	Description fonctionnelle des systèmes	
	Langage de modélisation graphique de système : SysML	Support de Cours
Connaissances visées : Approche fonctionnelle des systèmes, Représentations symboliques.		

1. Pourquoi SysML ?

De la même façon qu'il vaut mieux dessiner une maison avant de la construire, il vaut mieux modéliser un système avant de le réaliser.

Un système est un ensemble de constituants inter-reliés qui interagissent les uns avec les autres d'une manière organisée pour accomplir une finalité commune (NASA 1995).

SysML est l'acronyme de Systems Modeling Language, soit Langage de Modélisation de Systèmes.

SysML apporte une standardisation du vocabulaire. Ce nouveau langage, ajoute aussi la possibilité de représenter les exigences du système comme elles sont définies dans un cahier des charges, les éléments non-logiciels (mécanique, hydraulique, capteur...), les équations physiques, les flux continus (matière, énergie, etc.) et les allocations.

En résumé, l'intérêt de SysML est :

- Obtenir une modélisation de très haut niveau indépendante des langages et des environnements.
- Faire collaborer des participants de tous horizons autour d'un même document de synthèse.
- Générer des simulations de comportement d'un système.
- Documenter un projet.

2. Qui utilise SysML aujourd'hui ?



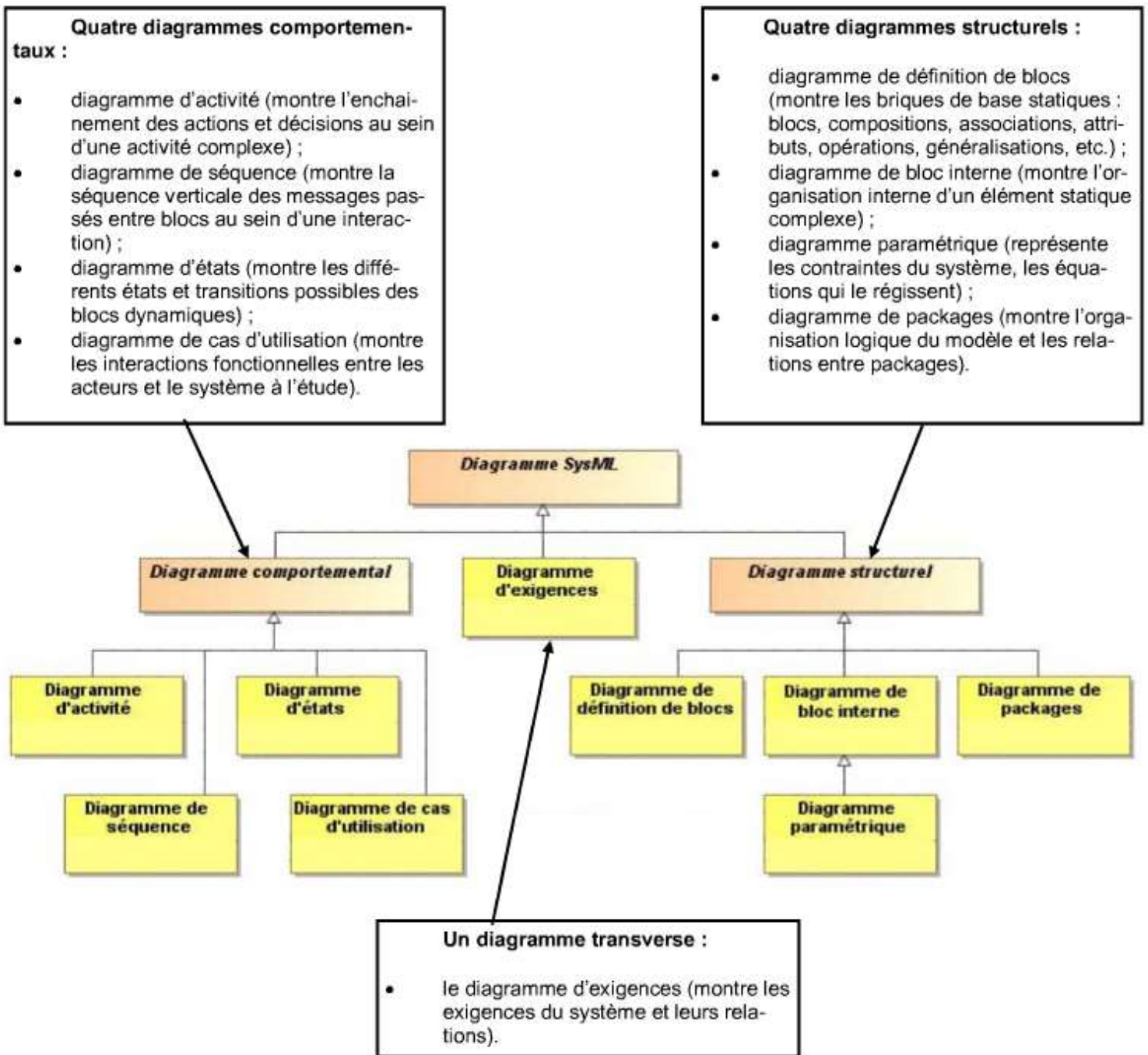
Airbus, CNES, NASA, Renault, BMW, VEGA Space GmbH, MIT Lincoln Laboratory, Lockheed Martin, US Army, ESO (European Organisation for Astronomical Research), Boeing, Raytheon, Thales, ESA (European Space Agency), Sopra Group, Rockwell Collins Inc., JPL (coentreprise avec la NASA), GE Aviation, NEWTEC LLC, BAE Systems, Siemens AG, Philips, Bombardier Transportation, SKF, Peugeot SA, Cyclocity (Vélib', Velô Toulouse...), Alstom, Michelin, Segway, Valeo, Dassault Systemes et bien d'autres ...

3. SysML et les différents diagrammes

SysML est un langage de modélisation permettant de décrire tout ou partie d'un système technique, d'un point de vue structurel, d'interactions ou comportemental.

SysML s'articule autour de 9 diagrammes, chacun d'eux étant dédié à la représentation des concepts particuliers d'un système. Dans ce chapitre, nous en étudierons 3 :

- Diagramme de cas d'utilisation
- Diagramme d'exigences
- Diagramme de séquence



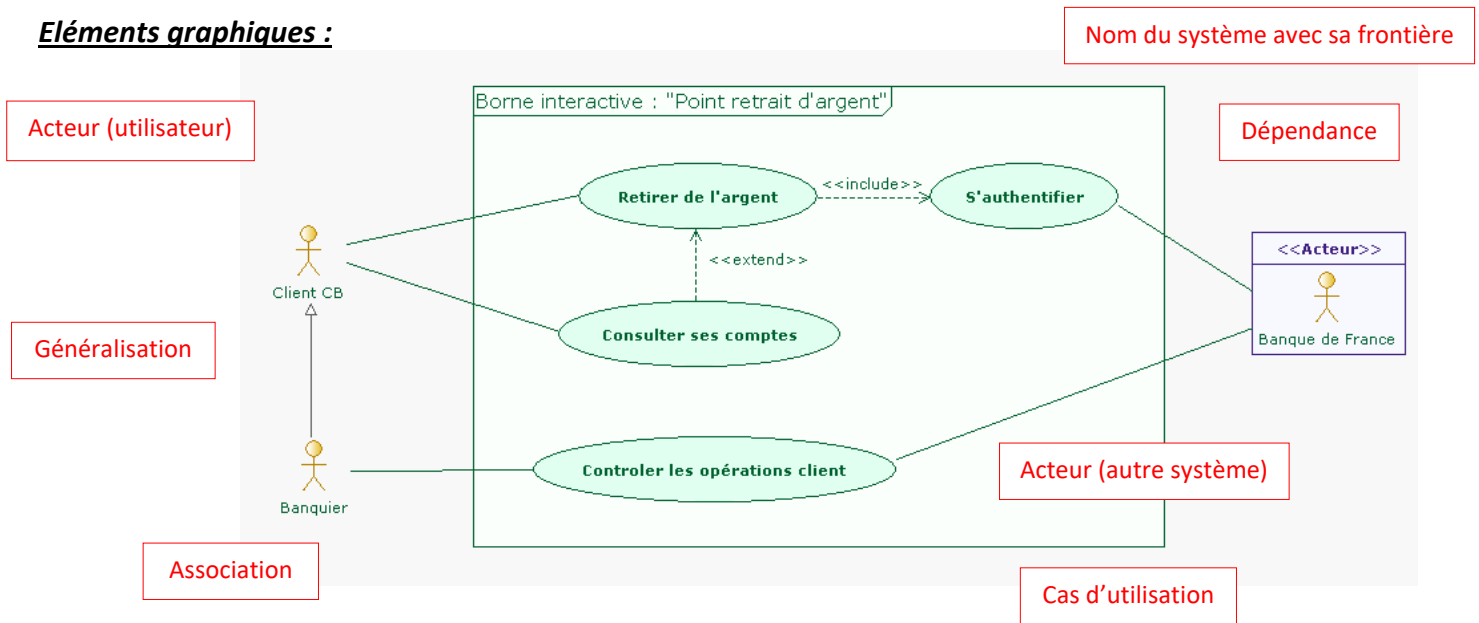
3.1 - Diagramme des Cas d'Utilisation (Use Case).

Il s'agit souvent du point de départ d'une conception. Le diagramme de cas d'utilisation permet d'identifier les possibilités d'interaction entre le système et les acteurs (intervenants extérieurs au système), c'est-à-dire toutes les fonctionnalités que doit fournir le système.

Ce type de diagrammes comporte :

- Des cas d'utilisation ;
- Des acteurs ;
- Des relations de dépendance, de généralisation et d'association.

