



C1 : MODÉLISATION DES SYSTÈMES PLURITECHNIQUES

C1-2 - Outils de l'analyse système : utilisation du langage SysML

5 septembre 2017

Table des matières

I Présentation du langage SysML	1
1 Intérêts et objectifs	1
2 Architecture du langage	1
3 Éléments de syntaxe	2
4 Présentation du support du cours	2
II Analyse du contexte et des exigences du système	4
1 Diagramme de contexte	4
2 Diagramme des exigences	5
3 Diagramme des cas d'utilisation	7
III Analyse structurelle du système	8
1 Diagramme de définition des blocs	8
2 Diagramme de blocs internes	10
3 Diagramme paramétrique	12
IV Analyse comportementale du système	12
1 Diagramme de séquence	13
2 Diagramme d'états	16
V Interaction entre les différents diagrammes	17
VI Modélisation et analyse structurelle : chaîne d'information et d'énergie	17
1 La chaîne d'information	17
2 La chaîne d'énergie	18
3 Les interfaces	18

Compétences

- **Analyser :**
 - Identifier le besoin et les exigences.
 - Appréhender les analyses fonctionnelles et structurelles.
- **Communiquer :** Rechercher et traiter les informations.

I. Présentation du langage SysML

1 Intérêts et objectifs

Le chapitre d'introduction précédent a mis en évidence l'intérêt de développer un langage commun de modélisation tel que le **SysML** (Systems Modeling Language). Cette modélisation a pour objectif de formaliser et d'appréhender la conception d'un système. Elle permet ainsi d'augmenter les capacités de conception des ingénieurs en minimisant les risques de retard ou de problèmes liés à l'élaboration du produit. Cette méthode permet de mettre en commun les spécifications, contraintes et paramètres de l'ensemble du système pour aboutir directement à la simulation d'un système complexe et ainsi autoriser la prévision de ses performances.

2 Architecture du langage

Le langage SysML s'articule autour de 9 diagrammes regroupés en 2 catégories principales (figure 1) :

- diagrammes comportementaux : cas d'utilisation, séquences, états, activité (pas explicitement au programme) ;
- diagrammes structurels : définition des blocs, blocs internes, paramétrique, paquetages (pas au programme de CPGE).

A ces diagrammes s'ajoute le diagramme transversal des exigences. Chacun de ces diagrammes comporte une abréviation qui provient de leur dénomination anglo-saxonne.

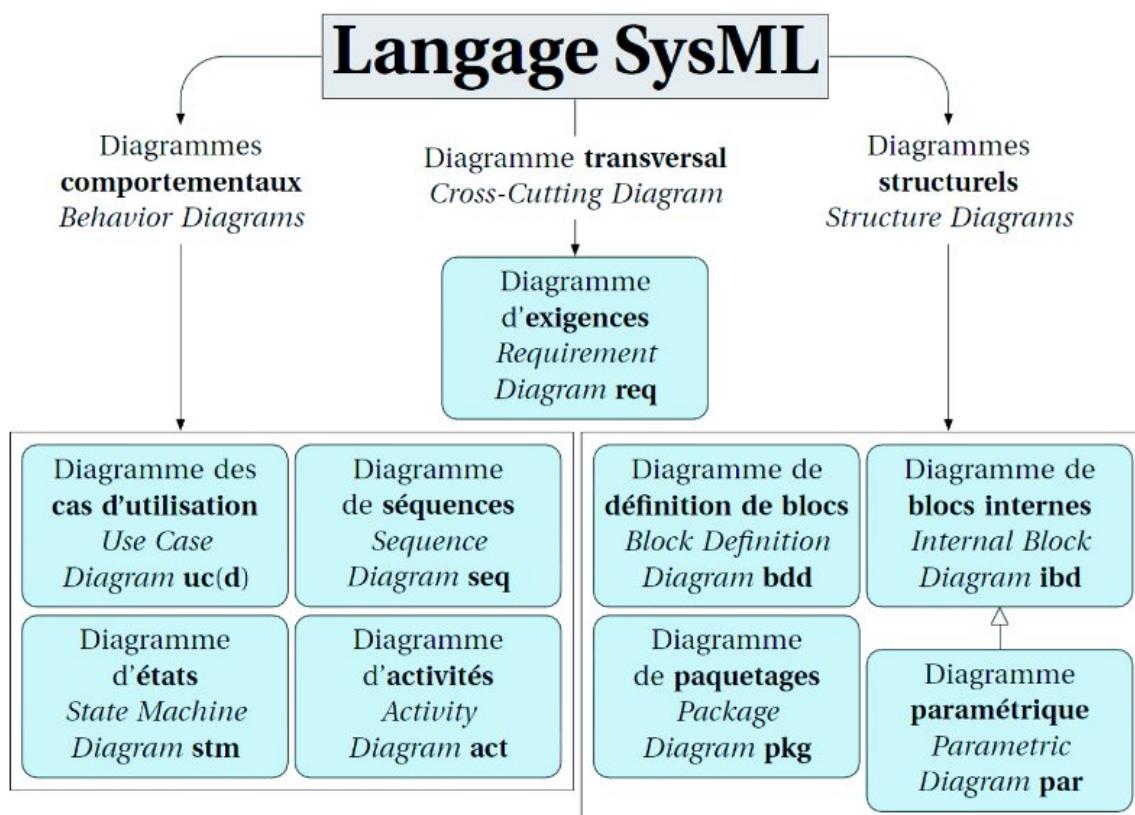


FIGURE 1 – Présentation de l'architecture du langage SysML

**Remarque 1 :**

A ces neuf diagrammes peut s'ajouter le **diagramme du contexte** qui est un cas particulier du diagramme de définition des blocs et qui permet de situer le contexte du système étudié.

3 Elements de syntaxe

Un diagramme SysML est implanté dans un “cartouche” précisant sa fonction et sa dénomination (figure 2).

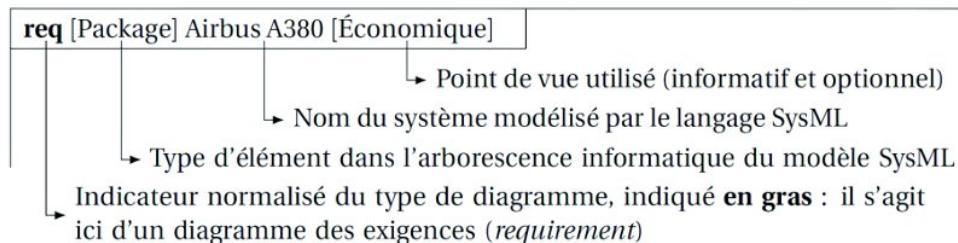


FIGURE 2 – Composition du cadre de description des diagrammes **SysML**

Les neuf diagrammes du langage **SysML** sont composés des mêmes types de formes géométriques : des rectangles, des ellipses et des lignes. Selon les diagrammes, tout ou partie de ces formes géométriques seront utilisées. Plusieurs types de relations peuvent être rencontrées entre les formes géométriques dans les diagrammes **SysML** : le tableau 1 regroupe les liens les plus classiques.

Pour chacun des liens, nous pourrons avoir différents types d'associations :

- **Extend** : le cas d'utilisation source est une extension possible du cas d'utilisation destination.
- **Include** : le cas d'utilisation source comprend obligatoirement le cas inclus.
- **Derive** : une ou plusieurs exigences sont dérivées d'une exigence.
- **DeriveReqt** : permet de relier une exigence d'un niveau général à une exigence d'un niveau plus spécialisé mais exprimant la même contrainte.
- **Satisfy** : un ou plusieurs éléments du modèle permettent de satisfaire une exigence.
- **Verify** : un ou plusieurs éléments du modèle permettent de vérifier et valider une exigence.
- **Refine** : un ou plusieurs éléments du modèle redéfinissent une exigence.

