

# *Le Guide Pratique des Cas d'Utilisation*

Auteur : Equipe Conseil Modeliosoft  
Version: 1.0  
Copyright: Modeliosoft

## **Modeliosoft**

21 avenue Victor Hugo  
75016 Paris

[www.modeliosoft.com](http://www.modeliosoft.com)

## Introduction aux Guides Pratiques

Les Guides Pratiques sont issus de l'expérience des consultants de Modeliosoft et destinés à faciliter la construction de modèles en bénéficiant des capacités de l'atelier Modelio. Ils sont délibérément courts, pour fournir l'essentiel de la pratique en peu de pages. L'équipe conseil de Modeliosoft est à votre disposition pour vous assister dans vos travaux liés à la définition d'architecture d'entreprise, modélisation des processus métier, modélisation d'architectures logicielles, SOA, et assistance dans vos projets informatiques.

Modeliosoft vous propose une offre packagée conseil / outil. Pour plus d'informations, visitez [www.modeliosoft.com](http://www.modeliosoft.com).

Sous [www.modeliosoft.com](http://www.modeliosoft.com), vous pouvez librement télécharger Modelio Free Edition, ateliers gratuit, ergonomique et sans limitations pour la modélisation UML et la modélisation métier (Architecture d'Entreprise, BPM, architecture logique SOA, architecture logiciel).

Sous [www.modeliosoft.com](http://www.modeliosoft.com), vous pouvez également évaluer et acheter Modelio Enterprise Edition, pour bénéficier d'une grande richesse fonctionnelle : support du travail en équipe, analyse des objectifs, définition du dictionnaire et analyse des besoins, génération de code, génération documentaire sur l'ensemble du cycle de vie, et ainsi de suite.

Les Guides Pratiques disponibles sont les suivants :

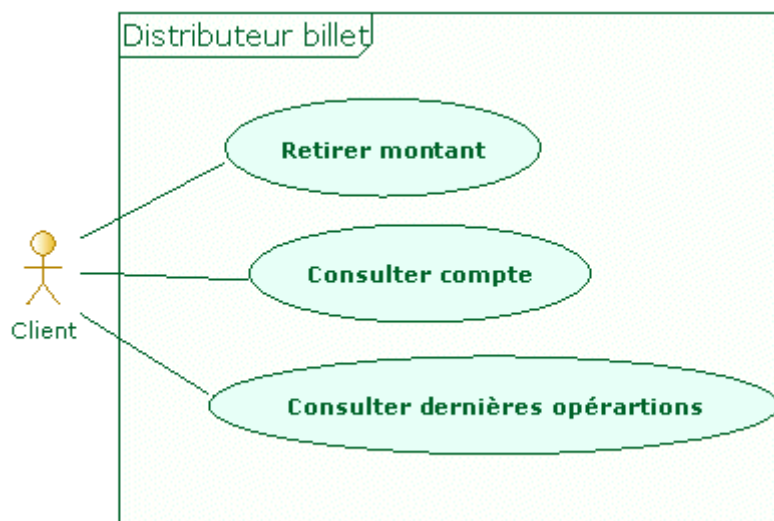
- Guide Pratique des Cas d'Utilisation,
- Guide Pratique des Processus Métiers,
- Architecture d'Entreprise : Guide Pratique de l'Architecture Logique,
- Guide Pratique de la Modélisation de l'Organisation d'une Entreprise.

D'autres guides pratiques seront fournis prochainement – n'oubliez pas de consulter régulièrement nos sites.

## Quoi

Un modèle de cas d'utilisation UML décrit et formalise les relations entre le système logiciel à réaliser et le monde extérieur. Cette description se place du point de vue externe (boîte noire) sans jamais entrer dans les structures internes du logiciel. L'objectif est de préciser les frontières du système et les différentes interactions mises en œuvre dans la réalisation des besoins métiers.

Le modèle est constitué par deux principaux types d'élément UML : les Acteurs et les Cas d'utilisation (voir figure ci-dessous). Le container qui apparaît dans le diagramme (sous forme rectangulaire) représente le système. Les cas d'utilisation sont visualisés à l'intérieur du système, les acteurs à l'extérieur du système.



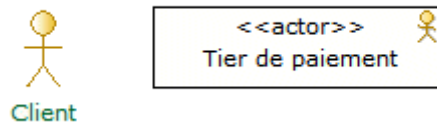
### *Exemple de cas d'utilisation avec 1 acteur et 3 cas d'utilisation*

Particulièrement employé au cours des phases amont du processus de développement, ce type de modèle est un outil de base pour la formalisation de besoins fonctionnels et une aide précieuse dans le dialogue avec les utilisateurs.

## Les acteurs

Un acteur est une entité externe au système qui est amenée à interagir directement avec celui-ci. Un acteur peut représenter aussi bien un utilisateur humain que tout dispositif matériel ou logiciel.

Exemples : Usager, Client, Progiciel de facturation, Machine de production.

