Chapitre 1

Diagramme de cas d'utilisation

UML permet de construire plusieurs modèles d'un système : certains montrent le système du point de vue des utilisateurs, d'autres montrent sa structure interne, d'autres encore en donnent une vision globale ou détaillée. Les modèles se complètent et peuvent être assemblés. 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). Dans ce chapitre, nous allons étudier un des modèles, en l'occurrence le premier à construire : le diagramme de cas d'utilisation. Il permet de recueillir, d'analyser et d'organiser les besoins. Avec lui débute l'étape d'analyse d'un système.

1. L'importance de bien recueillir les besoins

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.

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

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 ?

2. Le diagramme de cas d'utilisation

2.1. Les cas d'utilisation

Parlons à présent d'UML et voyons quelle aide il peut apporter lors du recueil des besoins. UML n'est qu'un langage et il ne sert ici 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. Ils permettent de recenser les grandes fonctionnalités d'un système.

Exemple

La figure 1.1 modélise une borne interactive qui permet d'accéder à une banque. Le système à modéliser apparaît dans un cadre (cela permet de séparer le système à modéliser du monde extérieur). Les utilisateurs sont représentés par des petits bonshommes, et les grandes fonctionnalités (les cas d'utilisation) par des ellipses.

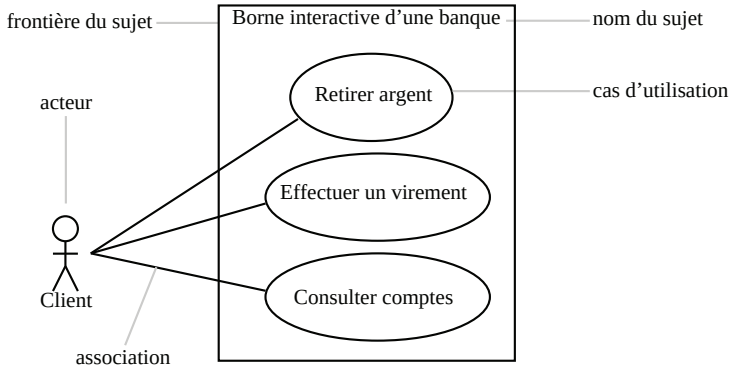


Figure 1.1
Diagramme de cas d'utilisation modélisant une borne d'accès à une banque.

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

Définition

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.

Notation et spécification

Un cas d'utilisation se représente par une ellipse (figure 1.2). Le nom du cas est inclus dans l'ellipse ou bien il figure dessous. Un stéréotype (voir la définition ci-après), peut être ajouté optionnellement au-dessus du nom, et une liste de propriétés placée au-dessous.
Un cas d'utilisation se représente aussi sous la forme d'un rectangle à deux compartiments : celui du haut contient le nom du cas ainsi qu'une ellipse (symbole d'un cas d'utilisation) ; celui du bas est optionnel et peut contenir une liste de propriétés (figure 1.3).
Un acteur se représente par un petit bonhomme ayant son nom inscrit dessous (figure 1.1) ou par un rectangle contenant le stéréotype acteur avec son nom juste en dessous (figure 1.4). Il est recommandé d'ajouter un commentaire sur l'acteur pour préciser son rôle.

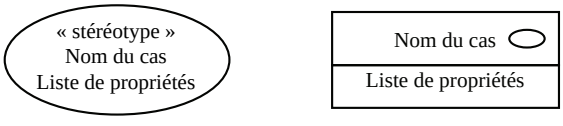
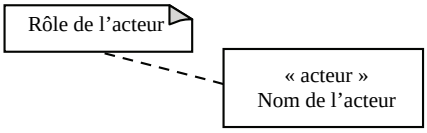


Figure 1.2 et 1.3
Représentations d'un cas d'utilisation.

Figure 1.4
Autre représentation d'un acteur.



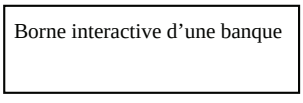
La figure 1.4 représente un acteur par un rectangle. UML utilise aussi les rectangles pour représenter des classes, et plus généralement des classeurs. Pour autant, la notation n'est pas ambiguë puisque le stéréotype acteur indique que le rectangle désigne un acteur. Les stéréotypes permettent d'adapter le langage à des situations particulières.

Définition

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

À un niveau d'abstraction plus élevé, UML permet de représenter tous les cas d'utilisation d'un système par un simple rectangle. La figure 1.5 montre comment un tel rectangle peut remplacer tous les cas d'utilisation de la figure 1.1.

Figure 1.5
Représentation des cas d'utilisation à un niveau d'abstraction élevé.



Le rectangle de la figure 1.5 est appelé « classeur ».

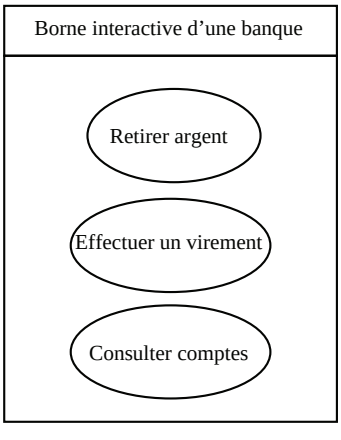
Définition

Un classeur est un élément de modélisation qui décrit une unité comportementale ou structurelle. Les acteurs et les cas d'utilisation sont des classeurs. Tout au long de ce livre, on retrouvera le terme « classeur » car cette notion englobe aussi les classes, des parties d'un système, etc.

Notation

Un classeur se représente par un rectangle contenant éventuellement des compartiments (la figure 1.6 montre comment il est possible de faire figurer explicitement des cas d'utilisation dans un classeur).

Figure 1.6
Les compartiments d'un classeur.



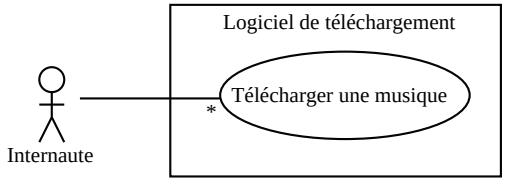
2.2. Relations entre acteurs et cas d'utilisation

Un acteur peut utiliser plusieurs fois le même cas d'utilisation.

Exemple

La figure 1.7 montre un internaute qui télécharge plusieurs morceaux de musique sur Internet.

Figure 1.7
Association avec multiplicité.



Le symbole * qui signifie « plusieurs » est ajouté à l'extrémité du cas et s'appelle une multiplicité. Plusieurs valeurs sont possibles pour la multiplicité : exactement n s'écrit tout simplement n, m..n signifie entre m et n, etc. Préciser une multiplicité sur une relation n'implique pas nécessairement que les cas sont utilisés en même temps.

2.3. 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.

- La relation d'inclusion. Un cas A est inclus dans un cas B si le comportement décrit par le cas A est inclus dans le comportement du cas B : on dit alors que le cas B dépend de A. Cette dépendance est symbolisée par le stéréotype inclut. Par exemple, l'accès aux informations d'un compte bancaire inclut nécessairement une phase d'authentification avec un mot de passe (figure 1.8).
- La relation d'extension. Si le comportement de B peut être étendu par le comportement de A, on dit alors que A étend B. Une extension est souvent soumise à condition. Graphiquement, la condition est exprimée sous la forme d'une note. La figure 1.8 présente l'exemple d'une banque où la vérification du solde du compte n'intervient que si la demande de retrait d'argent dépasse 20 euros.
- La relation de généralisation. Un cas A est une généralisation d'un cas B si B est un cas particulier de A. À la figure 1.8, la consultation d'un compte bancaire *via* Internet est un cas particulier de la consultation. Cette relation de généralisation/spécialisation est présente dans la plupart des diagrammes UML et se traduit par le concept d'héritage dans les langages orientés objet.

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

Exemple

À la figure 1.9, le modélisateur a jugé que la vente d'un article par correspondance est un problème complexe et qu'il est important de faire apparaître dans le modèle une décomposition en sous-cas.

Notation et spécification

Une dépendance se représente par une flèche pointillée. Un stéréotype est souvent ajouté au-dessus du trait.

Le stéréotype inclut indique que la relation de dépendance est une inclusion (figures 1.8 et 1.9). Le stéréotype étend indique une extension (figure 1.8). L'extension peut intervenir à un point précis du cas étendu ; ce point s'appelle le point d'extension ; il porte un nom, qui figure dans un compartiment du cas étendu sous la rubrique « point d'extension », et est éventuellement associé à une contrainte indiquant le moment où l'extension intervient. Une extension est souvent soumise à une condition (indiquée dans une note attachée à la flèche pointillée).

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 (figure 1.8).

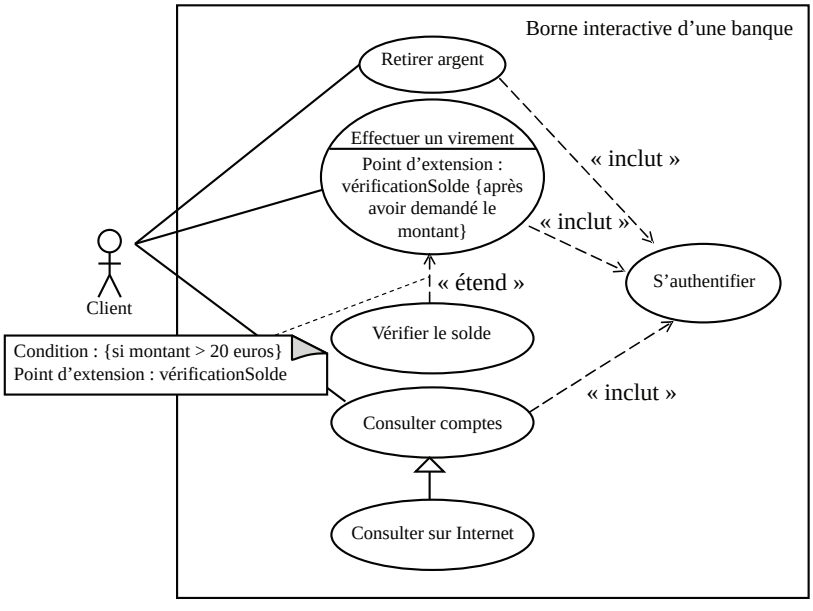


Figure 1.8
Relations entre cas dans un diagramme de cas d'utilisation.