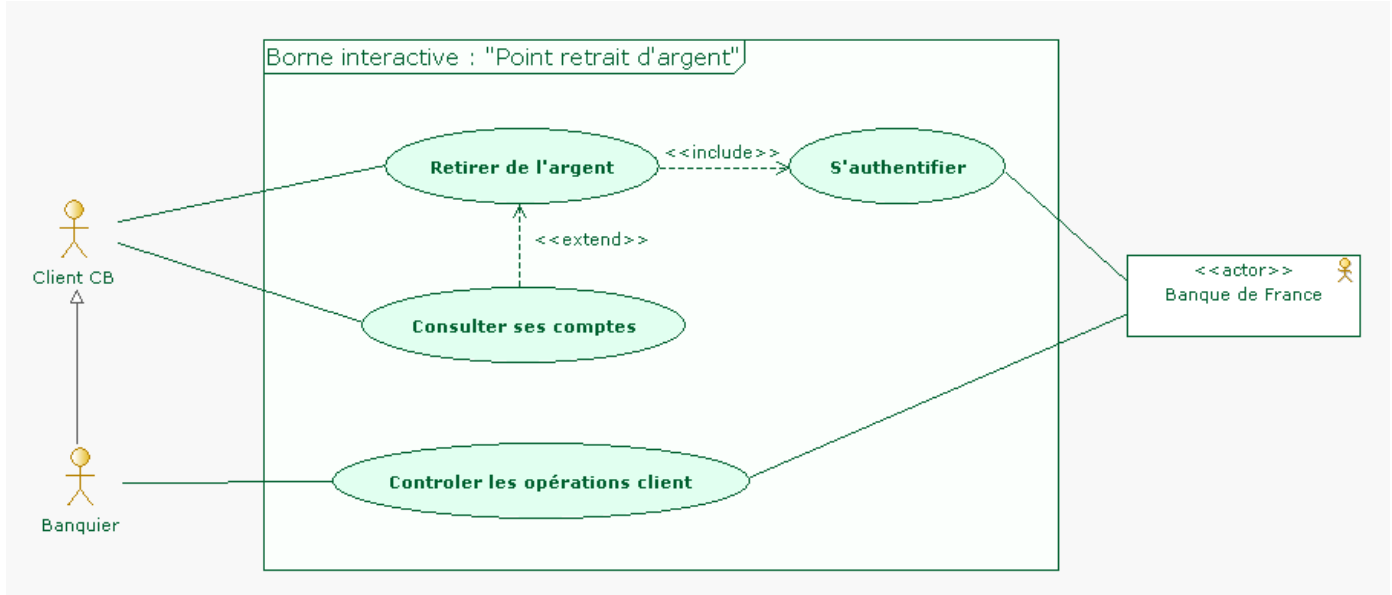
Eléments graphiques :



Interprétation des éléments graphiques :

<p>Les acteurs</p>	<p>Rôle joué par un utilisateur humain ou un autre système qui interagit directement avec le système étudié. Un acteur participe à au moins un cas d'utilisation.</p>
<p>Les cas d'utilisation</p>	<p>« Cas d'utilisation » est une fonction ou interaction entre acteur et système, dans le but de répondre à un besoin. Un cas d'utilisation doit être relié à au moins un acteur. Il est exprimé par un verbe à l'infinitif.</p>
<p>Les relations : de dépendance,</p>	<p>Dépendance : il existe deux types de dépendance :</p> <ul style="list-style-type: none"> - Inclusion (include): un cas intègre le comportement d'un autre (le cas d'utilisation « s'authentifier » est inclus dans « retirer de l'argent »). - Extension (extend): comportement optionnel du système (à partir de « retirer de l'argent », fonction du distributeur, on peut « consulter ses comptes »). <p><i>Les dépendances permettent de décomposer un cas complexe, mais l'abus de dépendance rend la lecture complexe.</i></p>
<p>de généralisation</p>	<p>Généralisation: relation hiérarchique entre deux acteurs ou cas d'utilisation. Le banquier est le spécialiste de la banque par rapport au client. La fonction « retirer de l'argent » est le cas général du système, « Consulter ses comptes » est un cas particulier.</p>
<p>et d'association</p>	<p>Association : chemin de communication entre un acteur et un cas d'utilisation.</p>

Interprétation de l'exemple :



À partir de l'exemple donné, répondre aux questions suivantes :

🔗 **Quel est le système étudié ?**

Borne interactive : « Point retrait d'argent »

🔗 **Quel est le cas d'utilisation principal du système ?**

Retirer de l'argent

🔗 **Quels sont les autres cas d'utilisation ?**

Consulter ses comptes

Contrôler les opérations client

🔗 **Quel est l'utilisateur (acteur) principal ? (cocher la bonne réponse)**

Banquier

Client CB

Banque de France

🔗 **Quel est le rôle du banquier par rapport au client ?**

Le banquier est le spécialiste de la banque par rapport au client

🔗 **Que représente la Banque de France ? (cocher la (ou les) bonne(s) réponse(s))**

Acteur principal

Acteur secondaire

Un autre système

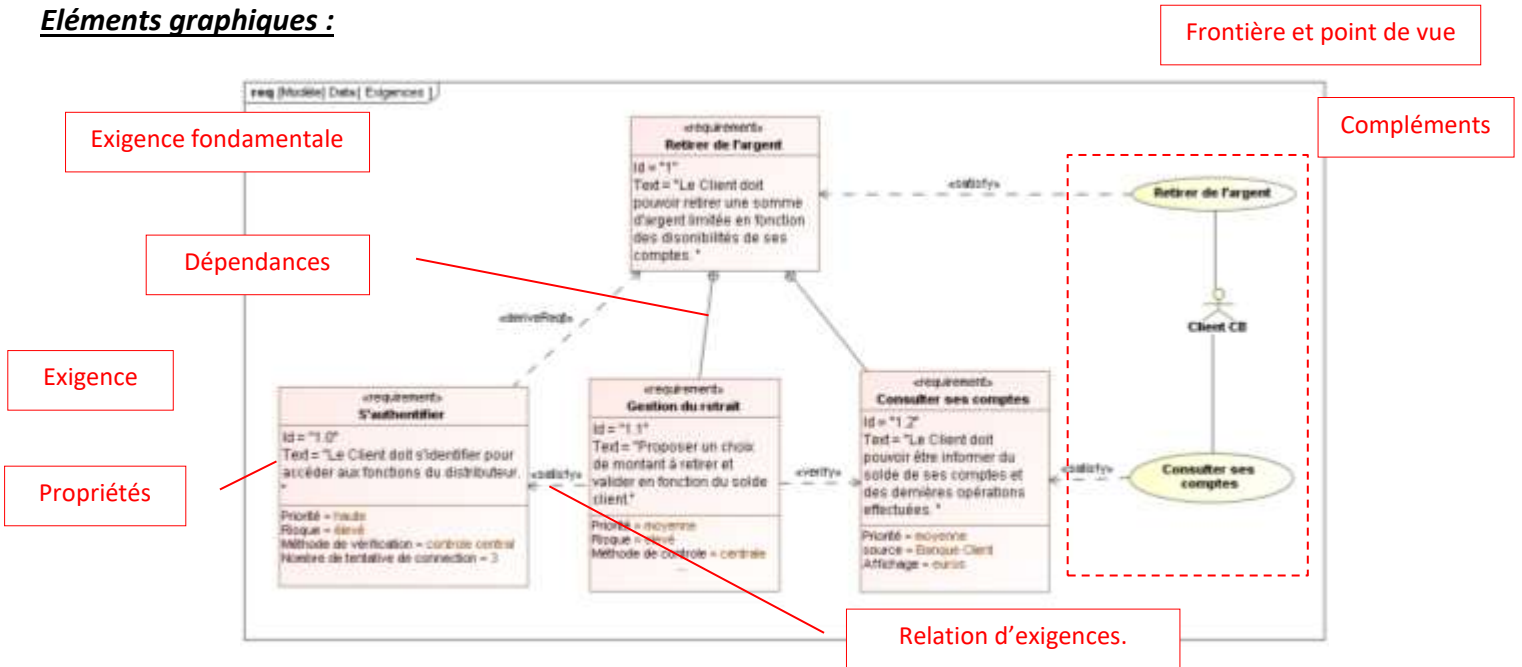
🔗 **Quelle est la différence graphique entre un acteur « humain » et un acteur « système » ?**

Pour différencier les deux, on encadre l'acteur système

3.2 - Diagramme d'exigences.

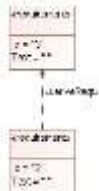
Le diagramme d'exigences permet **de représenter graphiquement les exigences du système en matière de contrainte ou de capacité afin de remplir une fonction et de satisfaire un besoin.**

Eléments graphiques :

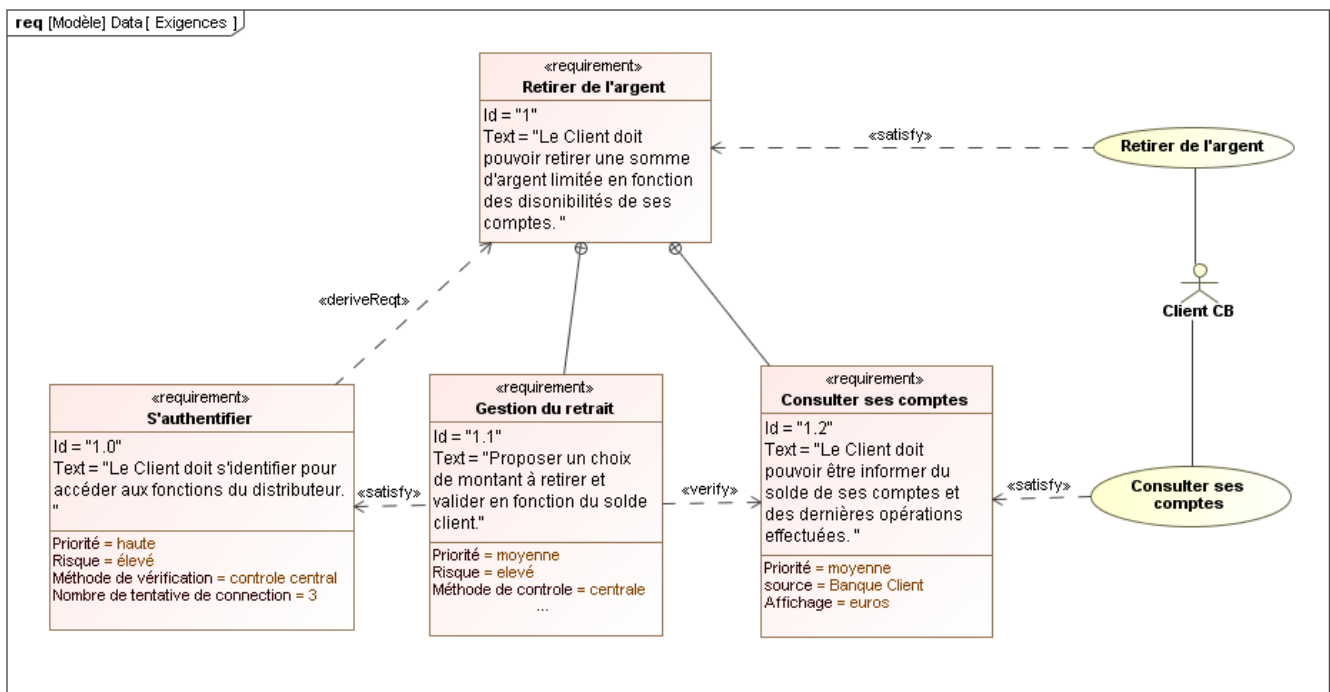


Interprétation des éléments graphiques :

<p>Exigence</p> <pre> «requirements» Consulter ses comptes Id = "3" Text = "Le Client doit pouvoir être informé du solde de ses comptes et des dernières opérations effectuées." Priorité = moyenne Source = Banque Client Affichage = euros </pre>	<p>Une exigence permet de spécifier une capacité ou une contrainte qui doit être satisfaite par un système. Elle peut spécifier une fonction à réaliser ou une condition de performance, de fiabilité, de sécurité...</p> <p>Les exigences représentent le contrat (C.d.C.F) entre le client et les concepteurs du système.</p> <p>Il est courant de définir d'autres propriétés quantifiées ou pas afin d'être précis sur les fonctions à remplir par le système.</p>
<p>Les dépendances</p>	<p>Les exigences peuvent être liées entre elles par plusieurs types de relation, nous en utiliserons trois principalement : relation de contenance, de raffinement ou de dérivation :</p>
	<p>la contenance (ligne terminée par un cercle contenant une croix du côté du conteneur) (« confinement ») : permet de décomposer une exigence composite en plusieurs exigences unitaires, plus faciles ensuite à tracer vis-à-vis de l'architecture ou des tests ;</p>
	<p>le raffinement (« refine ») : consiste en l'ajout de précisions, par exemple de données quantitatives ;</p>

	<p>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.</p>
<p>Compléments (facultatif)</p>	<p>Le diagramme d'exigences permet tout au long d'un projet de relier les exigences entre elles et avec d'autres types d'élément SysML (cas d'utilisation, bloc, état) par plusieurs types de relation.</p> <p>Relations d'exigence :</p> <p>« verify » : consiste à exiger une vérification de test entre deux éléments.</p> <p>« satisfy » : consiste à satisfaire un élément pour obtenir l'élément associé.</p>

Interprétation de l'exemple :



À partir de l'exemple donné, répondre aux questions suivantes :

👉 Pour l'exigence « s'authentifier », quelle est la valeur définissant la propriété « nombre de tentatives de connexions » ?

nombre de tentative de connexions : 3

👉 Quel sont les exigences les plus sensibles au risque de sécurité ?