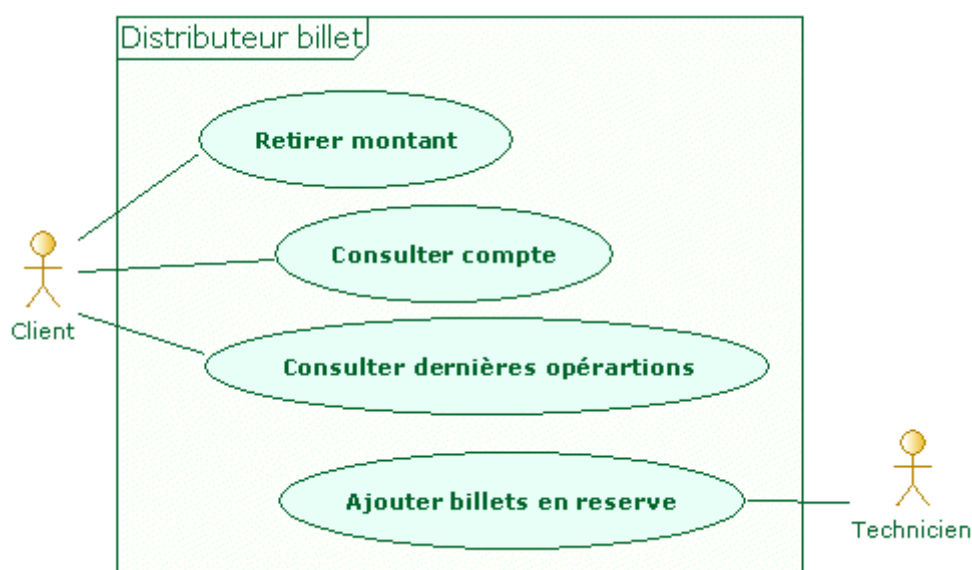


Un acteur représente un rôle joué vis-à-vis du système. Un utilisateur physique peut jouer successivement des rôles différents en fonction du mode d'utilisation du système.

Par exemple, un administrateur système est chargé d'installer une nouvelle version d'un cours dans la salle de formation. Il va d'abord jouer son rôle habituel (**Acteur Administrateur**) pour installer la version sur les postes de formation; puis il va se connecter en tant que **Acteur Stagiaire**, afin de vérifier le fonctionnement. Dans ce cas, il n'y a qu'une entité concrète externe: l'administrateur système. Pourtant il y a deux acteurs distincts : l'administrateur et le stagiaire.

Dans un modèle de cas d'utilisation, on cherche à identifier tous les acteurs qui interagissent avec le système. En plus des **acteurs principaux**, qui justifient la construction du système, on veillera à ne pas omettre les **acteurs secondaires** mais nécessaires au bon fonctionnement du système (administrateurs, réparateurs, etc.).

Dans la représentation des diagrammes de cas d'utilisation, les acteurs principaux sont représentés à gauche du système et les acteurs secondaires à droite du système.



**Acteurs Client et Technicien**

Dans l'exemple ci-dessus, le technicien chargé de renouveler régulièrement la réserve de billets (acteur secondaire) est absolument nécessaire et impacte fortement la conception du distributeur de billet.

### *Règle de validation*

- Tout acteur est lié à au moins un cas d'utilisation.

## Les cas d'utilisation

Un cas d'utilisation représente une interaction entre acteurs et système, dans le but de répondre à un besoin fondamental. Il est décrit par un ensemble de scénarios, qui précisent le dialogue entre le système et les acteurs.

Un cas d'utilisation **doit rendre un service réel complet**, apportant une valeur ajoutée à l'acteur. C'est sa caractéristique essentielle. A l'inverse, les fonctions comme "saisir son code secret" ou "choisir le montant" ne sont probablement pas des cas d'utilisation.

Un cas d'utilisation est un élément atomique, qui doit satisfaire aux critères suivants :

- **Unité de temps.** Le déroulement d'un scénario doit se réaliser dans un délai relativement bref. A l'inverse, une interaction ne peut pas durer plusieurs mois.
- **Unité de lieu.** On évite un changement de lieu au cours d'un déroulement d'un scénario (on ne débute pas le scénario au bureau pour le terminer au domicile).
- **Unité d'acteur (un acteur bénéficiaire).** Le service est rendu pour un seul acteur (acteur principal), qui est souvent la source du déclenchement du cas d'utilisation. Les autres acteurs qui interagissent avec le cas d'utilisation sont des acteurs secondaires. Ils participent aux scénarios, mais ne sont pas les bénéficiaires du service.
- **Non interruptible.** Il n'est pas possible (dans une utilisation normale) d'interrompre le déroulement d'un scénario, pour le reprendre plus tard. Le scénario s'arrête lorsque le service est rendu. Typiquement, un acteur ne va pas 'prendre ses congés' au milieu du déroulement d'un scénario.

Ces critères ci-dessus mettent l'accent sur le caractère atomique des cas d'utilisation. Il ne s'agit pas de critères formels, mais ils constituent une aide pour la compréhension, la construction et la validation des cas d'utilisation. Ils permettent également de différencier les cas d'utilisation avec d'autres types d'éléments, comme les processus métiers, les opérations ou les fonctions.