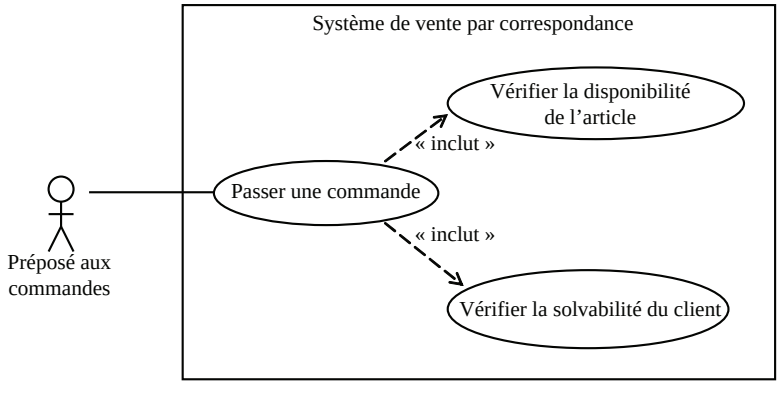


Figure 1.9
Relations entre cas pour décomposer un cas complexe.

Un cas relié à un autre cas peut ne pas être directement accessible à un acteur (figure 1.9). Un tel cas est appelé « cas interne ».

Définition

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

Les relations entre cas ne sont pas obligatoires. Elles permettent de clarifier et d'enrichir les cas d'utilisation. Par exemple, à la figure 1.8, rien n'empêche de regrouper les cas « Consulter comptes » et « Consulter sur Internet » en un seul cas. Cependant, indiquer dès la phase de recueil des besoins qu'il y a des cas particuliers apporte une information supplémentaire pertinente. La question à se poser est : faut-il la faire figurer dans le diagramme de cas d'utilisation ou la prendre en compte plus tard ? La réponse à cette question ne sera pas toujours la même selon le contexte du projet.

Remarque

Attention à l'orientation des flèches : si le cas A inclut B on trace la flèche de A vers B, mais si B étend A, la flèche est dirigée de B vers A.

2.4. 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).

Exemple

La figure 1.10 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.

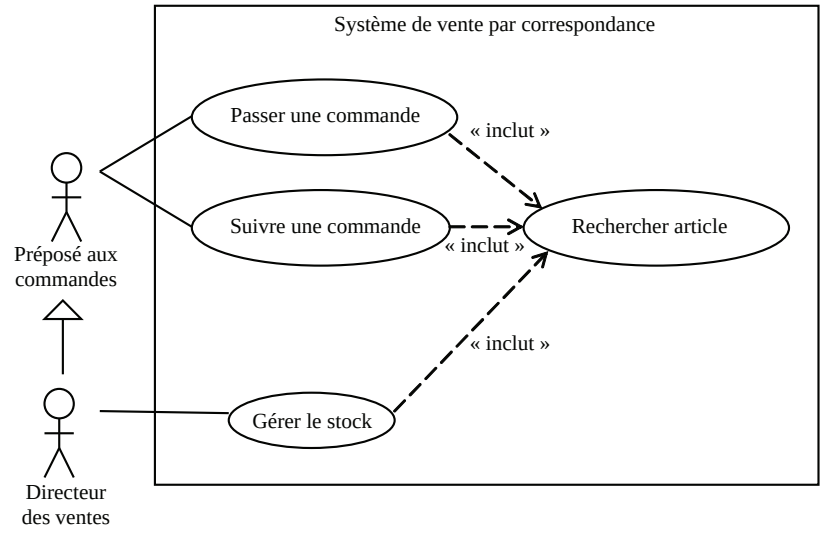


Figure 1.10
Relations entre acteurs.

Notation

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

2.5. Regroupement des cas d'utilisation en paquetages

UML permet de regrouper des cas d'utilisation dans une entité appelée « paquetage ». Le regroupement peut se faire par acteur ou par domaine fonctionnel. Un diagramme de cas d'utilisation peut contenir plusieurs paquetages et des paquetages peuvent être inclus dans d'autres paquetages.

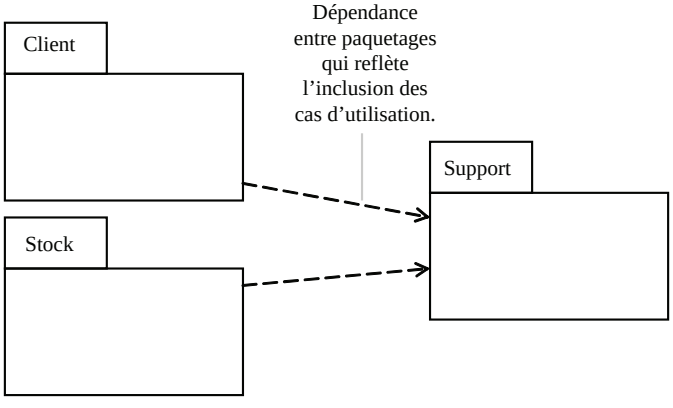
Définition

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

Exemple

À la figure 1.11, trois paquetages ont été créés : Client, Stock et Support. Ces paquetages contiennent les cas d'utilisation du diagramme de la figure 1.10 (Client contient les cas « Passer une commande » et « Suivre une commande », Stock contient le cas « Gérer le stock », tandis que le cas « Rechercher article » est inclus dans le paquetage Support.

Figure 1.11
Regroupement des cas d'utilisation en paquetage.



En tant que langage, UML est soumis à des règles de nommage qu'il faut strictement respecter : pour accéder au contenu de paquetages imbriqués les uns dans les autres, il faut utiliser des deux-points comme séparateur des noms de paquetage. Par exemple, si un paquetage B inclus dans un paquetage A contient une classe X, il faut écrire A::B::X pour pouvoir utiliser la classe X en dehors du contexte des paquetages.

3. Modélisation des besoins avec UML

3.1. Qui sont les acteurs ? Comment 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 Mme Dupont. 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.

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.

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'œuvre doit réaliser.

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.

Définition

L'acteur est dit « principal » pour un cas d'utilisation lorsque le cas d'utilisation rend service à cet acteur. Les autres acteurs sont dits « secondaires ». Un cas d'utilisation a au plus un acteur principal, et un ensemble – éventuellement vide – d'acteurs secondaires.
Un acteur principal obtient un résultat observable du système tandis qu'un acteur secondaire est sollicité pour des informations complémentaires.

3.2. 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 déterminez comment il se sert du système, dans quels cas il l'utilise, et à quelles fonctionnalités il doit avoir accès. Il faut éviter les redondances et limiter le nombre de cas en se situant au bon niveau d'abstraction (par exemple, ne pas réduire un cas à une action).

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.

L'ensemble des cas d'utilisation doit couvrir exhaustivement tous les besoins fonctionnels du système. L'étape de recueil des besoins est souvent longue et fastidieuse 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.

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 (voir la section « Description textuelle des cas d'utilisation »).

Remarque

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 (voir section 3.3).
L'intérêt des cas d'utilisation ne se limite pas au recueil des besoins. 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.

3.3. 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.

Exemple

L'exemple de la figure 1.12 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.

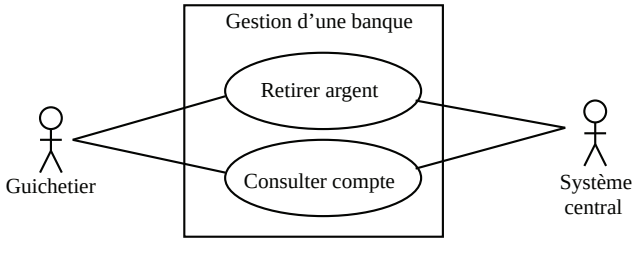


Figure 1.12
Diagramme de cas d'utilisation pour une banque.

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, tandis que 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.

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, vous devez décrire comment les acteurs interagissent avec le système.

Exemple

À partir du diagramme de cas d'utilisation de l'exemple précédent, la figure 1.13 montre une façon de décrire les interactions entre le guichetier et le système. On y voit clairement apparaître une séquence de messages.
Le graphisme utilisé fait partie du formalisme des diagrammes de séquence (voir chapitre 3).

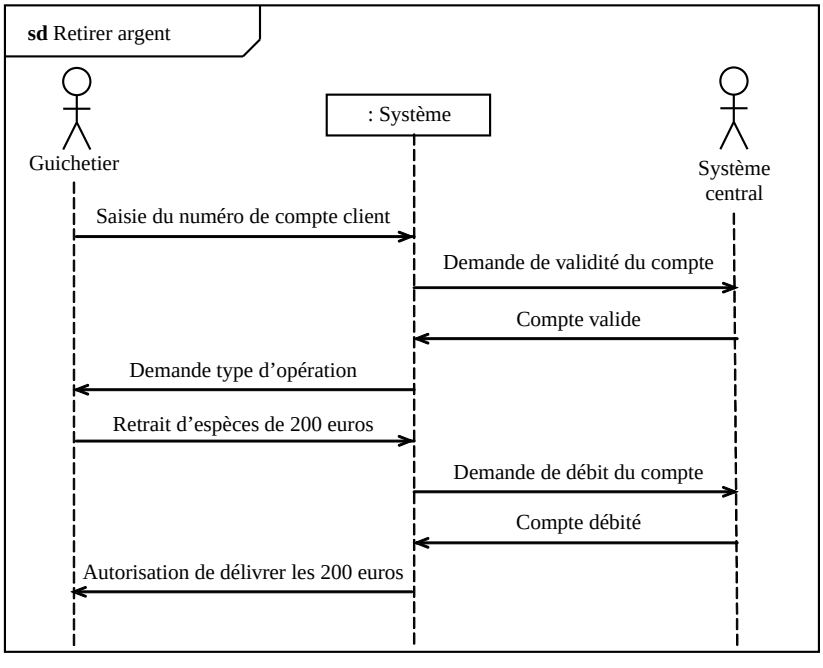


Figure 1.13
Description d'un cas d'utilisation par une séquence de messages.

Les différentes façons de décrire les cas d'utilisation

UML n'impose rien quant à la façon de décrire un cas d'utilisation. Au lieu d'utiliser une séquence de messages, il est possible d'utiliser les diagrammes d'activités d'UML (voir chapitre 5). Cette liberté de choix peut être déroutante, mais comme UML est censé pouvoir modéliser tout type de système, une manière unique de décrire un cas ne suffirait pas.

