Diagrammes de cas d'utilisation

Principaux éléments des diagrammes de cas d'utilisation

QUOI ?

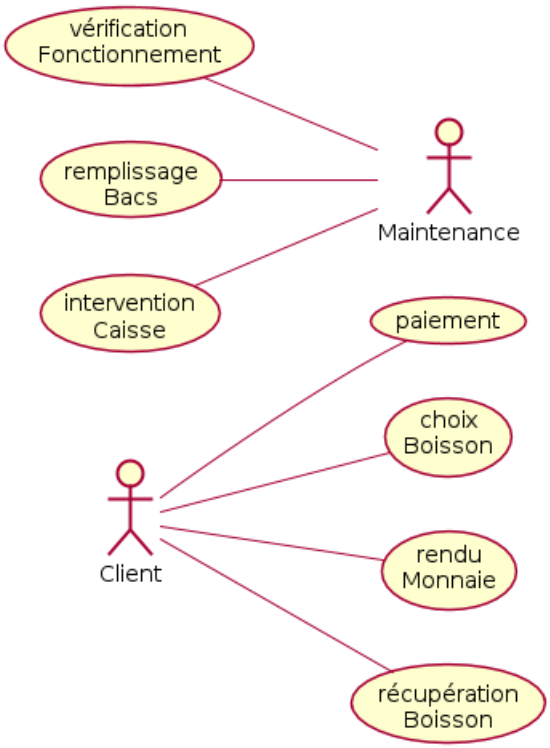
Avant tout développement, il convient de répondre à la question :

"A quoi va servir le logiciel ?"

sous peine de fournir des effeorts considérables en vain.

En UML, on établit des *Diagrammes de Cas d'Utilisation* pour répondre à cette question.

Diagrammes de Cas d'Utilisation



Principaux éléments :

- Acteurs (bonhommes)
- Cas d'utilisation (ellipses)

Acteurs

Un acteur est une entité extérieure au système modélisé, et qui interagit directement avec lui.

Acteurs non humains

- Les principaux acteurs sont les utilisateurs du système.
- En plus des utilisateurs, les acteurs peuvent être :
    - Des logiciels déjà disponibles à intégrer dans le projet ;
    - Des systèmes informatiques externes au système mais qui interagissent avec lui ;
    - tout élément *extérieur* au système et avec lequel il interagit

Pour identifier les acteurs, on se fonde sur les frontières du système.

Rôles et personnes physiques

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.

Exemples :

- Pierre, Paul ou Jacques sont tous les `Clients` du point de vue d'un distributeur automatique
- En tant que webmestre, Paul a certains droits sur son site, mais il peut également s'identifier en tant qu'utilisateur quelconque, sous une autre identité

Cas d'utilisation

Un cas d'utilisation est un service rendu à un acteur : c'est une fontionnalité de son point de vue.

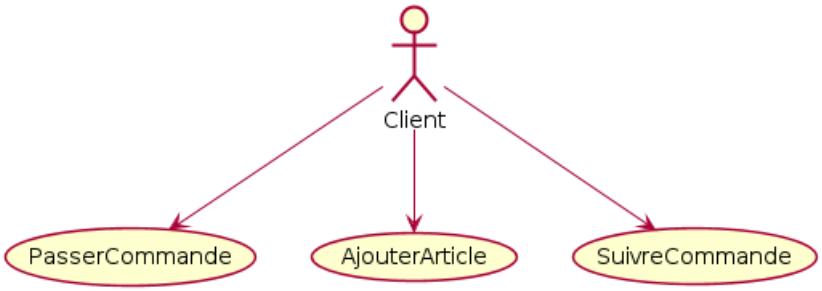
Relations liant les acteurs

Associations entre cas et acteurs

Les acteurs demandant des services aux systèmes, ils sont le plus souvent à l'initiative des échanges avec le système :

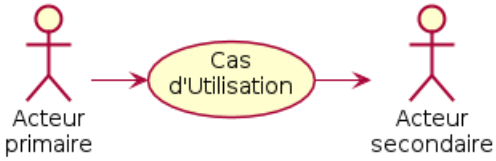
- ils sont dits *acteurs primaires*.

Lorsqu'ils sont sollicités par le système (dans le cas de serveurs externes par exemple), ils sont dits *acteurs secondaires*.



On représente une association entre un acteur et un cas d'utilisation par une ligne pleine.

Un acteur est souvent associé à plusieurs cas d'utilisation

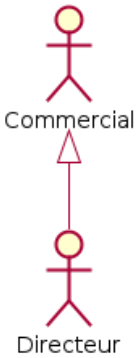


Relations entre acteurs

Il n'y a qu'un seul type de relation possible entre acteurs : la relation de *généralisation*.

Il y a généralisation entre un cas `A` et un cas `B` lorsqu'on peut dire : `A` *est une sorte de* `B`. Exemple :

- Un directeur *est une sorte de* commercial : il peut faire avec le système tout ce que peut faire un commercial, plus d'autres choses

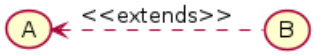


Relations entre cas d'utilisation

Types de relations possibles

- *Inclusion* : B est une partie obligatoire de A et on lit *A inclut B* (dans le sens de la flèche).

- *Extension* : B est une partie optionelle de A et on lit *B étend A* (dans le sens de la flèche).

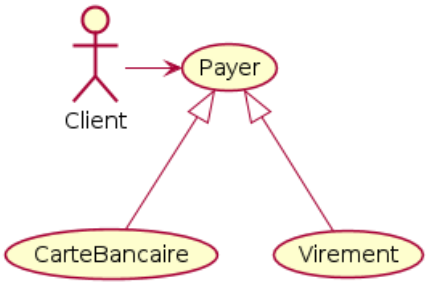


- *Généralisation* : le cas A est une généralisation du cas du cas B et on lit *B est une sorte de A*.