S'authentifier

Gérer le retrait

👉 Parmi les exigences précédentes, laquelle a la priorité la plus élevée ?

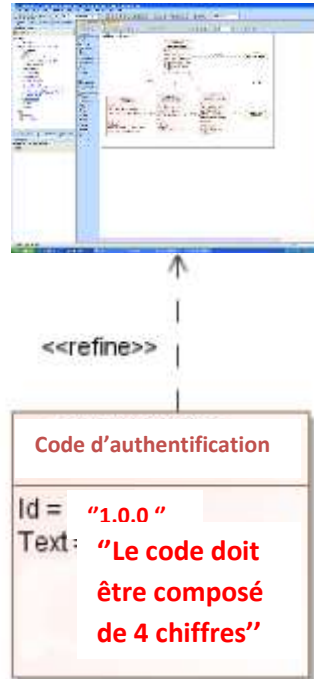
S'authentifier

👉 Pour l'exigence « Gestion du retrait », quelles relations d'exigence avons-nous ?

« verify »

« satisfy »

👉 Ajouter, sur le schéma ci-dessous, une dépendance de type « refine » à l'exigence « S'authentifier » :



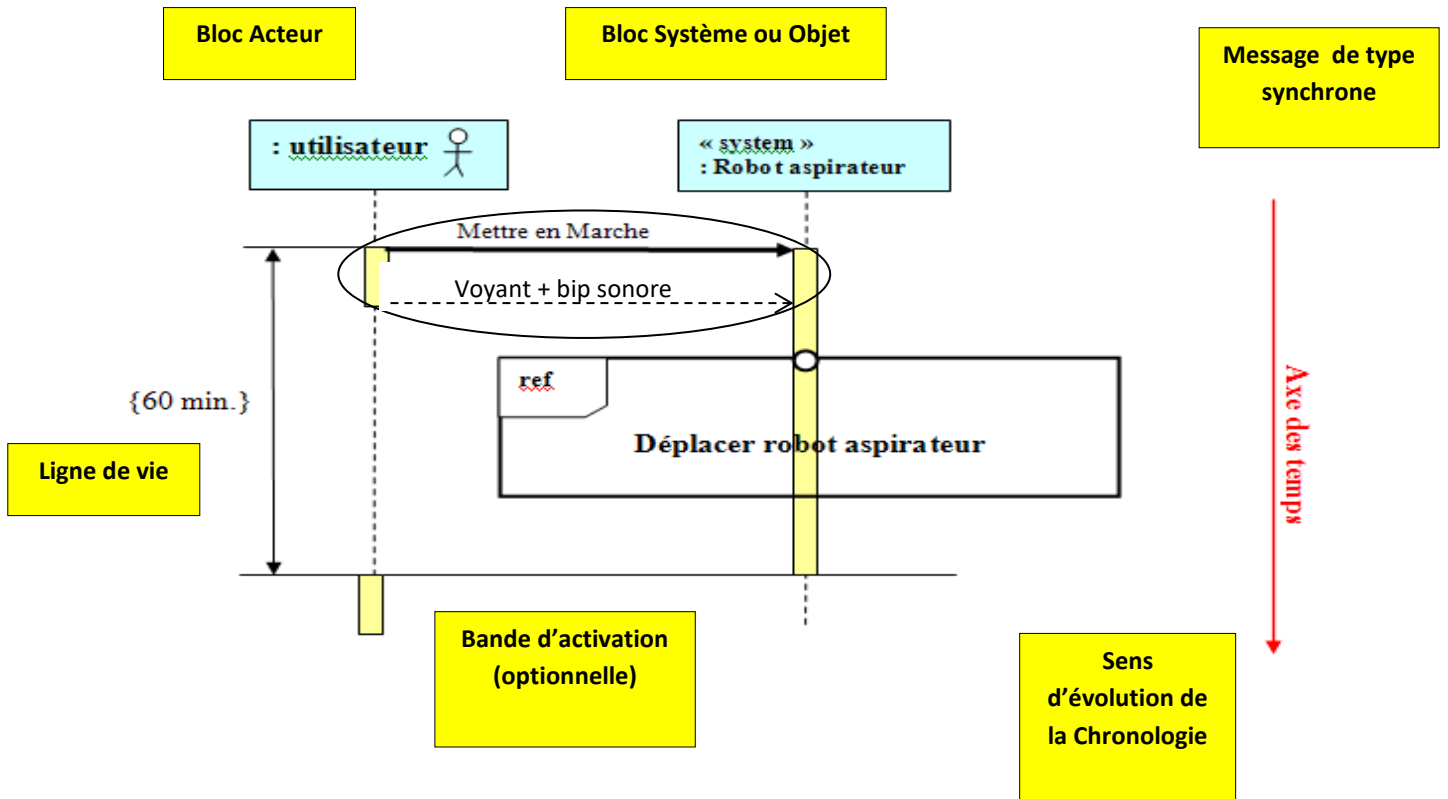
3.3 - Diagramme de séquence

Un cas d'utilisation représente les interactions entre un acteur et le système d'un point de vue « boîte noire », et comprend l'ensemble des scénarios identifiés. **Ces scénarios peuvent être détaillés par un diagramme de séquence.**

Le diagramme de séquence représente **les éléments intervenant dans le scénario ainsi que les messages échangés dans un ordre chronologique.**

Éléments graphiques :

Dans un premier temps, on peut choisir de représenter les interactions entre l'acteur et le système (vue boîte noire).



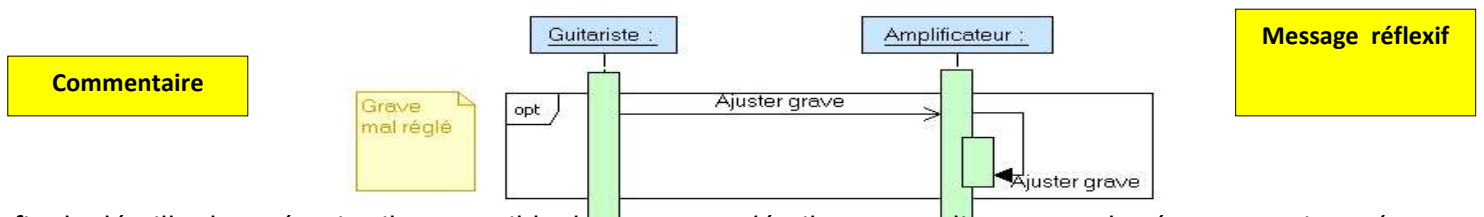
Un objet est représenté par un bloc et une ligne verticale appelée « ligne de vie de l'objet ».

On peut représenter explicitement la période d'activité (ou bande d'activation) d'un objet, période pendant laquelle il exécute une action.

L'axe du temps est vertical, le flot de contrôles est horizontal.

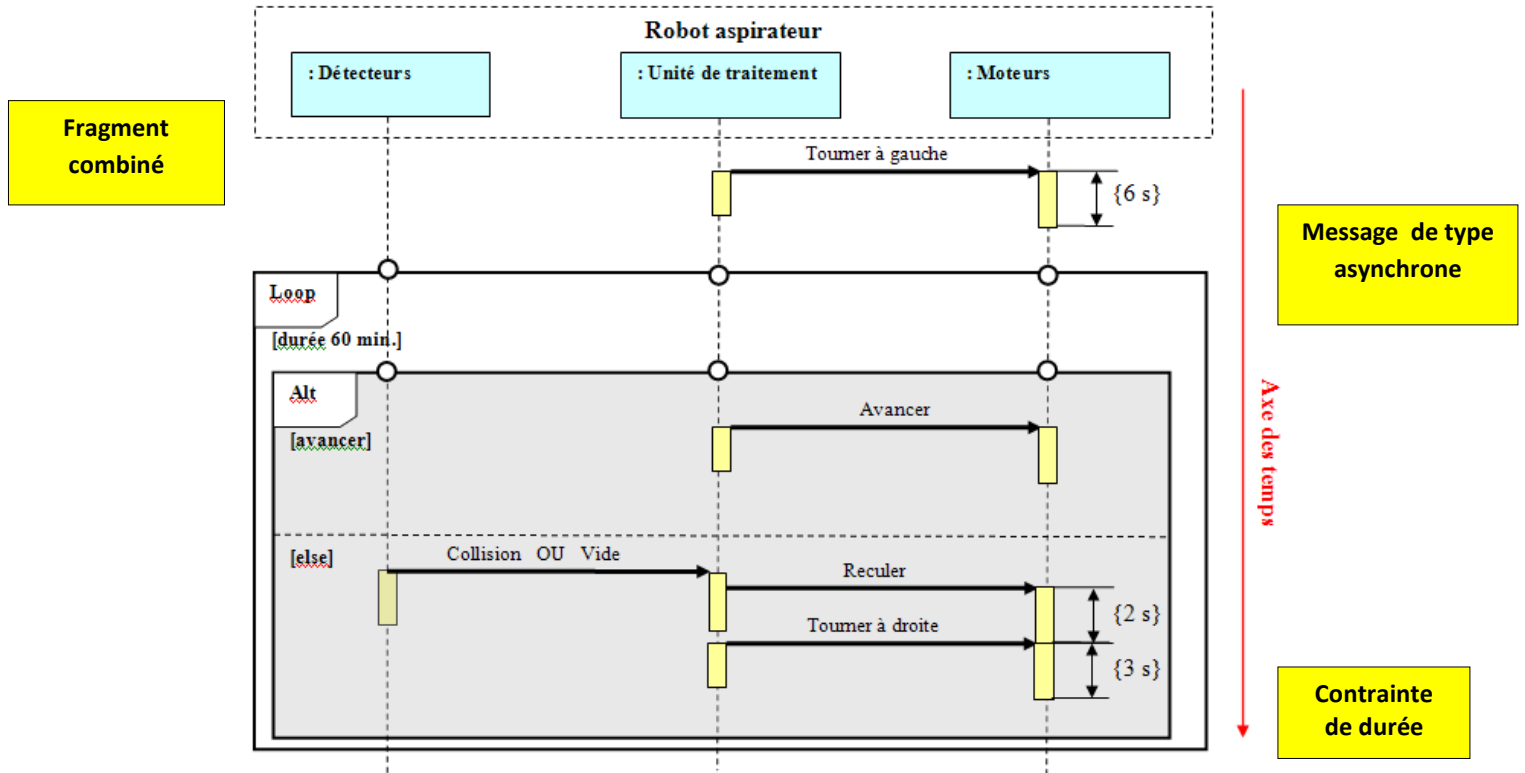
Un message synchrone (émetteur bloqué en attente de réponse) est représenté par une flèche pleine, alors qu'un message asynchrone est représenté par une flèche évidée.