Remarque

Une des forces de la notation UML est de proposer de nombreux types de diagrammes qui mettent en avant des aspects différents d'une description. Ainsi, le modélisateur peut utiliser le type de diagramme qui lui paraît le plus adapté pour représenter son problème (et sa solution), tout en restant dans la norme.

Description textuelle des cas d'utilisation

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 dans bien des situations. Une description textuelle couramment utilisée se compose de trois parties, comme le montre l'exemple suivant. La première partie permet d'identifier le cas. Elle doit contenir :

- 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.

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 dérouement de la séquence nominale (une transaction a été enregistrée par la banque par exemple).

Parfois la séquence correspondant à un cas a besoin d'être appelée dans une autre séquence (par exemple quand une relation d'inclusion existe entre deux cas d'utilisation). Signifier l'appel d'une autre séquence se fait de la façon suivante : « appel du cas X », où X est le nom du cas.

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. Ces deux types de séquences se décrivent de la même façon que la séquence nominale mais il ne faut pas les confondre. Une séquence alternative diverge de la séquence nominale (c'est un embranchement dans une séquence nominale) mais y revient toujours, alors qu'une séquence d'exception intervient quand une erreur se produit (le séquencement nominal s'interrompt, sans retour à la séquence nominale).

Exemple

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.

La dernière partie de la description 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).

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.

La séquence nominale, les séquences alternatives, les exceptions, etc., font qu'il existe une multitude de chemins depuis le début du cas jusqu'à la fin. Chaque chemin est appelé « scénario ». Un système donné génère peu de cas d'utilisation, mais, en général, beaucoup de scénarios.

Remarque

Parfois, les utilisateurs ont du mal à décrire un cas sous une forme textuelle. Il est alors judicieux de se servir d'une autre forme de description, en utilisant un organigramme ou encore en construisant des maquettes des interfaces homme-machine. Le dessin, même sommaire, de l'aspect des écrans des interfaces permet de fixer les idées ; il offre une excellente base pour la discussion avec le maître d'ouvrage, qui le considère comme plus « parlant ».

Conclusion

Les phases de recueil des besoins et d'écriture des spécifications sont longues et fastidieuses. Mais quand elles sont bien menées, elles permettent de lever toutes les ambiguïtés du cahier des charges et de recueillir les besoins dans leurs moindres détails. Les spécifications permettant d'approfondir les besoins (on parle d'ailleurs à juste titre de spécifications techniques des besoins), la frontière entre ces deux notions est floue.

Il n'est pas question à ce moment de la modélisation de limiter les besoins. Du côté du maître d'œuvre, la tendance est à les limiter à des fonctionnalités essentielles, tandis que le maître d'ouvrage, et surtout les utilisateurs, ont tendance à en exprimer bien plus qu'il n'est possible d'en réaliser. Le maître d'œuvre doit faire preuve ici de patience et mener la phase de spécifications de tous les besoins jusqu'à son terme. C'est à ce moment seulement, que des priorités sont mises sur les spécifications. Le maître d'œuvre tente alors d'agencer les besoins de façon cohérente et fait plusieurs propositions de solutions au maître d'ouvrage, qui couvrent plus ou moins de besoins. UML ne propose rien pour mettre des priorités sur les spécifications.

Le diagramme de cas d'utilisation est un premier modèle d'un système. Que savons-nous sur le système après avoir créé ce diagramme ? Sur le système lui-même, en interne, pas grand-chose à vrai dire. C'est encore une boîte noire à l'architecture et au mode de fonctionnement interne inconnus. Donc, a fortiori, à ce stade, nous ne savons toujours pas comment le réaliser. En revanche, son interface avec le monde qui l'entoure est partiellement connue : nous nous sommes placés du point de vue des acteurs pour définir les spécifications fonctionnelles. Il faut s'attarder un instant sur l'expression « partiellement connue » pour mesurer les limites du modèle des cas d'utilisation. Les spécifications fonctionnelles disent ce que le système doit faire pour les acteurs. En d'autres termes, nous connaissons précisément ce qui entre et ce qui sort du système, mais, en revanche, nous ne savons toujours pas comment réaliser cette interface avec l'extérieur.

L'objectif de cette phase de la modélisation est donc de clairement identifier les frontières du système et les interfaces qu'il doit offrir à l'utilisateur. Si cette étape commence avant la conception de l'architecture interne du système, il est en général utile, quand la réflexion est suffisamment poussée, de poser les bases de la structure interne du système, et donc d'alterner analyse des besoins et ébauche des solutions envisagées.

Aux spécifications fonctionnelles s'ajoutent des spécifications techniques qui peuvent être vues comme des exigences pour la future réalisation.

Le présent chapitre se poursuit par une série de problèmes corrigés. Le chapitre 2, quant à lui, présente le diagramme des classes, qui permet de modéliser la structure interne d'un système.

Problèmes et exercices

La construction d'un diagramme de cas d'utilisation débute par la recherche des frontières du système et des acteurs, pour se poursuivre par la découverte des cas d'utilisation. L'ordre des exercices suit cette progression. L'élaboration proprement dite d'un diagramme de cas d'utilisation est illustrée par plusieurs exercices. Ce chapitre se termine par des études de cas de complexités croissantes.

Exercice 1 : Identification des acteurs et recensement de cas d'utilisation simples

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.

1. 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. Qui est l'acteur du système ? Est-ce le client, le pistolet ou la gâchette ?
2. Le pompiste peut se servir de l'essence pour sa voiture. Est-ce un nouvel acteur ?
3. La station a un gérant qui utilise le système informatique pour des opérations de gestion. Est-ce un nouvel acteur ?
4. 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. Comment modéliser cela ?

Solution

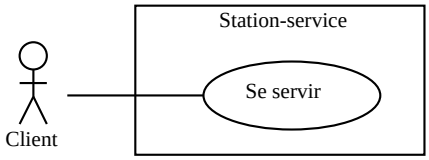
1. Pour le système informatique qui pilote la station-service, le pistolet et la gâchette sont des périphériques matériels. De ce point de vue, ce sont des acteurs. Il est néanmoins nécessaire de consigner dans le système informatique l'état de ces périphériques : dès qu'un client prend le pistolet par exemple, le système doit informer le pompiste en indiquant le type d'essence choisi. Pistolet et gâchette doivent donc faire partie du système à modéliser. Ici, nous sommes face à deux options contradictoires : soit le pistolet et la gâchette sont des acteurs, soit ils ne le sont pas. Pour lever cette ambiguïté, il faut adopter le point de vue du client. Le client agit sur le système informatique quand il se sert de l'essence. L'action de se servir constitue une transaction bien isolée des autres fonctionnalités de la station-service. Nous disons donc que « Se servir » est un cas d'utilisation.

   Le client, qui est en dehors du système, devient alors l'acteur principal, comme le montre la figure 1.14. Ce cas englobe la prise du pistolet et l'appui sur la gâchette. Ces périphériques ne sont plus considérés comme des acteurs ; s'ils l'étaient, la modélisation se ferait à un niveau de détails trop important.

   Le client est donc l'acteur principal du système. Or, bien souvent, le pompiste note le numéro d'immatriculation du véhicule du client dans le système informatique.

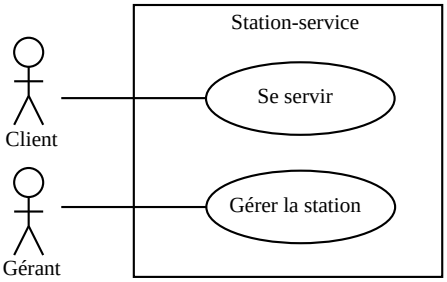
Le client doit alors être modélisé deux fois : la première fois en tant qu'acteur, et la seconde, à l'intérieur du système, pour y conserver un numéro d'immatriculation.

Figure 1.14
Le client comme acteur du cas « Se servir ».



2. Un acteur est caractérisé par le rôle qu'il joue vis-à-vis du système. Le pompiste, bien qu'étant une personne différente du client, joue un rôle identique quand il se sert de l'essence. Pour le cas « Se servir », il n'est pas nécessaire de créer un acteur supplémentaire représentant le pompiste.
3. La gestion de la station-service définit une nouvelle fonctionnalité à modéliser. Le gérant prend le rôle principal ; c'est donc un nouvel acteur (figure 1.15).

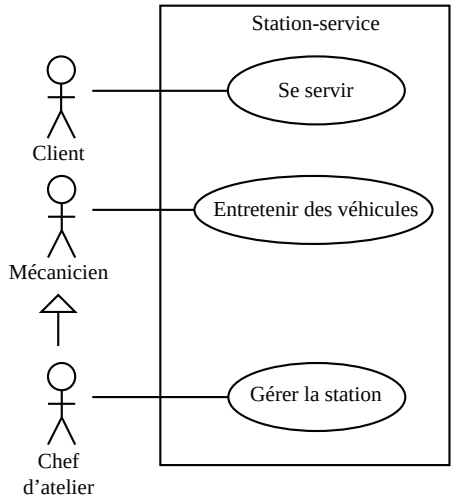
Figure 1.15
Deux acteurs pour deux rôles.



à

4. La station offre un troisième service : l'entretien des véhicules. Le système informatique doit prendre en charge cette fonctionnalité supplémentaire. Un nouvel acteur apparaît alors : le mécanicien. Le gérant est à présent un chef d'atelier qui est un mécanicien ayant la capacité de gérer la station. Il y a ainsi une relation de généralisation entre les acteurs Mécanicien et Chef d'atelier (figure 1.16) signifiant que le chef d'atelier peut, en plus d'assurer la gestion, entretenir des véhicules.

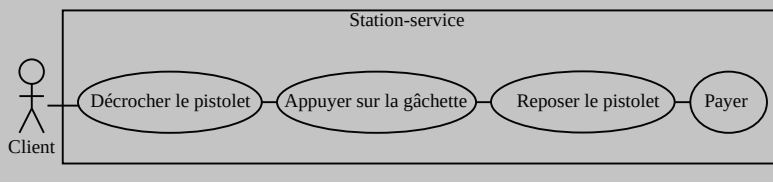
Figure 1.16
Relation de généralisation entre acteurs.