4 Présentation du support du cours

Pour ce cours, nous allons étudier le cas pratique d'un système complexe qu'est une **caméra numérique embarquée**.

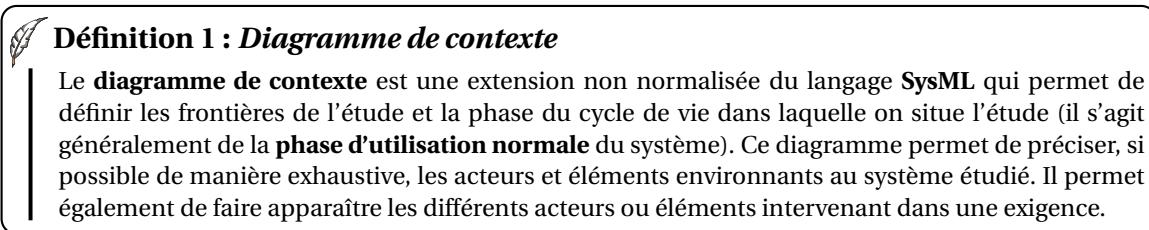


Liens	Significations et commentaires de la relation	Orientations
	Contenance : décompose une exigence en plusieurs autres plus faciles ensuite à identifier lors de la mise en place du système ou des tests.	Cercle contenant une croix du côté du conteneur
	Association : relie deux éléments considérés d'égale importance et indique qu'ils sont en lien sans en indiquer la nature	<ul style="list-style-type: none"> • Unidirectionnelle (une seule flèche); • bidirectionnelle (sans flèche).
	Inclusion (extension, raffinement ou dérivation)	<ul style="list-style-type: none"> • Du cas d'utilisation global vers un cas d'utilisation partiel inclus avec le mot clé <i>include</i> pour l'inclusion; • du cas d'utilisation partiel vers le cas d'utilisation global avec le mot clé <i>extend</i> pour l'extension; • de l'exigence partielle vers l'exigence globale avec le mot clé <i>refine</i> pour l'ajout de précisions, par exemple des données quantitatives, pour le raffinement; • de l'exigence partielle vers l'exigence globale avec le mot clé deriveReqt pour relier de manière dérivée des exigences de niveaux différents, par exemple entre un système et certains de ses sous-systèmes (dérivation).
	Généralisation : spécialisation d'un élément (cas d'utilisation, bloc, etc)	Pointe blanche orientée vers l'élément plus général.
	Composition : relie deux blocs et indique qu'un élément est structurellement indispensable à l'autre	Losange plein du côté du composé (ou système principal), l'autre extrémité du côté du composant.
	Agrégation : même rôle que la relation de composition mais elle a un sens moins fort : en général, elle indique que le composant est présent de manière optionnelle	Losange vide du côté du composé (ou système principal), l'autre extrémité du côté du composant

TABLE 1 – Principaux liens graphiques du langage SysML

II. Analyse du contexte et des exigences du système

1 Diagramme de contexte



La figure suivante illustre le contexte de la caméra numérique embarquée.

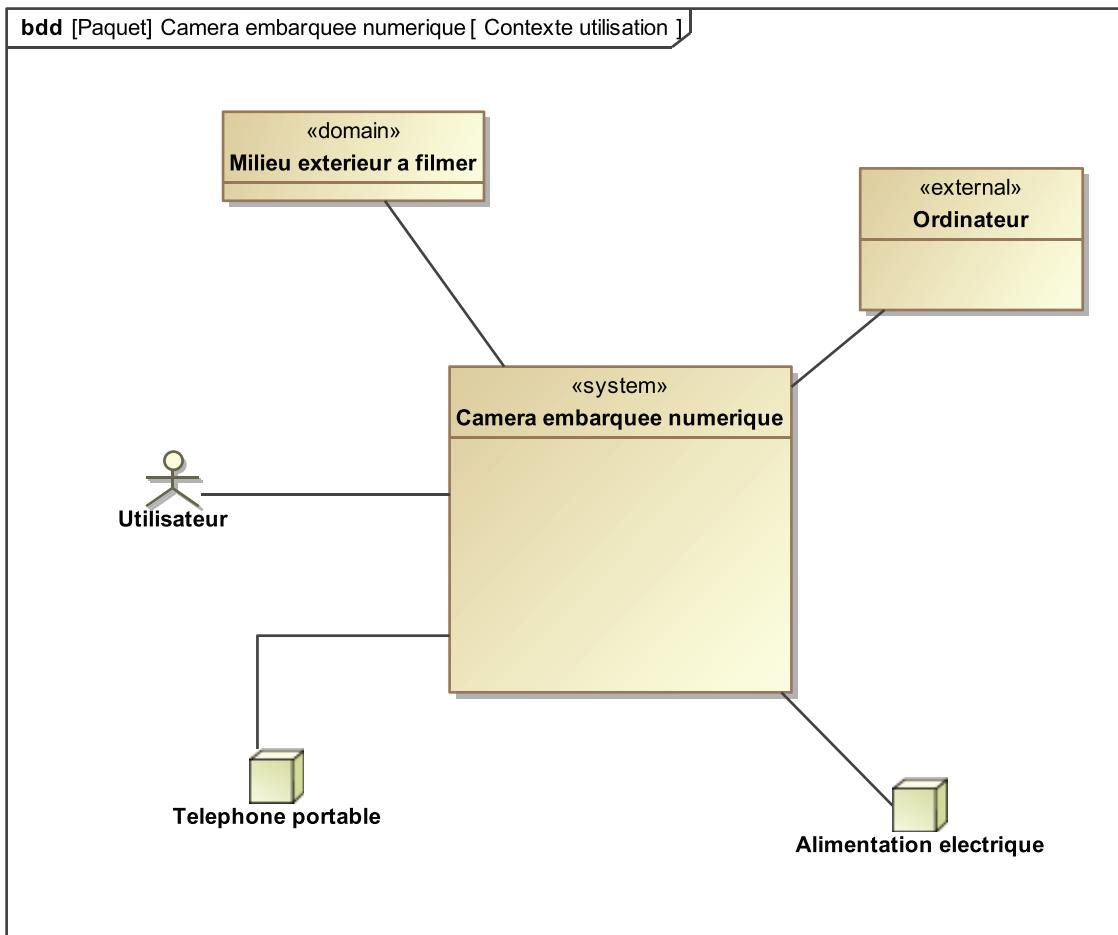


FIGURE 3 – Diagramme de contexte de la caméra numérique embarquée



Remarque 2 :

L'analyse de la syntaxe de ce diagramme est la même que pour le **diagramme de définition des blocs** qui viendra dans la partie 1.

2 Diagramme des exigences



Définition 2 : Exigence

Une **exigence** permet de spécifier un capacité ou une contrainte qui doit être satisfaite par le système. Les exigences servent à établir un contrat entre le client et les réalisateurs du futur système (cahier des charges).

De nombreux domaines peuvent être couverts, les plus classiques étant les exigences **environnementales, économiques, fonctionnelles ou techniques**.