**Astuce :** Le petit test suivant permet souvent de vérifier le critère essentiel (**doit rendre un service réel et complet**). On imagine que l'acteur s'arrête juste après la fin du scénario. Si on aboutit à une absurdité, il y a probablement une erreur d'identification. Par exemple, si on considère le scénario de "s'identifier", qui consiste à saisir son login et son mot de passe, si l'utilisateur quitte la pièce pour se consacrer à une autre occupation, cela n'a pas beaucoup de sens. Le cas d'utilisation "s'identifier" ne sera pas conservé.

### Règles de validation

- Tout cas d'utilisation est lié à au moins un acteur (\*).
- Tout cas d'utilisation comporte au moins un scénario.

(\*) Sauf pour les « faux cas d'utilisation » évoqués dans le chapitre Relations entre cas d'utilisation.

## Les scénarios

Les scénarios d'un cas d'utilisation sont constitués par un enchaînement de séquences qui décrivent un dialogue entre le système et un ou plusieurs acteurs.

Les scénarios s'expriment le plus souvent à l'aide de texte. Cependant, il n'y a pas de règles dans le standard UML sur ce point. Par exemple, on peut utiliser les diagrammes d'activités, les diagrammes de séquences ou tout autre moyen.

Pour un cas d'utilisation, on trouve au moins un scénario principal (ou nominal), qui représente l'interaction la plus significative du cas d'utilisation (chemin ou "tout se passe bien"). D'autres scénarios peuvent être ajoutés afin de décrire les autres interactions possibles.

Exemple :

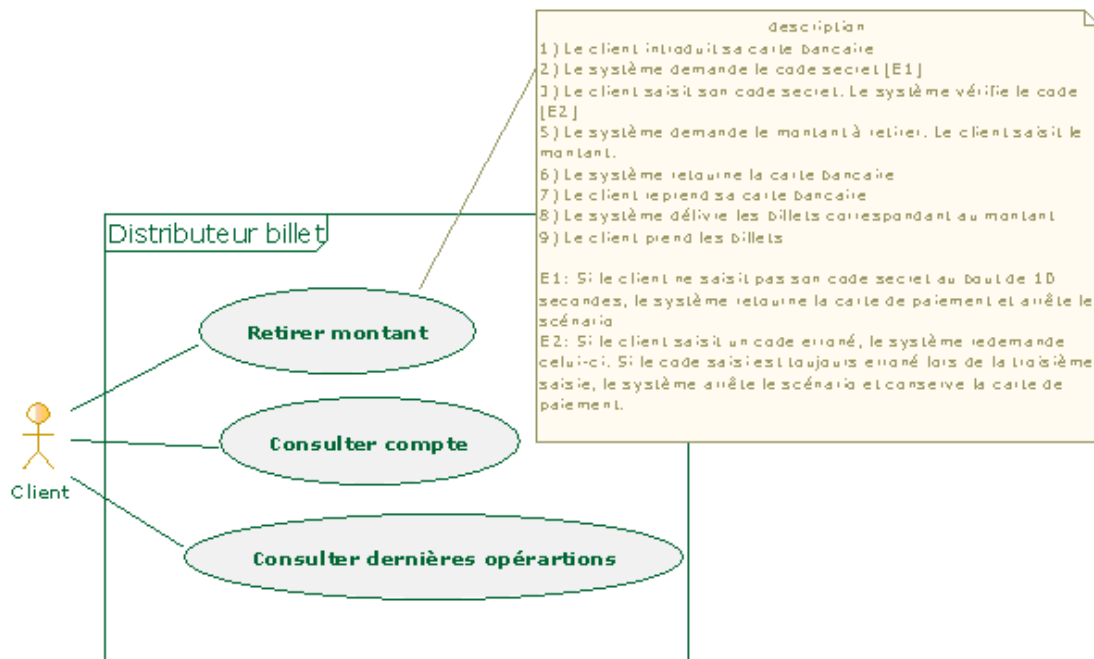
```
1) le CLIENT introduit sa carte de paiement
2) le système demande au CLIENT son code secret [E1]
3) le CLIENT saisit son code secret [E2]
4) Le système vérifie le code secret du CLIENT
5) le système demande au CLIENT le montant de son retrait
6) le CLIENT choisit le montant
7) le système rend la carte de paiement. Le client prend sa carte de
   paiement
8) le système délivre les billets correspondants au montant
9) le CLIENT retire les billets
```

Les erreurs, exceptions ou cas particuliers peuvent être ajoutés aux scénarios sous la forme de renvois associés à une séquence particulière.

