

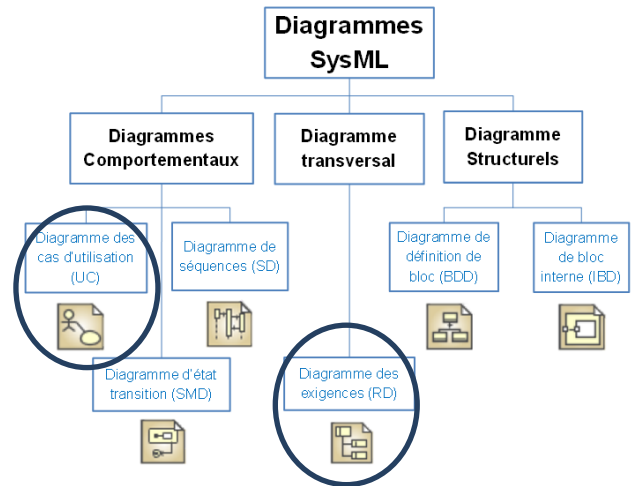
INGENIERIE SYSTEMES :

Compréhension - Exploitation des diagrammes SysML (Systems Modeling Language)

La modélisation fonctionnelle

La modélisation fonctionnelle d'un système est définie par :

- Le diagramme des cas d'utilisation
- Le diagramme des exigences



1. Diagramme des cas d'utilisation

Un cas d'utilisation représente une manière spécifique d'utiliser un système. Il correspond donc à un service rendu et se traduit généralement par une séquence d'actions observables avec un déclenchement, un déroulement et une fin. **Les acteurs sont à l'extérieur du système et modélisent tous ceux qui ont recours à ces services.**

Il est important de noter qu'un cas d'utilisation décrit ce que le système devra faire, **sans spécifier comment il le fera.**

Chaque cas d'utilisation est désigné par **un verbe à l'infinitif suivi d'un complément**. Il exprime le service rendu **du point de vue de l'acteur (et non du système)**. Il s'inscrit à l'intérieur d'une ellipse reliée à l'acteur par un trait.

Un acteur se représente par un stickman ayant son nom inscrit dessous. Le nom dépend du **rôle** joué par l'acteur en tant qu'utilisateur.

Remarque :

Pour une question de lisibilité, il est préférable de limiter le nombre de cas d'utilisation de base.

Les acteurs candidats sont systématiquement :

- les utilisateurs humains directs (administrateur, opérateur de maintenance ...) ;
- les autres systèmes connexes qui interagissent directement avec le système étudié.