Exercice 2 : Relations entre cas d'utilisation

Quel est le défaut du diagramme présenté à la figure 1.17 ?

Figure 1.17
Exemple d'un diagramme erroné.



Solution

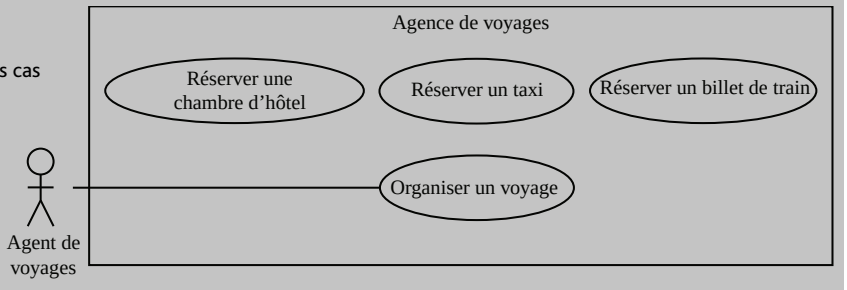
Il ne faut pas introduire de séquencement temporel entre des cas d'utilisation (cette notion apparaît lors de la description des cas). De plus, il est incorrect d'utiliser un trait plein pour relier deux cas. Cette notation est réservée aux associations entre les acteurs et les cas.

Exercice 3 : Relations entre cas d'utilisation – cas internes

Choisissez et dessinez les relations entre les cas suivants :

1. Une agence de voyages organise des voyages où l'hébergement se fait en hôtel. Le client doit disposer d'un taxi quand il arrive à la gare pour se rendre à l'hôtel.

Figure 1.18
Diagramme incomplet des cas d'utilisation d'une agence de voyages.

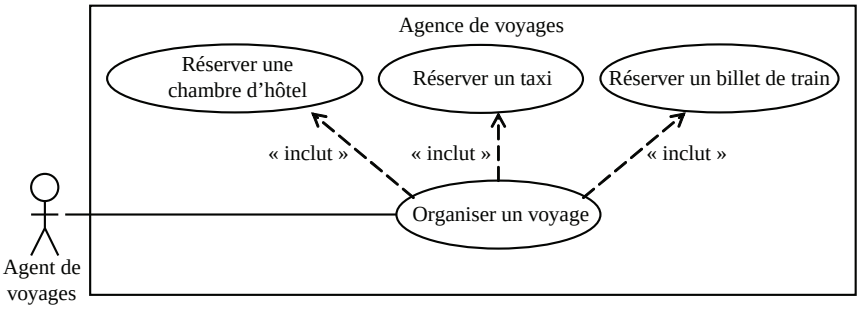


2. Certains clients demandent à l'agent de voyages d'établir une facture détaillée. Cela donne lieu à un nouveau cas d'utilisation appelé « Établir une facture détaillée ». Comment mettre ce cas en relation avec les cas existants ?
3. Le voyage se fait soit par avion, soit par train. Comment modéliser cela ?

Solution

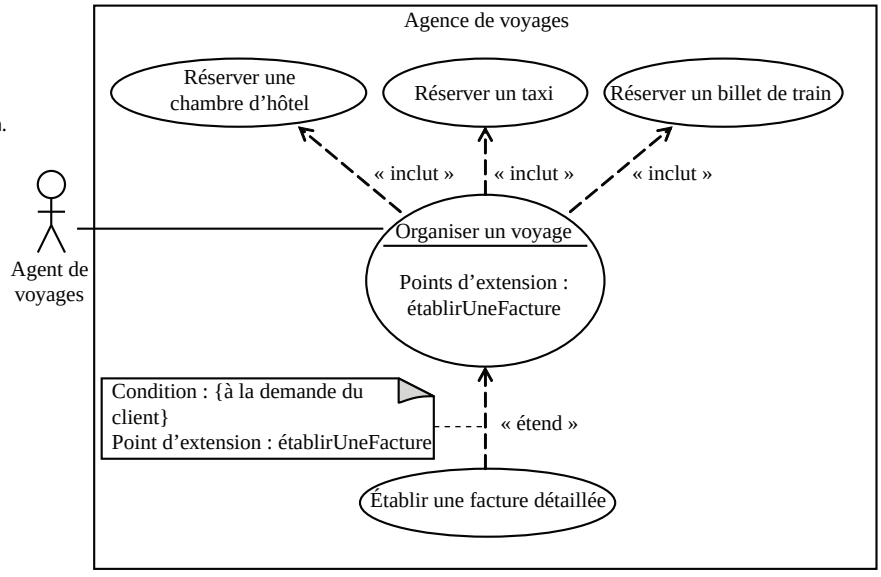
1. Le modélisateur a considéré que l'organisation d'un voyage est trop complexe pour être représentée par un seul cas d'utilisation. Il l'a donc décomposée en trois tâches modélisées par les trois cas d'utilisation « Réserver une chambre d'hôtel », « Réserver un taxi » et « Réserver un billet de train ». Ces trois tâches forment des transactions suffisamment isolées les unes des autres pour être des cas d'utilisation. De plus, ces cas sont mutuellement indépendants. Ils constituent des cas internes du système car ils ne sont pas reliés directement à un acteur.

Figure 1.19
Relations d'inclusion entre cas d'utilisation.



2. L'établissement d'une facture détaillée se fait uniquement sur demande du client. Ce caractère optionnel est modélisé par une relation d'extension entre les cas « Organiser un voyage » et « Établir une facture détaillée ». L'extension porte la condition « à la demande du client ».

Figure 1.20
Relation d'extension entre cas d'utilisation.



3. Il y a maintenant deux cas particuliers : le voyage se fait en train ou en avion. Ces cas particuliers sont modélisés par les cas « Réserver un billet de train » et « Réserver un billet d'avion ». Ceux-ci sont liés à un cas plus général appelé « Réserver un titre de transport ».

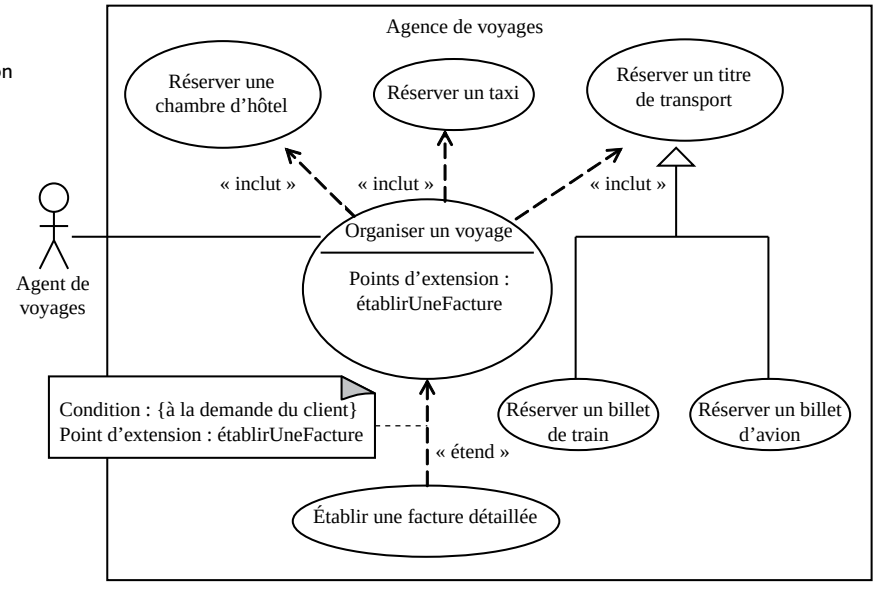


Figure 1.21
Relation de généralisation entre cas d'utilisation.

Exercice 4 : Identification des acteurs, recensement des cas d'utilisation et relations simples entre cas

Modélisez avec un diagramme de cas d'utilisation le fonctionnement d'un distributeur automatique de cassettes vidéo dont la description est donnée ci-après.

Une personne souhaitant utiliser le distributeur doit avoir une carte magnétique spéciale. Les cartes sont disponibles au magasin qui gère le distributeur. Elles sont créditées d'un certain montant en euros et rechargeables au magasin. Le prix de la location est fixé par tranches de 6 heures (1 euro par tranche). Le fonctionnement du distributeur est le suivant : le client introduit sa carte ; si le crédit est supérieur ou égal à 1 euro, le client est autorisé à louer une cassette (il est invité à aller recharger sa carte au magasin sinon) ; le client choisit une cassette et part avec ; quand il la ramène, il l'introduit dans le distributeur puis insère sa carte ; celle-ci est alors débitée ; si le montant du débit excède le crédit de la carte, le client est invité à régulariser sa situation au magasin et le système mémorise le fait qu'il est débiteur ; la gestion des comptes débiteurs est prise en charge par le personnel du magasin. On ne s'intéresse ici qu'à la location des cassettes, et non à la gestion du distributeur par le personnel du magasin (ce qui exclut la gestion du stock des cassettes).

Solution

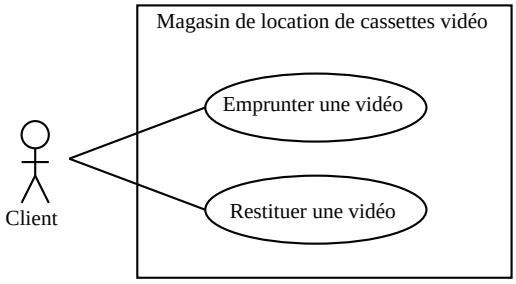
Le seul acteur est le client.

L'acquisition d'une carte et sa recharge ne se font pas *via* le distributeur : il faut aller au magasin. Ces fonctions ne donnent pas lieu à des cas d'utilisation.

Il ne faut pas faire apparaître un séquencement temporel dans un diagramme de cas d'utilisation. On ne fait donc pas figurer les étapes successives telles que l'introduction de la carte puis le choix d'une cassette, etc. Ce niveau de détails apparaîtra quand on décrira les cas d'utilisation (sous forme textuelle par exemple). Dans un diagramme de cas d'utilisation, il faut rester au niveau des grandes fonctions et penser en termes de transactions (une transaction est une séquence d'opérations qui fait passer un système d'un état cohérent initial à un état cohérent final).

