dans le document :

➔ UML-03_TP_UC_SeqSys_Acti_Etats.pdf

On traitera les exercices suivants en utilisant judicieusement des héritages, des include et extends.

1.1 Compteur (suite)
1.2 TV (suite)
1.3 Agence de voyage
1.4 Enchère (suite)
2.2 Médiathèque (suite)
2.3 Le portail (suite)

UC - ARCHITECTURES

1. Niveau de présentation des UC

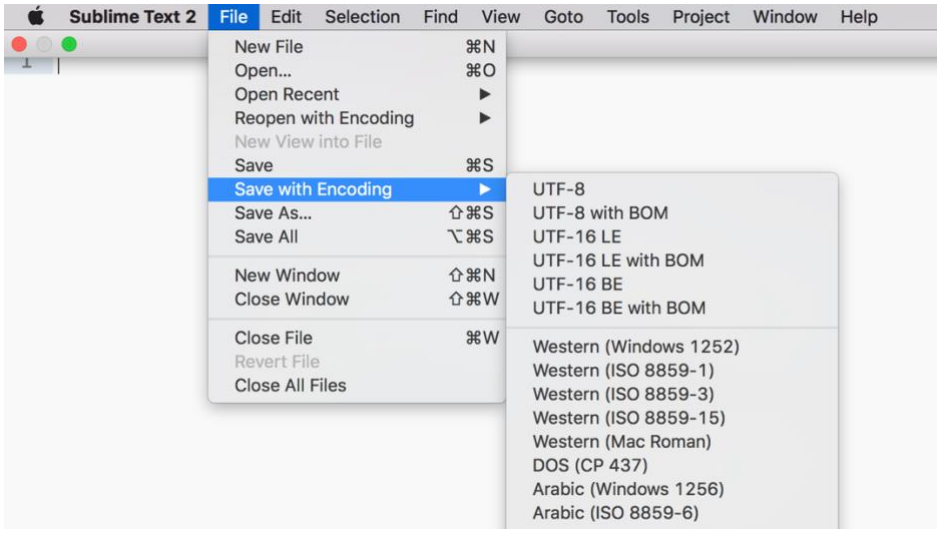
Niveaux de présentation des UC

Principes

- On peut proposer un seul diagramme avec tous les UC : s'il y en a trop, il risque d'être très embrouillé.
- On aura donc intérêt à présenter plusieurs diagrammes d'utilisation par niveau d'abstraction descendant.

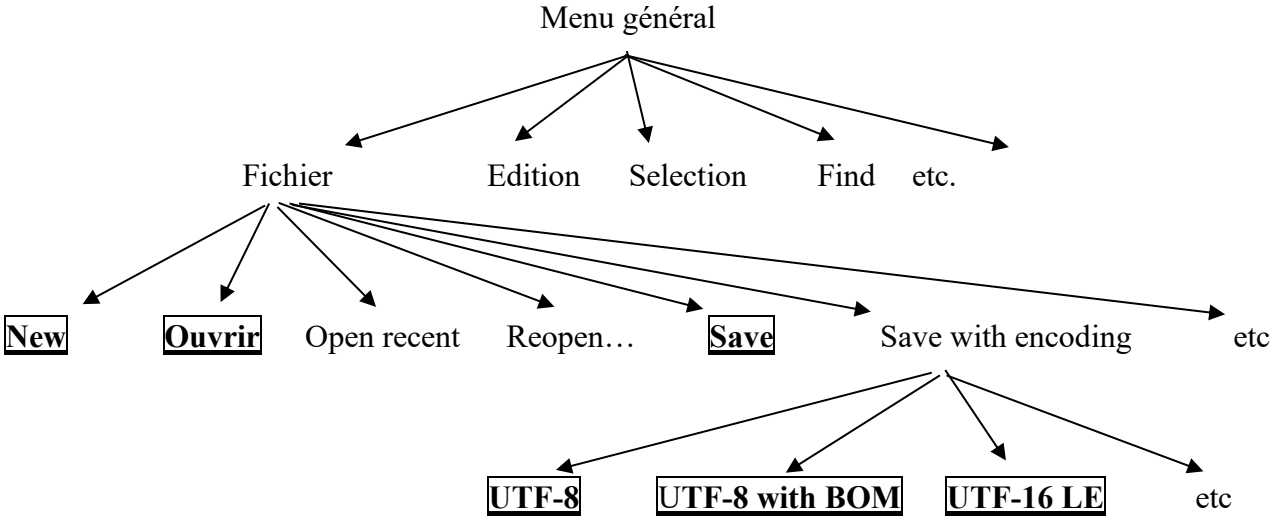
Exemple : l'arborescence des menus de Sublime Text

L'éditeur SublimeText et décrivons les UC.