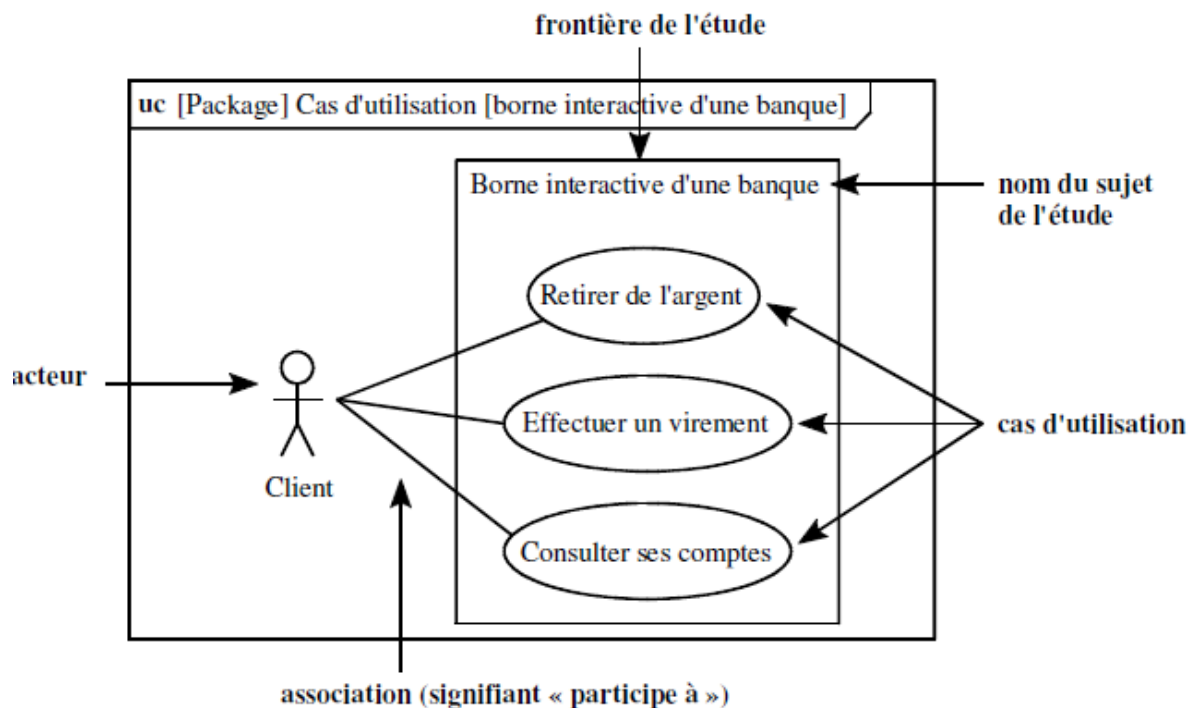
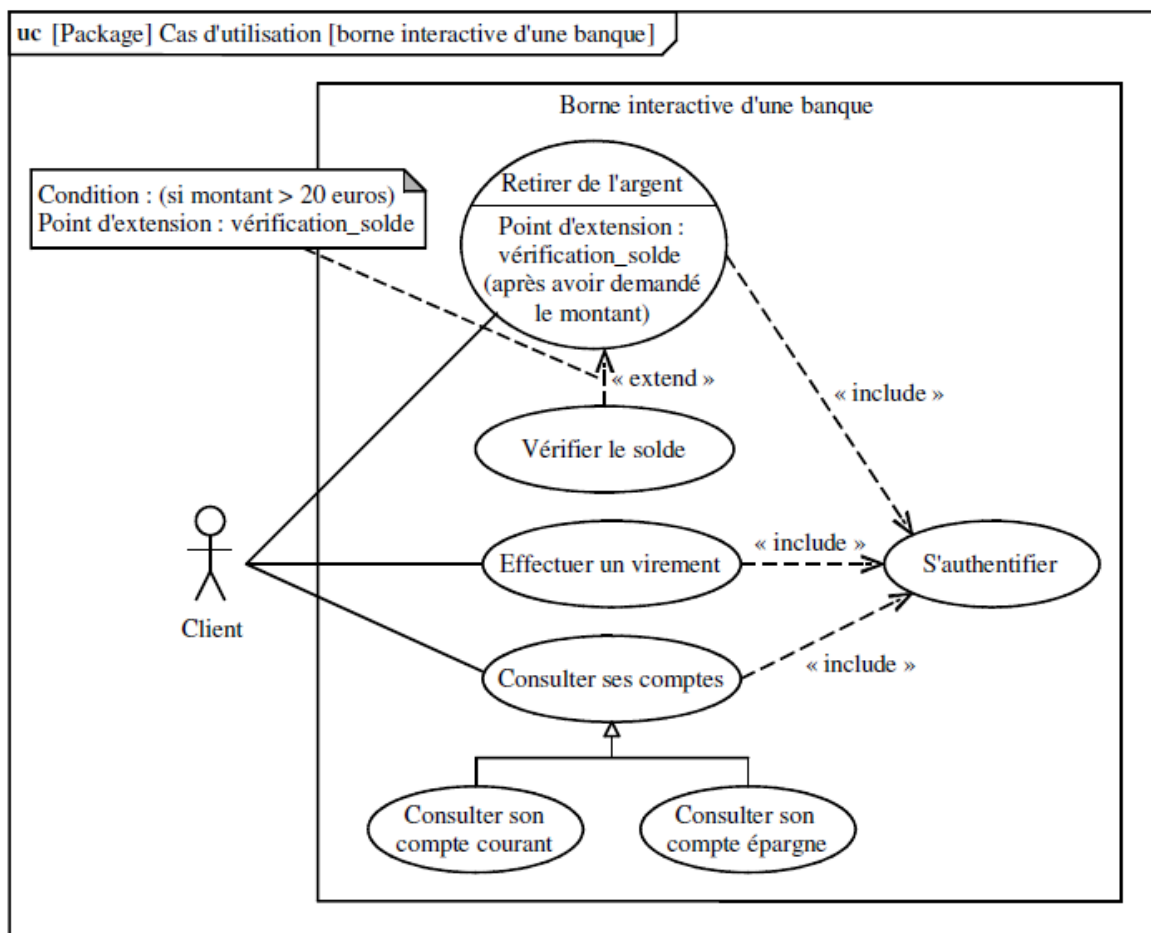