Les flèches en pointillés dénotent en fait une relation de *dépendance*, et les mentions *includes* et *extends* sont des *stéréotypes* et à ce titre placés entre guillemets.

Généralisation entre cas d'utilsation

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

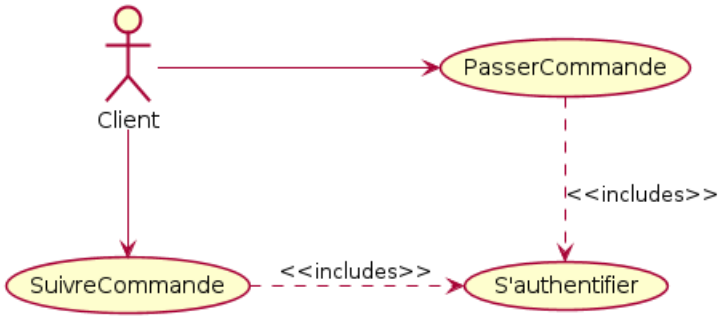


Cet héritage signifie que les éléments spécifiques *héritent* de tout ce qui caractérise l'élément général : - Les associations avec des acteurs - Les relations de dépendance - Les héritages déjà existant, dans lesquel l'élément général jour le rôle d'élément plus spécifique

Réutilisation de cas d'utilisation

Les relations d'inclusion et d'extension permettent d'isoler un service réutilisable comme partie de plusieurs autres cas d'utilisation :

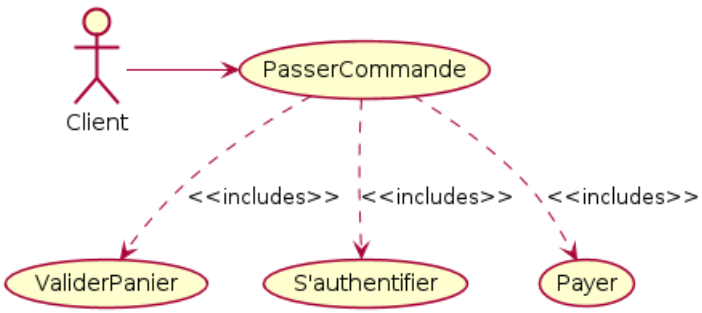
- On parle alors de *réutilisation*.
- Le code développé pour implémenter le cas d'utiliation réutilisé est d'emblée identifié comme ne devant être développé qu'une seule fois, puis réutilisé.



Décomposition de cas d'utilisation

Un cas d'utilisation ne doit jamais se réduire à une seule action : il doit occasionner des traitements d'une complexité minimale.

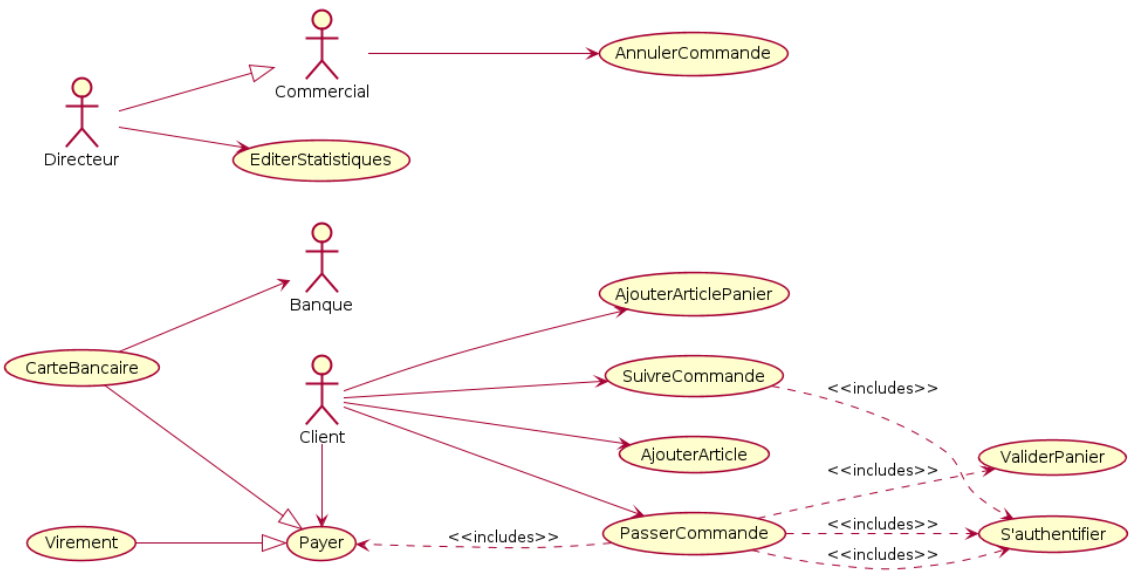
Toutefois, il peut arriver qu'un cas d'utilisation recouvre un ensemble très important d'achanges et de traitements. Dans ce cas, on peut utiliser les relations d'inclusion et d'extension.



Attention : ne jamais décomposer en première analyse, mais lorsqu'on a suffisamment avancé dans la conception ou le codage pour être certain que la décomposition est utile.

Synthèse

Diagramme complet



Description textuelle de cas d'utilisation

Un nom ne suffit pas à comprendre le détail de ce que recouvre un cas d'utilisation.

Il est donc nécessaire d'adjoindre à chaque cas d'utilisation une description détaillée. Cette description est parfois textuelle et composée de plusieurs rubriques dont les plus importantes sont :

- Le *scénario nominal* : enchaînement d'actions typiques dans le cas où les choses se passent comme prévu.
- Les *enchaînements alternatifs* : enchaînements dans des cas particuliers.