Définition 3 : Diagramme des exigences (req)

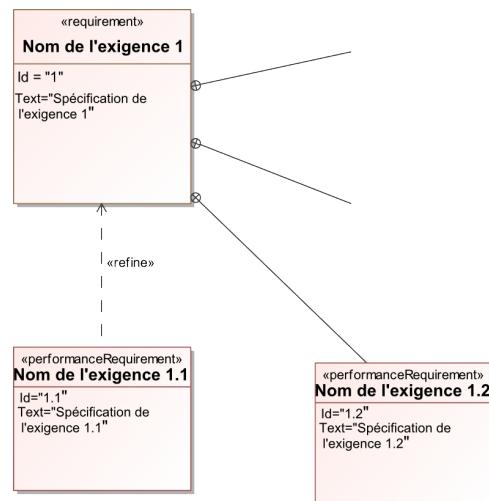
Le diagramme des exigences, appelé *Requirement Diagram (req)* dans le langage **SysML**, est le seul diagramme transversal du langage **SysML** (voir figure 1).

L'objectif de ce diagramme est de modéliser les exigences devant être vérifiées par le système en liant les solutions mises en oeuvre sur le système avec les besoins définis dans le cahier des charges. Ce diagramme traduit, par des fonctionnalités ou des contraintes, ce qui doit être satisfait par le système.

Chaque exigence est représenté dans un rectangle (constituant un bloc) séparé en deux partie :

- un **intitulé** de l'exigence sous «*requirement*» (exigence en anglais) pour la partie supérieure;
- un **identifiant** sous forme de numéro et une **description textuelle** libre mais concise.

Il est possible, mais non obligatoire, de relier les exigences entre elles par des liens tels que ceux présentés dans le tableau 1.

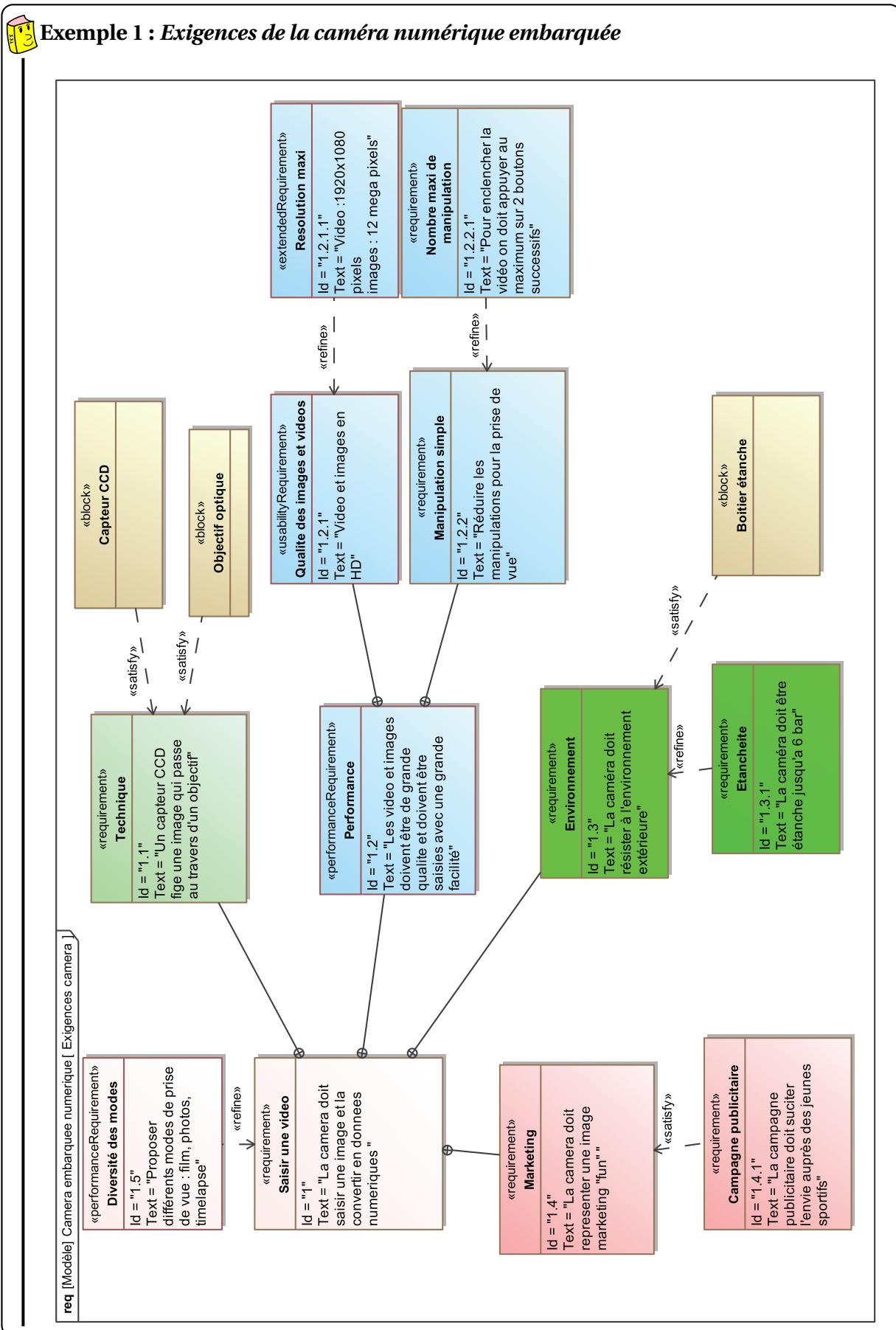


Remarque 3 :

- Le diagramme doit être le plus lisible possible et donc le plus simple (il est possible de réaliser plusieurs diagrammes pour alléger les schémas).
- Il est possible d'associer des propriétés aux exigences telles que :
 - une **priorité**, par exemple haute, moyenne ou basse;
 - une **indication de la “source”**, par exemple client, législation ou concurrence;
 - un **statut**, par exemple proposé, validé, implanté;
 - ou, **de manière générale**, toute donnée pouvant se rapporter à une exigence devant être validée à un niveau du cycle de vie du produit.

Pour l'exemple de la caméra numérique embarquée il est possible de mettre en place le diagramme des exigences suivant. On peut alors décliner l'exigence principale en 4 autres exigences :

- Technique;
- Marketing;
- Performance;
- Environnement.



3 Diagramme des cas d'utilisation



Définition 4 : Diagramme des cas d'utilisation (uc ou ucd)

Le diagramme des cas d'utilisation est un *diagramme comportemental*, appelé **Use Case Diagram** (**uc** ou **ucd**) dans le langage **SysML**. L'objectif de ce diagramme est de montrer les fonctionnalités offertes par un système en identifiant les services qu'il rend : il permet donc de modéliser les exigences selon un point de vue complémentaire à celui exposé par le diagramme des exigences (voir partie 2). L'énoncé d'un cas d'utilisation doit se faire hors technologie, puisque il est défini en termes de résultats attendus.

