

Formation IRIS-SE

«Introduction à SysML»



Philippe LE GAL

Philippe.Le-Gal@ac-nantes.fr



Sommaire



Introduction

Les diagrammes SysML

Présentation de DARwIn-OP

Avant de commencer...

Diagramme de contexte

Diagramme des cas d'utilisation

Diagramme d'exigences

Diagramme de séquence

Diagramme de définition de bloc

Diagramme de bloc interne

Diagramme de machine d'état

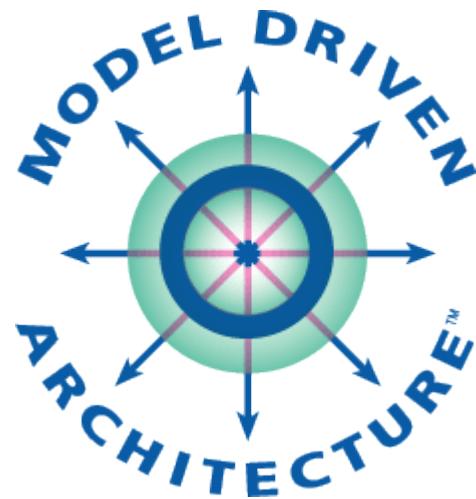
Diagramme d'activité

Diagramme paramétrique



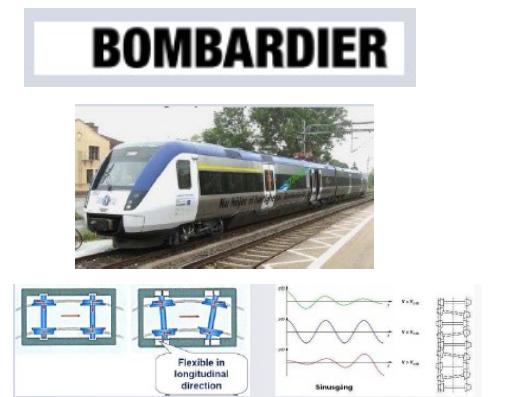
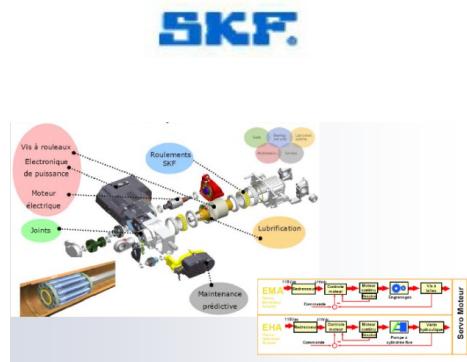
Introduction (1) : Ingénierie Système

- Ingénierie Système (IS), ou Systems Engineering en anglais (SE) : démarche méthodologique dont le but est de formaliser et d'appréhender la conception de **systèmes complexes** avec succès.
- L'Ingénierie Système s'adresse aux secteurs suivants : systèmes embarqués, automobile, ferroviaire, aéronautique, espace, militaire, télécoms, médical, production d'énergie, etc.
- Les méthodes de l'Ingénierie Système (IS) reposent sur des approches de **modélisation** et de **simulation** pour valider les exigences ou pour évaluer le système à concevoir.



Introduction (2) : SysML

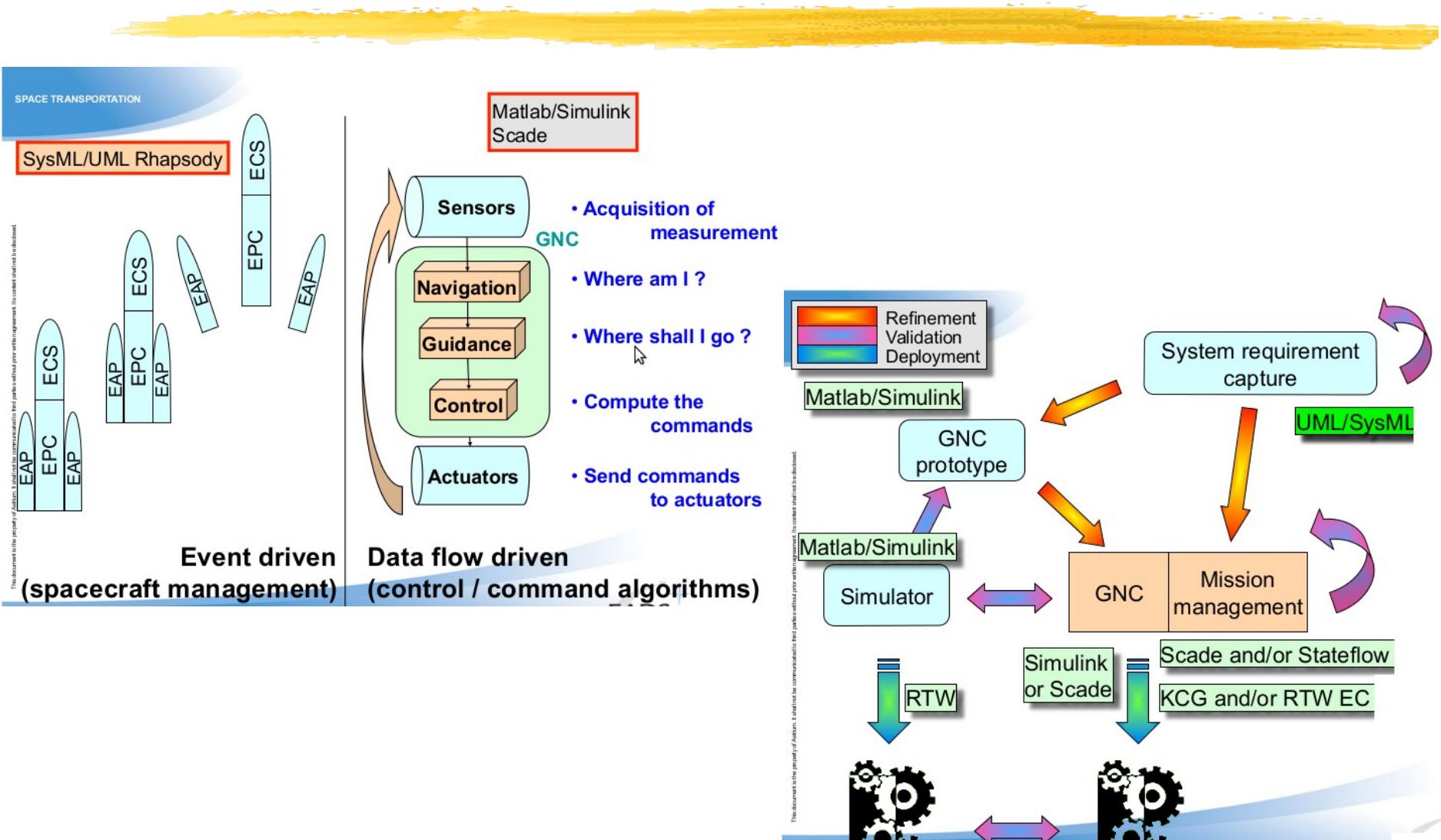
- La complexité des systèmes augmente :
 - diversité des composants
 - diversité des spécialistes
- Il faut adapter (simplifier) les représentations du système (modèles) au besoin à un moment donné.
- Unifier les outils : SysML
- SysML n'est pas une méthode de conception, c'est un ensemble d'outils graphiques associé à un méta-langage



Introduction (3) : Avantages

- Collaboration **transdisciplinaire**
- Partage et interprétation des informations (stockage, mise à jour, etc.)
- Modélisation à toute les étapes de la vie du système
- **Intégration** des composantes techniques d'un système dans un même modèle (logiciel et actionneur mécanique par exemple)
- Validation de solutions par la **simulation** (diagrammes paramétriques)
- Conception ou description des systèmes existants.