Exemples :

```
E1: Si le client ne saisit pas son code secret au bout de 10 secondes, le
système retourne la carte de paiement et arrête le scénario.

E2: Si le client saisit un code erroné, le système redemande celui-ci. Si le
code saisi est toujours erroné lors de la troisième saisie, le système arrête le
scénario et conserve la carte de paiement.
```



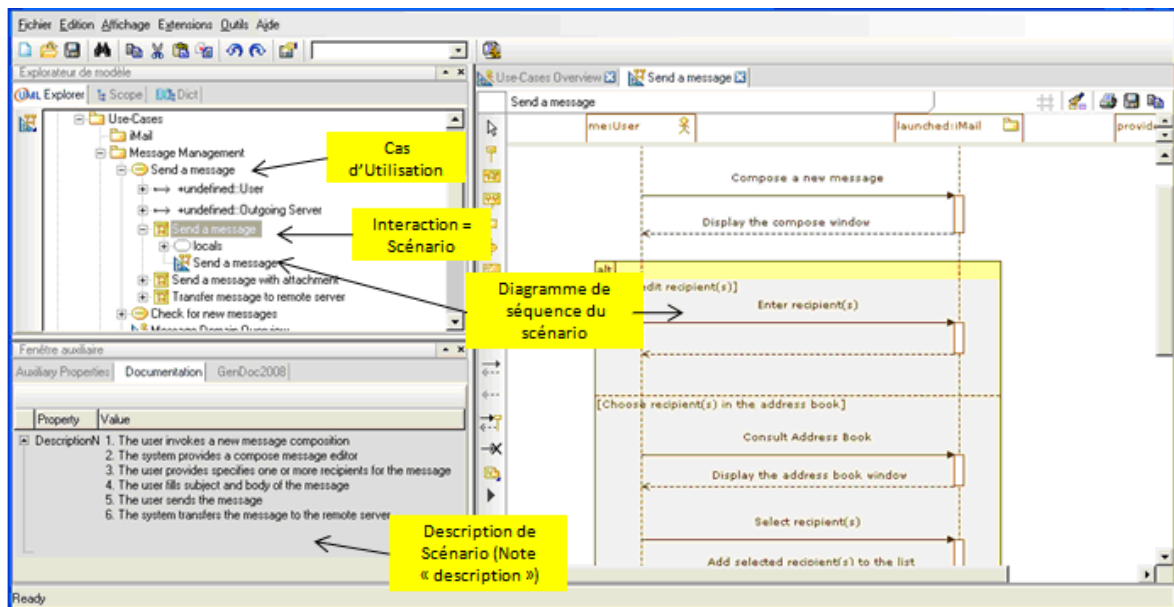
### Scénario représenté par une note associée au cas d'utilisation

Dans la pratique, la plupart des cas d'utilisation sont décrits par un scénario (nominal avec des renvois pour les erreurs et les exceptions).

La notion de scénario n'existe pas dans UML. Nous recommandons de créer une interaction sur le cas d'utilisation pour représenter le scénario.



Il est possible de réaliser un diagramme de séquence (lié à l'interaction) pour modéliser le scénario. Dans la pratique, la plupart des scénarios se contentent d'une description textuelle, insérée dans la note "description" sous Modelio.



### Définition d'un scénario sous Modelio

Les scénarios de doivent pas comporter des indications IHM (Interface Homme Machine) ou des éléments techniques :

Recommandé	A proscrire
L'opérateur demande la liste des dossiers. Le système retourne la liste des dossiers en cours.	L'opérateur clique sur le bouton "Liste". Le système affiche la liste dans la fenêtre 'dossiers' de couleur verte.
Le système enregistre le profil du membre.	Le système écrit le profil du membre dans la base de données Oracle 8.1 par un SQL INSERT statement.

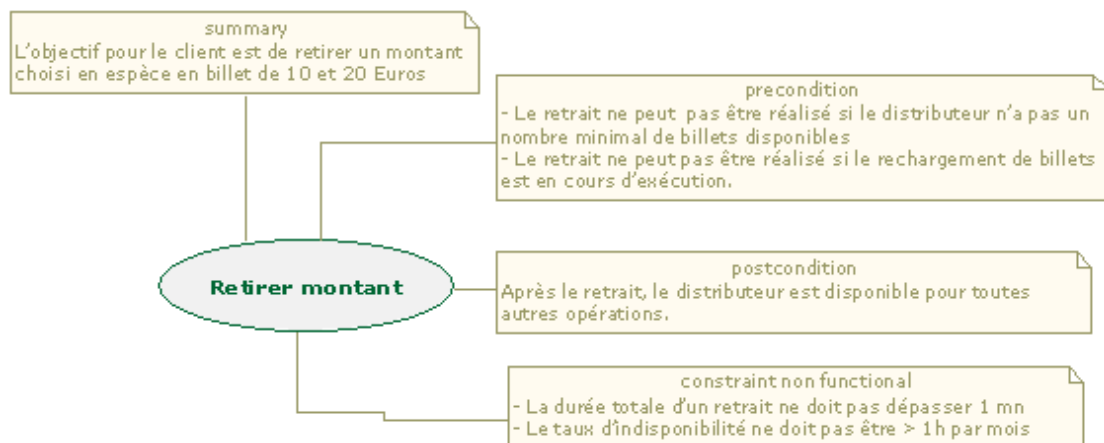
## Fiche détaillée d'un cas d'utilisation

En plus des scénarios, chaque cas d'utilisation est décrit par une fiche détaillée qui contient un ensemble de caractéristiques :

- Service attendu
- Pré-condition
- Post-condition
- Contraintes non fonctionnelles
- Règles métiers applicables
- Fréquence d'exécution

Cette liste n'est pas exhaustive et peut être adaptée et enrichie en fonction des réalités projet ou des termes habituellement employés (voir notamment A. Cockburn. *Writing Effective Use Cases*. Addison-Wesley Longman Publishing Co).