La flèche qui boucle (message réflexif) permet de représenter un comportement interne.



Afin de détailler les scénarios il est possible de rentrer en détails avec un diagramme de séquence qui représente les blocs internes du système intervenant (pour un message émis par l'acteur, le diagramme décrit l'enchaînement des messages échangés entre les blocs internes du système).

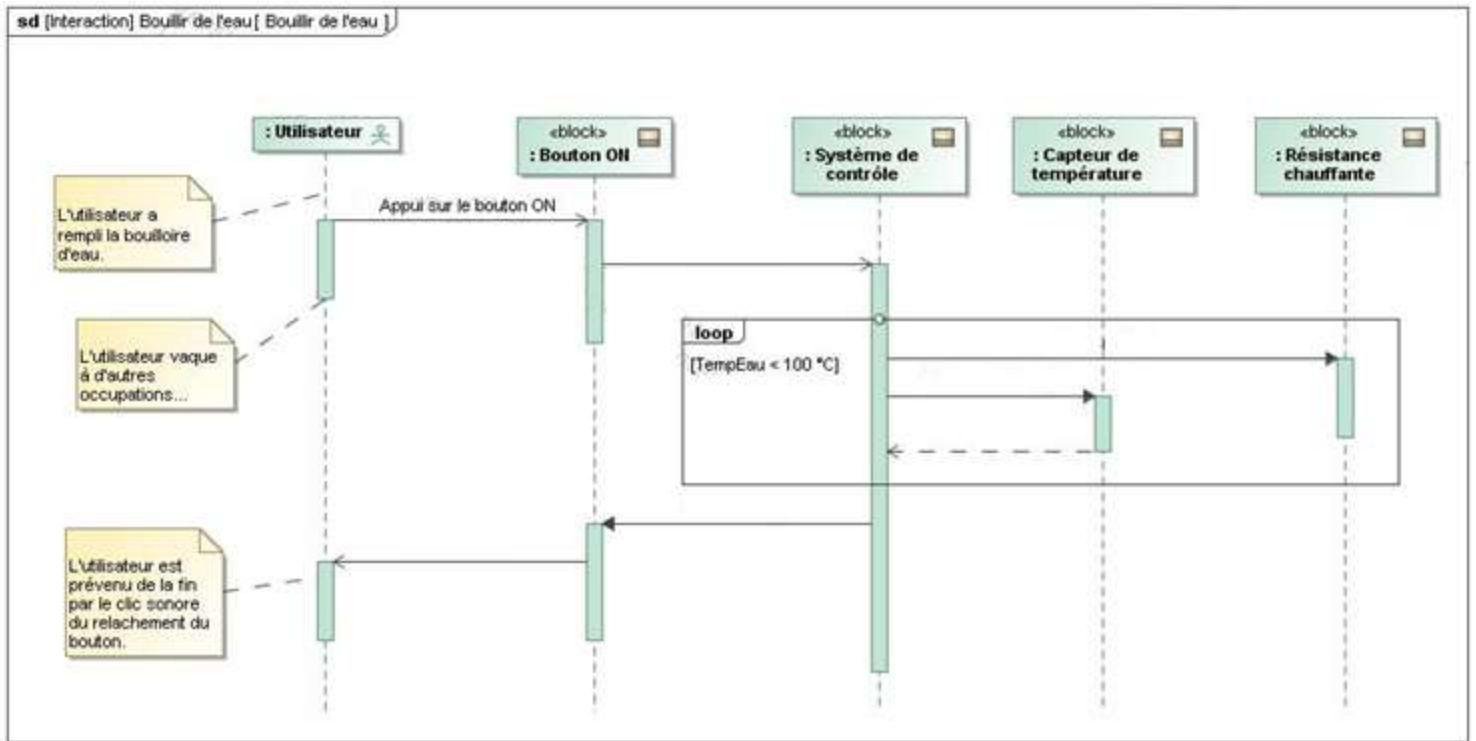
On parle ainsi de la vue boîte blanche (comportement du système).



Les principaux fragments combinés :

- LOOP** : boucle. Le fragment peut s'exécuter plusieurs fois et la condition explicite l'itération.
- ALT** : alternatif. Seul le fragment possédant la condition vraie s'exécute.
- OPT** : optionnel. Le fragment ne s'exécute que si la condition est vraie.
- REF** : référence. Un diagramme de séquence peut faire référence à un autre diagramme de séquence.
- PAR** : parallèle. Plusieurs opérations peuvent être exécutées en parallèle.

Interprétation d'un exemple :



De quel système le diagramme de séquence ci-dessus décrit-il le scénario ?

Une bouilloire d'eau électrique

Qui est l'acteur ?

L'utilisateur

Quelles sont les blocs internes du système intervenant dans ce diagramme de séquence ?

Bouton ON/Système de contrôle/Capteur de température/Résistance chauffante

Placer les séquences ci-dessous dans l'ordre chronologique.

(Chauffer/Relâcher/Température d'eau atteinte/Clic sonore/Mesurer la température/Démarrer)

1- **Appui sur le bouton ON**

2- **Démarrer**

3- **Chauffer**

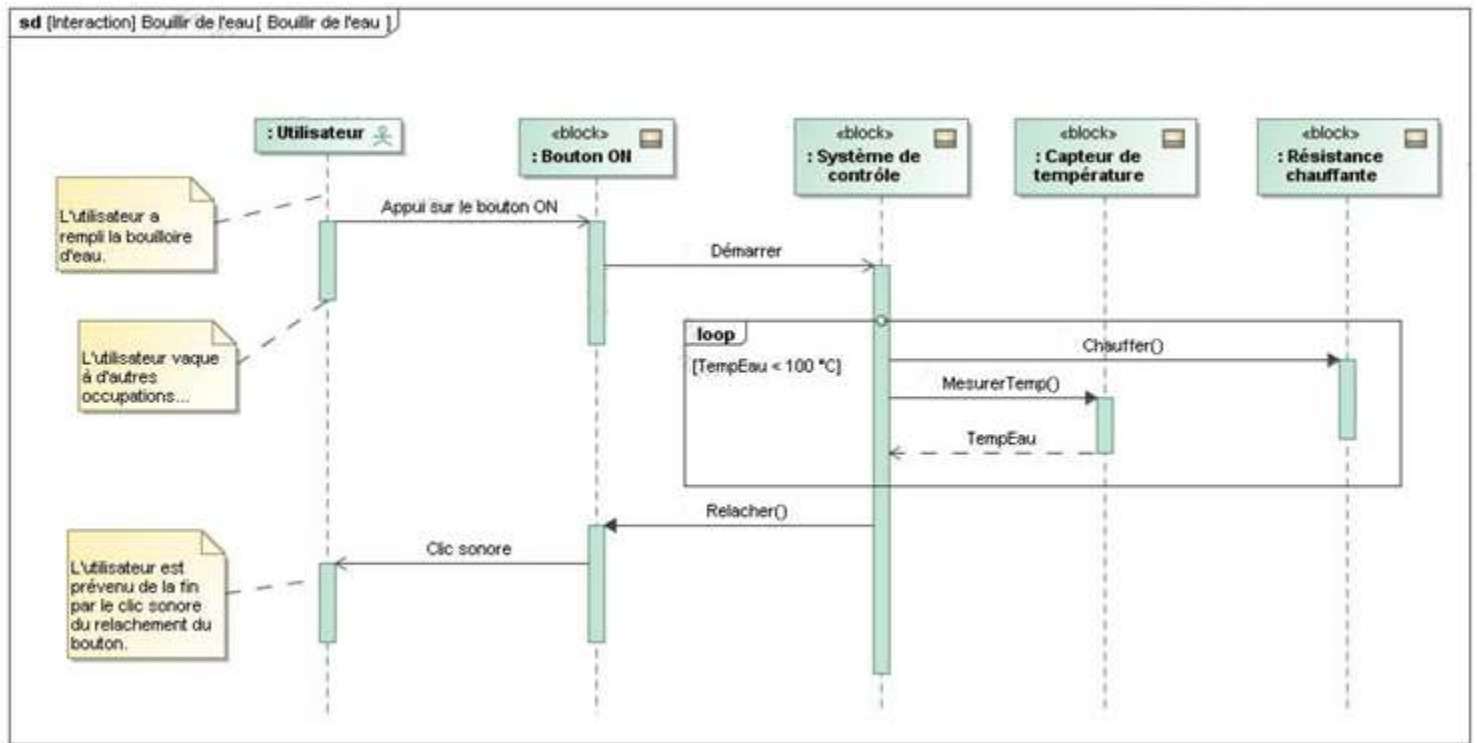
4- **Mesurer la température**

5- **Température d'eau atteinte**

6- **Relâcher**

7- **Clic sonore**

👉 A partir de la chronologie définie précédemment, compléter le diagramme de séquence ci-dessus.



👉 Décrire le fragment combiné : « loop »

Tant que la température est inférieure à 100°C, le bloc « Système de contrôle » envoie l'ordre de chauffer au bloc « résistance ».

3.4. Diagramme de définition de blocs

Le diagramme de définition de bloc (BDD, ou Block Definition Diagram en anglais) permet de représenter une vue statique du système, c'est-à-dire sa structure.