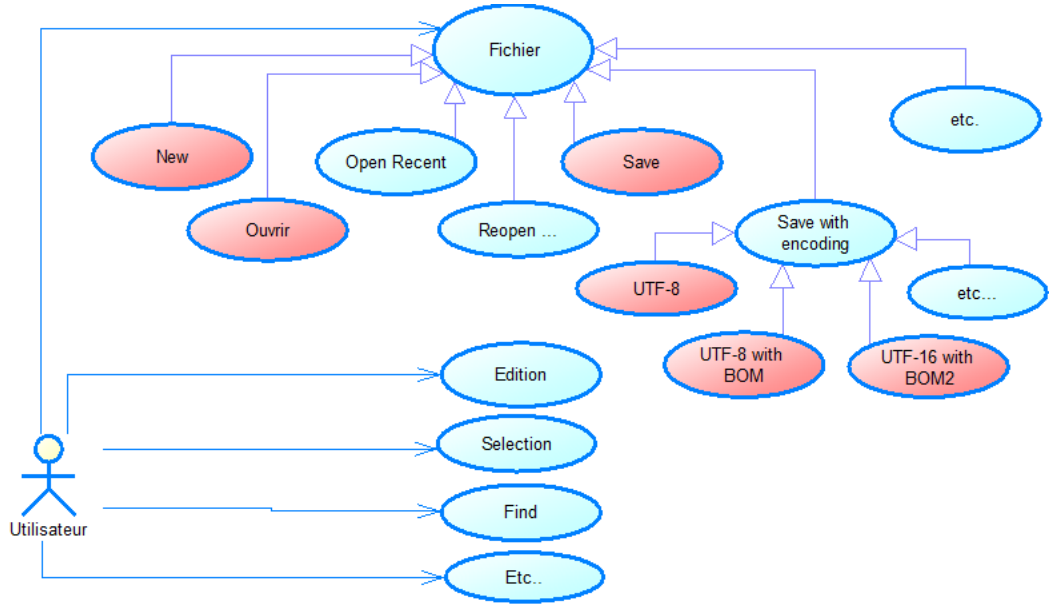
- Les UC sont donnés par les menus.
- Les **UC abstraits** sont les menus qui contiennent des sous-menus.
- Les **UC concrets** sont les menus qui conduisent à une activité.

Arborescence des menus



- Les **UC concrets** correspondent aux **feuilles de l'arbre**. Ils sont soulignés, gras, encadrés.
- Les **UC abstraits** correspondent aux **nœuds non-feuilles**.

UC des menus



➢ ***UC concrets***

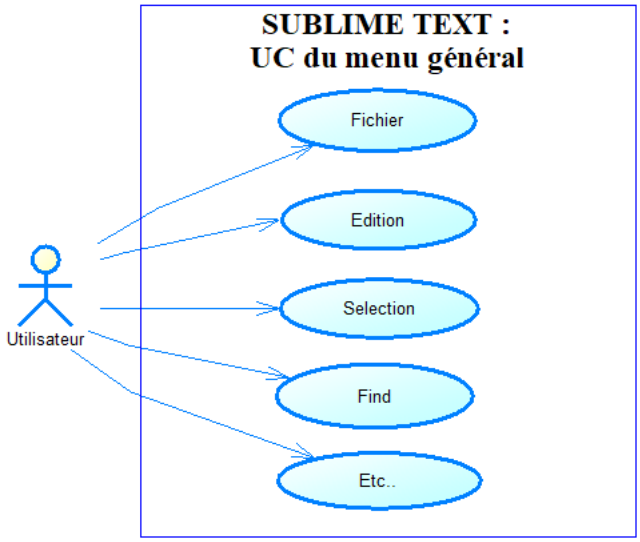
- Il y a 6 UC concrets : « New», « Ouvrir », « Sauver », « UTF-8 », « UTF-8 whith BOM», et « UTF-8 whith BOM». Ils sont présentés en rouge. Ce n'est pas un standard UML.
- Les UC concrets correspondent à un usage concret du logiciel.