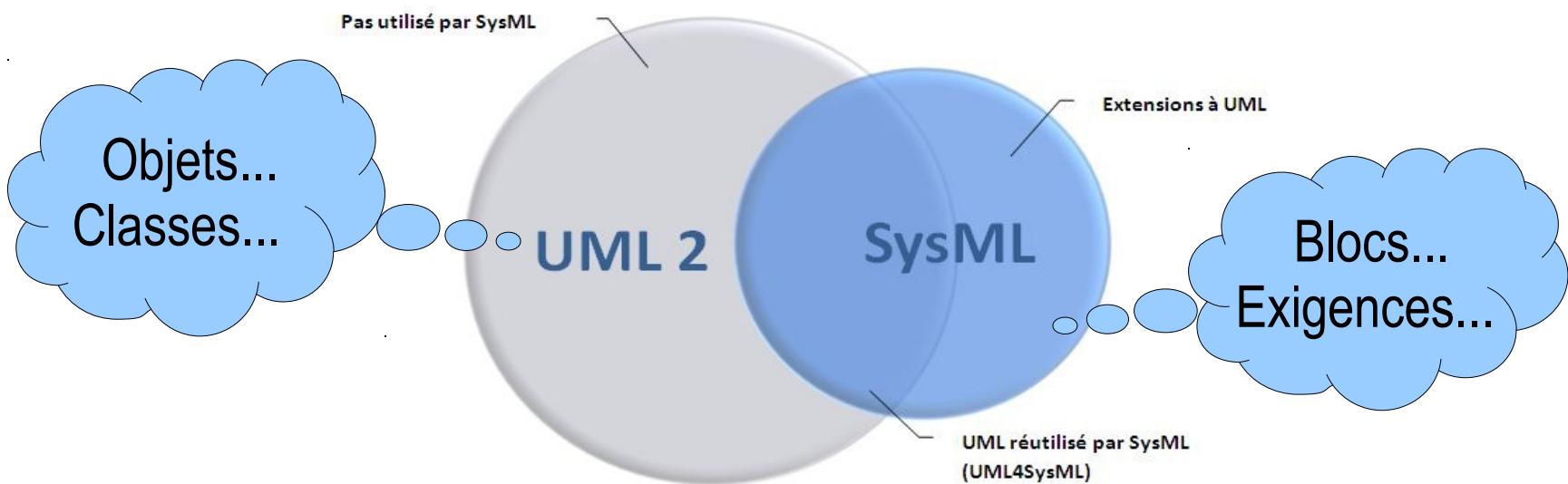
Introduction (4) : Exemple EADS



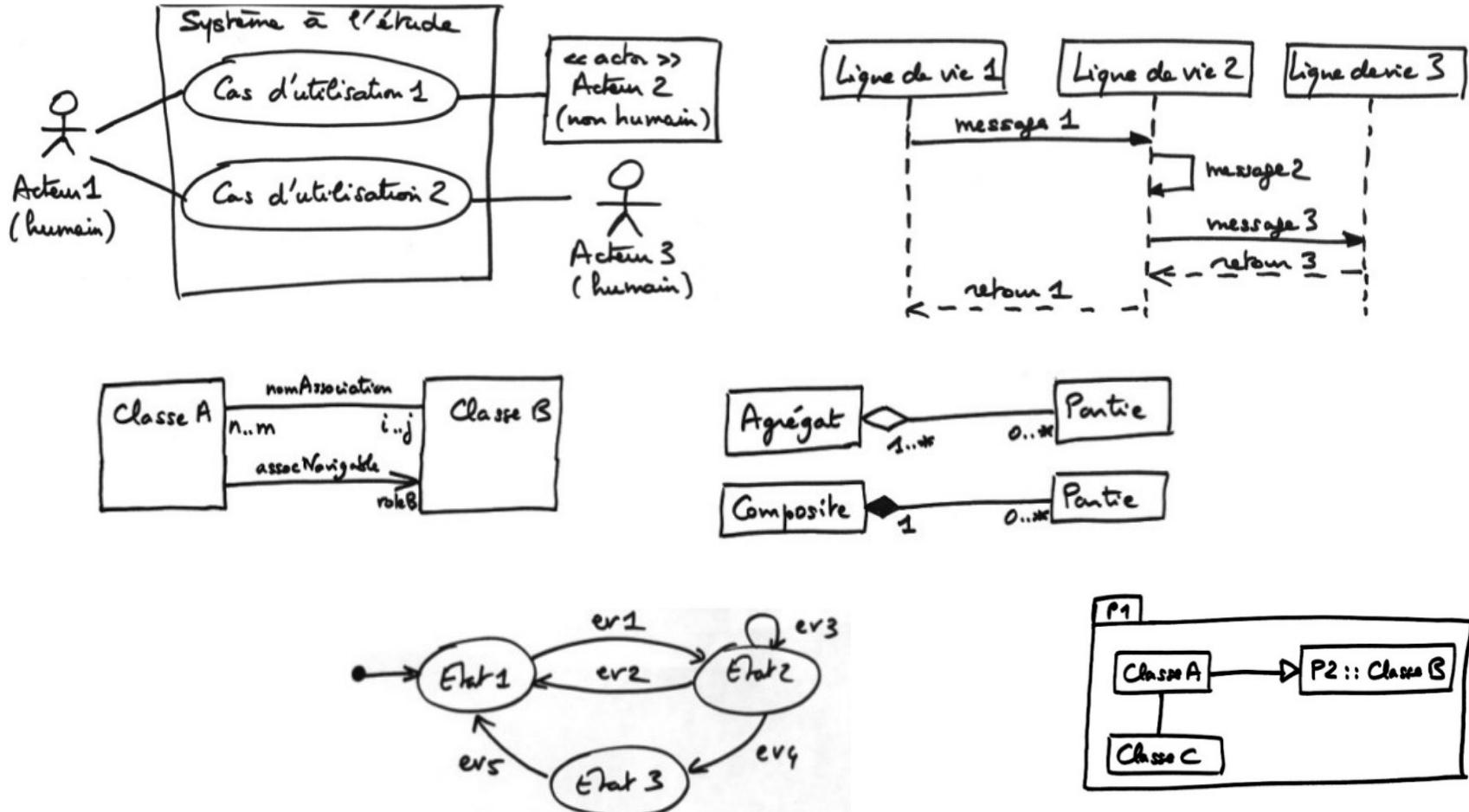


■ Spécification courante : 1.3 (juin 2012)

- SysML = Profile UML
- UML (Unified Modeling Language) : langage de modélisation des données et des traitements utilisé dans le **génie logiciel** dans le cadre de la **conception objet**.
- UML évolue depuis 1995. Spécification courante : 2.4.1



Introduction (6) : SysML vs UML



Quelques diagrammes UML...

Introduction (7) : SysML vs UML

- SysML est basé sur UML et remplace la modélisation de **classes** par des **blocs** pour un vocabulaire plus adapté à l'Ingénierie Système.
- Un **bloc** englobe tout concept **logiciel**, **matériel**, **données** et structures de données, **processus**, moyen de **transmission** et même gestion des personnes.
- Les spécifications :



732 pages...



272 pages...