Diagramme de cas d'utilisation modélisant une borne d'accès à une banque

1.1. Relations entre cas particuliers

Pour clarifier ce diagramme(cas d'utilisation), SysML permet d'établir trois types de relations entre les cas d'utilisation.

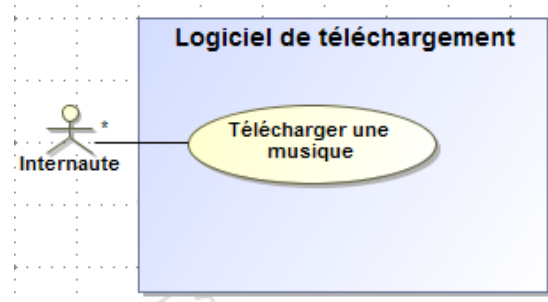
- une **relation d'inclusion** précisée par le **stéréotype « include »**. Un cas A est inclus dans un cas B si la réalisation A entraîne obligatoirement celle de B. Par exemple, l'accès aux informations d'un compte bancaire inclut nécessairement une phase d'authentification avec un code d'accès.
- une **relation d'extension** précisée par le **stéréotype « extend »**. Un cas B peut être étendu au cas A si la réalisation de A entraîne, sous certaine condition, celle de B. La réalisation de B n'est donc plus systématique et dépend d'une condition appelée « extension point ». Par exemple, le retrait d'argent entraîne la vérification du solde si le montant dépasse 20 euros.
- une relation de **généralisation / spécialisation** représentée par une flèche blanche. Un cas A est une généralisation d'un cas B si B est un cas particulier de A. Les cas d'utilisation descendants héritent de la description par leur parent commun. Par exemple, consulter son compte courant ou son compte épargne sont des cas particuliers de consulter ses comptes.



1.2. Multiplicité

Un acteur peut avoir recours plusieurs fois au même cas d'utilisation.

Le diagramme ci-contre montre un internaute qui télécharge plusieurs morceaux de musique sur Internet.



Le symbole * indiqué sur la liaison signifie « plusieurs », il représente une **multiplicité**.

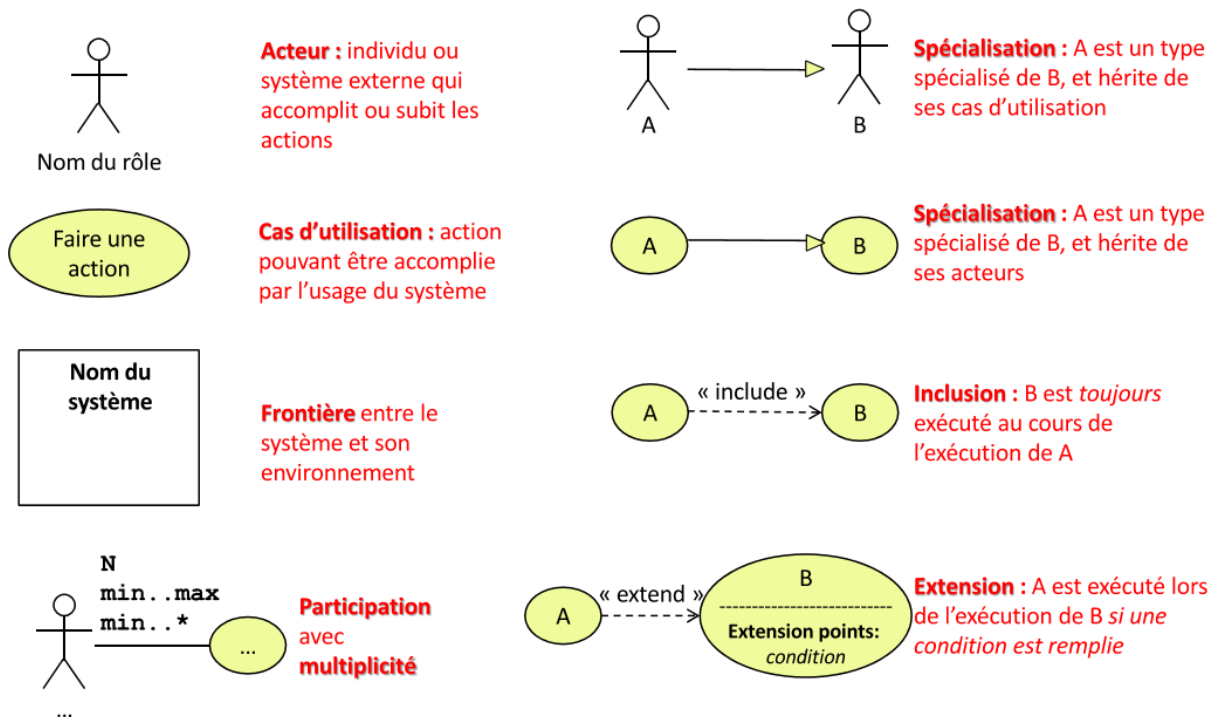
Une multiplicité est un intervalle compris entre une borne inférieure et une borne supérieure :