➢ ***UC abstraits***

- Tous les autres UC sont abstraits : soit ils regroupent des UC concrets, soit on n'a pas présenter le détail et les UCs concrets qu'ils regroupent.

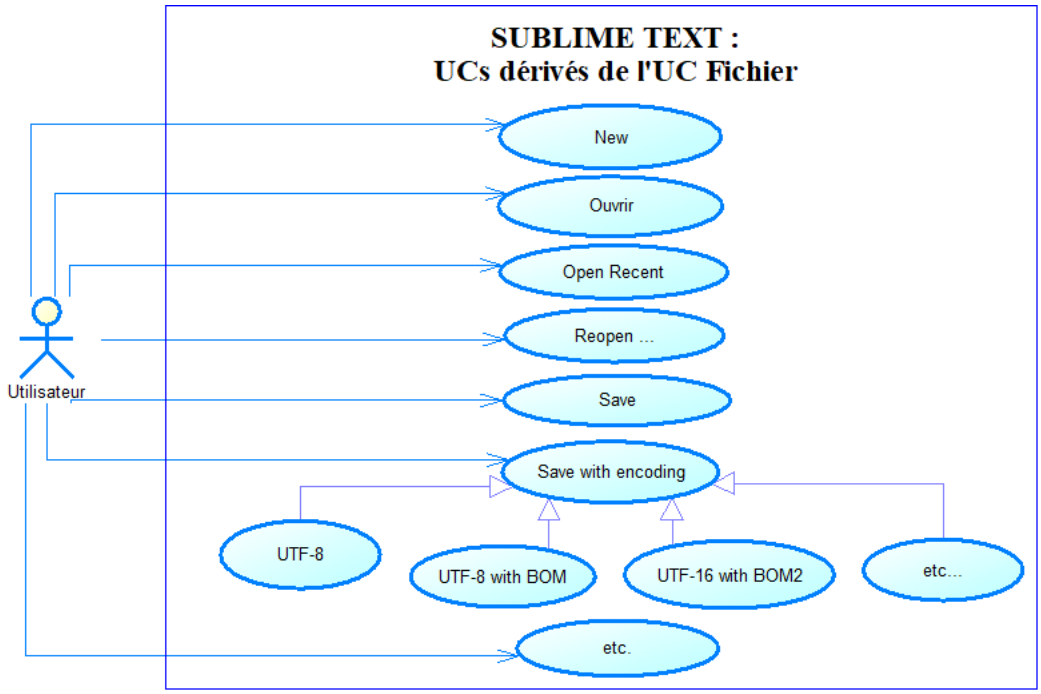
UC du menu général : que les UC « généraux »

- On précise dans le système (le rectangle), ce qu'on décrit.