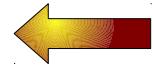
<http://www.omg.org/spec/UML/2.4.1/>

<http://www.sysml.org/specs/>

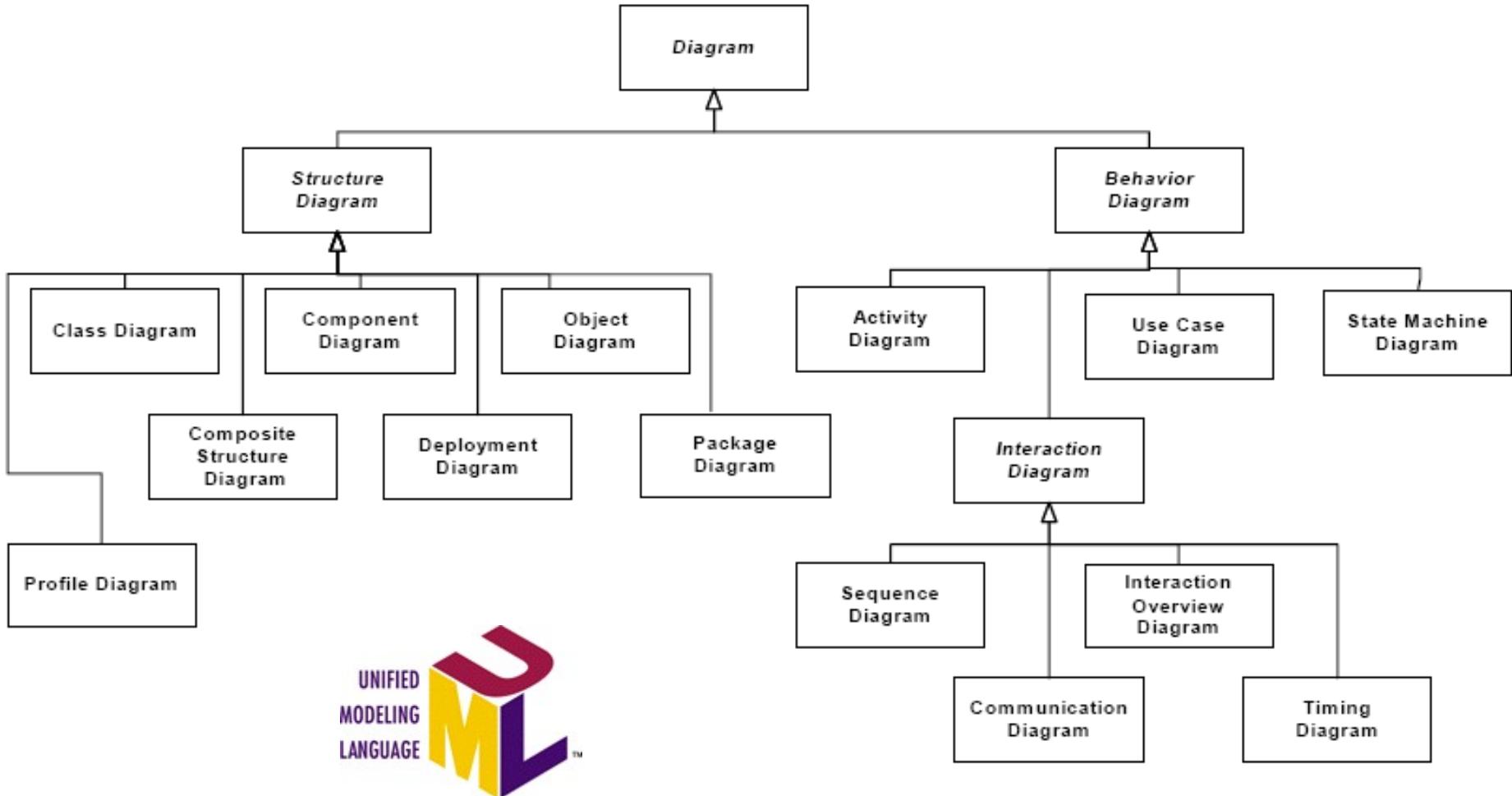
Sommaire



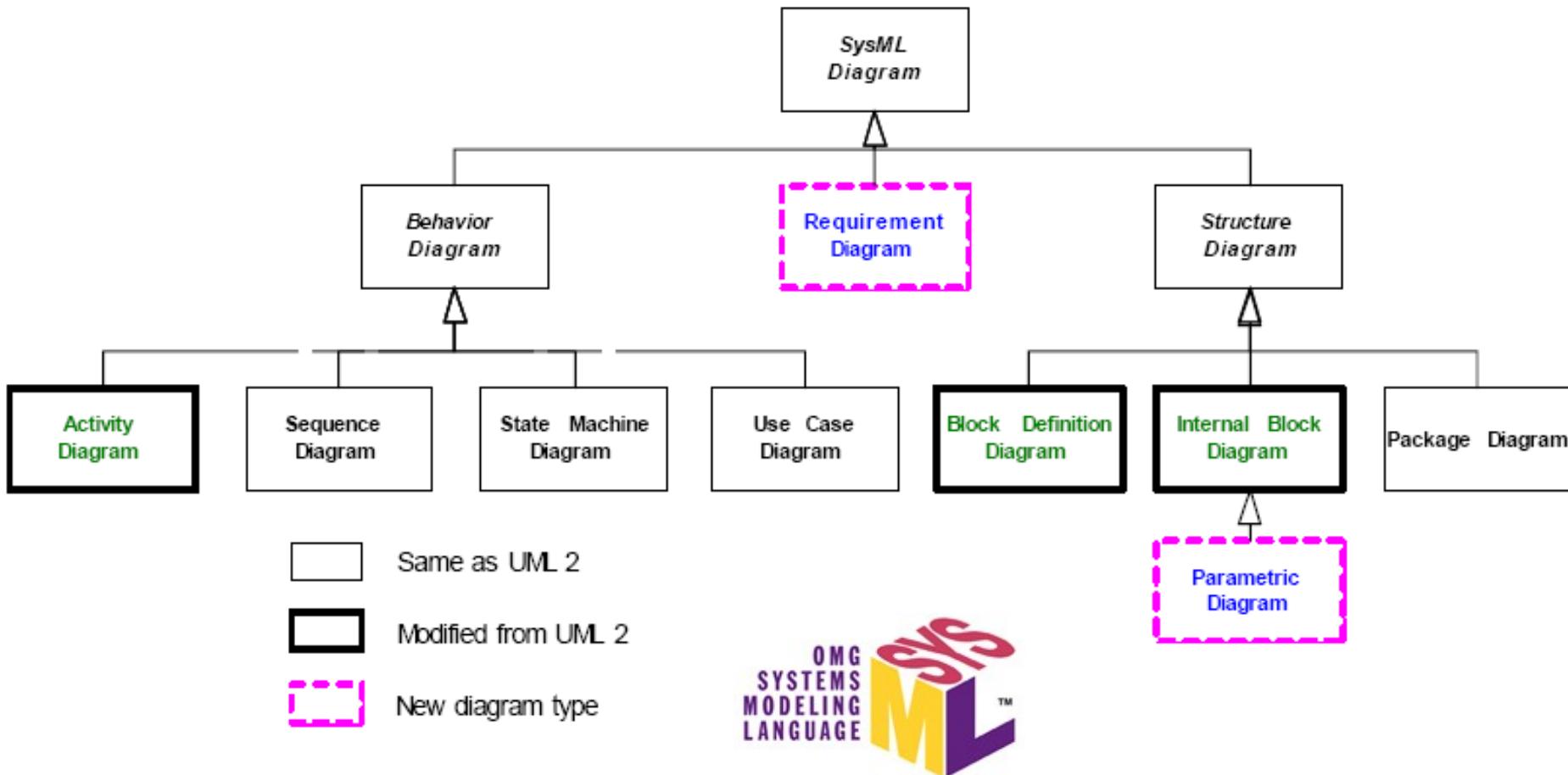
- Introduction
- Les diagrammes SysML
- Présentation de DARwIn-OP
- Avant de commencer...
- Diagramme de contexte
- Diagramme des cas d'utilisation
- Diagramme d'exigences
- Diagramme de séquence
- Diagramme de définition de bloc
- Diagramme de bloc interne
- Diagramme de machine d'état
- Diagramme d'activité
- Diagramme paramétrique



Les diagrammes UML (1)



Les diagrammes SysML (1)



Les diagrammes SysML (2)

- Chaque diagramme possède un cadre, avec une zone de contenu, un entête et une description du diagramme :

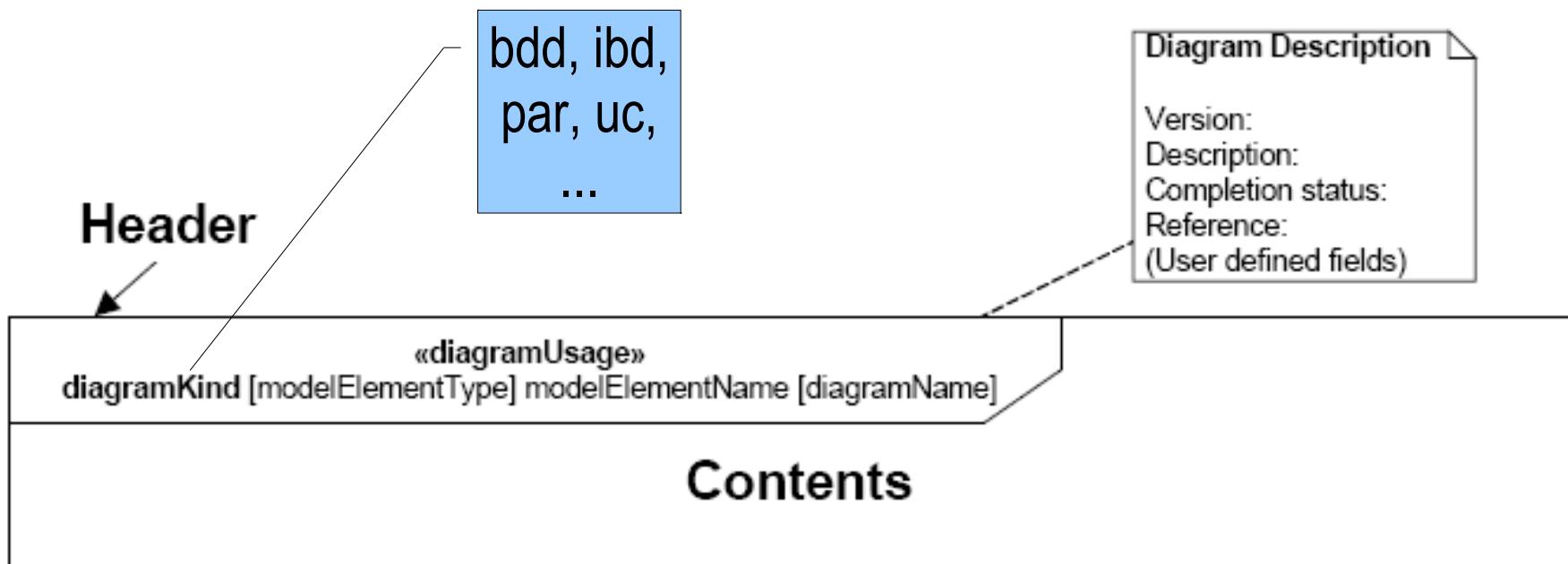


Figure A.2 - Diagram Frame

Système_MonSystème(MSYS)

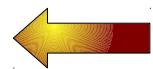
- MSYS_Contexte
- MSYS_Exigences
 - Fonctionnelles
 - Technologiques
 - Opératoires
- MSYS_Structure
 - MSYS_Types_Port
 - MSYS_Interface
 - Sous-Système_MonSous-Système(MSSYS)
 - MSSYS_Exigences
 - MSSYS_Structure
 - ...
- MSYS_Vue_des_cas_d_utilisation
 - Acteurs
 - Cas d'utilisation
- MSYS_Vue_comportementale

Arborescence de Paquets

Sommaire



- Introduction
- Les diagrammes SysML
- Présentation de DARwIn-OP
- Avant de commencer...
- Diagramme de contexte
- Diagramme des cas d'utilisation
- Diagramme d'exigences
- Diagramme de séquence
- Diagramme de définition de bloc
- Diagramme de bloc interne
- Diagramme de machine d'état
- Diagramme d'activité
- Diagramme paramétrique



Présentation de DARwIn-OP (1)

Dynamic Anthropomorphic Robot with Intelligence – Open Platform



Hauteur : 45,45 cm

Poids : 2,9 kg

Processeur principal Intel Atom Z530 à 1,6 GHz
avec 4GO mémoire flash SSD

Carte contrôleur CM-730 avec ARM Cortex M3 à
72MHz

Vitesse de marche par défaut : 24cm.s⁻¹, modifiable
par l'utilisateur

OS : Linux

Présentation de DARwIn-OP (2)



Démonstrations

- Mode Interactif
- Mode Football
- Mode Vision
- Marche
- RobotPlus

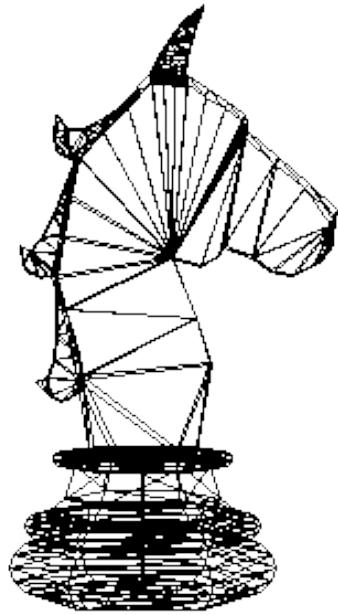
Sommaire



- Introduction
- Les diagrammes SysML
- Présentation de DARwIn-OP
- Avant de commencer...
- Diagramme de contexte
- Diagramme des cas d'utilisation
- Diagramme d'exigences
- Diagramme de séquence
- Diagramme de définition de bloc
- Diagramme de bloc interne
- Diagramme de machine d'état
- Diagramme d'activité
- Diagramme paramétrique



Notion de bloc et d'instance de bloc (1)

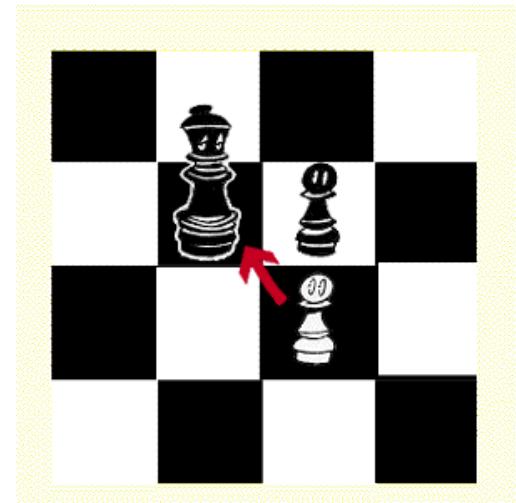
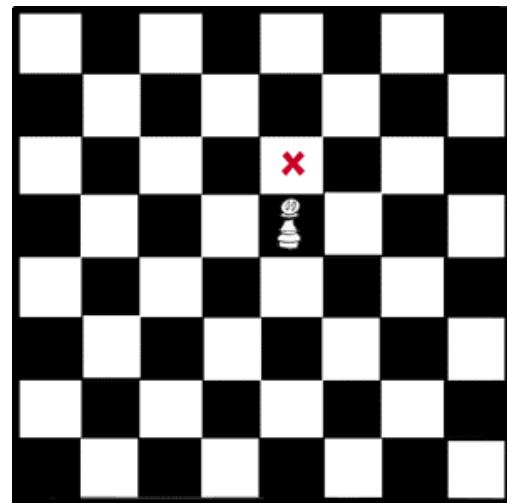
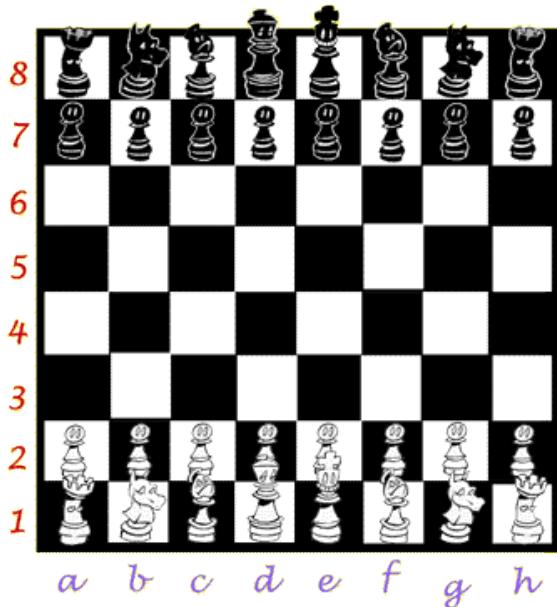