Les éléments graphiques utilisés dans ce diagramme sont principalement :

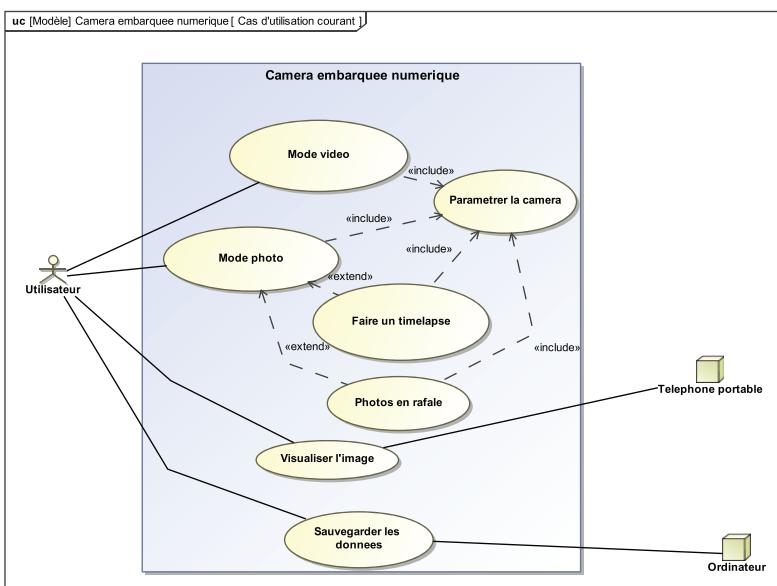
- Les acteurs, entités extérieures au système et en interaction avec lui, sont représentés par le pictogramme “*bonhomme bâton*” et sont reliés à un ou plusieurs cas d'utilisation par une ligne simple appelée association.
- Les cas d'utilisation sont représentés sous forme d'ovales. Ils donnent les fonctionnalités du système et sont énoncés du point de vue de l'acteur.
- La frontière du système permet de symboliser les limites du modèle et est représentée par un simple rectangle englobant les cas d'utilisation, les acteurs étant à l'extérieur, à gauche si ils sont considérés comme “principaux”, à droite si ils sont considérés comme “secondaires”.



Exemple 2 : Cas d'utilisation de la caméra numérique embarquée

Pour le système étudié voici le diagramme mis en place :

- un acteur qui interagit avec le système (liens avec les cas d'utilisation);
- les cas d'utilisation relié à l'acteur et rédigé selon le point de vue de ce dernier;
- la frontière du système qui contient tous les éléments permettant d'atteindre les objectifs terminaux.



Remarque 4 :

Les fonctionnalités d'un système correspondent à des cas d'utilisation, c'est-à-dire à des services rendus par le système. Il n'apparaîtra donc pas ce qui ne peut être fait par des acteurs extérieurs : ainsi, par exemple, le lavage, la recharge, le recyclage, la réparation, etc. ne doivent pas apparaître si le système n'a pas été développé expressément pour cela.

III. Analyse structurelle du système

1 Diagramme de définition des blocs



Définition 5 : Diagramme de définition des blocs (bdd)

Le **diagramme de définition de blocs** est un *diagramme structurel* appelé **Block Definition Diagram (bdd)** dans le langage **SysML**.

L'objectif de ce diagramme est de décrire le système via des blocs (blocks dans le langage **SysML**) et représentant des éléments matériels (cas le plus fréquent) mais également des entités abstraites (regroupement logique d'éléments) ou des logiciels.

Ce diagramme représente les caractéristiques principales de chaque bloc ainsi que les liens entre eux : il permet donc une **modélisation de l'architecture du système**.



Propriété 1 : Représentation graphique d'un bloc

D'un point de vue graphique, on représente un bloc par un rectangle avec le stéréotype «*block*» comprenant un titre et des compartiments étagés regroupant des propriétés particulières. Les plus significatives sont :

- *value* : exprime une caractéristique quantifiable;
- *part* : représente ce qui compose le bloc (équivalent à un lien de composition);

La figure ?? donne un exemple de constitution d'un bloc.

«*block*»

Voiture

parts

roue : Roue [4]

values

kilométrage



Exemple 3 : Diagramme de définition des blocs de la caméra numérique embarquée

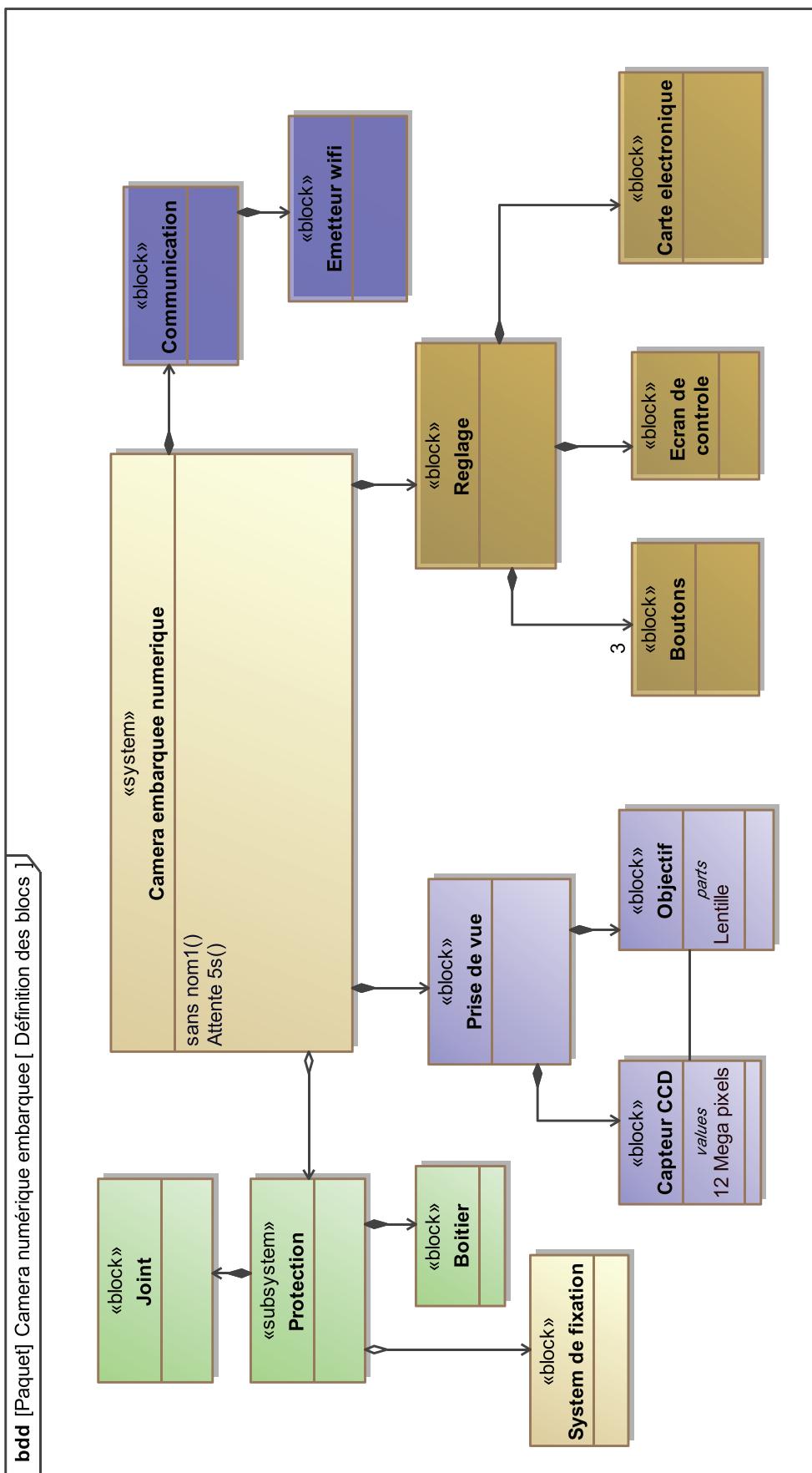
Pour le système de la caméra numérique embarquée (éclaté représenté en figure ci-contre) il est possible de mettre en place le diagramme de définition de blocs présenté sur la figure suivante qui fait apparaître une hiérarchie de blocs indiquant ce dont chaque bloc est composé ainsi que les relations entre les blocs.

- Relations de **composition** (trait avec losange plein reliant deux blocs entre eux).
- Relation d'**agrégation** (trait avec losange vide) : le système de fixation est optionnel pour ce système.
- Relation d'**association** : l'objectif et le capteur CCD sont associés à un niveau d'importance équivalent (absence de flèche).