Les UC de l'UC fichier

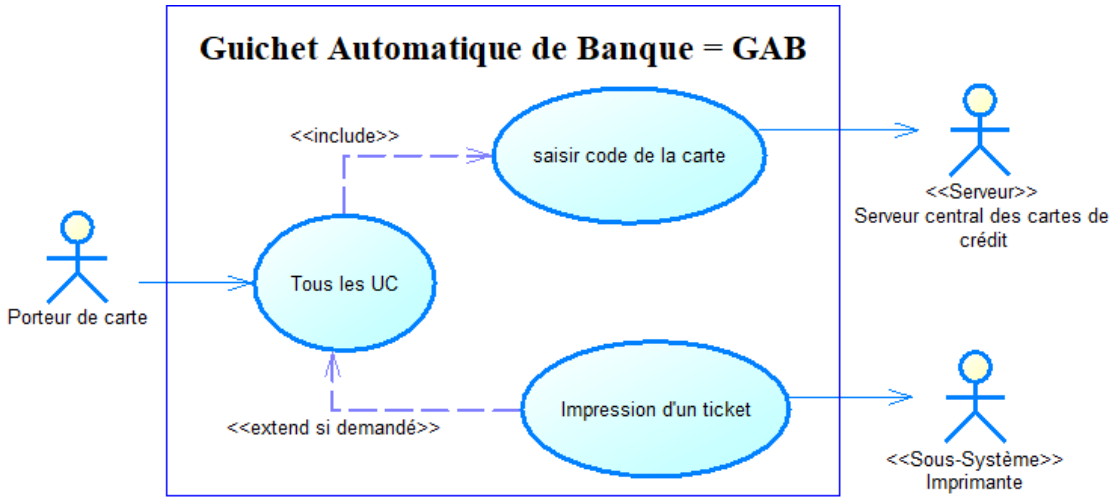
- On peut ensuite présenter les UCs dérivé pour chaque UC général du menu général.
- On précise dans le système (le rectangle), ce qu'on décrit : ici, les UCs dérivés de l'UC Fichier.



UC du menu Gestion de fichier

2. Présenter des « include » ou des « extend » communs

- Dans le cas du GAB, tous les UC nécessitent de commencer par une identification.
- De plus, tous les UC permettent de demander une impression de ticket d'information.
- On va donc avoir un include de la saisie du code de la carte et un extend d'impression de ticket pour tous les UC généraux (=directs).
- Pour alléger le diagramme d'UC, on peut présenter le diagramme général suivant :



- l'acteur Imprimante est stéréotypé « sous-système ».
  - Un sous-système est une partie du système. Ici l'imprimante fait partie du système GAB.
  - On le présente pour signaler une communication particulière dans le système GAB : celle qui passera avec le sous-système Imprimante et son protocole de communication.

3. Architecture fonctionnelle et diagramme d'UC

Architecture fonctionnelle

Composant fonctionnel = poste de travail = PT

- Un poste de travail est un système (une machine ou un humain) où des UC sont accessibles aux utilisateurs.
- Par exemple : dans une bibliothèque, on peut accéder à des services auprès du bibliothécaires, sur des postes dédiées à des recherches, sur des postes automatiques dédiés à des emprunts ou des retours.
- Chaque **poste de travail** est un **composant fonctionnel** du système complet, autrement dit un **sous-système fonctionnel** : un sous-système fournissant des fonctionnalités pour un utilisateur externe.
- L'architecture fonctionnelle définit les postes de travail = PT du système.
- L'architecture fonctionnelle ne concerne pas les composants internes au système (les serveurs, les API, etc.).
- L'architecture fonctionnelle ne concerne pas non plus les interfaces avec des machines ou logiciels externes dont le système a besoin pour fonctionner : ce sont des acteurs passifs.

Relation plusieurs à plusieurs entre UC et PT

- Un poste de travail propose en général plusieurs UC. Un UC peut être proposé par plusieurs postes de travail.
- Par exemple : l'UC « rendre un livre » existe au niveau du poste automatique de retour des livres et au niveau du poste de travail du bibliothécaire.

Analyse des sous-systèmes fonctionnels

- Dans l'analyse fonctionnelle, on peut faire un diagramme de cas d'utilisation qui considère le système comme un tout.
- On pourra aussi faire une analyse par sous-système fonctionnel.
- Les sous-système fonctionnel peuvent partager des cas d'utilisation. Techniquement, ils peuvent aussi partager des composants techniques (comme le serveur de BD par exemple).