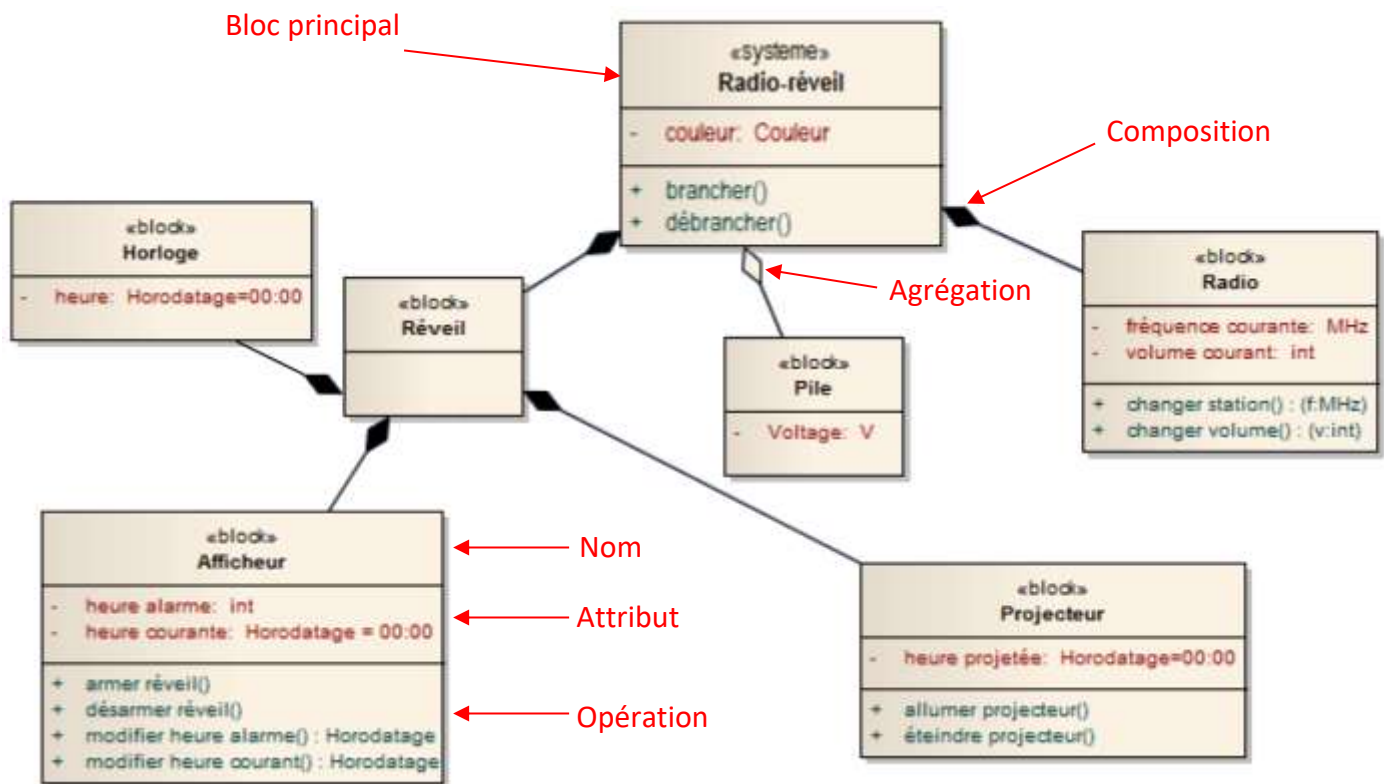
Il s'agit d'une vision « boîte noire » de ce que l'on modélise.

Le BDD est similaire à la première page d'une notice d'un système, indiquant la liste des éléments et des pièces assemblés.



Éléments graphiques : Radio réveil projecteur

Le diagramme de définition de blocs ci-dessous, montre les différents éléments (blocs) d'un radio réveil projecteur.



Interprétation des éléments graphiques :

Bloc	
<pre> «block» Radio - fréquence courante: MHz - volume courant: int + changer station(): (f:MHz) + changer volume(): (v:int) </pre>	<p>Bloc : Un système se décompose en plusieurs parties ou composantes. Chacune d'entre elles est modélisée par un bloc. Il est nécessaire d'avoir un bloc qui représente l'ensemble du système. Il est constitué de 3 parties :</p> <ul style="list-style-type: none"> - Nom : nom de l'élément du système, - Attribut : représente les propriétés qui caractérisent ce bloc, - Opération : représente ce que l'on peut demander au bloc.

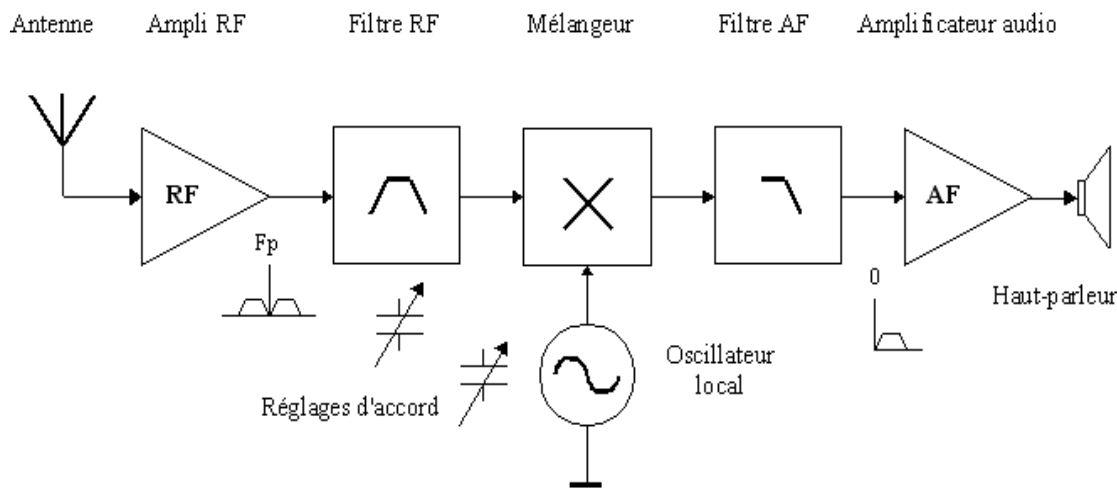
<p>Relations</p>	<p><u>Composition</u> : représente une imbrication, le fait qu'un élément se décompose en plusieurs composants.</p> <p><u>Agrégation</u> : représente un lien quelconque, un rapport entre un élément du système et son environnement.</p>
-------------------------	--

À partir de l'exemple donné, répondre aux questions suivantes :

☞ *Citer les différents composants du radio-réveil :*

réveil radio pile

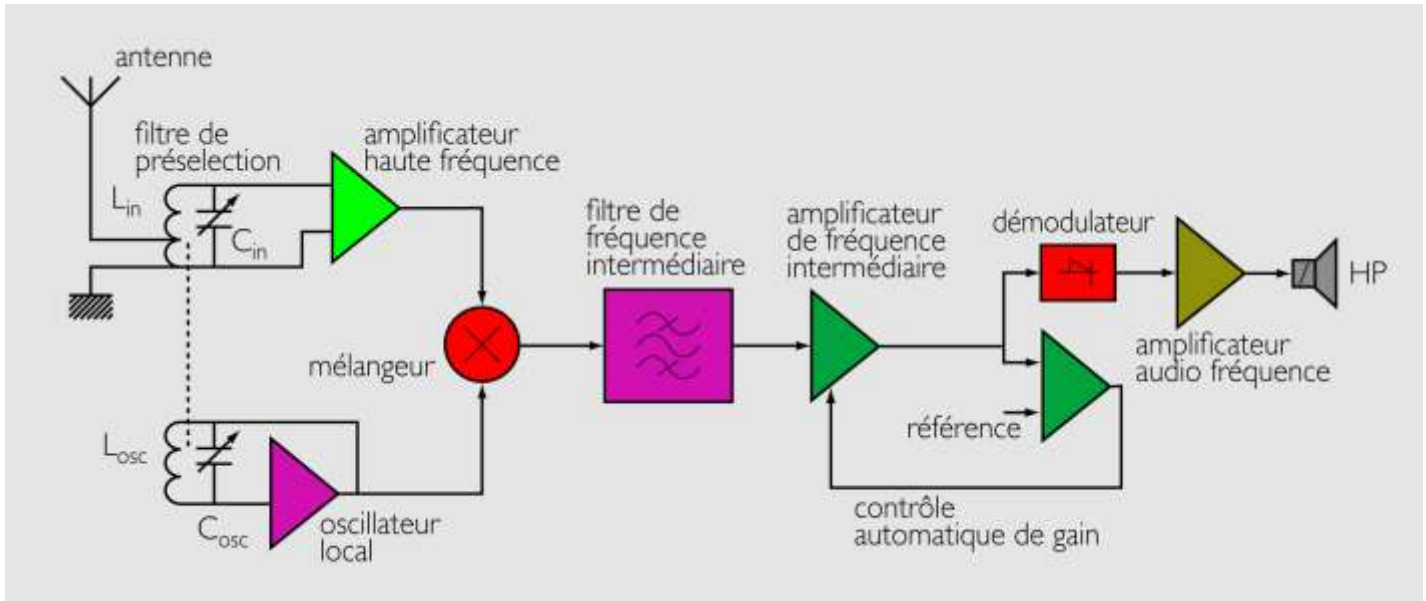
Voici la structure simplifiée du bloc radio :



- Amplificateur RF (radiofréquence) : il amplifie le signal d'antenne,
- Amplificateur audio : il amplifie le signal audio qui provient de la démodulation du signal capté par l'antenne,
- Un filtre RF élimine les signaux indésirables,
- La sélectivité est déterminée par le filtre AF (audiofréquence),
- Le tuner comprend l'[oscillateur](#) et le [mélangeur](#).