Exemple 3 : Diagramme de définition des blocs de la caméra numérique embarquée



2 Diagramme de blocs internes



Définition 6 : Diagramme de blocs internes (ibd)

Le **diagramme de blocs internes** est un *diagramme structurel* appelé **Internal Block Diagram (ibd)** dans le langage **SysML**.

Le diagramme de blocs internes est rattaché à un bloc issu du diagramme de définition de blocs présenté dans la partie précédente, le cadre du diagramme représentant la frontière d'un bloc.

Il introduit la notion fondamentale de "**port**" qui correspond à un point d'interaction avec l'extérieur du bloc.

Les connecteurs (traits) entre les ports indiquent soit les associations soit les **flux de matière, d'énergie et d'information** entre les différents blocs.



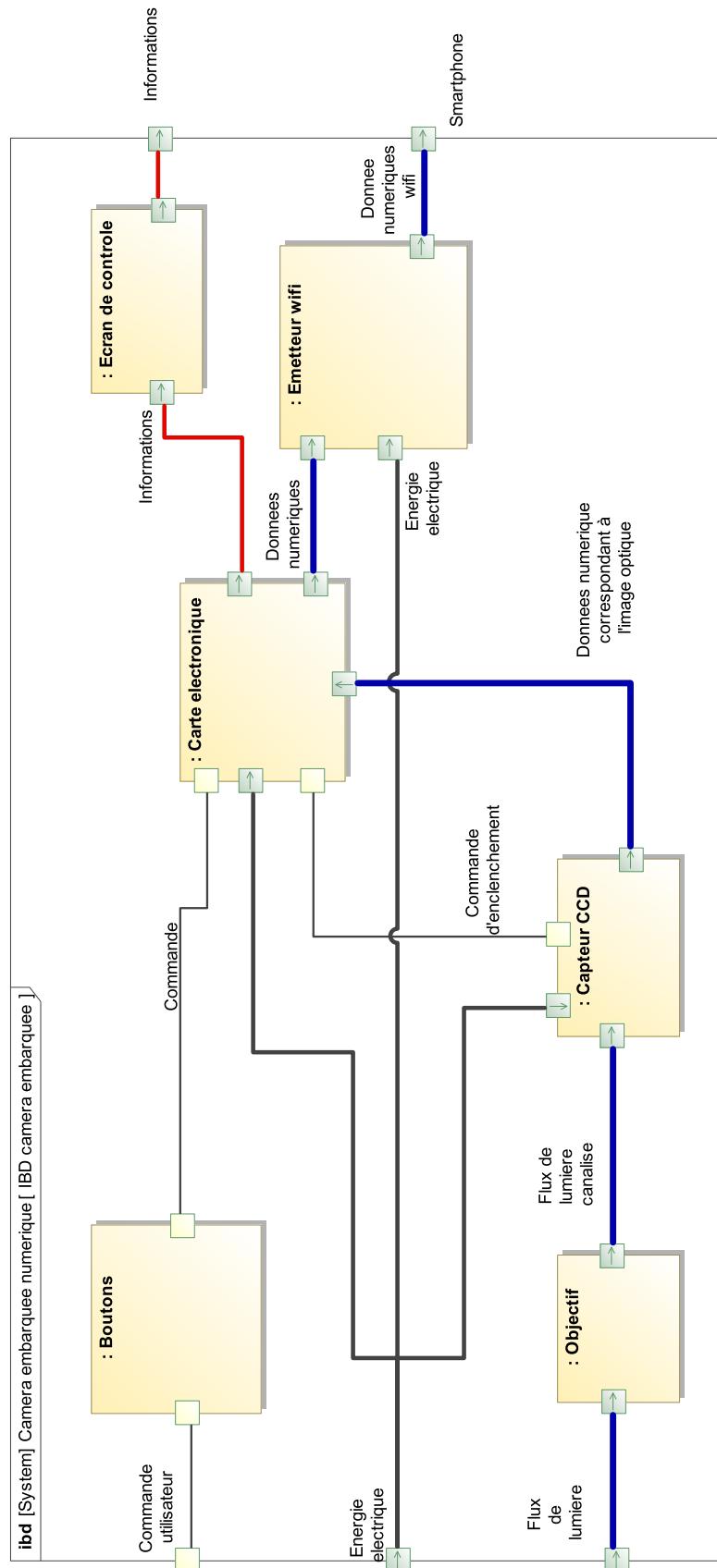
Propriété 2 : Représentation graphique

- Les blocs se représentent de la même manière que précédemment.
- Les ports se représentent par un carré placé sur le contour du bloc :
 - les ports *flux* : indiquent les échanges de matière, d'énergie et d'information entre blocs : ce type de port contient une flèche dont le sens (entrante, sortante ou bidirectionnelle) indique celui du flux;
 - les ports *standards* : indiquent la logique de commande et les interfaces d'un bloc : ce type de port ne contient pas d'indication particulière.



Exemple 4 : Diagramme de blocs internes de la caméra numérique embarquée

Pour le système de la caméra numérique embarquée, il est possible de mettre en place le diagramme des blocs interne suivant.



3 Diagramme paramétrique



Définition 7 : Diagramme paramétrique (par)

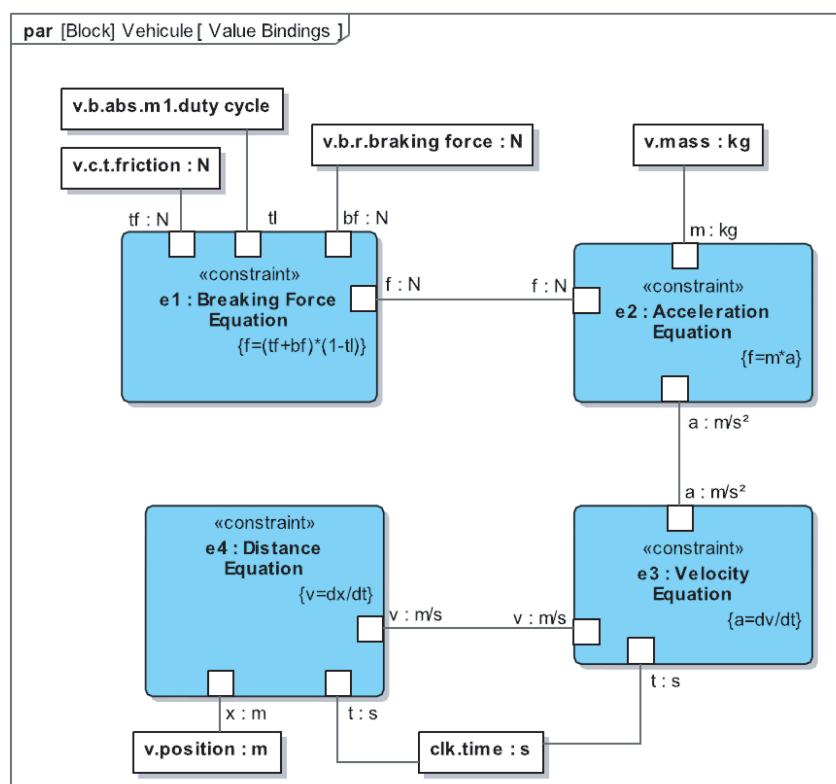
Le **diagramme paramétrique** est un *diagramme structurel* appelé **Parametric Diagram (par)** dans le langage SysML.

Ce diagramme est une extension du diagramme de blocs internes (**ibd** présenté précédemment) et il partage donc les mêmes éléments graphiques. Il présente la particularité de pouvoir connecter entre elles des contraintes ajoutées au diagramme de blocs par le biais d'un bloc particulier, dit "**de contraintes**" (*constraint block*) qui contient des paramètres et une relation, en général *mathématique*, les reliant.