Bloc Cavalier



Instance de bloc (=« part »)
cavalier noir N°2
en position g8

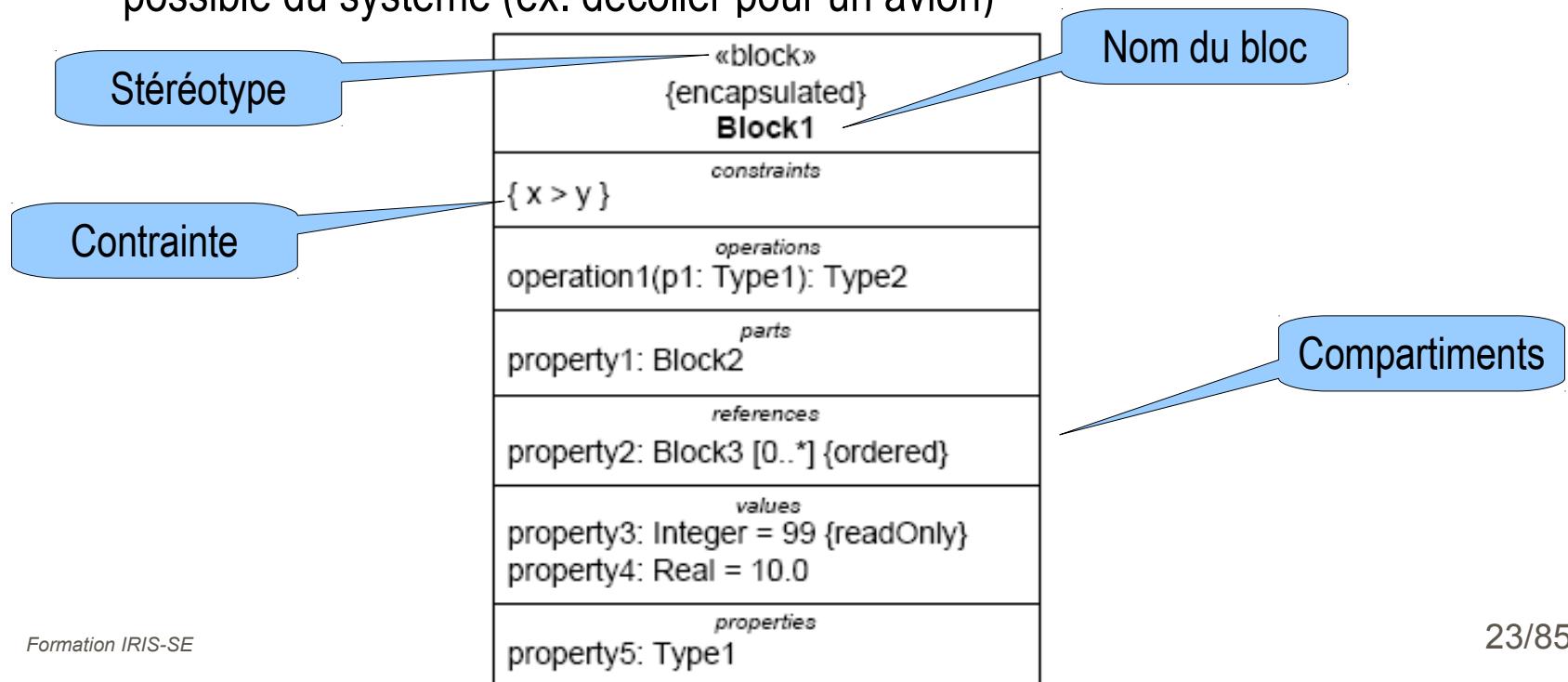
Notion de bloc et d'instance de bloc (2)



Quels états caractérisent les instances de bloc ?
Quelles opérations sont possibles sur ces instances ?

Notion de bloc et d'instance de bloc (3)

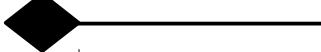
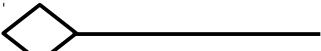
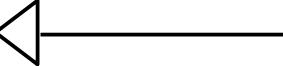
- Bloc : Unités **modulaires** de la description d'un système.
- Chaque bloc définit un ensemble de caractéristiques du système :
 - caractéristiques **structurelles** : les propriétés spécifiant l'état du système (ex. la pression pour un pneu)
 - caractéristiques **comportementales** : les opérations spécifiant le comportement possible du système (ex. décoller pour un avion)



Notion de bloc (2) : Caractéristiques principales

- Compartiment "**Operations**" : actions ou fonctions réalisées par le bloc.
 - ex : Marcher(), Shooter(), LocaliserBalle()
- Compartiment "**Parts**" : autre(s) instance(s) de bloc(s) composant le bloc (relation de composition)
 - ex : jambeGauche : ServoMoteur[5]
- Compartiment "**References**" : autre(s) instance(s) de bloc(s) ne faisant pas partie intégrante du système ou pouvant être partagé(s).
 - ex : bat : Batterie, interface entre deux blocs : ex. Ethernet
- Compartiment "**Values**" : caractéristiques spécifiant l'état du bloc
 - ex : couleur : string = « noir », enMouvement : boolean = false ;
- Compartiment "Constraints" {} : contrainte(s) pouvant être appliquée(s) au bloc.

Relations possibles entre les blocs :

- **Composition** : 
- **Agrégation** : 
- **Généralisation** : 
- **Association bidirectionnelle** : 
- **Association monodirectionnelle** : 
- **Multiplicité** : Aux extrémités de la relation peut apparaître des chiffres qui indiquent le nombre d'instances de bloc en jeu dans la relation (pour chaque sens utile de la relation)

Notion de Value Type

- Les types de valeurs (**Value Type**) sont utilisées pour « typer » les propriétés des blocs.

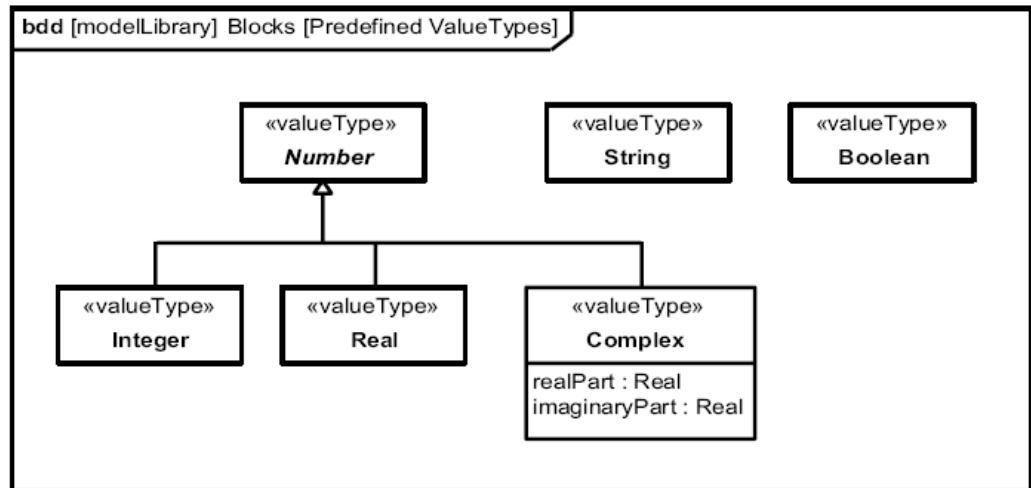


Figure 8.7 - Model Library for Blocks

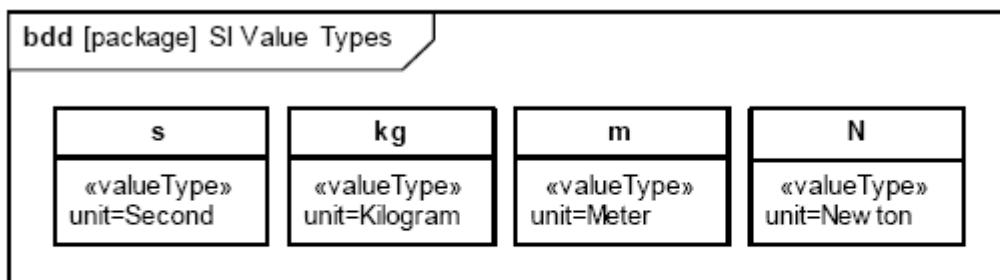


Figure 8.10 - Defining Value Types with units of measure from the International System of Units (SI)

Sommaire



- Introduction
- Les diagrammes SysML
- Présentation de DARwIn-OP
- Avant de commencer...
- Diagramme de contexte
- Diagramme des cas d'utilisation
- Diagramme d'exigences
- Diagramme de séquence
- Diagramme de définition de bloc
- Diagramme de bloc interne
- Diagramme de machine d'état
- Diagramme d'activité
- Diagramme paramétrique