Ces éléments peuvent être saisis à l'aide de notes associées au cas d'utilisation (voir figure ci-dessous), ou dans un document externe.



### Caractéristiques d'un cas d'utilisation définies à l'aide de notes

La structure et les caractéristiques prises en compte dans des fiches de cas d'utilisation ne font pas partie du standard UML. Il est conseillé de définir clairement un contenu cadre partagé pour un groupe de projets dans l'entreprise, de façon à obtenir une formulation homogène.

Par défaut, Modelio fournit ainsi les types de note suivants pour les cas d'utilisation :

- Description
- Pre-Conditions
- Post-Conditions
- Exceptions
- Functional Constraints
- Non-Functional Constraints

## Service attendu

Pour chaque cas d'utilisation, une description du service attendu par l'acteur qui en est bénéficiaire.

Exemple :

- L'objectif pour le client est de retirer un montant choisi en espèce (billet en Euro) à partir de son compte courant.

## Les pré et post conditions

Les pré-conditions décrivent les conditions qui doivent être vérifiées pour le déclenchement d'un des scénarios.

Exemple :

- Le retrait ne peut pas être réalisé si le distributeur n'a pas un nombre minimal de billets disponibles.
- Le retrait ne peut pas être réalisé si le rechargement de billets est en cours d'exécution.

Les post-conditions décrivent les conditions vérifiées après l'arrêt des scénarios (sauf en cas d'erreur ou de traitement exceptionnels).

Exemple :

- Après le retrait, le distributeur est disponible pour toute autre opération.

D'autres pré et post conditions peuvent être définies au niveau de chaque scénario, pour spécifier des conditions particulières.

## Les contraintes non-fonctionnelles

Les contraintes non fonctionnelles spécifient un ensemble d'exigences additionnelles. Il s'agit d'exigences liées aux performances, disponibilité, volumétrie, délai d'exécution etc.

Exemples :

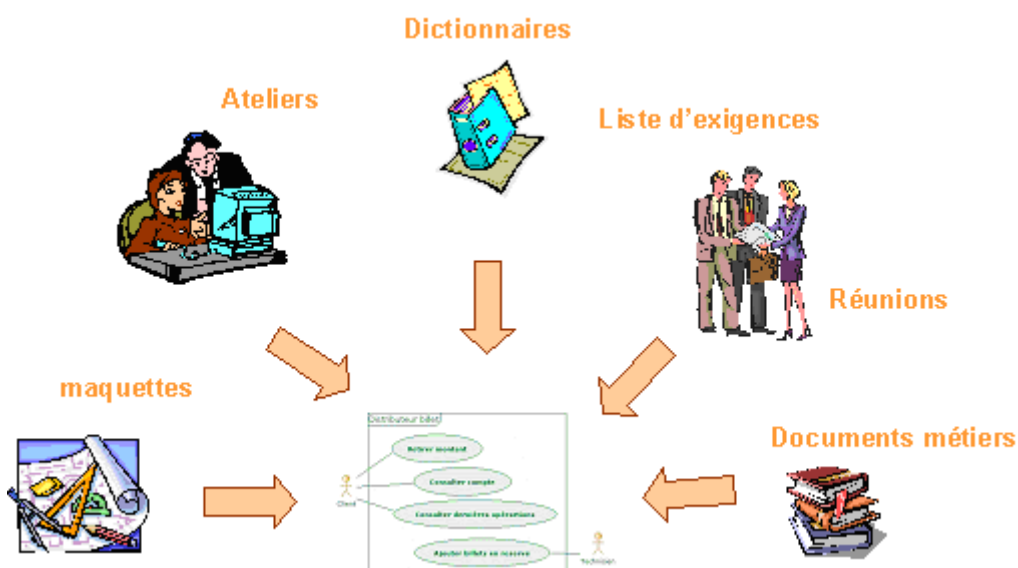
- La durée totale d'un retrait ne doit pas dépasser 1 mn.
- Le taux d'indisponibilité ne doit pas être > 1h par mois.

## Elaboration du modèle de cas d'utilisation

Les modèles de cas d'utilisation sont élaborés en relation étroite avec la maîtrise d'ouvrage (le côté métier et utilisateur). Les différentes ressources métiers, les dictionnaires ou listes d'exigences (s'ils existent) pourront être utilisés, sans les considérer comme sources uniques de la connaissance. Le module *Modelio Scope Manager* permet de définir et modéliser les exigences, et de les tracer avec le reste du modèle, notamment les Cas d'Utilisation. Pour chaque exigence fonctionnelle, la question doit être posée de savoir si elle correspond à un Cas d'Utilisation (voir les règles précédemment fournies).