Il n'y a donc que deux cas : « Emprunter une vidéo » et « Restituer une vidéo » (figure 1.22).

Figure 1.22
Diagramme de cas d'utilisation d'un distributeur de cassettes vidéo.

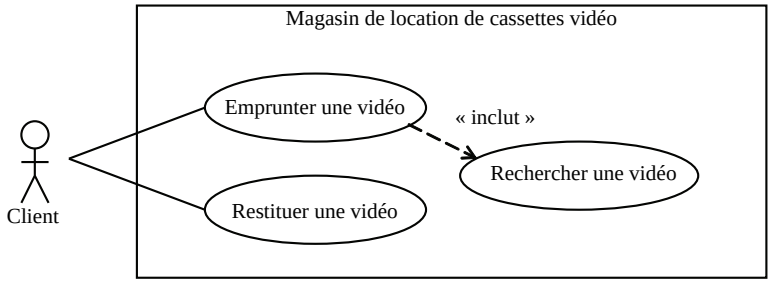


Nom de l'acteur	Rôle
Client	Représente un client du magasin de location de cassettes vidéo.

Pour décrire le cas « Emprunter une vidéo », imaginons un scénario possible. Le client introduit sa carte. Il doit ensuite pouvoir choisir une vidéo. Quels sont les critères de choix ? L'énoncé ne précise pas ces critères. Ce problème arrive fréquemment en situation réelle.

Le maître d'ouvrage dans un projet informatique de cette ampleur est bien souvent le propriétaire du magasin de location. Il sait rarement rédiger un cahier des charges. C'est le rôle du maître d'œuvre d'obliger le maître d'ouvrage à bien formuler ses besoins. Choisir une vidéo peut être complexe : la recherche se fait-elle par genres, par titres ou par dates de sortie des films en salles ? Si la recherche se fait par genres, quels sont-ils ? Rechercher un film semble plus complexe qu'on ne l'imaginait au premier abord. De plus, cette fonctionnalité peut être isolée de la location proprement dite, qui concerne plutôt la gestion de la carte. Ces remarques incitent à créer le cas supplémentaire « Rechercher une vidéo ». L'emprunt d'une vidéo inclut sa recherche. Une relation d'inclusion intervient donc entre les cas « Emprunter une vidéo » et « Rechercher une vidéo », comme le montre la figure 1.23.

Figure 1.23
Diagramme de cas d'utilisation complété par la recherche d'une vidéo.



À ce point de la modélisation, deux remarques s'imposent :

- Le succès d'UML en tant que langage de modélisation s'explique, entre autres, par le fait qu'il oblige le modélisateur à poser les bonnes questions au bon moment ; la modélisation vient à peine de commencer que déjà des questions se posent. Il faut cependant veiller à rester au niveau du recueil des besoins et des spécifications et ne faire aucun choix de conception du système. Il faut se contenter de décrire les fonctions du système sans chercher à savoir comment les réaliser.
- Le problème des critères de recherche a conduit à une révision du diagramme de cas d'utilisation. Cet « aller-retour » sur les modèles est nécessaire. Chacun d'eux apporte des informations complémentaires qui peuvent remettre en cause les modèles existants. Une fois tous les modèles établis, la modélisation sera alors aboutie.

Exercice 5 : Description d'un cas d'utilisation

Décrivez sous forme textuelle les cas d'utilisation « Emprunter une vidéo » et « Rechercher une vidéo » du diagramme présenté à la figure 1.23. La recherche d'une vidéo peut se faire par genres ou par titres de film. Les différents genres sont action, aventure, comédie et drame. Quand une liste de films s'affiche, le client peut trier les films par titres ou par dates de sortie en salles.

Solution

Avant de présenter la solution, donnons quelques indications :

- Tous les cas d'utilisation d'un système doivent être décrits sous forme textuelle (dans la suite de ce chapitre, nous omettrons éventuellement de le faire pour des raisons de place ou d'intérêt).
- Quand une erreur (exception) est détectée dans un cas, une séquence d'erreurs est activée (par exemple, voir la séquence E1 dans la description suivante). La séquence nominale n'est pas reprise et le cas s'interrompt aussitôt.

Description du cas « Emprunter une vidéo »

Identification
Nom du cas : « Emprunter une vidéo ».
But : décrire les étapes permettant au client du magasin d'emprunter une cassette vidéo *via* le distributeur automatique.
Acteur principal : Client.
Acteur secondaire : néant.
Date de création : le 31/12/2004.
Date de mise à jour : le 1/1/2005.
Responsable : M. Dupont.
Version : 1.1.

Séquencement
Le cas d'utilisation commence lorsqu'un client introduit sa carte.

Description du cas « Emprunter une vidéo » (suite)

Pré-conditions
Le client possède une carte qu'il a achetée au magasin.

Le distributeur est alimenté en cassettes.

Enchaînement nominal

1. Le système vérifie la validité de la carte.
2. Le système vérifie que le crédit de la carte est supérieur ou égal à 1 euro.
3. Appel du cas « Rechercher une vidéo ».
4. Le client a choisi une vidéo.
5. Le système indique, d'après la valeur de la carte, pendant combien de temps (tranches de 6 heures) le client peut garder la cassette.
6. Le système délivre la cassette.
7. Le client prend la cassette.
8. Le système rend la carte au client.
9. Le client prend sa carte.

Enchaînements alternatifs

A1 : Le crédit de la carte est inférieur à 1 euro.
L'enchaînement démarre après le point 2 de la séquence nominale :

3. Le système indique que le crédit de la carte ne permet pas au client d'emprunter une vidéo.
4. Le système invite le client à aller recharger sa carte au magasin.

La séquence nominale reprend au point 8.

Enchaînements d'exception

E1 : La carte introduite n'est pas valide.
L'enchaînement démarre après le point 1 de la séquence nominale :

2. Le système indique que la carte n'est pas reconnue.
3. Le distributeur éjecte la carte.

E2 : La cassette n'est pas prise par le client.
L'enchaînement démarre après le point 6 de la séquence nominale :

7. Au bout de 15 secondes le distributeur avale la cassette.
8. Le système annule la transaction (toutes les opérations mémorisées par le système sont défaites).
9. Le distributeur éjecte la carte.

E3 : La carte n'est pas reprise par le client.
L'enchaînement démarre après le point 8 de la séquence nominale :

9. Au bout de 15 secondes le distributeur avale la carte.
10. Le système consigne cette erreur (date et heure de la transaction, identifiant du client, identifiant du film).

E4 : Le client a annulé la recherche (il n'a pas choisi de vidéo).
L'enchaînement démarre au point 4 de la séquence nominale :

5. Le distributeur éjecte la carte.

Post-conditions
Le système a enregistré les informations suivantes :

- La date et l'heure de la transaction, à la minute près : les tranches de 6 heures sont calculées à la minute près.
- L'identifiant du client.
- L'identifiant du film emprunté.

Rubriques optionnelles

Contraintes non fonctionnelles
Le distributeur doit fonctionner 24 heures sur 24 et 7 jours sur 7.
La vérification de la validité de la carte doit permettre la détection des contrefaçons.

Contrainte liée à l'interface homme-machine
Avant de délivrer la cassette, demander confirmation au client.

Description du cas « Rechercher une vidéo »

Identification

Nom du cas : « Rechercher une vidéo ».
But : décrire les étapes permettant au client de rechercher une vidéo *via* le distributeur automatique.
Acteur principal : néant (cas interne inclus dans le cas « Emprunter une vidéo »).
Acteur secondaire : néant.
Date de création : le 31/12/2004.
Responsable : M. Dupont.
Version : 1.0.

Séquencement

Le cas démarre au point 3 de la description du cas « Emprunter une vidéo ».

Enchaînement nominal (le choix du film se fait par genres)

1. Le système demande au client quels sont ses critères de recherche pour un film (les choix possibles sont : par titres ou par genres de film).
2. Le client choisit une recherche par genres.
3. Le système recherche les différents genres de film présents dans le distributeur.
4. Le système affiche une liste des genres (les choix possibles sont action, aventure, comédie et drame).
5. Le client choisit un genre de film.
6. Le système affiche la liste de tous les films du genre choisi présents dans le distributeur.
7. Le client sélectionne un film.

Enchaînements alternatifs

A1 : Le client choisit une recherche par titres.

L'enchaînement démarre après le point 1 de la séquence nominale :

2. Le client choisit une recherche par titres.
3. Le système affiche la liste de tous les films classés par ordre alphabétique des titres.

La séquence nominale reprend au point 7.

Enchaînements d'exception

E1 : Le client annule la recherche.

L'enchaînement peut démarrer aux points 2, 5 et 7 de la séquence nominale :

Appel de l'exception E4 du cas « Emprunter une vidéo ».

Post-conditions

Le système a mémorisé le film choisi par le client.

Rubriques optionnelles

Contraintes non fonctionnelles

Contraintes liées à l'interface homme-machine

Quand une liste de films s'affiche, le client peut trier la liste par titres ou par dates de sortie en salles.
Le client peut se déplacer dans la liste et la parcourir de haut en bas et de bas en haut.
Ne pas afficher plus de 10 films à la fois dans la liste.

Exercice 6 : Relations d'extension entre cas d'utilisation, regroupement de cas d'utilisation en paquetages

Modélisez à l'aide d'un diagramme de cas d'utilisation un système informatique de pilotage d'un robot à distance.

Le fonctionnement du robot est décrit ci-après.

Un robot dispose d'une caméra pour filmer son environnement. Il peut avancer et reculer grâce à un moteur électrique capable de tourner dans les deux sens et commandant la rotation des roues. Il peut changer de direction car les roues sont directrices. Il est piloté à distance : les images prises par la caméra sont envoyées vers un poste de télépilotage. Ce dernier affiche l'environnement du robot sur un écran. Le pilote visualise l'image et utilise des commandes pour contrôler à distance les roues et le moteur du robot. La communication entre le poste de pilotage et le robot se fait *via* des ondes radio.

Solution

Le système informatique est composé de deux sous-systèmes :

- le sous-système du robot ;
- le sous-système du poste de télépilotage.