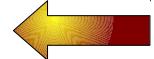


Diagramme de contexte (1) : BDD

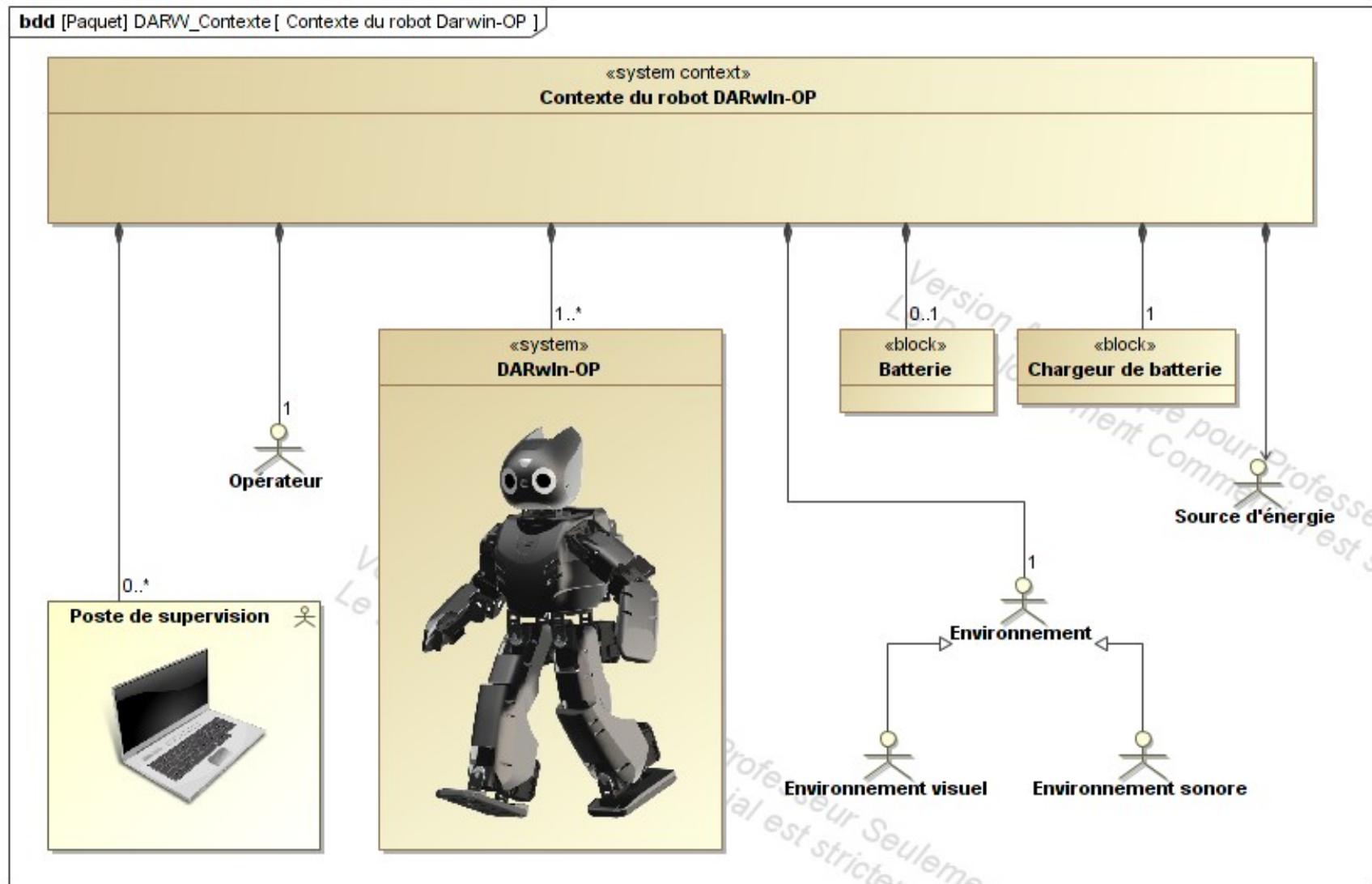
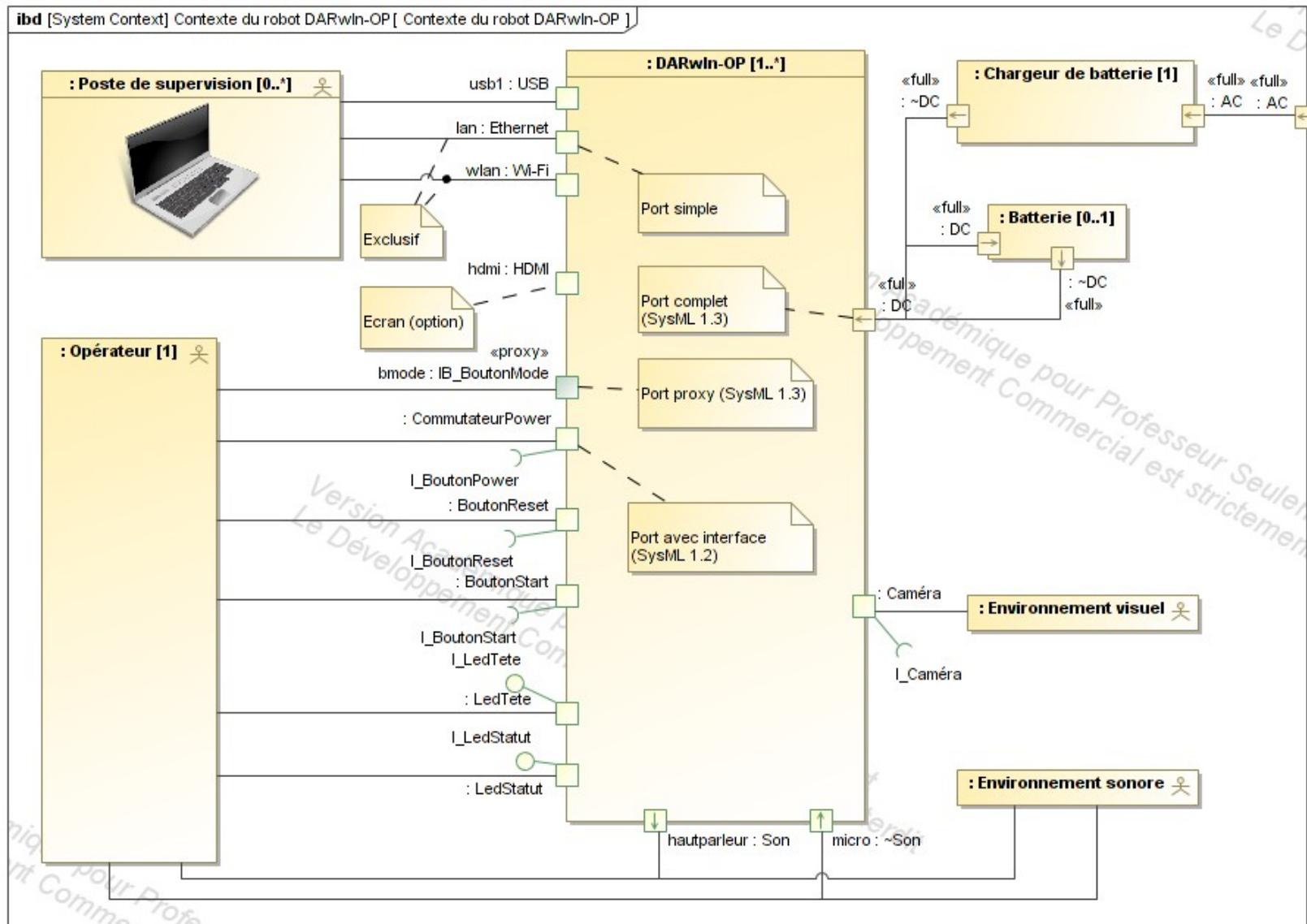


Diagramme de contexte (2)

- Le diagramme de contexte exprime **l'environnement** du système dans une situation donnée. Il définit l'ensemble des **acteurs** et **autres systèmes** qui échangent des **flux** (matière, énergie et information) avec le système étudié.
- Il peut être modélisé à l'aide d'un diagramme de définition de bloc (**bdd**) complété par un diagramme de bloc interne (**ibd**).
- On retrouve le système étudié (bloc avec stéréotype « system »), les acteurs en interaction (ex. l'opérateur) et les blocs utiles dans le contexte (ex. batterie).
- On peut ajouter des éléments externes (stéréotype « **external** ») qui ne sont pas des acteurs (pas d'intervention dans les cas d'utilisation) mais qui peuvent préciser le contexte de l'utilisation du système (ex. domicile de la personne utilisant le robot, etc.)

Diagramme de contexte (3) : IBD



Notion de port

- Port : point d'interaction entre un bloc et son environnement.
L'interface associée à ce port spécifie les interactions possibles à travers ce port :
 - **provided interface** : requêtes que l'environnement peut faire sur le classificateur.
Les requêtes arrivant sur ce port sont prisent en charge par un comportement du bloc pour atteindre les propriétés de l'instance (ex: appel de méthode d'une instance de bloc)
 - **required interface** : requêtes possibles du bloc vers son environnement
 - **flow property** : ce qui peux circuler entre deux ports dans le cas général (ex : fluide)
 - **item flow** : ce qui circule entre deux ports dans un cas particulier (ex: eau, essence, etc.)

Sommaire



- Introduction
- Les diagrammes SysML
- Présentation de DARwIn-OP
- Avant de commencer...
- Diagramme de contexte
- Diagramme des cas d'utilisation
- Diagramme d'exigences
- Diagramme de séquence
- Diagramme de définition de bloc
 - Diagramme de bloc interne
- Diagramme de machine d'état
- Diagramme d'activité
- Diagramme paramétrique

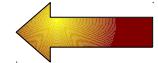


Diagramme des cas d'utilisation : cas des systèmes simples

- Sur des systèmes simples, il est souvent possible d'établir directement le diagramme des cas d'utilisation.