- la borne inférieure peut-être 0 (optionnelle) ou n'importe quel entier positif ;
- la borne supérieure peut être 1, plusieurs (noté *), ou un entier positif.

Remarque :

Si les bornes sont égales, on n'écrit qu'une valeur et la valeur par défaut en SysML est 1.

1.3. Syntaxe



2. Diagramme des exigences

2.1. Caractéristiques

Une exigence sert à spécifier une capacité ou une contrainte qui doit être satisfaite par un système.

Elle peut spécifier une fonction que le système devra réaliser ou une condition de performance, de fiabilité, de sécurité, etc.

Les exigences servent à établir un contrat entre le client et les réalisateurs du futur système.

Les deux propriétés de base d'une exigence sont :

- un identifiant unique (pour ensuite gérer la traçabilité avec l'architecture...) ;
- un texte descriptif comportant obligatoirement le verbe « doit ».

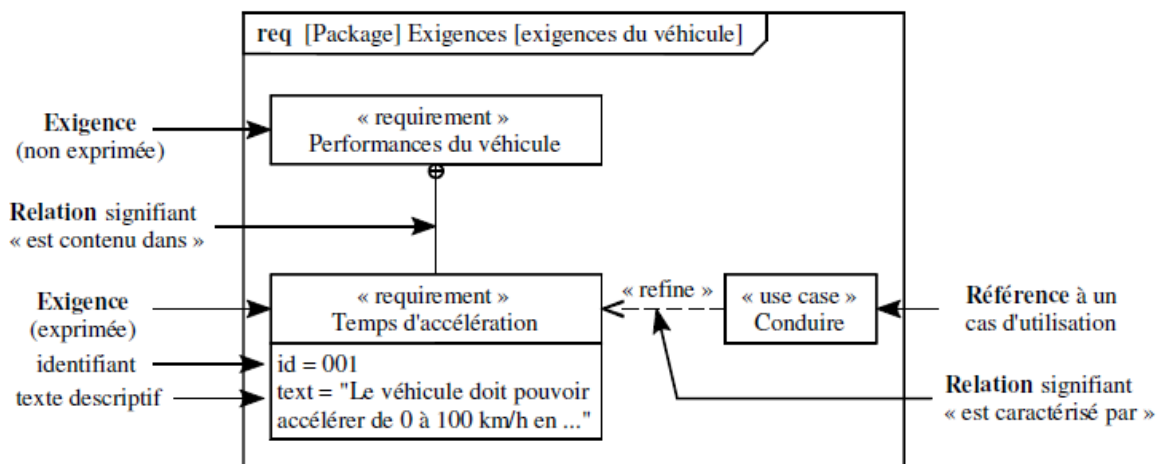


Diagramme partiel des exigences d'un véhicule montrant une exigence particulière

Il est courant de définir d'autres propriétés pour les exigences, typiquement :

- priorité (par exemple : haute, moyenne, basse) ;
- source (par exemple : client, marketing, technique, législation...) ;
- risque (par exemple : haut, moyen, bas) ;
- statut (par exemple : proposé, validé, implémenté, testé, livré ...) ;
- méthode de vérification (par exemple : analyse, démonstration, test...).