D'où l'idée de faire deux diagrammes de cas d'utilisation – un par sous-système – et de placer chaque diagramme dans un paquetage. La figure 1.24 montre deux paquetages : un pour le sous-système du robot et un pour le sous-système du poste de pilotage. La relation de dépendance entre les paquetages signifie que le système de pilotage utilise le robot.

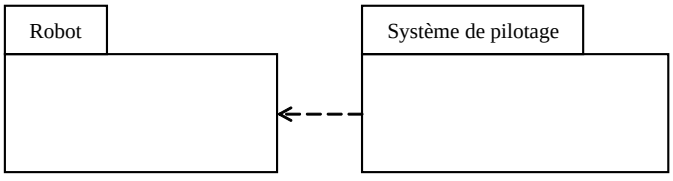


Figure 1.24
Représentation du système de télépilotage d'un robot par des paquetages.

Commençons par modéliser le robot. Ses capteurs (caméra, moteur et roues) sont à l'extérieur du système informatique et ils interagissent avec lui. Ils correspondent *a priori* à la définition d'acteurs. Reprenons chaque capteur pour l'étudier en détail :

- Le système doit demander la capture d'une image à la caméra et réaliser la capture. La caméra est donc un acteur associé à un cas d'utilisation appelé « Capturer images » (figure 1.25).
- Le sens de rotation du moteur peut être commandé. Le moteur est l'acteur ; il est associé à un cas appelé « Commander le moteur ».

- La direction des roues peut être modifiée, d'où la création du cas d'utilisation « Commander la direction » relié à l'acteur Direction.

Pour pouvoir envoyer les images au poste de pilotage et recevoir les commandes en retour, il faut un capteur supplémentaire, émetteur/récepteur d'ondes radio. Le système informatique ne va pas se charger d'envoyer et de recevoir des ondes radio – c'est le rôle des périphériques d'émission et de réception – mais il doit s'occuper du transcodage des images pour les envoyer *via* des ondes. Le système informatique doit-il réaliser lui-même ce transcodage ou bien les fonctions de transcodage sont-elles fournies avec les périphériques ? Pour répondre à cette question, il faudrait réaliser une étude de marché des émetteurs/récepteurs radio. Cela dépasse le cadre de cet ouvrage. Considérons que le système informatique intervient, ne serait-ce que pour appeler des fonctions de transcodage. Cela constitue un cas d'utilisation. Deux flots d'informations distincts doivent être envoyés au poste de pilotage : des images et des commandes. Cette dernière remarque incite à créer deux cas d'utilisation : un pour émettre des images (« Envoyer les images ») et un pour recevoir les commandes (« Recevoir les commandes »). En outre, selon l'utilisation du robot, la transmission des images s'effectue plus ou moins vite : si les déplacements du robot sont rapides par exemple, la transmission doit l'être aussi. Ces contraintes de réalisation font partie des spécifications techniques du système. Elles doivent figurer dans la description textuelle du cas d'utilisation. Sur le diagramme de cas d'utilisation, il est possible de placer une relation d'extension entre les cas « Envoyer les images » et « Capturer images », en indiquant comme point d'extension à quel moment de la capture et à quelle fréquence sont envoyées les images.

Nom de l'acteur	Rôle
Caméra	Permet de capturer des images de l'environnement du robot.
Direction	Permet de diriger les roues du robot.
Moteur	Permet de faire avancer ou reculer le robot.
Émetteur	Permet d'envoyer des ondes radio.
Récepteur	Permet de recevoir des ondes radio.

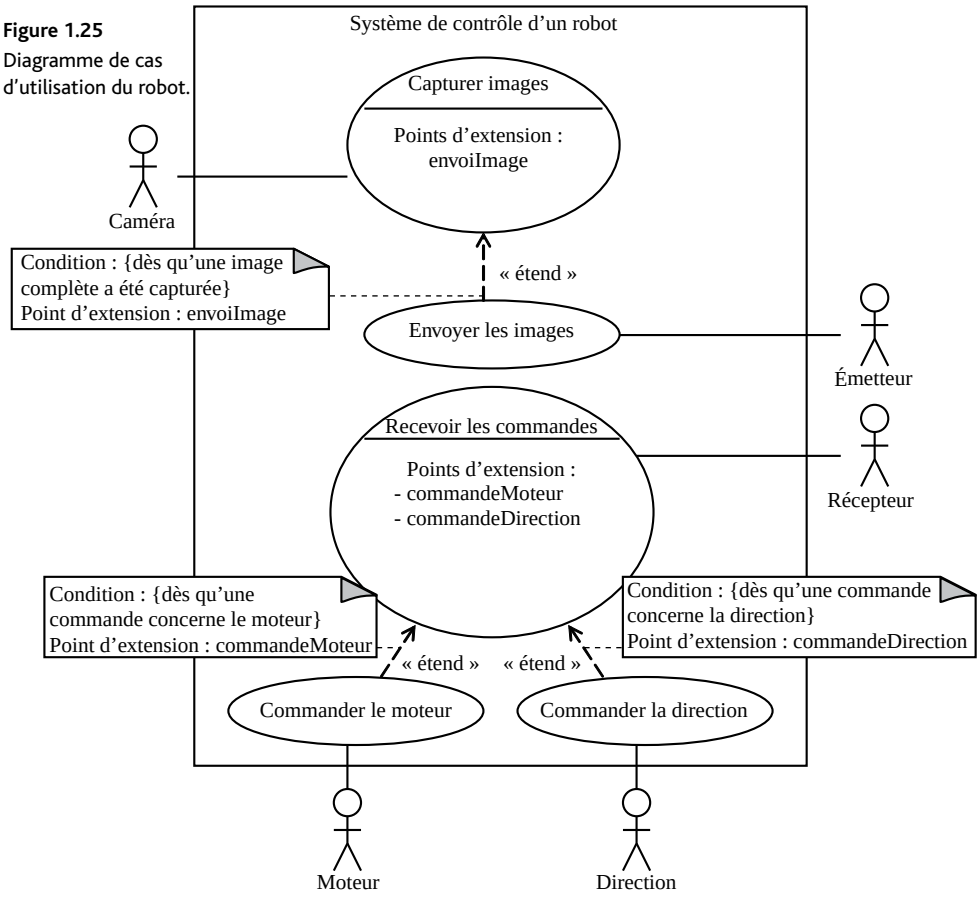


Figure 1.25
Diagramme de cas d'utilisation du robot.

Intéressons-nous à présent au sous-système de pilotage. La figure 1.26 présente le diagramme de cas d'utilisation, qui se déduit sans problème du diagramme précédent.

Nom de l'acteur	Rôle
Pilote	Représente un pilote qui agit sur les commandes à distance.
Émetteur	Permet d'envoyer des ondes radio.
Récepteur	Permet de recevoir des ondes radio.

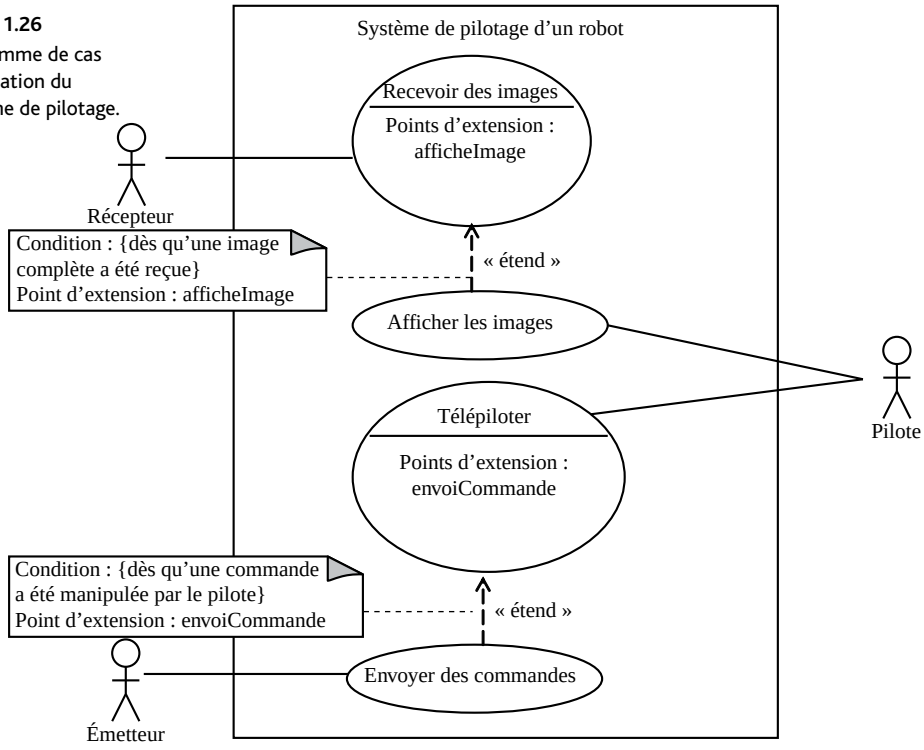


Figure 1.26
Diagramme de cas d'utilisation du système de pilotage.

Exercice 7 : Relations entre acteurs, extensions conditionnelles entre cas d'utilisation

Modélisez à l'aide d'un diagramme de cas d'utilisation une médiathèque dont le fonctionnement est décrit ci-après.

Une petite médiathèque n'a qu'une seule employée qui assume toutes les tâches :

- la gestion des œuvres de la médiathèque ;
- la gestion des adhérents.

Le prêt d'un exemplaire d'une œuvre donnée est limité à trois semaines. Si l'exemplaire n'est pas rapporté dans ce délai, cela génère un contentieux. Si l'exemplaire n'est toujours pas rendu au bout d'un an, une procédure judiciaire est déclenchée.

L'accès au système informatique est protégé par un mot de passe.

Solution

La médiathèque n'emploie qu'une employée. Néanmoins, un acteur est déterminé par le rôle qu'il joue vis-à-vis du système à modéliser. Ici, l'employée a deux rôles essentiels :

- le rôle de bibliothécaire qui gère les œuvres ainsi que les adhérents ;
- le rôle de gestionnaire des contentieux ayant les connaissances juridiques suffisantes pour déclencher des procédures judiciaires.