Traduction UML

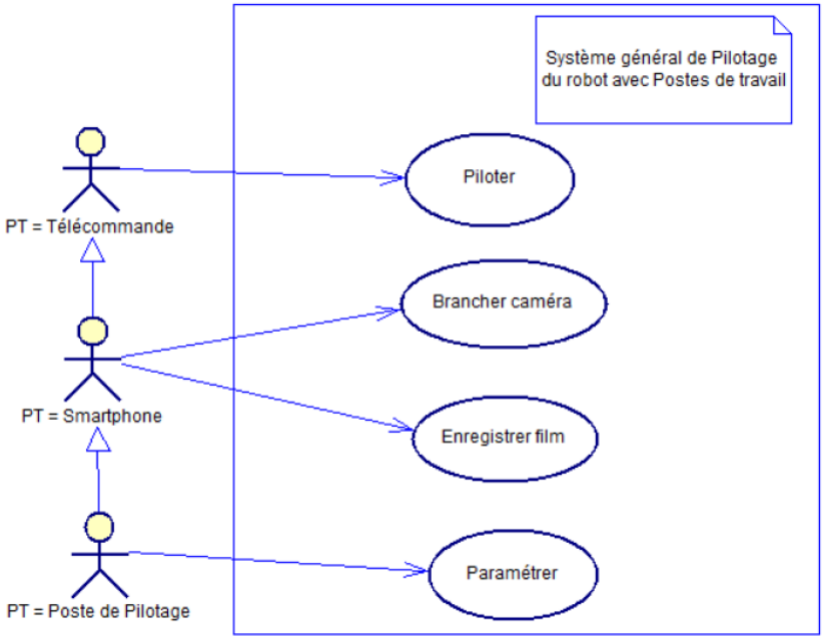
Deux solutions pour présenter les postes de travail en UML.

- **1) Un acteur correspond à un PT** et éventuellement des héritages entre PT.
- **2) Un diagramme d'UC par PT**.