☞ *Compléter le bloc radio ci-dessous :*



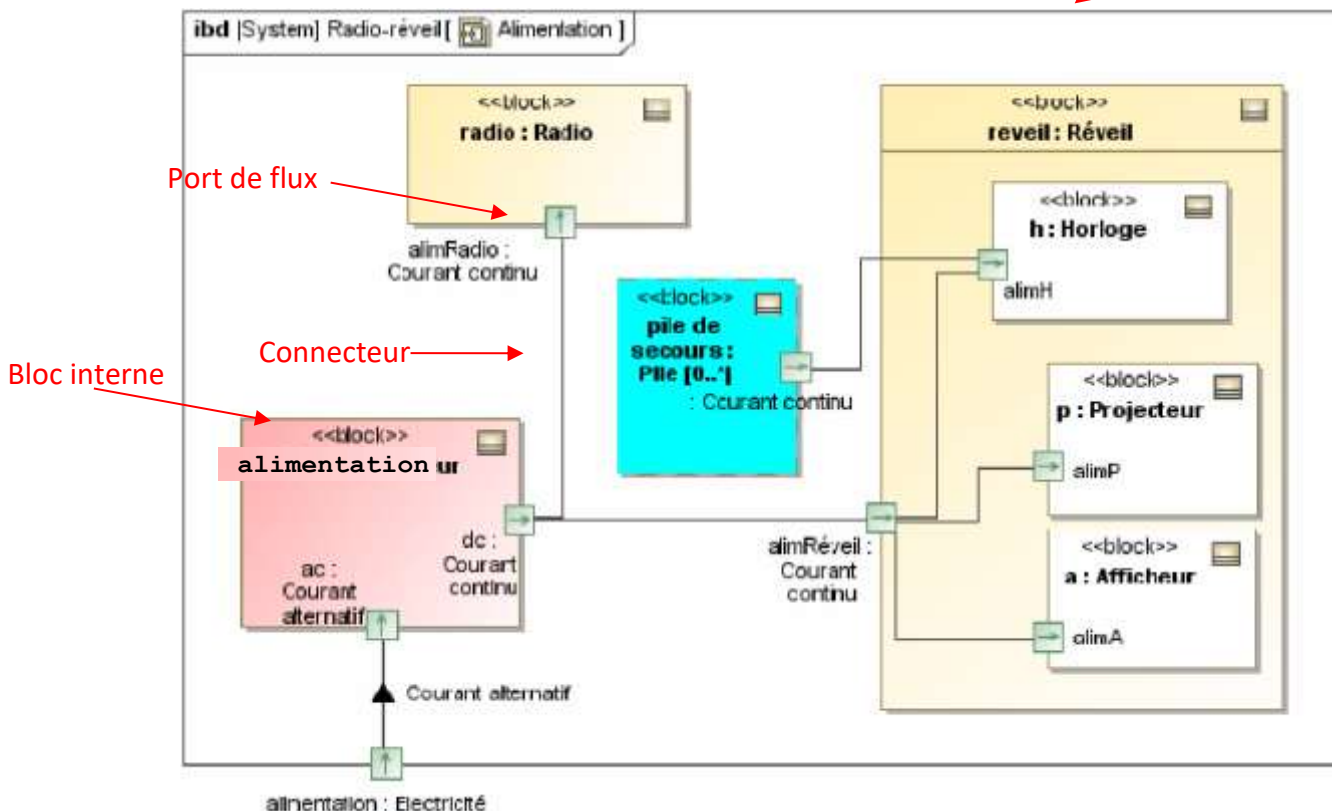


3.5 Diagramme de bloc interne

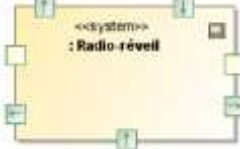
Le diagramme de bloc interne (IBD, ou Internal Block Diagram) décrit la vue interne d'un bloc et se base sur le diagramme de définition de bloc pour assembler les blocs qui composent le bloc principal. Ces parties sont assemblées par des **connecteurs** qui relient leurs **ports** (ports standards avec interfaces exposées et/ou ports de flux).

Éléments graphiques : *Radio réveil projecteur (point de vue flux d'énergie)*

Bloc principal

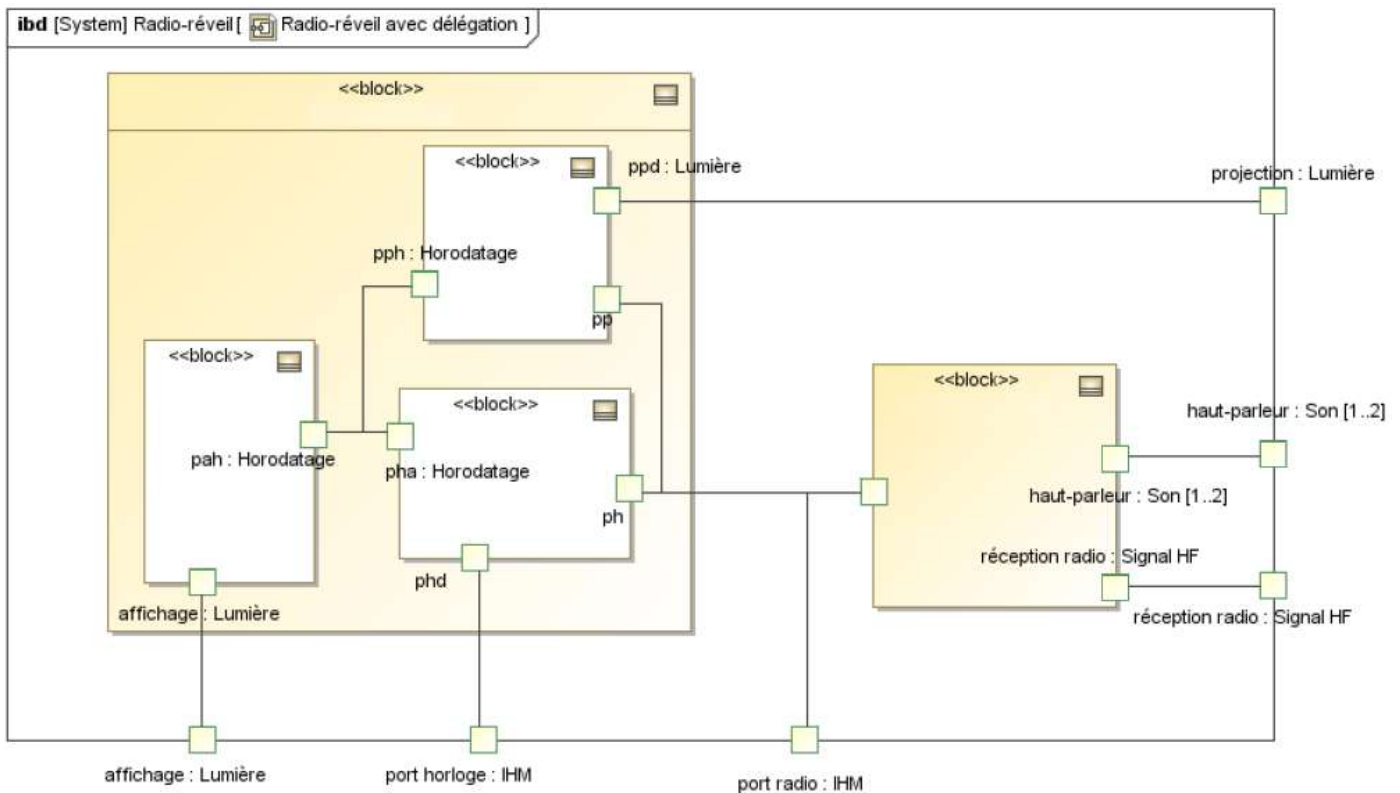


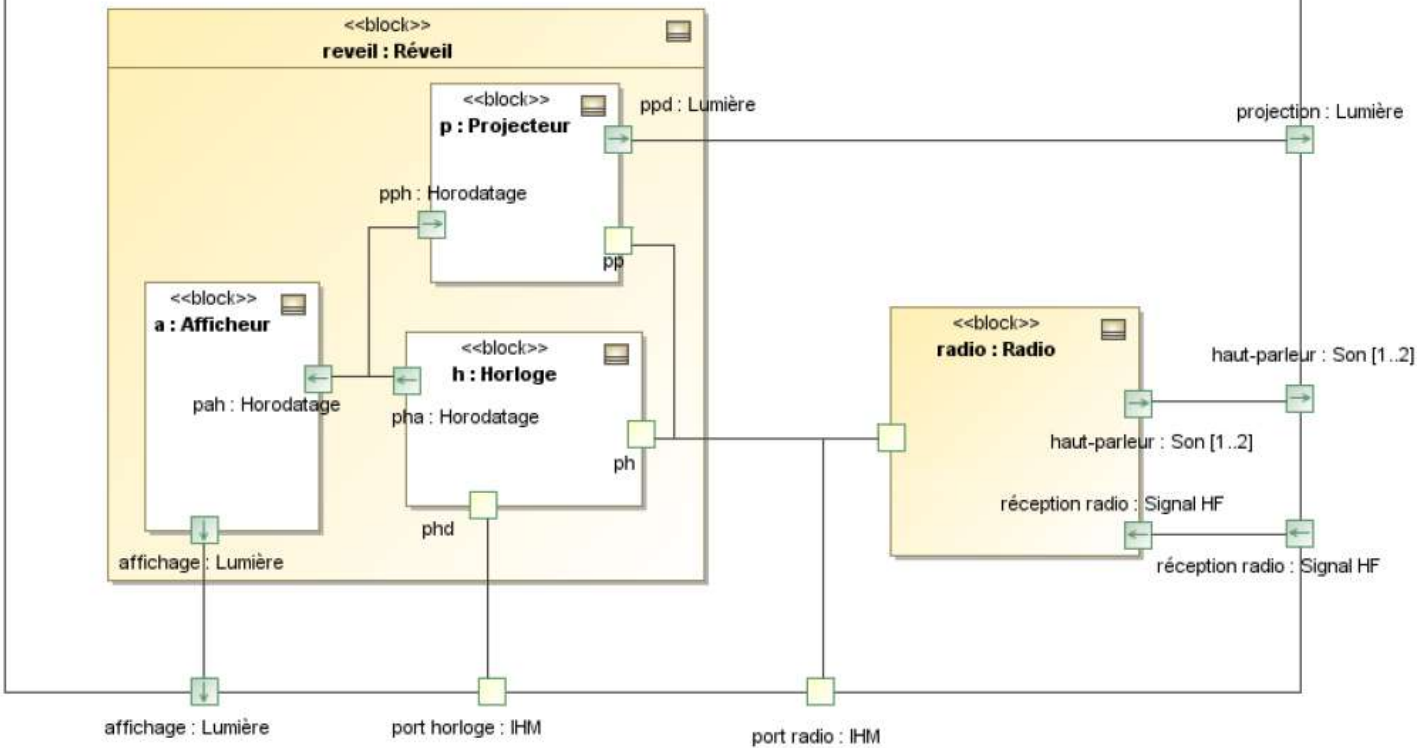
Interprétation des éléments graphiques :

<p>Bloc interne</p> 	<p>Un <u>bloc interne</u> peut avoir plusieurs ports qui spécifient des points d'interaction différents.</p>
<p>Connecteurs</p> <hr/>	<p>Les <u>connecteurs</u> représentent les chemins de communication entre les éléments. Lorsqu'un connecteur relie deux ports de flux, il représente un échange de matière, d'énergie ou d'information (MEI) et quand il relie deux ports standards, il représente l'appel à un service proposé.</p>
<p>Ports</p> <p>□</p> <p>⇒ } ⇐ }</p>	<p>2 types de ports existent :</p> <ul style="list-style-type: none"> - <u>Port standard</u> : ce type de port autorise la description de services logiques entre les blocs, au moyen d'interfaces regroupant des opérations. - <u>Port de flux (flow port)</u> : ce type de port autorise la circulation de flux physiques entre les blocs. La nature de ce qui peut circuler est de type MEI (). Exemple : fluides, énergie électrique, données ... <p>Quand deux blocs interagissent (échange de flux dans les 2 sens), les ports sont dits "non atomiques". Représentés par deux crochets se faisant face (<>).</p> <p>Quand le flux est monodirectionnel les ports sont dits "atomiques" et représentés par un crochet (< ou >) indiquant le sens du flux.</p>

À partir de l'exemple donné :