Exemple 5 : Diagramme paramétrique d'un véhicule

Voici un exemple de diagramme paramétrique du freinage d'un véhicule.



IV. Analyse comportementale du système

L'analyse comportemental d'un système peut se présenter suivant deux approches.

- La première à l'aide du **diagramme de séquence** permet une **analyse globale** du système.
- La deuxième avec les **diagrammes d'état** représente le système d'un **point de vue interne** (Cette deuxième partie sera plus détaillée durant le semestre 2).

1 Diagramme de séquence



Définition 8 : Diagramme de séquence (seq)

Le **diagramme de séquence** est un *diagramme comportemental* appelé **Sequence Diagram (seq)** dans le langage **SysML**.

L'objectif de ce diagramme est de décrire les interactions existant entre plusieurs entités, celles-ci pouvant être des acteurs, le système ou ses sous-systèmes. Le diagramme ne montre donc que l'enchaînement séquentiel des différentes interactions.

Un diagramme de séquence est rattaché à un cas d'utilisation et décrit ce dernier en entier ou en partie, ce qui correspond à un scénario de fonctionnement possible, défini dans un cadre précis : il peut donc aboutir tout aussi bien à des évolutions positives (fonctionnement normal) ou négatives (gestion des problèmes), en particulier dans la phase de démarrage avant le fonctionnement normal.



Propriété 3 : Représentation graphique

Les éléments graphiques utilisés dans ce diagramme sont principalement :

- Des *traits verticaux* en pointillés appelés "**lignes de vie**" avec l'indication des propriétaires (en général des acteurs, le système et tout ou partie de ses sous-systèmes) sur la partie supérieure. Le temps se déroule du haut vers le bas, sans échelle particulière.
- Les **messages** sont les entités qui peuvent transits d'une ligne de vie à l'autre (*traits horizontaux*). La réception d'un **message** provoque un événement chez le récepteur.
 - La flèche pointillée représente un retour. Cela signifie que le message en question est le résultat direct du message précédent.
 - Un message synchrone (émetteur bloqué en attente de réponse) est représenté par une flèche pleine;
 - un message asynchrone est représenté par une flèche évidée.
 - La flèche qui boucle (message réflexif) permet de représenter un comportement interne.



Remarque 5 :

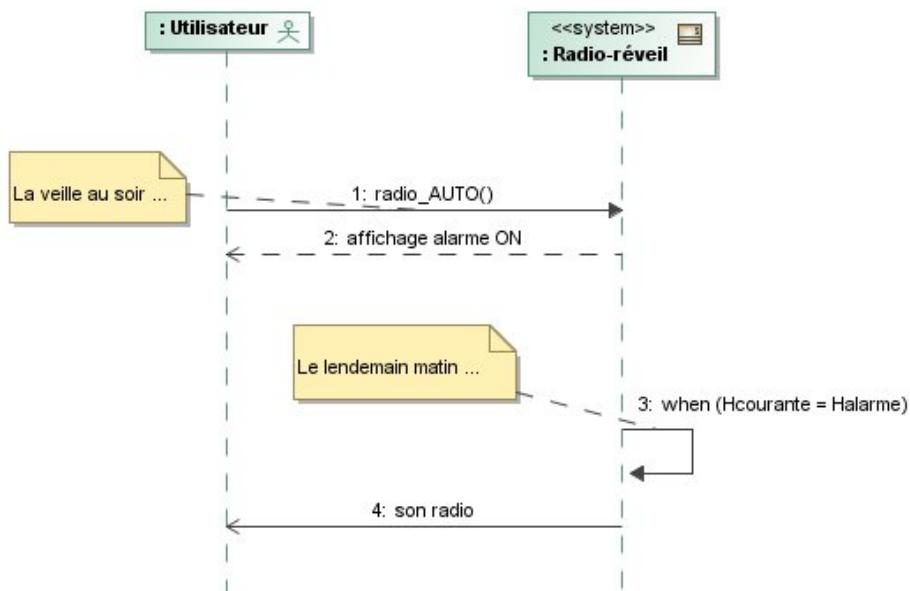
- Ce diagramme comportemental est en forte interaction avec le diagramme de cas d'utilisation.
- On construit généralement un diagramme de séquence par scénario.
- Ce diagramme permet de montrer les interactions entre les différentes parties non visibles dans un diagramme de cas d'utilisation qui n'indique que l'association entre l'acteur et un cas d'utilisation.



Exemple 6 : Radio réveil

Pour le cas d'utilisation “Être réveillé à l'heure en musique”.

- Le premier message est un message synchrone, donnant lieu à un retour : l'affichage d'un point à côté de l'heure indiquant que l'alarme est positionnée.
- Le fait que radio-réveil détecte que l'heure courante devient égale à l'heure d'alarme est représenté par un message réflexif avec le mot-clé when.
- Le dernier message est un signal asynchrone.



Définition 9 : Fragments combinés

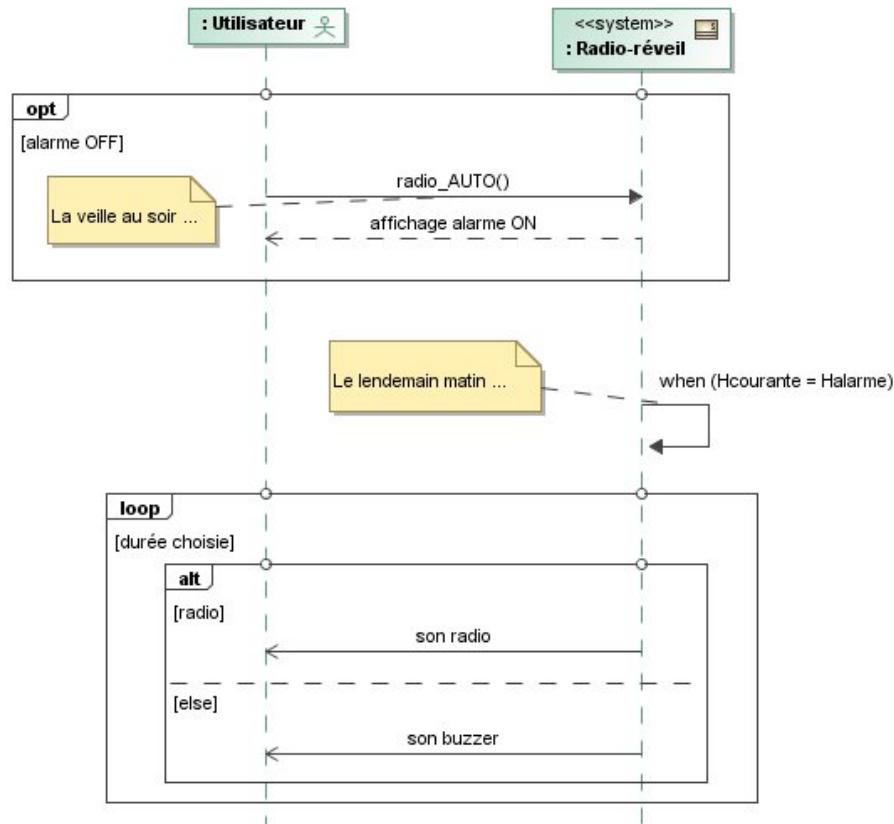
Chaque fragment possède un opérateur et peut être divisé en opérandes. Les principaux opérateurs sont :

- **loop** - boucle. Le fragment peut s'exécuter plusieurs fois, et la condition de garde explicite l'itération;
- **opt** - optionnel : le fragment ne s'exécute que si la condition fournie est vraie;
- **alt** - fragments alternatifs : seul le fragment possédant la condition vraie s'exécutera.