Ces rôles sont modélisés par deux acteurs : Bibliothécaire et Gestionnaire des contentieux. Un gestionnaire de contentieux est un bibliothécaire avec pouvoir. Les acteurs correspondants sont reliés par une relation de généralisation (figure 1.27). Ainsi, l'acteur Gestionnaire des contentieux peut utiliser les cas associés à l'acteur Bibliothécaire. A contrario, l'acteur Bibliothécaire ne peut pas utiliser les cas relatifs à la gestion des contentieux.

Jusqu'à présent la médiathèque fonctionne avec une seule employée. Si, à l'avenir, plusieurs employés devenaient nécessaires, le système informatique pourrait fonctionner avec deux groupes d'utilisateurs : un premier groupe dont le rôle serait limité à celui des bibliothécaires et un deuxième groupe susceptible de gérer les contentieux en plus d'avoir un rôle de bibliothécaire. L'authentification du groupe auquel appartient un utilisateur du système doit être contrôlée par un mot de passe. La gestion des mots de passe requiert la présence d'un administrateur du système. Pour UML, peu importe si cette personne fait partie ou non du groupe des bibliothécaires ou des gestionnaires de contentieux. Comme un nouveau rôle apparaît dans le système, cela justifie la définition d'un acteur supplémentaire : Administrateur. Tous les cas d'utilisation liés aux acteurs incluent la procédure d'authentification matérialisée par le cas « S'authentifier ».

Dans le diagramme, la gestion des adhérents et la gestion des emprunts sont séparées : « Gérer les adhérents » consiste à ajouter, à supprimer ou à modifier l'enregistrement d'un adhérent dans la médiathèque, tandis que « Gérer les emprunts » consiste à prêter des exemplaires aux adhérents déjà inscrits.

La gestion des contentieux a deux degrés d'alerte :

- Un exemplaire n'a pas été rendu au bout de trois semaines.
- Un exemplaire n'a toujours pas été rapporté au bout d'un an.

Cela correspond à deux fonctionnalités distinctes puisque, dans le deuxième cas seulement, il faut déclencher une procédure judiciaire. Nous représentons cela par deux cas d'utilisation : « Gérer les contentieux » et « Déclencher une procédure judiciaire ». Ces deux cas sont liés par une relation d'extension soumise à la condition « si le retard dépasse un an ».

Nom de l'acteur	Rôle
Bibliothécaire	Représente un bibliothécaire. Il gère les œuvres, les adhérents et les emprunts.
Gestionnaire des contentieux	Représente un bibliothécaire qui peut gérer les contentieux.
Administrateur	Représente un gestionnaire des droits d'accès au système.

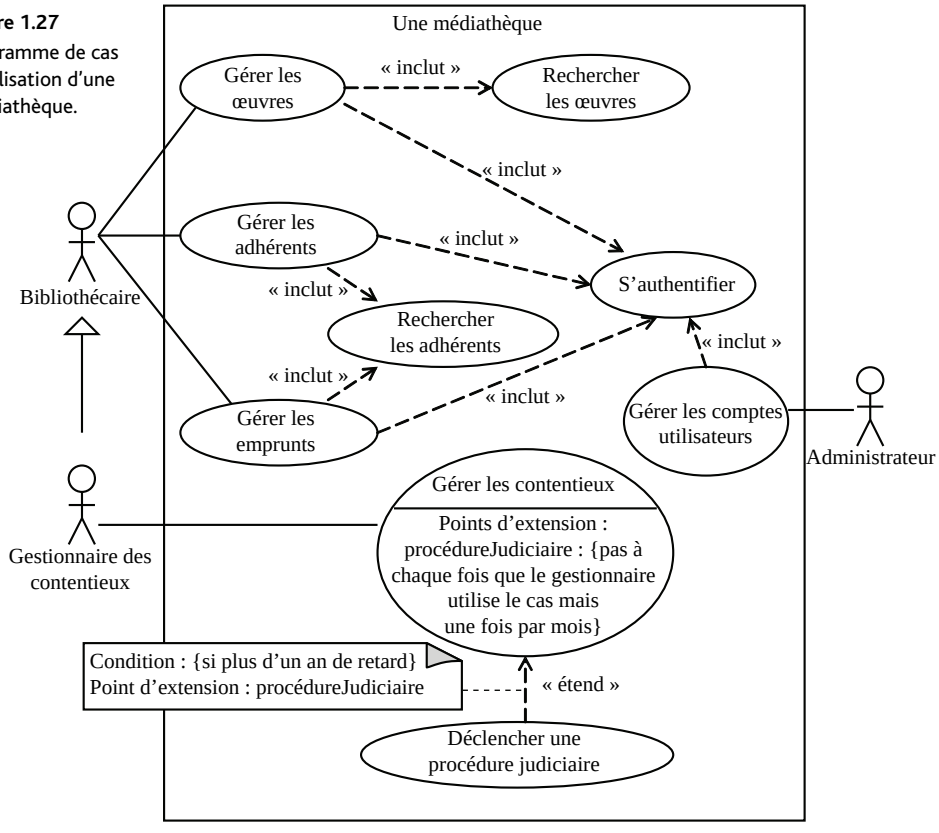


Figure 1.27
Diagramme de cas d'utilisation d'une médiathèque.

Exercice 8 : Identification des acteurs, recensement des cas d'utilisation internes et relation de généralisation entre cas

Modélisez à l'aide d'un diagramme de cas d'utilisation le système informatique qui gère la distribution d'essence dans une station-service. Le fonctionnement de la distribution de l'essence est décrit ci-après.

Avant de pouvoir être utilisée par un client, la pompe doit être armée par le pompiste. La pompe est ainsi apprêtée, mais ce n'est que lorsque le client appuie sur la gâchette du pistolet de distribution que l'essence est pompée. Si le pistolet est dans son étui de rangement et si la gâchette est pressée, l'essence n'est pas pompée. La distribution de l'essence à un client est terminée quand celui-ci remet le pistolet dans son étui. La mesure de l'essence distribuée se fait par un débitmètre.

Quatre types de carburants sont proposés : diesel, sans plomb avec un indice d'octane de 98, sans plomb avec un indice d'octane de 95, et plombé.

Le paiement peut s'effectuer en espèces, par chèque ou par carte bancaire. En fin de journée, les transactions sont archivées.

Le niveau des cuves ne doit pas descendre en dessous de 5 % de la capacité maximale. Sinon les pompes ne peuvent plus être armées.

Solution

Pour un système complexe, il vaut mieux se concentrer sur les fonctions essentielles puis passer aux cas moins importants.

Identification des principaux acteurs et recensement des cas d'utilisation essentiels

L'acteur principal est évidemment le client. Il utilise le système pour obtenir de l'essence. Il est associé au cas d'utilisation « Se servir ». Rappel : dans un diagramme de cas d'utilisation, on ne détaille pas la séquence des opérations. Par exemple, le type d'essence choisi sera pris en compte quand on décrira le cas.

Imaginons le fonctionnement de la pompe : l'essence ne peut être distribuée que si la pompe est armée ; le client prend un pistolet ; sur le pupitre du pompiste est indiqué le type d'essence choisi ; le pompiste arme la pompe en appuyant sur un bouton de son pupitre, ce qui initialise la pompe.

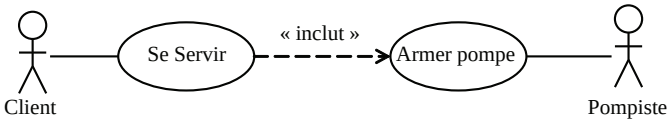
Ainsi, le cas « Se servir » doit inclure l'armement de la pompe. Deux solutions sont possibles :

- La première utilise un cas unique (Se servir), et deux acteurs (Client et Pompiste), comme le montre la figure 1.28.

Figure 1.28
Se servir de l'essence : solution avec un cas unique.

- La deuxième solution met en œuvre deux cas : « Se servir » et « Armer pompe » (figure 1.29).

Figure 1.29
Se servir de l'essence : solution avec deux cas.



L'armement de la pompe est indispensable, sinon l'essence ne peut être distribuée, d'où la relation d'inclusion entre les cas « Se servir » et « Armer pompe ». L'armement de la pompe se fait en une seule action pour le pompiste : celle d'armer la pompe en appuyant sur un bouton. La description de l'armement se résume donc à une séquence très sommaire d'actions. Faut-il alors représenter cela par un cas d'utilisation ? L'argument qui fait pencher pour le maintien du cas « Armer pompe » est que l'armement est une opération bien isolée des autres fonctions : il s'agit d'initialiser la pompe et donc de piloter des périphériques (mécaniques, électroniques…). Vu sous cet angle, l'armement est suffisamment complexe et bien isolé des autres fonctions pour en faire un cas (figure 1.31).

Le pompiste est un acteur secondaire du cas « Armer pompe » (c'est un cas interne pour lequel le pompiste est consulté). L'armement de la pompe n'est possible que si le niveau de la cuve est suffisant. Un détecteur de niveau (périphérique externe au système informatique) est nécessaire. Ce périphérique est représenté par l'acteur « Capteur niveau

cuve pour armement ». Il est secondaire car l'information sur le niveau de la cuve ne lui est pas destinée. Si le niveau est trop bas, c'est le pompiste qui doit en être informé. Il saura ainsi ce qui empêche l'armement de la pompe. La vérification du niveau de la cuve est importante pour le système. De plus, cette opération constitue une transaction bien isolée des autres fonctions (il s'agit de contrôler un périphérique matériel). C'est la raison pour laquelle on décide de créer un cas « Vérifier niveau cuve pour armement ». Pour transmettre le niveau de la cuve au pompiste, il faut relier l'acteur Pompiste au cas « Vérifier niveau cuve pour armement ». Le pompiste est informé que le niveau est trop bas, mais la vérification doit être automatique pour que les pompes soient éventuellement bloquées. Une relation d'inclusion intervient donc entre les cas « Armer pompe » et « Vérifier niveau cuve pour armement » (figure 1.31).

Recensement des cas de moindres importances

Après avoir trouvé les principaux cas, il est temps de se consacrer à ceux de moindre importance. Il ne faut pas oublier de couvrir tous les besoins et ne pas hésiter à introduire des cas auxquels le maître d'ouvrage n'a pas pensé. Il validera ou pas le modèle a posteriori.