- ☞ **Compléter, sur le diagramme ci-dessous, chaque bloc en précisant son nom (composants du système).**
- ☞ **Compléter chaque port en lui attribuant (si nécessaire) un ou deux crochets afin de définir le sens du flux.**



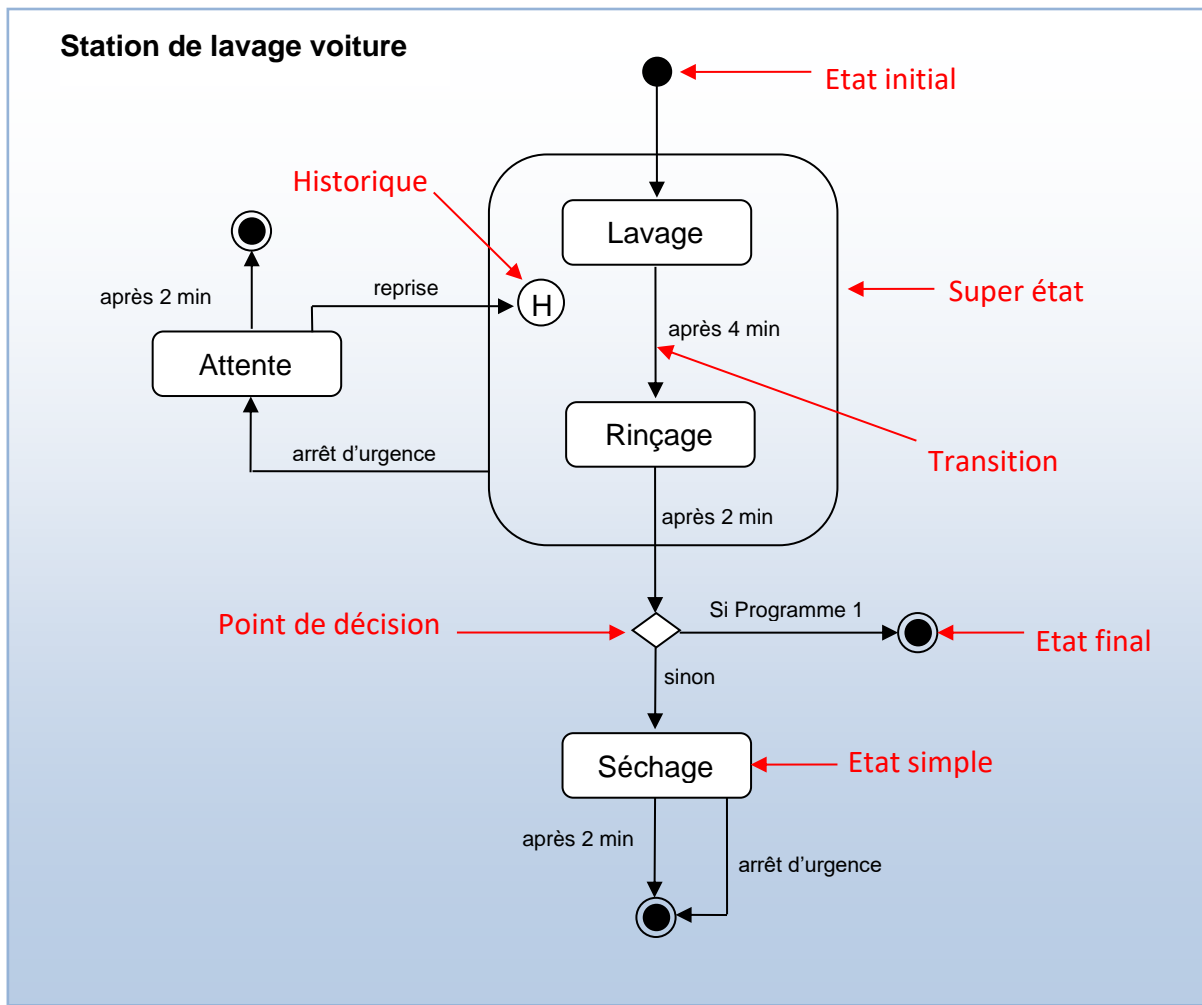


3.6 Diagramme d'états transitions



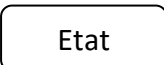
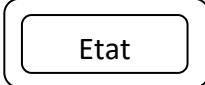

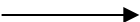
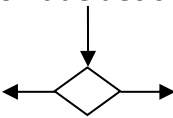
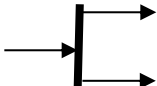
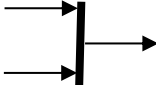
Le diagramme d'états transitions représente la vue dynamique du système. C'est un graphe permettant de décrire les changements d'états d'un système ou d'un composant, en réponse aux interactions avec d'autres systèmes/composants ou avec des acteurs.

Eléments graphiques : Station de lavage voiture

Le diagramme d'états-transitions ci-dessous, montre les différents états par lesquels passe une station de lavage de voitures.



Interprétation des éléments graphiques :

<p>Etat</p>     	<p><u>Etat initial</u> : cet élément marque le début du programme. Il y a au moins un élément « état initial » dans un programme. Cet élément est actif au démarrage du programme.</p> <p><u>Etat final</u> : cet élément marque la fin du programme. Il peut ne pas être utilisé si le programme ne doit jamais se terminer.</p> <p><u>Etat simple</u> : représente un état du système, une configuration des variables du système ou encore le statut du système.</p> <p><u>Super état (ou état composite)</u>: il s'agit de regroupements d'états.</p> <p><u>L'état historique</u> est un pseudo-état qui se place au sein d'un super état. Cet état va permettre de retourner immédiatement à l'endroit de l'état composite que l'on a quitté.</p>
<p>Transition</p> 	<p><u>Une transition</u> représente le passage instantané d'un état vers un autre. La transition est déclenchée par un événement. Elle relie les états entre eux. Elle peut :</p> <ul style="list-style-type: none"> - relier simplement deux états, - être réflexive (l'état de départ est le même que celui d'arrivée), - arriver sur un super-état (dans ce cas, elle arrive à l'état initial du super-état), - être conditionnée : exemple la durée est indiquée.
<p>Point de décision</p>  <p>Barre de synchronisation</p>  	<p><u>Point de décision</u> : une transition arrive sur un point de décision (ou point de choix) et plusieurs transitions en repartent. Chacune doit porter une condition (garde).</p> <p><u>Barre de fraction</u> : une transition arrive sur une barre de fraction et plusieurs transitions en partent. Les transitions qui partent de cette barre sont des chemins d'exécution qui se réalisent en parallèle et indépendamment.</p> <p><u>Barre de jonction</u> : tous les chemins d'exécution arrivent sur cette barre, chacun à l'aide d'une transition. Une seule transition repart de cette barre. Pour que cette transition puisse s'exécuter, il faut que toutes les transitions qui arrivent sur la barre aient été franchies.</p>

À partir de l'exemple donné, répondre aux questions suivantes :

👉 **Quelle est la durée du lavage ? 4 min**

👉 **En phase de lavage ou de rinçage, si le client appui sur le bouton d'arrêt d'urgence, que se passe-t-il ?**
S'il appuie sur ce bouton, la station se met en attente. Il a alors deux minutes pour reprendre le cycle (la station continue en phase de lavage ou de séchage, suivant l'état dans lequel elle a été interrompue), sans quoi la station s'arrête.

↳ En phase de séchage, si le client appui sur le bouton d'arrêt d'urgence, que ce passe-t-il ?

En phase de séchage le client peut aussi interrompre la station. Mais dans ce cas, la station s'arrêtera définitivement.

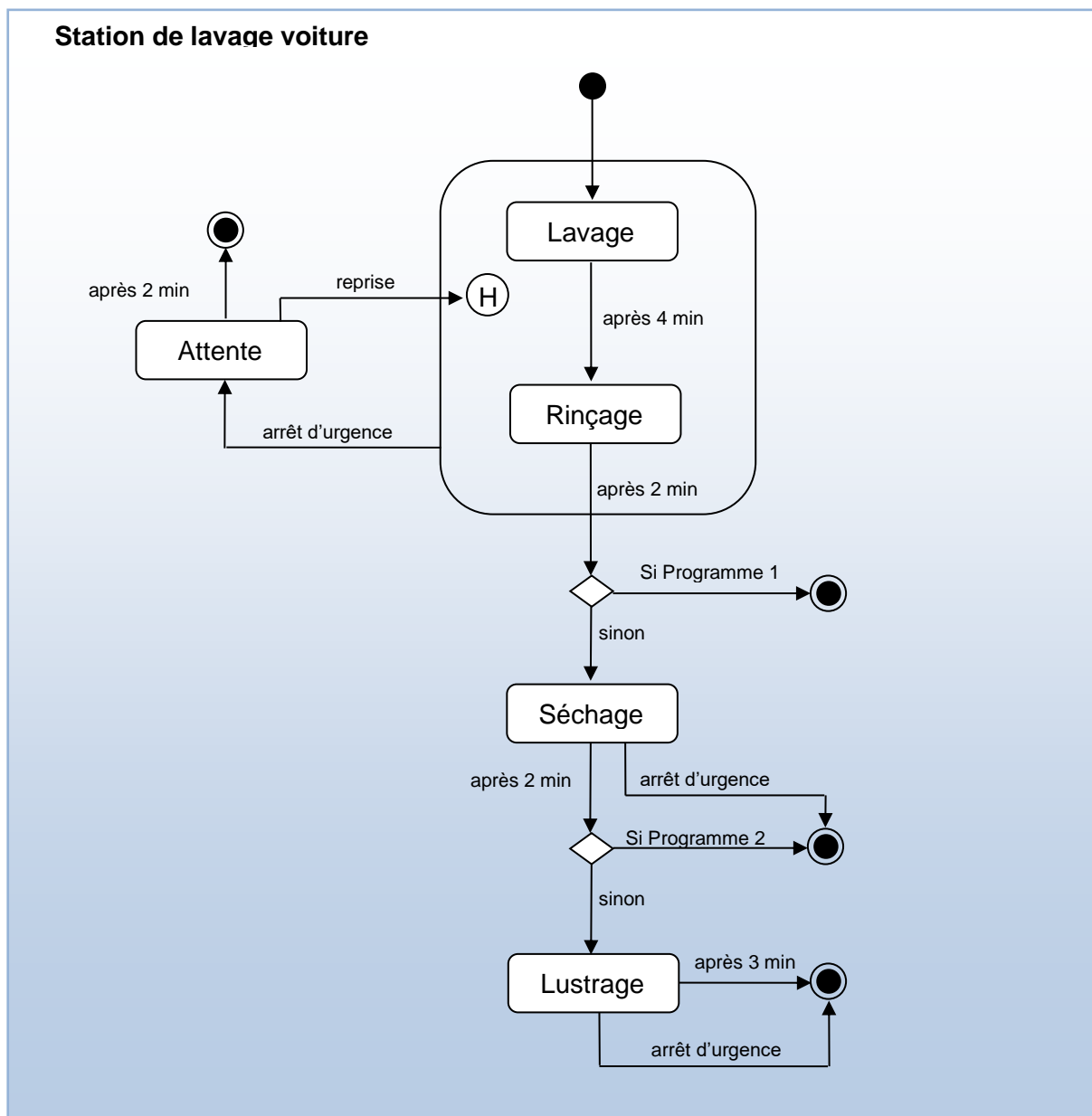
↳ Donner les étapes du Programme 1 :