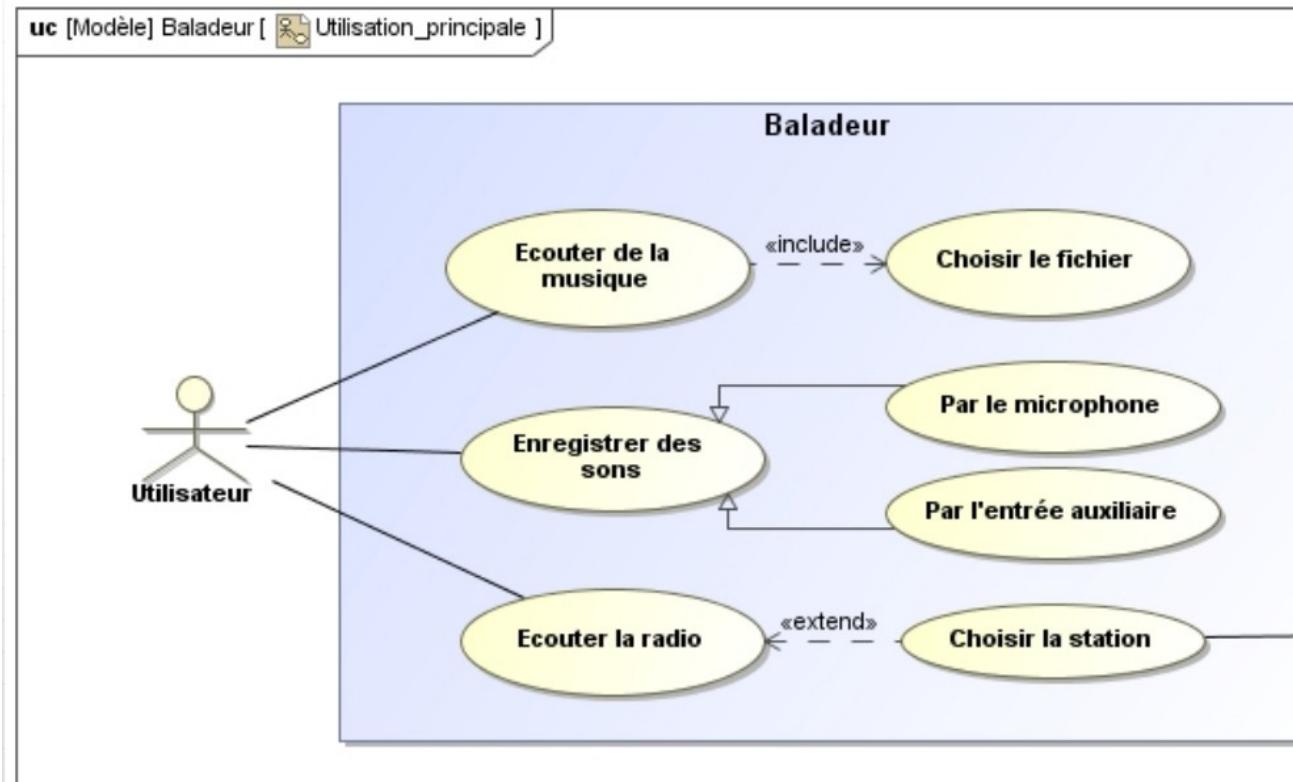


Diagramme des cas d'utilisation : cas de DARwIn

uc [Paquet] Cas d'utilisation DARwIn-OP [Simuler un être humain détaillé]

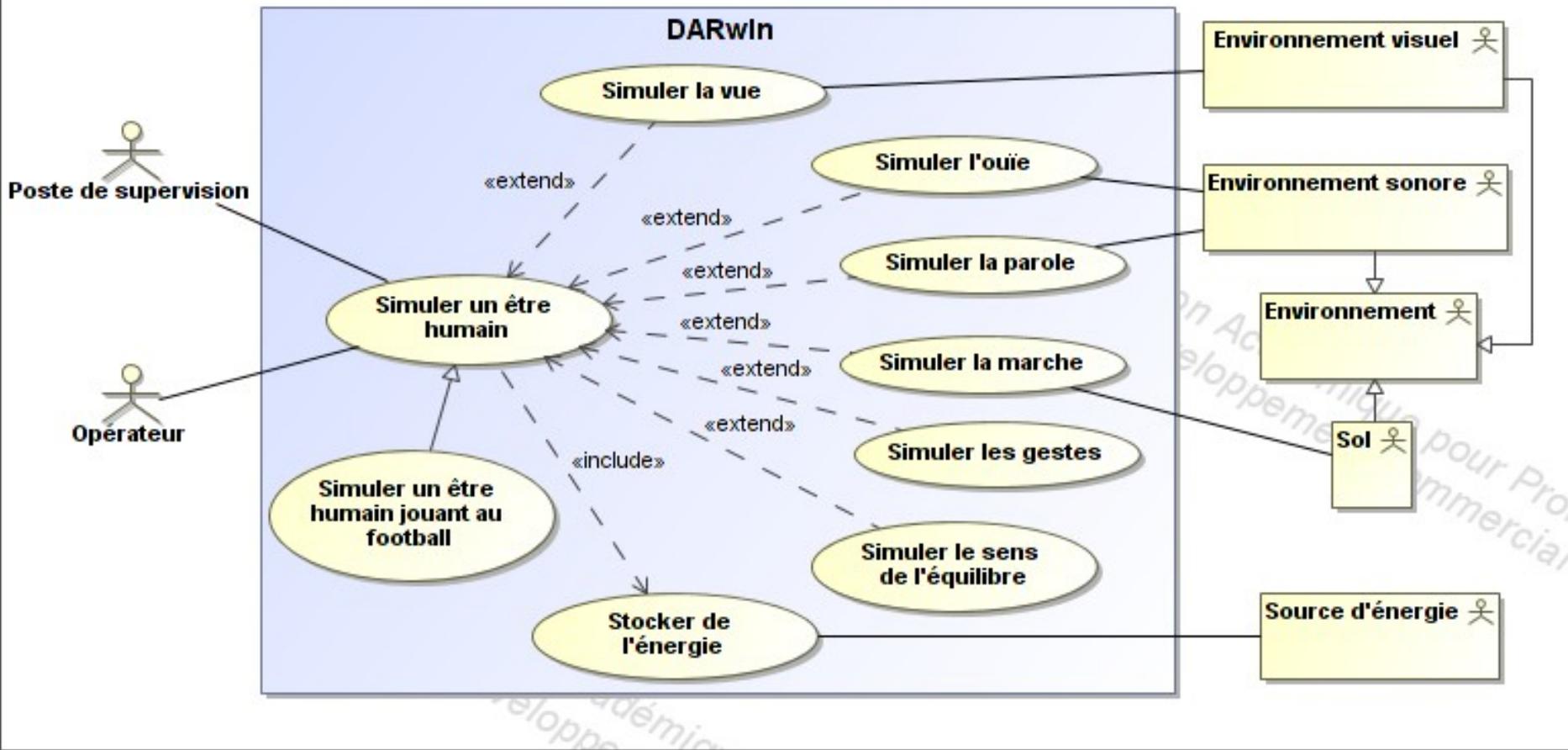


Diagramme de mode d'utilisation : cas des systèmes complexes

- Pour des systèmes complexes, il est souhaitable de définir les principaux modes d'utilisation du système.
- Quelques exemples :
 - Mode de test à la mise sous tension (PowerOnSelfTest)
 - Mode Nominal
 - Mode Dégradé
 - Mode Défaillance
 - Mode Maintenance
 - etc...
- On peut utiliser un diagramme de machine d'état pour représenter les conditions de passage entre ces différents modes.
- On définira dans un deuxième temps des diagrammes de cas d'utilisation par mode d'utilisation