Si l'on dispose d'une modélisation des processus métier (voir le guide sur la modélisation de processus), l'ensemble des activités identifiées supportées par le système sont candidates à devenir des cas d'utilisation.

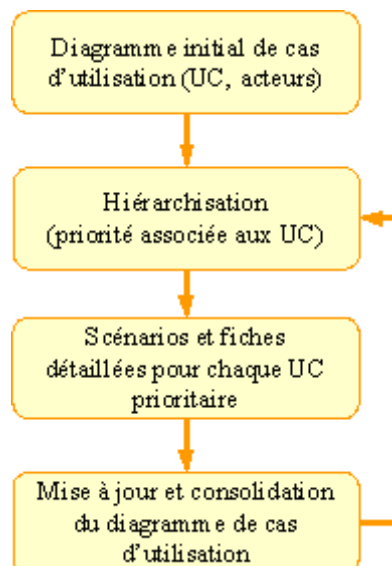
Le dialogue nécessaire s'effectue en employant tous les moyens adaptés en fonction de la situation, en privilégiant la communication directe (réunions, ateliers, maquettes etc).



Les cas d'utilisation pourront également servir de support sans imposer le formalisme UML si les intervenants sont réticents. Par exemple, les scénarios textuels pourront être échangés tel quels, sans les diagrammes UML de cas d'utilisation.

Globalement cette élaboration sera effectuée de manière progressive suivant un schéma "en largeur" plutôt qu'un schéma "en profondeur". On évitera de découper le travail suivant un ordre guidé par les éléments (d'abord les acteurs, ensuite les cas d'utilisation, pour finir par les scénarios). Les acteurs, cas d'utilisation et scénarios sont intimement liés, et se construisent en parallèle.

En fonction du volume (nombre de cas d'utilisation), il est préférable de découper les travaux suivant les priorités données aux cas d'utilisation (en premier les cas d'utilisation orientés vers le client, liés aux acteurs principaux), en poursuivant toujours le dialogue avec la maîtrise d'ouvrage. Le but est de converger petit à petit vers un modèle cohérent, valide et conforme à la vision métier.



### ***Démarche générale pour l'élaboration des cas d'utilisation***

Au cours de l'élaboration des premiers modèles, il est conseillé de ne pas utiliser les relations entre cas d'utilisation (include, extend ou héritage), et de mettre l'accent sur les scénarios.

## Identification et nommage des acteurs

L'identification des acteurs, leur nommage correct n'est pas toujours simple et peut s'avérer délicat.

Les acteurs représentent des rôles que peut jouer à un moment une entité externe. Par ailleurs, une entité peut jouer plusieurs rôles suivant les circonstances.

La distinction entre la notion de rôle et les notions de poste ou fonction dans une entreprise peut être confuse. Dans la pratique, on n'attend pas le nommage définitif des acteurs pour construire le modèle avec les cas d'utilisation. On utilise par exemple des noms temporaires (en utilisant les postes occupés par exemple) qui seront remplacés plus tard par une terminologie plus adaptée.

De plus, l'identification de tous les acteurs qui interviennent n'est pas forcément immédiate. On utilise les scénarios des cas d'utilisation pour contrôler et ajouter éventuellement les acteurs manquants. Par exemple, dans le scénario du cas d'utilisation "Retirer un montant" :

```
1) le CLIENT introduit sa carte de paiement
2) le système demande au CLIENT son code secret
3) le CLIENT saisit son code secret
4) le système vérifie le code secret du CLIENT
5) le système demande au CLIENT le montant de son retrait
```

A la séquence 4, la formulation implique que c'est le système qui vérifie le code secret du client. Or, après discussion avec la maîtrise d'ouvrage, il s'avère que cela est erroné. Dans la réalité, le système demande à la carte de paiement le contrôle du code (La carte de paiement contient son logiciel de vérification). Si l'on considère que la carte de paiement ne fait pas partie du système (le distributeur), on doit ajouter un nouvel acteur.