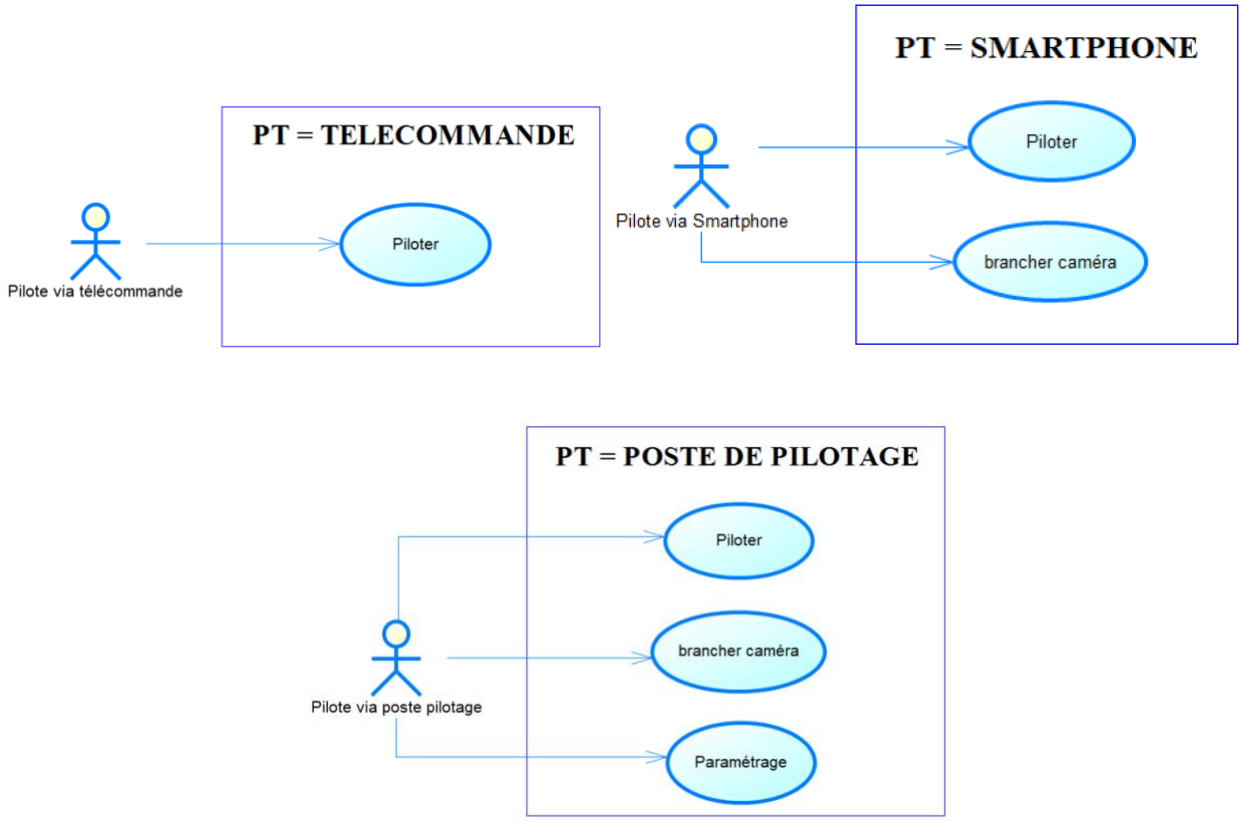
Exemple

- On pilote un robot :
  - soit « en visuel » avec une télécommande,
  - soit « à distance » via un PC ou un smartphone. Le pilotage à distance permet de brancher la caméra qui renvoie l'environnement du robot.
  - soit via le PC qui permet de faire en plus des réglages spécifiques du robot.
- On peut considérer qu'on a 3 postes de travail : la télécommande, le smartphone et le PC.

➢ ***Version 1 : un acteur par PT avec héritage entre acteur***



➢ ***Version 2 : 3 diagrammes d'UC : 1 par PT :***



4. Troisième série d'exercices : poste de travail

La troisième série d'exercices met en œuvre la notion de poste de travail (=PT = sous-système fonctionnel).

Elle reprend certains exercices déjà abordés en deuxième partie et en ajoute d'autres.

L'objectif est de bien comprendre la notion de sous-système fonctionnel = PT.

Les sujets d'exercices sont dans le document :

➔ UML-03_TP_UC_SeqSys_Acti_Etats.pdf

On traitera :

2.1 Le robot

2.2 Médiathèque (suite)

2.3 Le portail (suite)

UC – CONCLUSION

1. Méthodes de construction d'un diagramme des UC

Recherche des acteurs

- Est acteur du système tout ce qui est à l'extérieur du système et qui interagit avec le système.
- Les principaux acteurs sont **les utilisateurs** du système.
- Mais il ne faut pas oublier **ceux qui l'administrent** d'une façon ou d'une autre.
- Ni **les acteurs mécaniques** :
  - **Les périphériques** manipulés par le système (imprimantes, système de télétransmission, etc.)
  - **Les logiciels** interfacés ou intégrés au système (serveur, etc.)

Recherche des événements

- L'analyse des événements est une technique permettant la mise au jour des cas d'utilisation et des acteurs.
  - **Evénement externe** : ceux qui sont initiés par un acteur
  - **Evénement temporel** : un moment externe particulier qui déclenche une action du système.
  - **Evénement état** : un état interne particulier qui déclenche une action du système.

Garantir le caractère concret d'un UC

- Pour s'assurer qu'un UC est concret, on essaye de le décomposer avec des « include » ou des « extend ».

Approche par le concret et généralisation : de l'UC concret à l'UC abstrait

- On cherche les UC concrets les plus évidents et on cherche de quel UC abstrait on peut le faire dériver.
- Ensuite on réfléchit à d'autre UC concret dérivant de l'UC abstrait trouvé.

Approche par abstraction : de l'UC abstrait à l'UC concret

- On cherche d'abord les UC les plus généraux (abstrait).
- Ensuite on cherche des UC dérivés concrets.

Approche par les données

- Si on connaît les données manipulées par le système (la BD), on peut concevoir les UC qui permettent de manipuler ces données.
- L'analyse du cycle de vie des données permet de définir des UC : comment créer la donnée, comment la modifier, comment la supprimer ?
- L'analyse plus précise des caractères obligatoire et modifiable des attributs permet d'affiner encore la recherche d'UC concrets.

Approche par le diagramme des flux MERISE

Pour déterminer les UC et les acteurs externes, on peut reprendre la technique de l'analyse des flux de la méthode MERISE.

Cela correspond aux diagrammes organisationnels des flux, diagrammes de contexte et diagrammes d'activité à travée.

2. Compléments, limites et alternatives du diagramme des cas d'utilisation

Diagramme de cas d'utilisation

Le diagramme des cas d'utilisation est un diagramme efficace permettant de représenter clairement les usages et les acteurs.