Diagramme de mode d'utilisation : exemple d'un système complexe

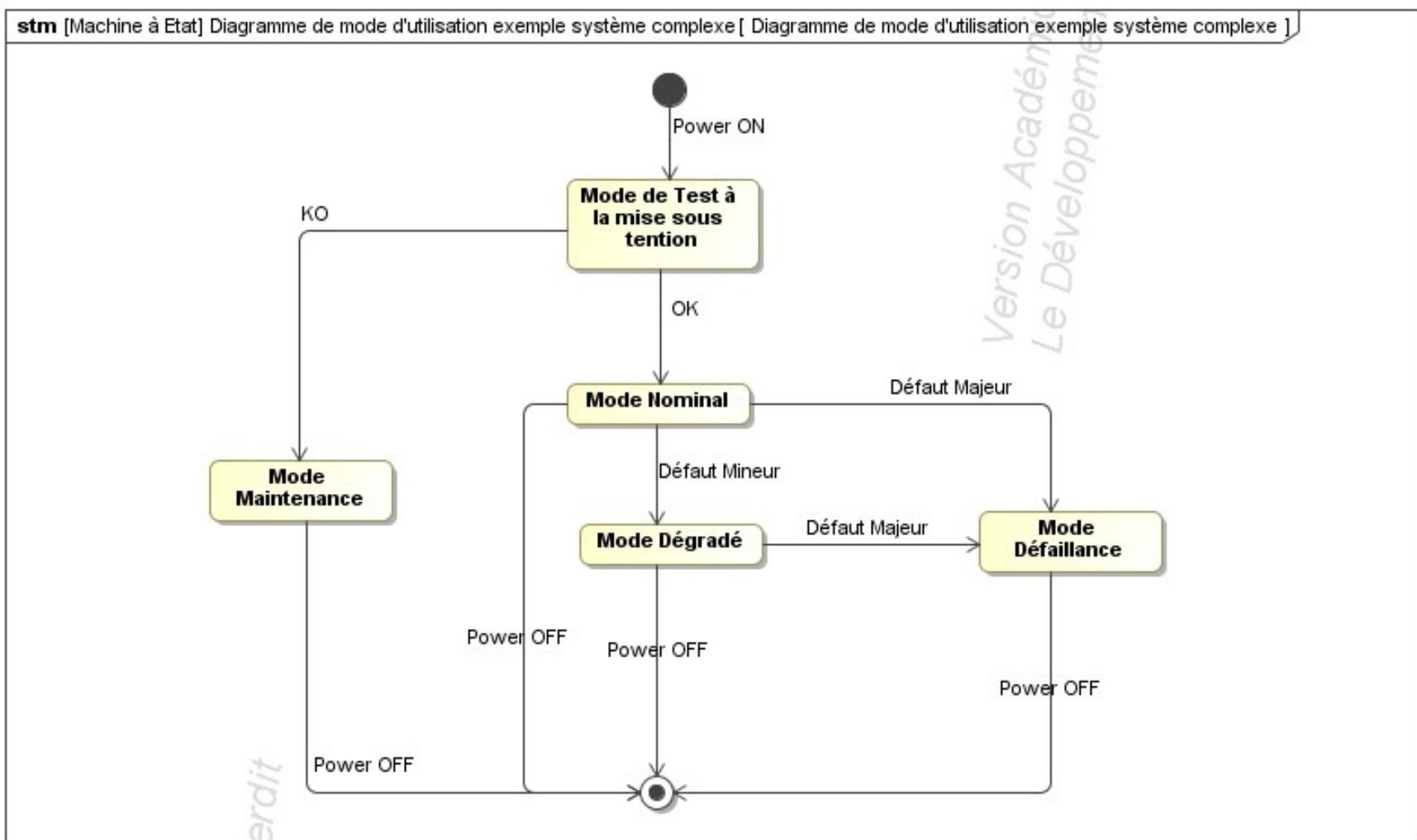


Diagramme de mode d'utilisation : exemple de DARwIn

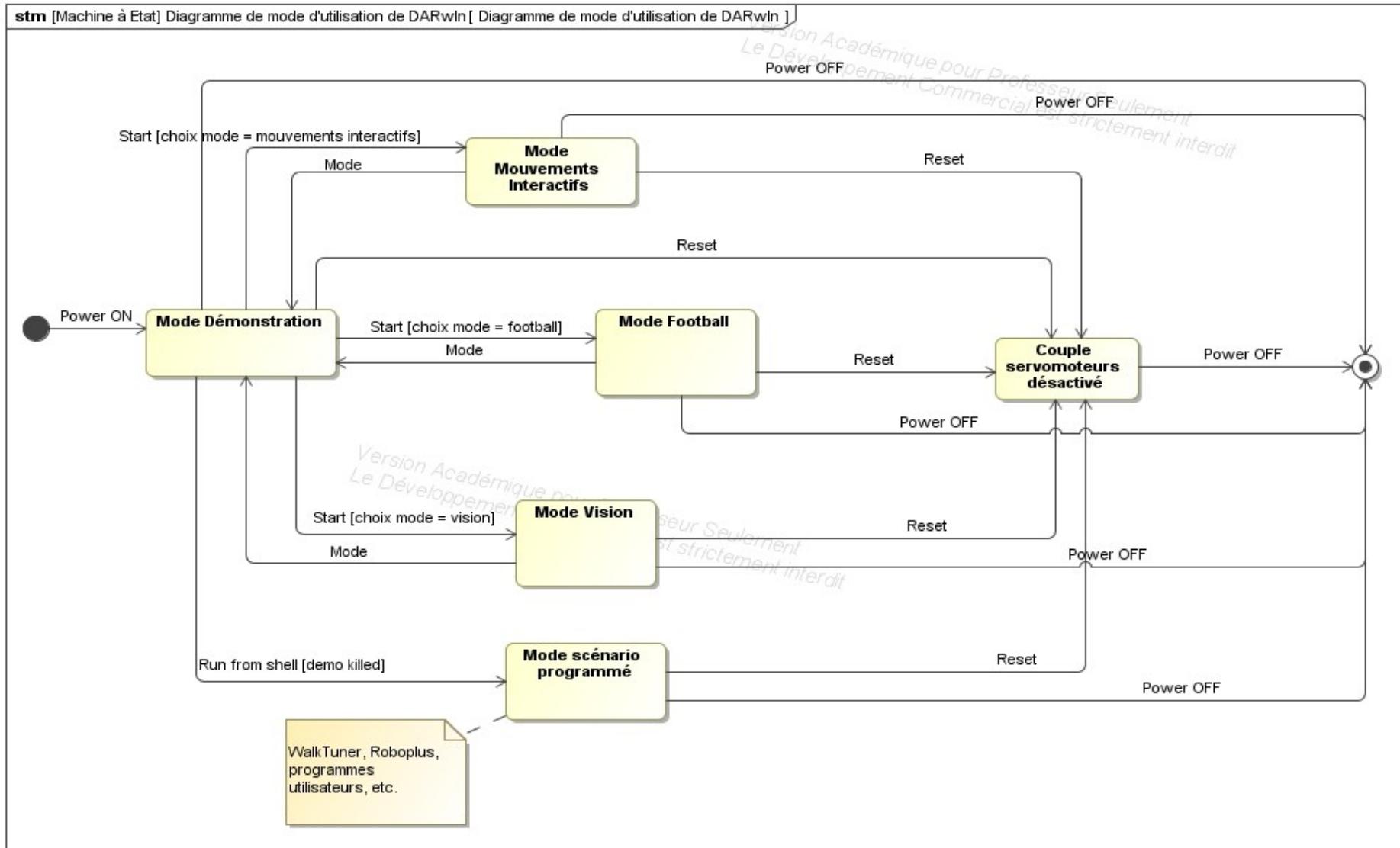


Diagramme des cas d'utilisation : Mode Football

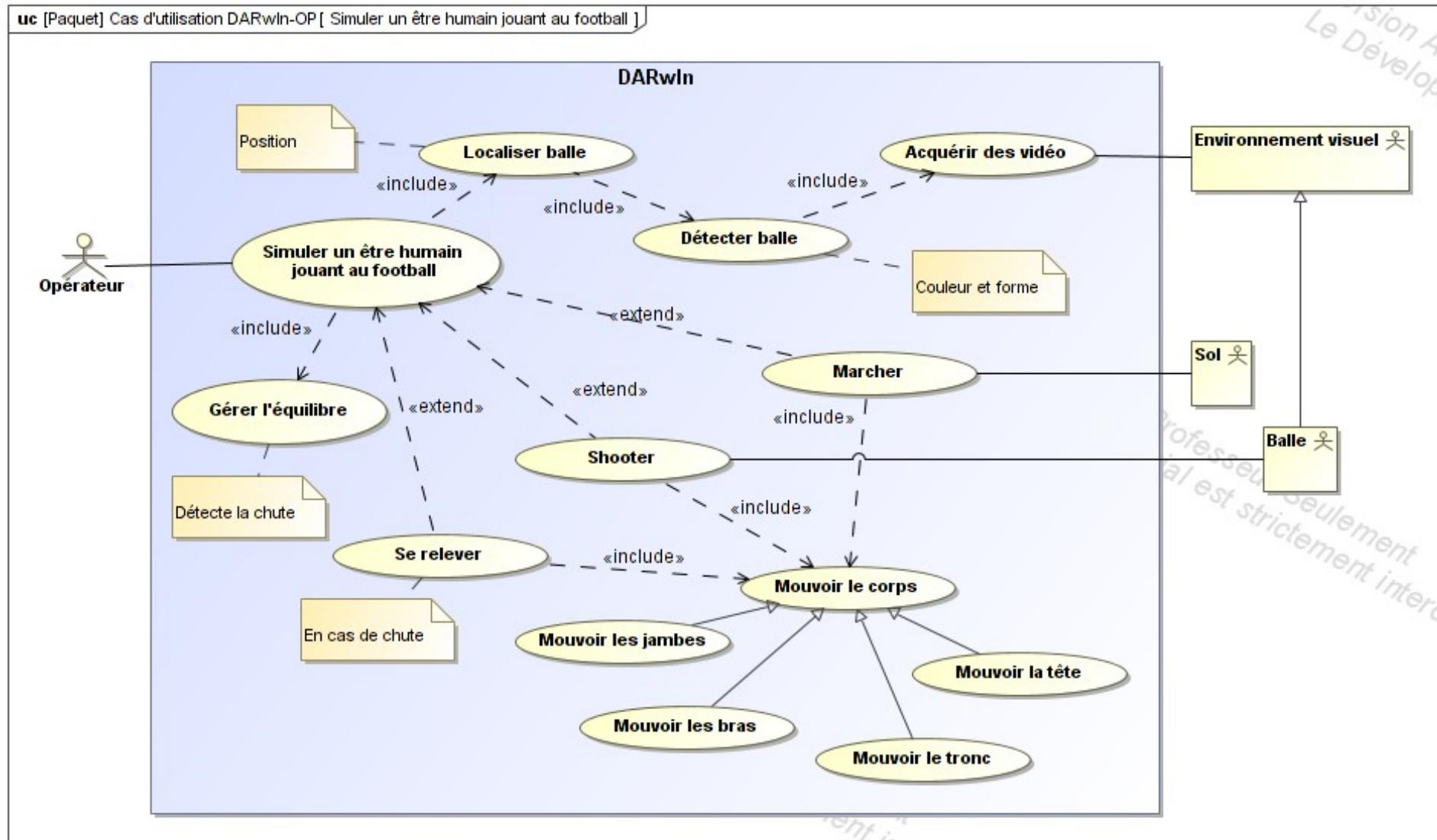
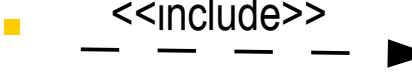
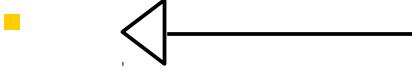


Diagramme des cas d'utilisation (1)

- Il permet de représenter les **fonctionnalités** du système et les acteurs avec lesquels le système interagit.
- Les cas d'utilisation sont des fonctions d'usage offertes par le système aux acteurs afin de **satisfaire un besoin**.
- L'ensemble des cas d'utilisations forme toutes les façons que le système pourra être utilisé.
- Un cas d'utilisation est un ensemble de séquences d'actions effectuées par le système qui mène à un résultat tangible pour un acteur.
- Il est possible de détailler un cas d'utilisation par une décomposition en plusieurs cas d'utilisation (deux niveaux maximum).