Exemple 7 : Radio réveil

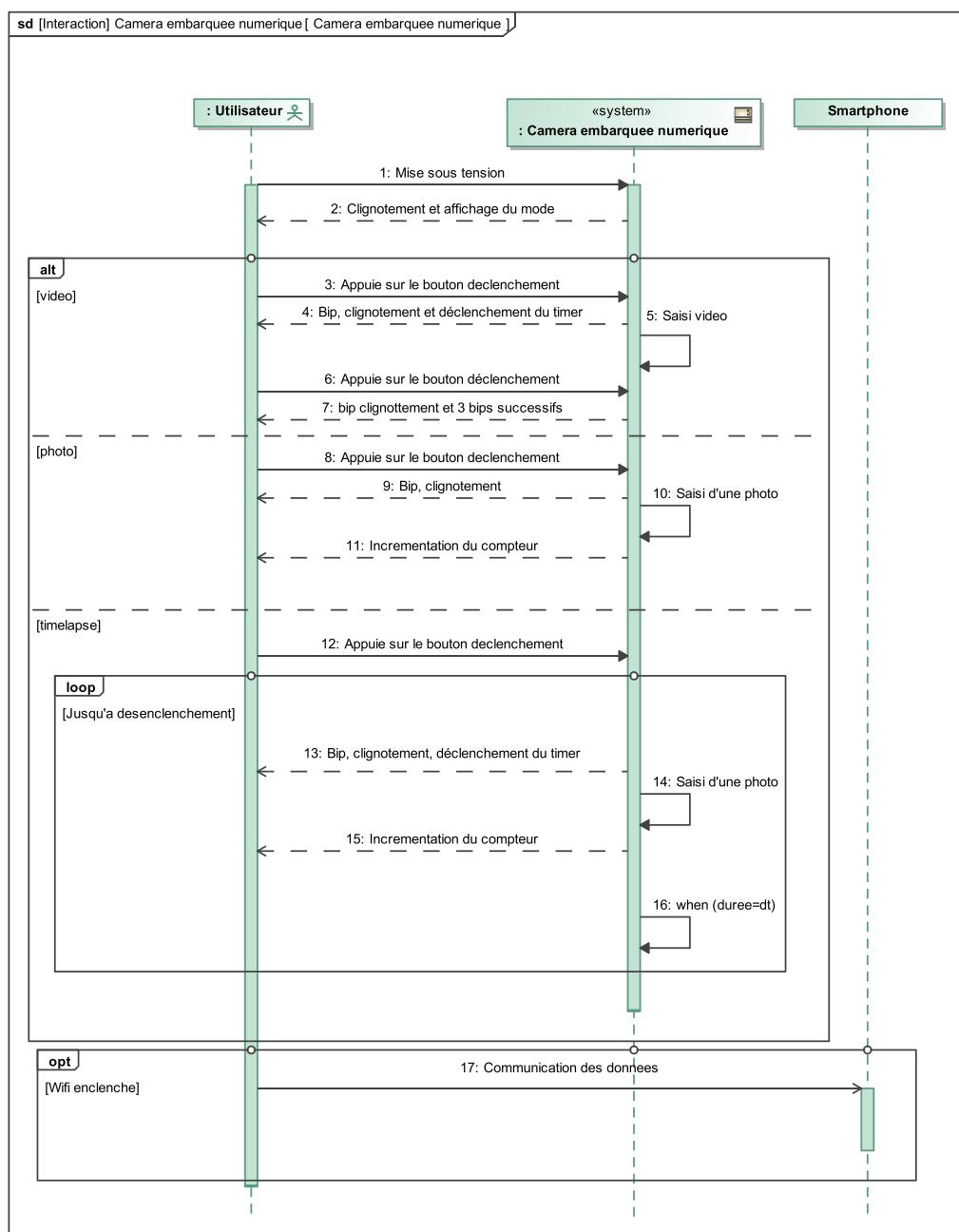
- Le son qui sort de la radio est continu pendant plusieurs minutes, ce n'est pas un simple signal unitaire : pour le représenter, positionnons un fragment de boucle.
- L'utilisateur sera réveillé par la radio ou le buzzer suivant son choix. Nous pouvons donc ajouter un fragment **alt** avec deux opérandes.
- Enfin, le premier message n'est pas nécessaire si l'alarme était déjà positionnée la veille : il est donc optionnel.





Exemple 8 : Diagramme de séquence de la caméra embarquée numérique

On peut décrire le fonctionnement du système étudié plus complexe à l'aide du diagramme suivant entre l'utilisateur et la caméra embarquée numérique.



2 Diagramme d'états

Le diagramme d'états (ainsi que le diagramme d'activité) fera l'objet d'une étude plus poussée au semestre 2 mais nous pouvons déjà le définir. Ils permettent de modéliser le comportement d'un système à des fins de programmation.

V. Interaction entre les différents diagrammes

Conclusion :

La modélisation SysML d'un système permet de décrire de manière ordonnée un système. La richesse et la polyvalence des diagrammes donne une bonne vision d'ensemble du système. De plus cette modélisation permet d'aller jusqu'à la simulation des systèmes et permet ainsi d'aider les ingénieurs dans leur démarche de conception.

Le synoptique ci-dessous (figure 4) donne une possibilité de mise en place d'une modélisation SysML, depuis la définition des exigences et/ou des cas d'utilisation jusqu'à la mise en oeuvre du diagramme paramétrique en passant par les différents diagrammes et les simulations associées.

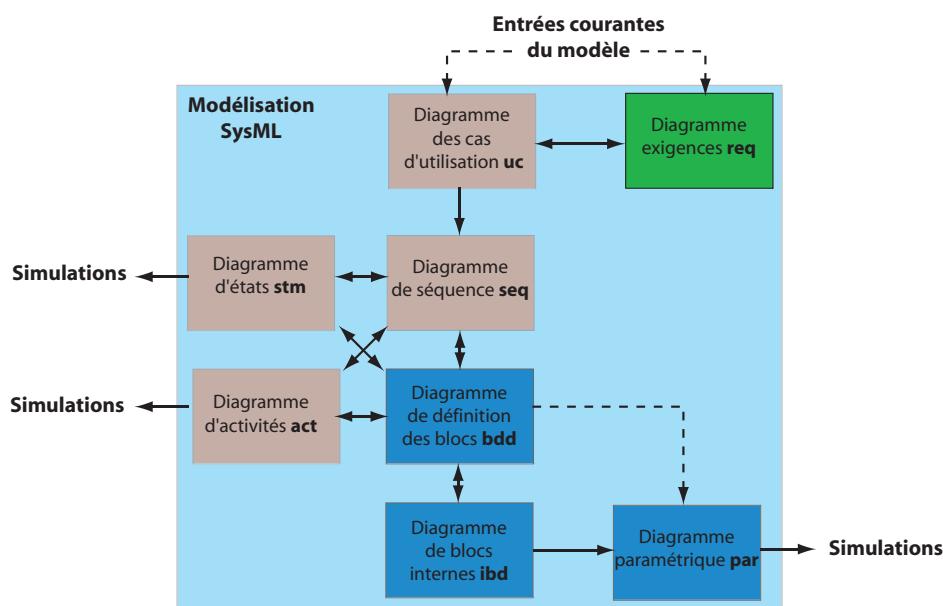


FIGURE 4 – Représentation synoptique des liens entre les différents diagrammes

VI. Modélisation et analyse structurelle : chaîne d'information et d'énergie

Il existe une représentation générique permettant d'organiser les composants en fonction de leur fonction dans un système.