L'énoncé n'aborde pas le problème du remplissage des cuves d'essence. Cette opération est réalisée par une entreprise tierce de livraison d'essence. Elle doit cependant être consignée dans le système informatique de la station-service. Une solution consiste à alerter le pompiste dès que le niveau des cuves descend au-dessous d'un certain seuil. Ce deuxième seuil, appelé « seuil de remplissage des cuves », doit être supérieur au seuil des 5 % de l'énoncé (celui qui empêche les pompes d'être armées). Quand il est atteint, le pompiste prévient l'entreprise de livraison d'essence de sorte à éviter de tomber au seuil des 5 %. Si la station-service est sur une autoroute, la livraison d'essence doit être garantie 24 heures sur 24. Il faut peut-être contacter la société de livraison automatiquement – sans passer par l'intermédiaire du pompiste – dès que le niveau devient trop bas. Le capteur du niveau de remplissage est représenté par un acteur appelé « Capteur niveau cuve pour remplissage » (figure 1.31). Il est associé au cas « Vérifier niveau cuve pour remplissage », lui-même associé à l'acteur Pompiste.

Abordons à présent le problème du paiement. De concert avec le maître d'ouvrage, le maître d'œuvre imagine un fonctionnement possible du système : dès que le client raccroche le pistolet, le montant à payer est calculé ; il s'affiche sur le pupitre du pompiste ; le client qui vient payer indique son mode de paiement (espèces, chèque ou carte bancaire) ; le pompiste sélectionne le mode de paiement choisi. À partir de là, les cas divergent :

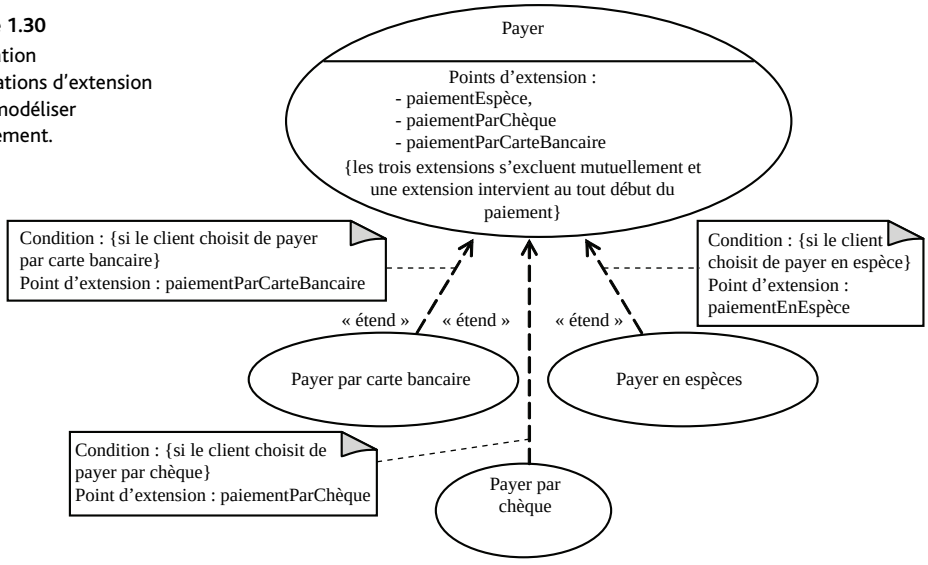
- Pour un paiement en espèces, le pompiste encaisse le montant demandé, puis valide la transaction, qui est mémorisée dans le système informatique (le montant, la date de la transaction et le mode de paiement sont conservés).
- Si le paiement se fait par chèque, ce dernier est rempli automatiquement, puis le pompiste l'encaisse. La transaction est mémorisée dans le système informatique (en plus de la date, du montant et du mode paiement, sont conservés les références d'une pièce d'identité ou le numéro d'immatriculation du véhicule).
- Pour un paiement par carte bancaire, la banque de la station-service réalise une transaction avec la banque du client. Les seules informations à conserver dans le système

informatique de la station-service sont le montant, la date de la transaction et le mode de paiement.

Comment représenter cela dans un diagramme de cas d'utilisation ? Une première solution (présentée à la figure 1.31) consiste à créer un cas général appelé « Payer », et trois cas particuliers reliés par une relation de spécialisation au cas « Payer ». Chacun des trois cas représente un mode de paiement.

Une autre solution (présentée à la figure 1.30) repose sur un cas, « Payer », qui se déroule jusqu'au choix du mode de paiement, puis, selon le type de paiement, un des trois modes est activé (« Payer en espèces », « Payer par chèque » ou « Payer par carte bancaire »). Ces trois cas sont typiquement des extensions du cas « Payer », où l'extension est soumise à condition.

Figure 1.30
Utilisation de relations d'extension pour modéliser le paiement.



Ces deux solutions sont possibles. Il est difficile de dire laquelle est la meilleure. La suite de l'exercice se fonde arbitrairement sur la représentation avec des relations de généralisation (figure 1.31). Le modélisateur se retrouve régulièrement face à ce dilemme. La réponse est souvent la même : peu importe la façon de modéliser un système du moment que le modèle est correct – un modèle est correct s'il montre une solution qui satisfait le maître d'ouvrage ainsi que les futurs utilisateurs du système.

Aboutir à une modélisation correcte

Il faut prendre le temps d'élaborer le diagramme de cas d'utilisation, bien qu'il soit généralement simple à bâtir, afin d'éviter les a priori qui peuvent conduire à une modélisation erronée.

À la relecture du fonctionnement du paiement tel qu'il est décrit précédemment, le pompiste devient l'acteur principal du cas de paiement. C'est un peu surprenant car on pourrait croire au premier abord qu'il s'agit du client. Or, la seule fois où le client intervient directement sur le système informatique de la station-service est quand il saisit son numéro de carte bancaire. Toutes les autres situations nécessite l'intervention du

pompiste. Attardons-nous un instant encore sur le paiement par carte bancaire. Il faut de toute évidence faire figurer un acteur supplémentaire qui représente la banque, car elle interagit avec le système sans en faire partie. C'est un acteur secondaire qui est sollicité uniquement pour confirmer le bon déroulement de la transaction. Quel est le rôle de cet acteur ? Le plus souvent, les solutions de paiement par carte bancaire sont disponibles clés en main sur le marché. Elles incluent un lecteur de cartes ainsi qu'un logiciel pour le piloter. Le maître d'œuvre doit proposer plusieurs solutions présentes sur le marché au maître d'ouvrage, et éventuellement une solution propriétaire si aucune solution du marché ne lui convient. Le maître d'ouvrage décidera. Dans le cadre de cet exercice, à défaut de pouvoir dialoguer avec un maître d'ouvrage, nous choisissons l'achat d'une solution clés en main pour le paiement par carte bancaire. Dans ce cas, le client, lorsqu'il saisit le code de sa carte, n'interagit plus directement avec le système informatique de la station-service. Ainsi, quel que soit le mode de paiement, tout passe par l'intermédiaire du pompiste. Le client n'est donc pas un acteur du système.

Le calcul automatique du montant à payer peut se faire dès que le client raccroche le pistolet (ce qui intervient à la fin du cas « Se servir »). Pour indiquer le moment où le montant est calculé, on peut ajouter une relation d'extension entre les cas « Se servir » et « Payer », avec un point d'extension qui précise le moment où le calcul du montant intervient, comme le montre la figure 1.31. Le paiement peut aussi intervenir avant de se servir. Dans ce cas, il est possible d'ajouter un deuxième point d'extension.

Modélisation des concepts abstraits

Parfois, le langage UML peut paraître limité. Il faut alors trouver, parmi les éléments du langage, celui qui convient le mieux à une situation donnée.

Un dernier besoin n'est pas décrit par le diagramme de cas d'utilisation : c'est l'archivage en fin de journée des transactions. Faut-il prendre en compte ce cas et, si oui, quel acteur l'utilise ? Un cas d'utilisation est déclenché par un événement. Les événements peuvent être classés en trois catégories :

- **Les événements externes.** Ils sont déclenchés par des acteurs.
- **Les événements temporels.** Ils résultent de l'atteinte d'un moment dans le temps.
- **Les événements d'états.** Ils se produisent quand quelque chose dans le système nécessite un traitement.

Ici, il s'agit d'un événement temporel. Il est difficile de définir un acteur qui déclencherait cet événement. Comment le temps peut-il interagir avec un système ? Cela dit, l'archivage quotidien est une fonctionnalité essentielle. Il faut la faire figurer dans la modélisation du système. À quelle étape de la modélisation doit-on prendre en compte cette fonctionnalité ? Comme nous en sommes à produire le premier modèle du système et que cette fonctionnalité est importante, nous choisissons, pour ne pas l'oublier, d'en faire un cas d'utilisation. Pour déclencher ce cas, nous introduisons un acteur appelé Timer qui, une fois par jour, déclenche un événement temporel. Timer est un acteur, il est donc en dehors du système. Cela signifie que l'heure est donnée par une horloge externe à notre système informatique (par exemple, l'horloge du système d'exploitation du système informatique).

La prise en compte des événements d'états est plus délicate puisque, par définition, ils se produisent à l'intérieur d'un système. Ils ne peuvent donc pas être déclenchés par un

acteur, qui est forcément en dehors du système. Il est toutefois possible qu'un événement d'état active un cas d'utilisation à condition que ce cas soit interne au système. Une façon de représenter un cas de ce type consiste à le relier à d'autres cas *via* des relations d'extension et à faire figurer comme condition d'extension l'événement d'état. C'est cette solution qui a été adoptée entre les cas « Se servir » et « Payer », en considérant ce dernier comme un cas interne vis-à-vis du cas « Se servir ».

Nom de l'acteur	Rôle
Client	Acteur principal du cas d'utilisation « Se servir ». Représente le client qui se sert de l'essence.
Pompiste	Acteur principal des cas « Payer » et « Vérifier niveau cuve pour remplissage ». Acteur secondaire pour le cas « Armer pompe ».
Capteur niveau cuve pour armement	Acteur secondaire du cas « Vérifier niveau cuve pour armement ».
Capteur niveau cuve pour remplissage	Acteur secondaire du cas « Vérifier niveau cuve pour remplissage ».
Timer	Acteur secondaire du cas « Archiver les transactions ».
Banque	Acteur secondaire du cas « Payer par carte bancaire ».

Figure 1.31
Diagramme de cas d'utilisation d'une station-service.