Il permet de choisir le niveau d'abstraction en présentant ou pas les cas d'utilisation dérivés.

Il permet aussi de présenter les sous-systèmes et les interactions entre les sous-systèmes.

C'est donc un diagramme qu'on a intérêt à utiliser.

Alternative textuelle

Le diagramme des cas d'utilisation et les diagrammes de séquence peuvent être remplacés par une analyse purement textuelle.

En général, on conserve au moins l'analyse des cas d'utilisation sous forme graphique pour faciliter la communication avec le maître d'ouvrage (le client, l'utilisateur final).

Complément 1 : déroulement des cas d'utilisation

Le diagramme des cas d'utilisation est complété par un ensemble de diagrammes de séquence système qui permettent de détailler le déroulement effectif d'un scénario.

A ces diagrammes de séquence on peut aussi associer des diagrammes d'activité.

On peut aussi y ajouter des diagrammes d'états-transition pour modéliser l'évolution de certains paramètres.

Complément 2 : maquettage de l'IHM

L'aspect visuel de l'utilisation n'est pas décrit par les diagrammes UML.

Un maquettage des écrans prévus permet de clarifier l'usage réel du logiciel.

Complément 3 : méthode agile et incrément

Dans une méthode itérative et incrémentale de type « agile », on peut réaliser un premier incrément du logiciel mettant œuvres quelques UC seulement.

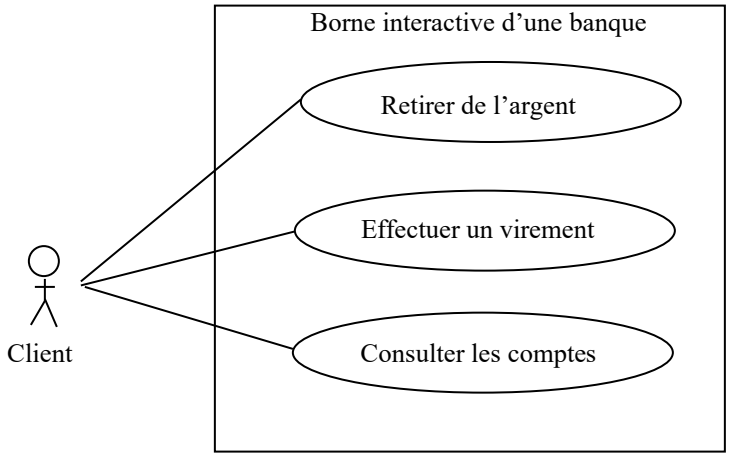
Ca permet de se donner une idée précise de l'aspect du résultat final.

Ca permet au client de mieux réfléchir à ses besoins.

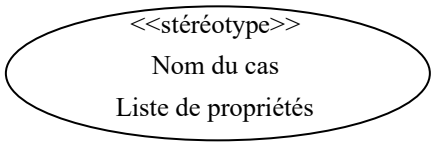
3. Syntaxe UML des Use Case

Description des UC

Exemple



UC



Stéréotype

Un stéréotype définit un type particulier pour un élément de modélisation.

Propriétés

Les propriétés peuvent être ajoutées. Elles peuvent définir par exemples des conditions d'utilisation. C'est peu utilisé.

Le classeur

Un classeur est un élément de modélisation qui décrit une unité comportementale ou structurelle.

C'est la forme la plus simple du regroupement.

Un classeur est représenté par un rectangle.

Le système complet est un classeur.

Un menu peut être regroupé dans un classeur : menu fichier, menu édition, etc.

L'acteur

<<stéréotype>>

Nom de l'acteur

L'association entre acteur et UC

L'association est un lien entre un acteur et un UC.

Par défaut, l'association n'est pas orientée : cela signifie que la communication se fait dans les deux sens.

En orientant l'associant dans un sens, on signifie la priorité d'un sens de communication sur un autre. Ce n'est qu'une priorité. Elle n'exclut pas la communication dans l'autre sens.

- ***Usages***

Pour les acteurs actifs, on laisse une association non orientée.