Lavage + rinçage

↳ Donner les étapes de du Programme 2 :

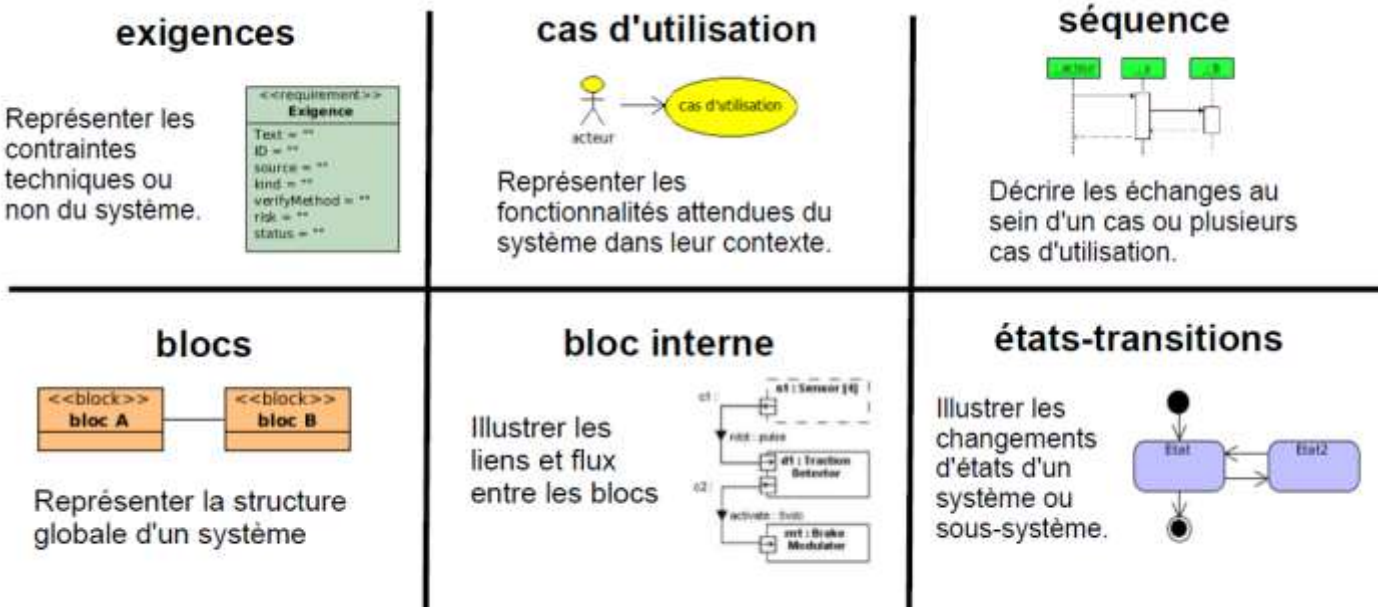
Lavage + rinçage + séchage

↳ Le Programme 3 qui est le « Programme 2 + un lustrage de 3 min » n'a pas été représenté. Compléter le diagramme d'état ci-dessous pour faire apparaître ce dernier programme (si le client appui sur l'arrêt d'urgence, le programme s'arrête définitivement) :



4. Synthèse

4.1 Les 6 diagrammes les plus utilisés en SYSML

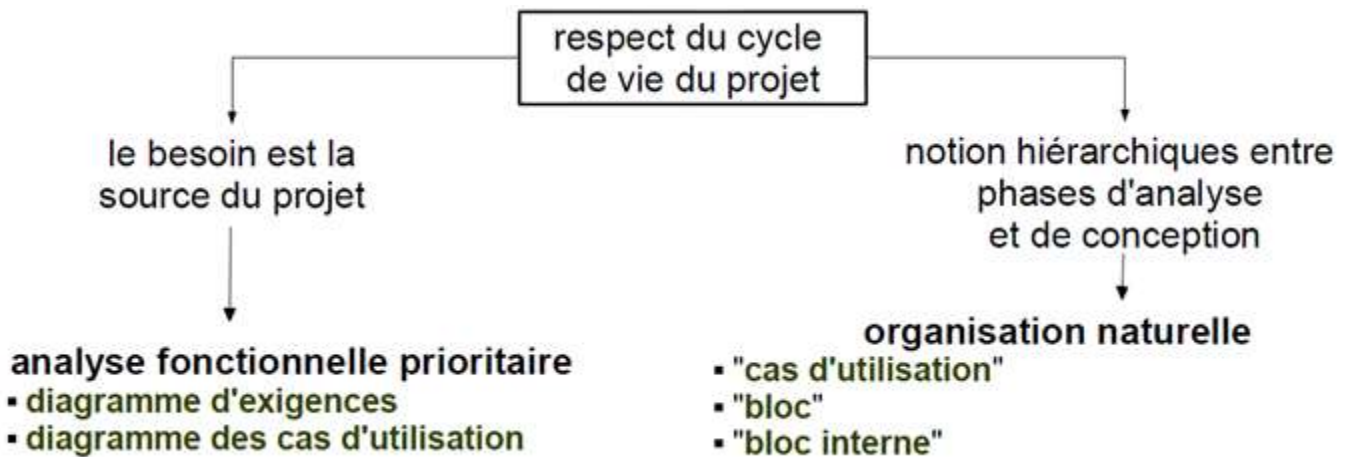


4.2 Les principales relations

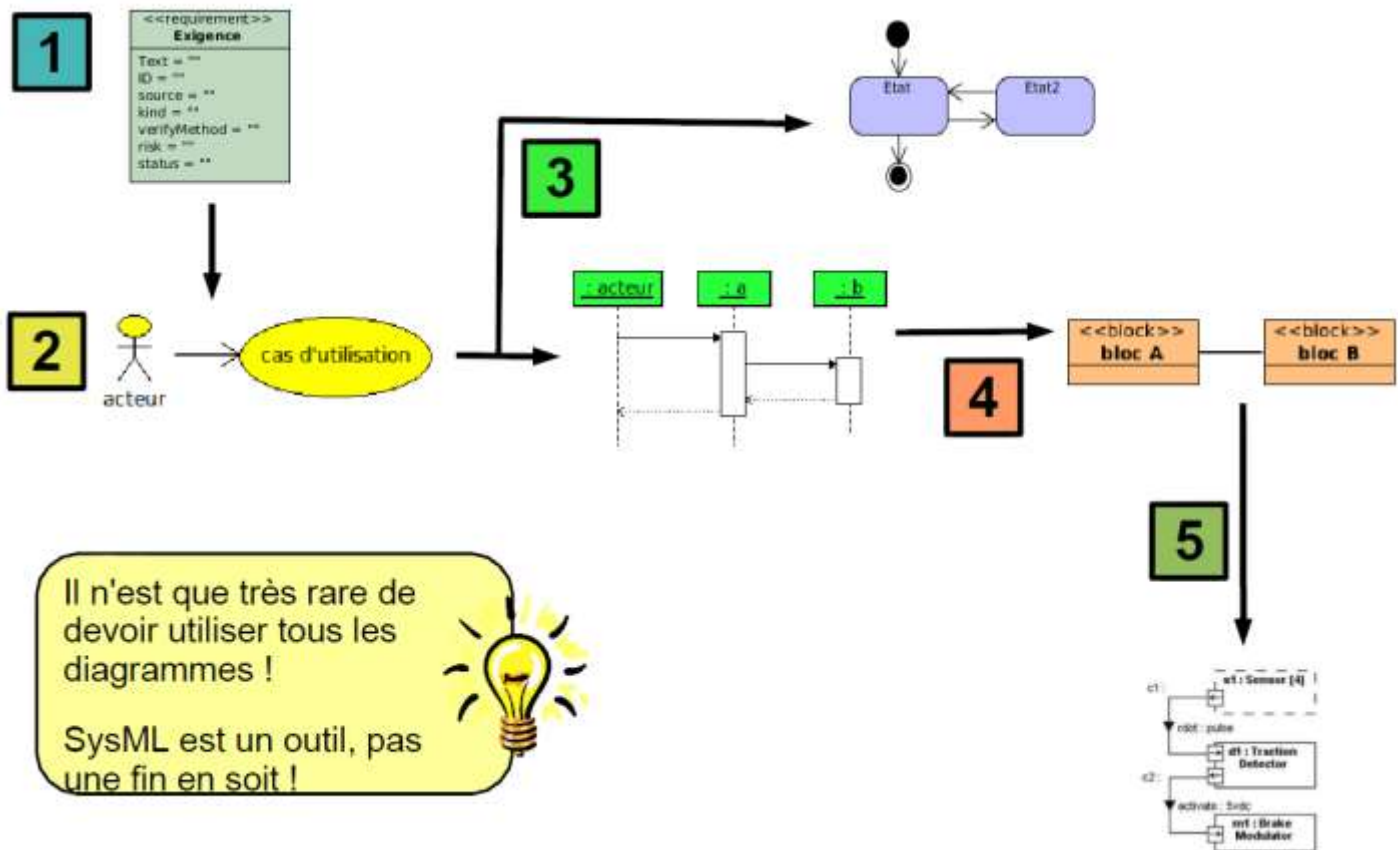
- A —> B **association** : relation d'égal à égal entre deux éléments.
 → **A utilise B**
 • 2 diagrammes : cas d'utilisation, blocs
- A -.-> B **dépendance**: deux items indépendants mais dont l'un dépend de l'autre.
 -.-> **A dépend de B**
 • 4 diagrammes : exigences, cas d'utilisation, blocs, paquetages
- A —◇ B **agrégation**: un élément est une composante facultative d'un autre.
 —◇ **A entre dans la composition de B, sans être indispensable à son fonctionnement**
 • 2 diagrammes : exigences, blocs
- A —◆ B **composition**: un élément est une composante obligatoire d'un autre.
 —◆ **A entre dans la composition de B et est lui est indispensable**
 • 2 diagrammes : exigences, blocs
- A —▷ B **généralisation**: dépendance de type 'filiation' entre 2 items
 —▷ **A est une sorte de B**
 • 3 diagrammes : cas d'utilisation, blocs, paquetages
- A —⊕ B **conteneur**: relation d'inclusion entre deux items
 —⊕ **B contient A**
 • 4 diagrammes : exigences, cas d'utilisation, blocs, paquetages

4.3 Le processus de modélisation

SysML n'est pas une méthode mais un outil de représentation donc il n'y a pas de méthode. Cependant la pratique habituelle peut se résumer ainsi :



4.4 Exemple de démarche possible



Il n'est que très rare de devoir utiliser tous les diagrammes !
 SysML est un outil, pas une fin en soit !