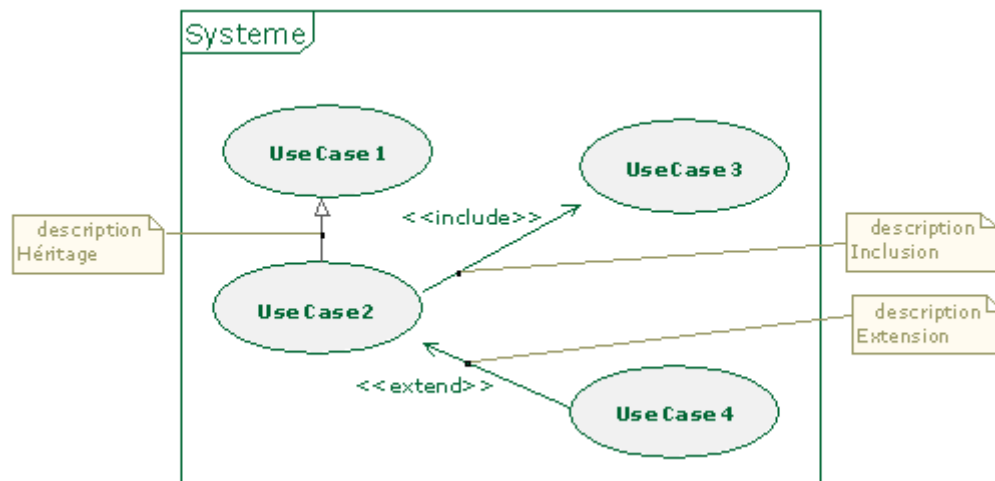
Le scénario est reformulé de la manière suivante :

```
1) le CLIENT introduit sa carte de paiement
2) le système demande au CLIENT son code secret
3) le CLIENT saisit son code secret
4) Le système vérifie le code secret du CLIENT
4.1) Le système demande à la CARTE DE PAIEMENT la vérification du code secret
5) le système demande au CLIENT le montant de son retrait
```

## Relations entre cas d'utilisation

Le standard UML définit un ensemble de relations entre cas d'utilisation :

- Les relations d'inclusion (« include ») : pour factoriser
- Les relations d'extension (« extend »)
- Les liens d'héritage

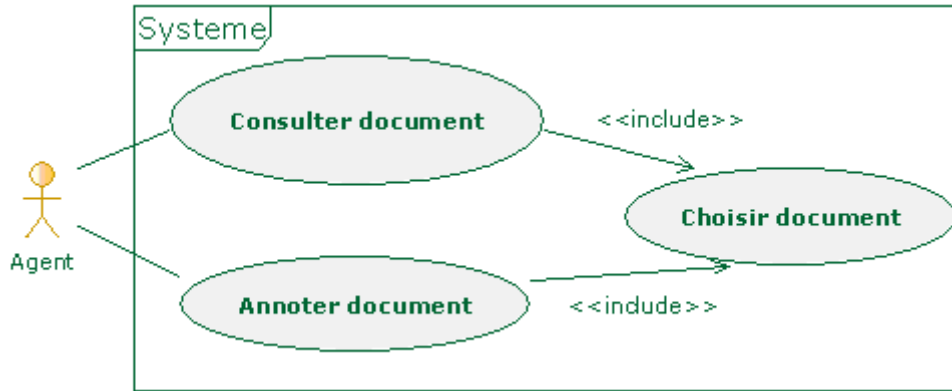


**Relations entre cas d'utilisation : UC2 hérite de UC1, UC2 inclut UC3, UC4 étend UC2**

Bien que partie intégrante du standard, l'emploi de ses relations n'est pas clair et peut amener des difficultés. Nous conseillons vivement de limiter leur emploi (voir détails dans le chapitre Recommandations et limitations).

## La relation d'inclusion

L'inclusion peut être employée lorsque plusieurs cas d'utilisation comportent des enchaînements de séquences identiques. Un nouveau cas d'utilisation qui déclare cette partie commune est utilisé par référence par les cas d'utilisation et factorise cette partie.



### Relation "include" entre cas d'utilisation

Dans l'exemple ci-dessus, les scénarios des deux cas d'utilisation "Consulter document" et "Annoter document" débutent avec le même enchaînement :

- 1) L'Agent demande la liste des documents disponibles
- 2) L'Agent sélectionne un document dans la liste
- 3) L'Agent visualise le contenu du document
- x) ...
- y) ...

Pour éviter les répétitions, le cas d'utilisation "Choisir document" déclare dans son scénario les trois premières séquences, puis les scénarios des deux cas d'utilisation sont modifiés pour référencer celui-ci.

- 1) include: Choisir document
- x) ...
- y) ...

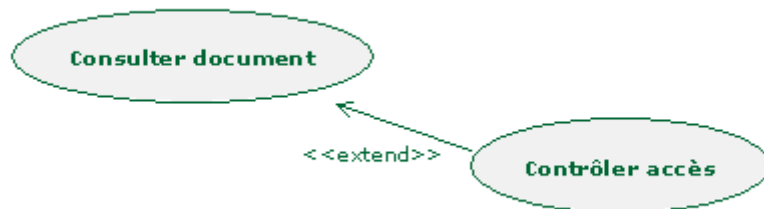
**Note :** Le standard UML n'indique pas le langage ou la technique pour référencer un cas d'utilisation dans un scénario.



## La relation d'extension

Cette relation permet d'introduire un chemin particulier dans un scénario, souvent associé à une condition exceptionnelle (effet de seuil, cas particulier ...) appelé point d'extension.

Un cas d'utilisation contient cet enchaînement exceptionnel et étend le cas d'utilisation initial.