- **la chaîne d'information** qui transfère stocke et transforme l'information (appelée également "partie commande");
- **la chaîne d'énergie** qui transforme l'énergie et permet d'agir sur le système physique (appelée également "partie opérative").

Chaque chaîne fonctionnelle est composée d'un nombre limité de fonctions qui pourront être étudiées séparément selon leur technologie (mécanique, électrique, automatique, etc. - voir cours suivant). Ces deux entités sont reliées par des **interfaces**. Le tout est appelé **chaîne fonctionnelle** (voir figure 5)

1 La chaîne d'information

La chaîne d'information permet de **communiquer** avec le milieu extérieur (utilisateur, ou autres systèmes ayant leur propre chaîne fonctionnelle) et de **commander** les éléments de la chaîne d'énergie. Elle permet :

- **d'ACQUERIR** des informations :
 - sur l'état d'un produit ou de l'un de ses éléments à l'aide de capteurs (cf. flèche "grandeur physique à acquérir", fig.5),
 - issues d'interfaces Homme/machine (boutons, ordinateur, etc.),
 - élaborées par d'autres chaînes d'informations,
 - sur des processus gérés par d'autres systèmes (bases de données, partage de ressources, etc.);
- de **TRAITER** ces informations à l'aide de matériels tels que des automates programmables, ordinateurs, etc;
- de **COMMUNIQUER** les informations traitées pour :
 - donner des ordres à la chaîne d'énergie,
 - élaborer des messages destinés aux interfaces Homme/Machine, ou à d'autres chaînes d'informations.

2 La chaîne d'énergie

La chaîne d'énergie, associée à sa commande, assure la réalisation d'une fonction de service dont les caractéristiques sont spécifiées dans le cahier des charges. Elle permet **d'agir** sur la matière d'œuvre. Les fonctions qui la composent sont :

- **ALIMENTER** : recevoir une énergie d'entrée (qui n'est pas forcément celle de la chaîne d'information) pour l'adapter à la machine. Par exemple : fournir une énergie électrique à une tension donnée ou fournir une énergie pneumatique à une pression donnée.
Matériel possible : prise réseau, raccord réseau, etc.
- **STOCKER** : stocker une énergie d'entrée (qui n'est pas forcément celle de la chaîne d'information) pour l'adapter à la machine. *Matériel possible : batterie.*
- **MODULER/DISTRIBUER** : l'élément "*distributeur*" (ou **pré-actionneur**) reçoit les ordres de la chaîne d'information pour distribuer l'énergie vers les actionneurs au moment souhaité.
Matériel possible : contacteur, relais, distributeur, electro-vannes, etc.
- **CONVERTIR** : l'élément convertisseur d'énergie s'appelle l'**actionneur**. Il permet de transformer l'énergie distribuée en énergie utile (en générale en énergie mécanique de rotation ou de translation).
Par exemple : moteur électrique, vérin, etc...
- **TRANSMETTRE** : l'énergie est amenée jusqu'à la matière d'œuvre via un **transmetteur**. Ce dernier, en plus de transmettre, peut modifier l'énergie selon les besoins (adapter la vitesse, transformer le mouvement, etc.).
Matériel possible : limiteur de couple, engrenages, système vis-écrou.
- **AGIR** : c'est un élément essentiel qui va finaliser la fonction opérative en réalisant directement l'action sur la matière d'œuvre pour lui fournir sa valeur ajoutée. Il doit être adapté au contact avec la matière d'œuvre. On l'appelle également l'**effecteur**.
Matériel possible : tapis roulant, ventouse, roue, balais, etc.

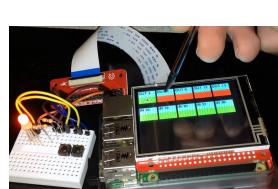
Les éléments qui composent la chaîne fonctionnelle (préactionneur, actionneur, etc.) seront présentés en TD et TP.

3 Les interfaces

Définition 10 : *Les interfaces*

Les interfaces permettent :

- d'acquérir certaines données de la chaîne d'énergie vers la chaîne d'information (capteurs).
- d'acquérir ou de communiquer des informations de l'utilisateur du système



Interface de communication sur une *raspberry-pi*



Centrale inertielle de drone

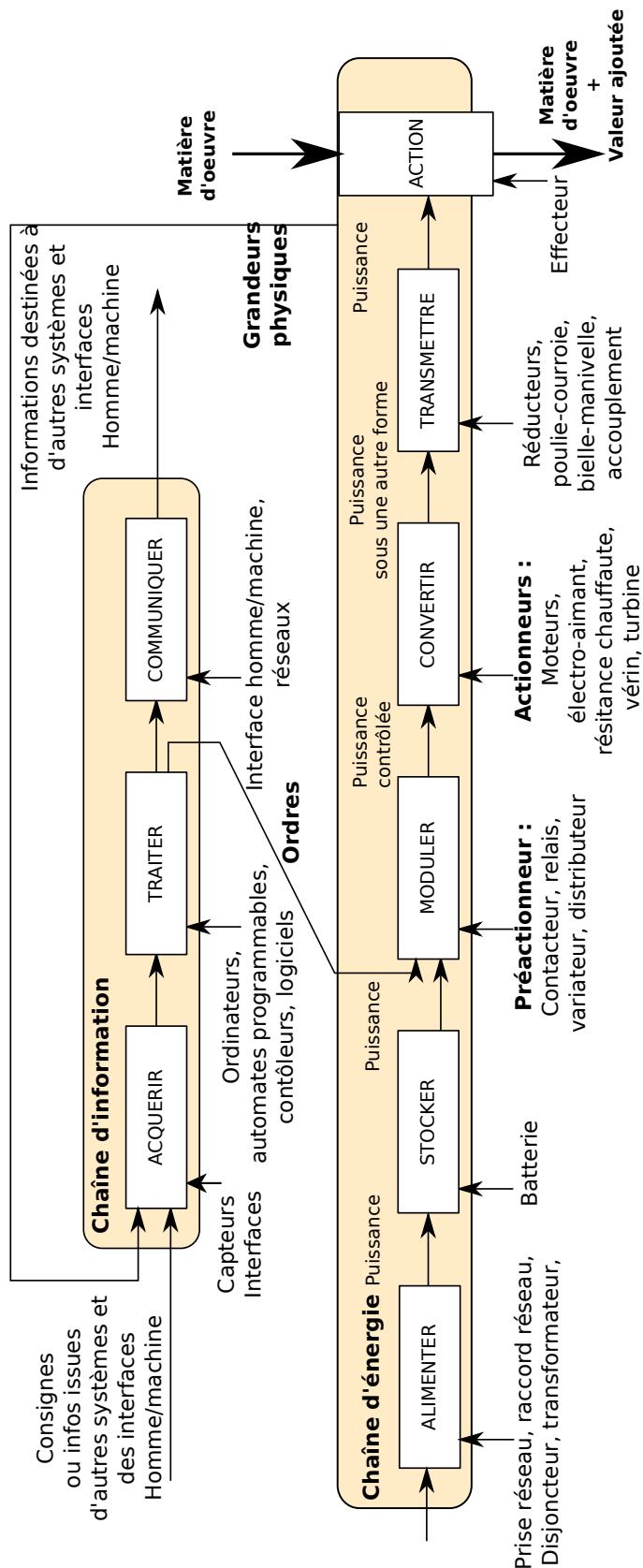


FIGURE 5 – Chaîne fonctionnelle