Exigences du véhicule avec propriétés :

« requirement » Temps d'accélération
id = 001 text = "Le véhicule doit pouvoir accélérer de 0 à 100 km/h en ..."
priorité : haute risque : moyen source : marketing statut : validée

Les exigences peuvent être reliées entre elles par des relations de contenance, de raffinement ou de dérivation.

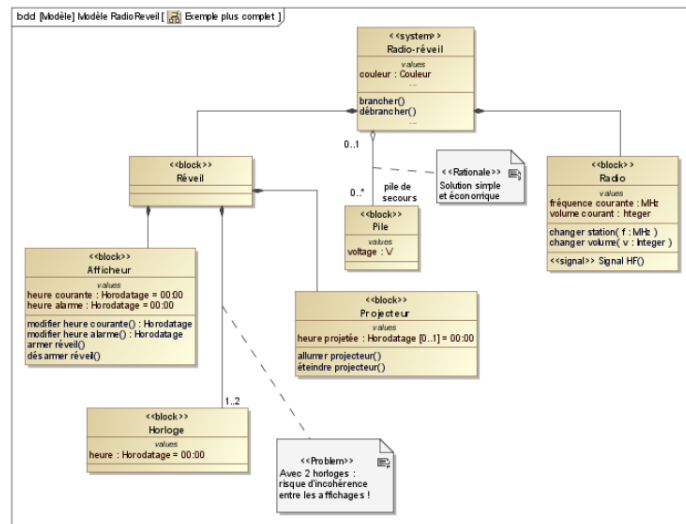
- **La contenance** (ligne terminée par un cercle contenant une croix du côté du contenant) sert à décomposer une exigence composite en plusieurs exigences unitaires, plus faciles ensuite à tracer vis-à-vis de l'architecture ou des tests.
- **Le raffinement** (« refine ») consiste en l'ajout de précisions, par exemple des données quantitatives.
- **La dérivation** (« deriveReq ») consiste à relier des exigences de niveaux différents, par exemple des exigences système à des exigences de niveau sous-système, etc. Elle implique généralement des choix d'architecture.

2.2. Compléments

SysML permet d'utiliser des notes graphiques sur tous les types de diagrammes (forme de Post-it).

Deux mots-clés particuliers ont été définis afin de représenter :

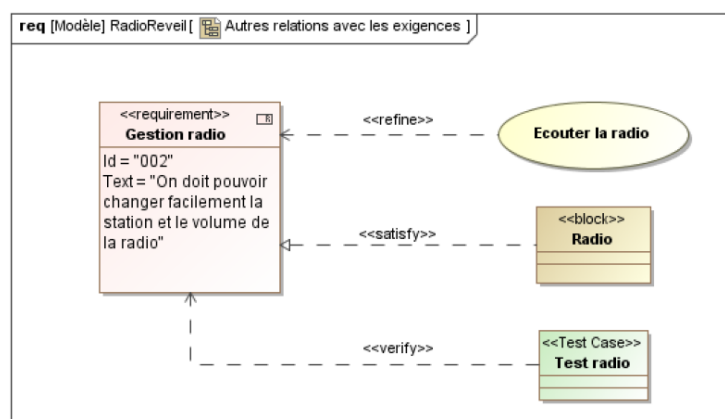
- des problèmes à résoudre (« problem ») ;
- des justificatifs (« rationale »).