### Relation "extend" entre cas d'utilisation

Dans l'exemple ci-dessus, s'il s'agit d'un document critique, un point d'extension (PEA) est inséré dans le scénario du cas d'utilisation "Consulter document".

```
1) include: Choisir document
   (PEA) Si document critique
2)
x) ...
y) ...
```

L'enchaînement correspondant est défini dans le scénario du cas d'utilisation "Contrôler acces".

```
(PEA)
1) Le système demande le mot de passe à l'Agent
2) L'Agent saisit le mot de passe
3) Le système vérifie le mot de passe
```

**Note :** Le standard UML n'indique pas le langage ou la technique pour insérer ou définir un point d'extension.

Dans la pratique, on évite les extensions entre cas d'utilisation (voir chapitre Recommandations et limitations).

## La relation d'héritage

L'héritage entre cas d'utilisation est représenté comme un héritage entre classes. Il peut être employé pour définir des "sous-cas d'utilisation".

Dans la pratique, on évite les héritages entre cas d'utilisation (voir chapitre Recommandations et limitations).

Pour le regroupement de cas d'utilisation (par catégories fonctionnelles par exemple), on utilisera le Package (voir chapitre Structuration des cas d'utilisation).

## Recommandations et limitations

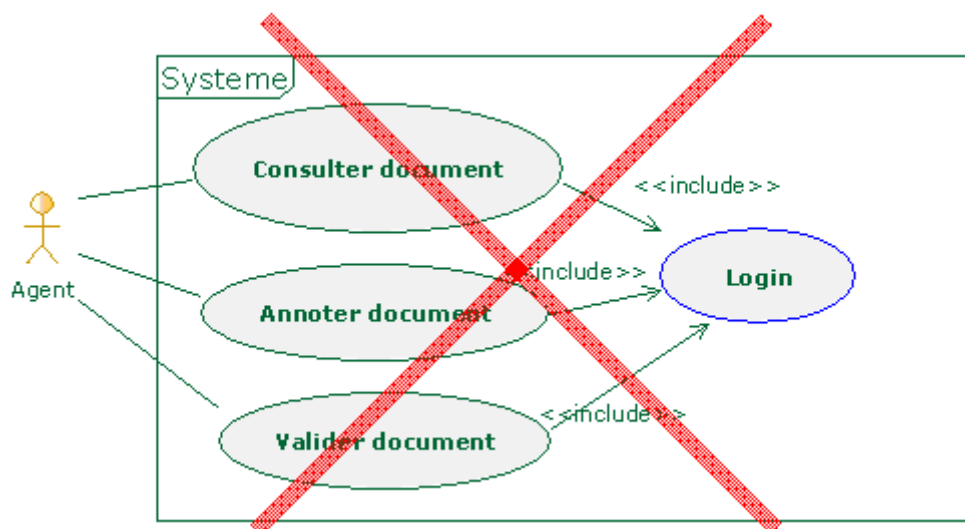
Eviter l'emploi des relations d'extension et les héritages. La sémantique n'est pas très rigoureuse et l'absence de langage standard tend à accroître les risques de confusions. Les extensions peuvent être représentées dans la plupart des cas par de simples renvois dans les scénarios.

*"I strongly suggest that you ignore them (extend and inheritance). I've seen too many situations in which teams can get terribly hung up on when to use different use case relationships, and such energy is wasted."* UML Distilled third edition, martin Fowler, Addison-Wesley.

Les relations d'inclusions peuvent être employées lorsqu'il y a réelle factorisation. Cependant cette utilisation doit rester minoritaire et apporter une valeur ajoutée.

De plus, l'emploi de ces relations tend à alourdir les diagrammes avec l'introduction de "faux cas d'utilisation" qui ne respectent pas les propriétés déjà présentées.

Par exemple, dans le cas fréquent de l'authentification (login mot de passe) systématique pour chaque utilisation du logiciel, on évite le schéma suivant (voir figure ci-dessous). Dans ce cas, on préférera indiquer au début du document les éléments de sécurité d'accès.



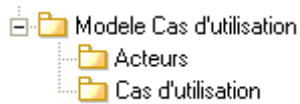
**Schéma à éviter**

Le cas d'utilisation "Login" n'est pas un réel cas d'utilisation (il n'y a pas de service complet rendu à l'utilisateur).

## Structuration des cas d'utilisation

Suivant le volume du problème à traiter, il est nécessaire de choisir une structuration adaptée.

A la racine, on définit deux packages container pour les acteurs et les cas d'utilisation.



Suivant leur nombre, les cas d'utilisation sont ensuite regroupés en packages fonctionnels.

## Documentation des cas d'utilisation

L'atelier Modelio fournit un plan type documentaire ("UseCases") dédié aux cas d'utilisation. Pour le mettre en œuvre, il faut :

- Documenter les diagrammes de cas d'utilisation (note "description" sur les diagrammes),
- Documenter les cas d'utilisation (notes sur Use Case "description", "pre-condition", etc.),
- Documenter les acteurs (notes "description"),
- Créer des interactions pour les scénarios sous les cas d'utilisation,
- Documenter les scénarios (note "description") ou éventuellement modéliser sous forme de diagrammes de séquence.

Il suffit alors de produire la documentation à partir d'un package englobant le modèle des Use Case pour obtenir un document conforme à la démarche.

---