Pour les acteurs passifs, on oriente l'association vers l'acteur.

Les 3 relations entre les UC

3 relations possibles entre les UC :

- généralisation (ou héritage)
- inclusion
- extension

Inclusion et extension sont deux formes de la relation de composition.

La généralisation

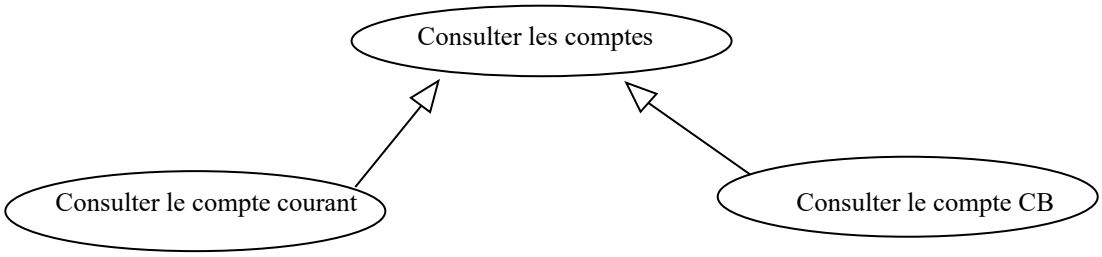
Principe

Un UC est un « film » qui peut se dérouler avec un début et une fin.

Quand on a un héritage, cela veut dire que l'UC enfant déroule le « film » parent en plus de son propre film.

Exemple et formalisme

Ce formalisme vaudra pour toutes les relations de généralisation / spécialisation.



- ***Explications***

Le UC « consulter le compte courant » est une espèce du UC « consulter les comptes ».

Quand on consulte le compte courant, on sélectionne le compte, il s'affiche et on a des usages possibles.

C'est pareil avec consulter le compte CB.

L'inclusion

Le UC de départ inclut le comportement d'un autre UC : l'UC inclus.

L'UC inclus est une étape obligée de l'UC de départ.

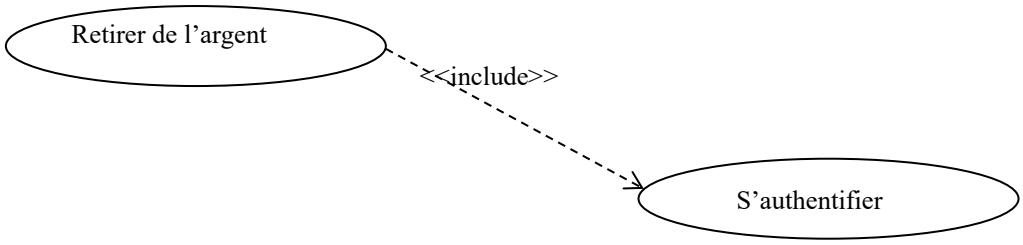
La flèche va de l'UC de départ à l'UC inclus.

Par exemple

Le UC « Retirer de l'argent» inclut le UC « s'authentifier ».

« S'authentifier » est une étape obligée de « retirer de l'argent ».

Formalisme



L'extension

Une extension c'est une inclusion sous condition.

Le UC extension ajoute son comportement à l'UC de départ sous certaines conditions.

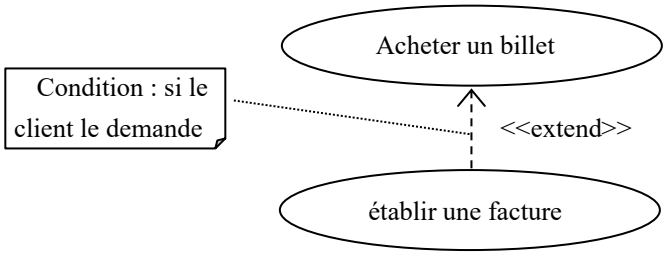
La flèche va de l'UC extension à l'UC de départ (c'est l'inverse de l'inclusion).

Par exemple

Le UC « Acheter un billet» est étendu par le UC « établir une facture».

En effet, le UC « établir une facture» sera effectué uniquement à la demande du client.

Formalisme



On peut mettre la condition dans un commentaire attaché à l'extension.