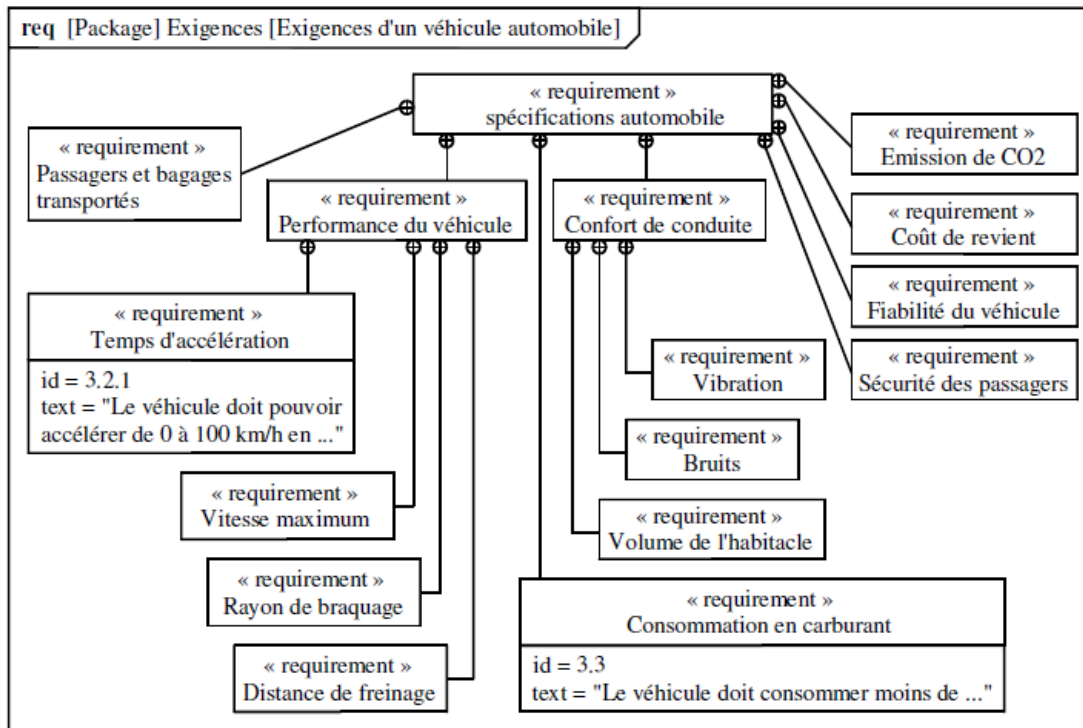
2.3. Traçabilité des exigences

Pour faciliter la gestion des exigences au cours du déroulement du projet, les exigences sont reliées avec d'autres types d'éléments SysML par plusieurs relations.

- La relation entre une exigence et un élément comportemental (cas d'utilisation, diagramme d'états...) se note par le mot-clé « **refine** ».
- La relation entre une exigence et un bloc d'architecture se note par le mot-clé « **satisfy** ».
- La relation entre une exigence et un cas de test se note par le mot-clé « **verify** ». Un cas de test représente une méthode de vérification de la satisfaction d'une exigence. Il est représenté en SysML par un rectangle avec le mot-dé « **Test Case** ».

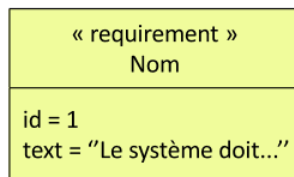


Exemple de relation entre les exigences



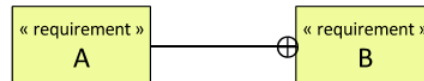
Exemple de hiérarchisations entre les exigences

2.4. Syntaxe



Exigence : qualité ou caractéristique attendue du système ou de son comportement

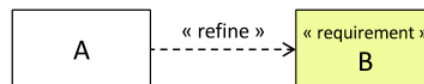
Justification de l'exigence ou de la relation désignée



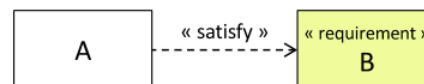
Inclusion : B contient A (la satisfaction de A est nécessaire à celle de B)



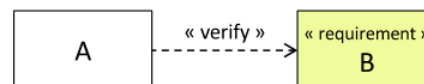
Dérivation : A a été identifiée comme un moyen de satisfaire B



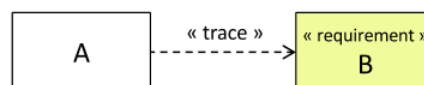
Raffinement : A raffine B (B concerne A)



Satisfaction : les propriétés réalisées de A sont conformes à B



Vérification : A permet de déterminer si B est réalisée ou non



Traçabilité : A est à l'origine de B