Diagramme des cas d'utilisation (2)

Les relations :

-  : relation d'association directe entre acteurs et cas d'utilisation (non dirigée)
-  ► : relation d'incorporation obligatoire (flèche dirigée vers le cas d'utilisation inclus)
-  ► : relation d'incorporation facultative (flèche dirigée vers le cas d'utilisation principal)
-  : relation de généralisation (spécialisation), apportant des précisions sur un cas d'utilisation (la flèche est dirigée vers le cas d'utilisation)

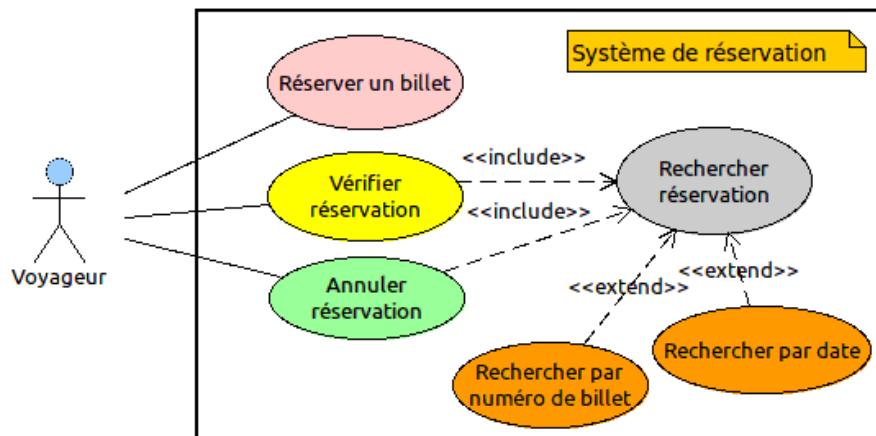


Diagramme des cas d'utilisation (3)

- Un cas d'utilisation regroupe une **famille de scénarios** d'utilisation.
- Un **scénario** vise à mettre en évidence le cas particulier d'une utilisation du système, répondant à un besoin du système.
- Un scénario est une « **instance** » d'un cas d'utilisation
- Pour chaque cas d'utilisation, on veillera à décrire plusieurs scénarios où l'on trouvera au minimum :
 - Le point de départ du scénario (déclencheur)
 - La description textuelle ou sous la forme de diagramme de séquence des actions du scénario nominal ou d'un des scénarios alternatifs
 - L'état du système à la fin du scénario.
- Le scénario peut décrire une procédure normale d'utilisation, mais également des procédures d'utilisation alternatives (se terminant de façon normale) ou d'erreur (aboutissant à un échec ou à une erreur)

Sommaire



- Introduction
- Les diagrammes SysML
- Présentation de DARwIn-OP
- Avant de commencer...
- Diagramme de contexte
- Diagramme des cas d'utilisation
- Diagramme d'exigences
- Diagramme de séquence
- Diagramme de définition de bloc
- Diagramme de bloc interne
- Diagramme de machine d'état
- Diagramme d'activité
- Diagramme paramétrique

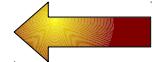


Diagramme d'exigences (1)

req [Paquet] DARW_Exigences [Exigences]

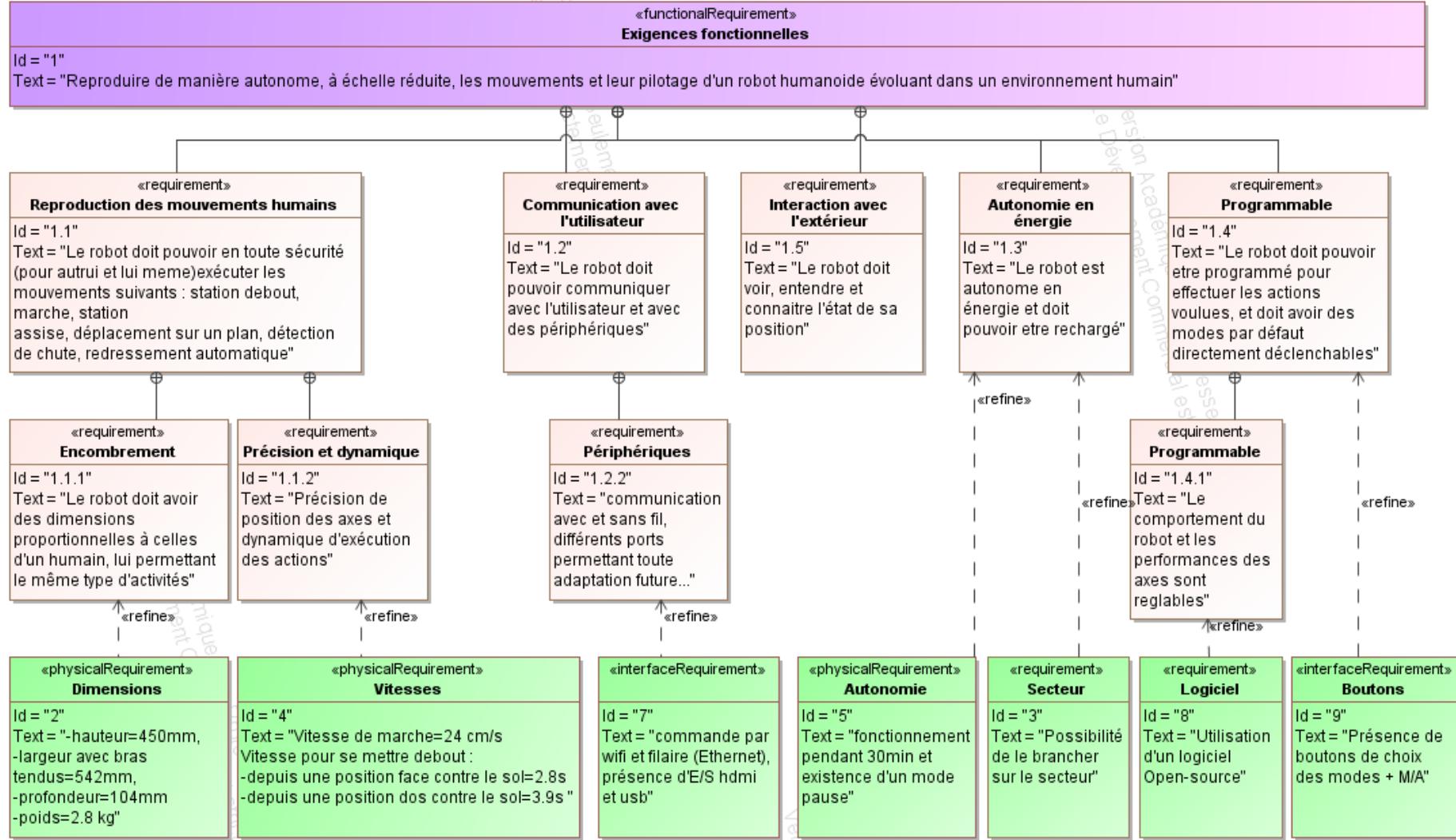


Diagramme d'exigences (2)

- Il permet de représenter graphiquement les exigences fonctionnelles et de performances du système ainsi que les contraintes qui doivent être respectées par le système (normes, recyclage, matériels imposés, etc.)
- C'est une traduction du **cahier des charges**
- Pour répondre au cahier des charges, toutes les exigences doivent être satisfaites
- On peut classer les exigences en plusieurs catégories :
 - Exigences **fonctionnelles** (affine les cas d'utilisation)
 - Contraintes **technologiques** (sécurité, norme, etc.)
 - Contraintes **opératoires** (autonomie, conditions de fonctionnement, performance, encombrement, etc.)
 - Exigences commerciales, etc.

Diagramme d'exigences (3)

- SysML définit une représentation graphique et visuelle des exigences textuelles, permet une organisation hiérarchique et l'association avec les éléments du modèle.
- SysML définit de nouveaux types de d'associations (liens de dépendance stéréotypés) :
 - **Derive** : une ou plusieurs exigences sont dérivées d'une exigence
 - **Satisfy** : un ou plusieurs éléments du modèle (par exemple un bloc) permettent de satisfaire une exigence
 - **Verify** : un ou plusieurs éléments du modèle (par exemple un « test case ») permettent de vérifier et valider une exigence
 - **Refine** : un ou plusieurs éléments du modèle, par exemple un cas d'utilisation, redéfinit une exigence

Diagramme d'exigences (4)

- SysML définit de nouveaux commentaires stéréotypés permettant d'associer une explication à des associations ou éléments du modèle :
 - Problem** : commentaire dont la description pose le problème ou le besoin qui a donné lieu à la création de l'association ou de l'élément associé
 - Rationale**: commentaire dont la description indique la raison ou la justification par rapport à l'élément ou l'association associé

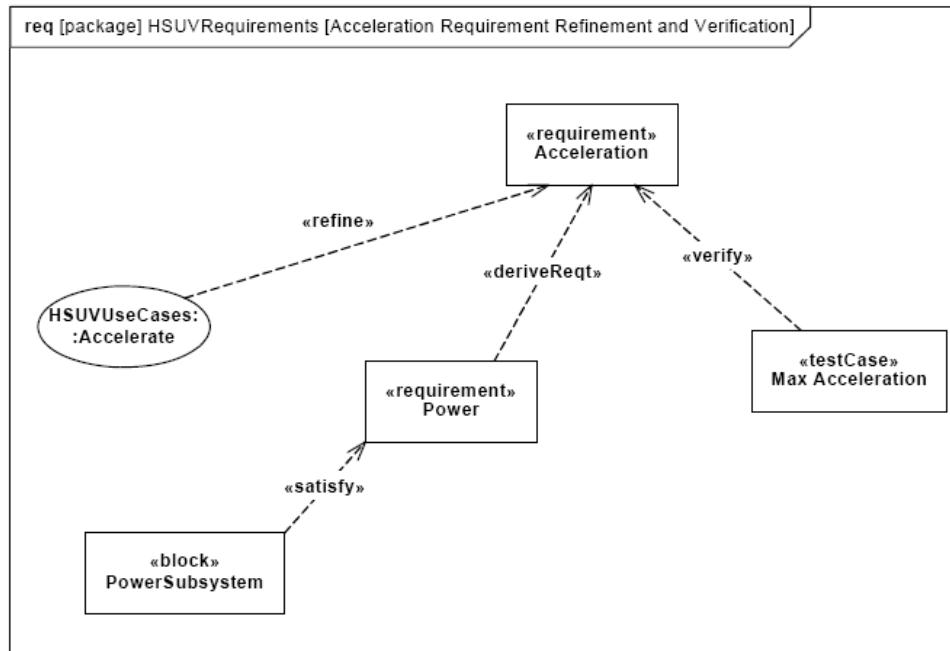


Figure B.13 - Acceleration Requirement Relationships (Requirements Diagram)

Sommaire



- Introduction
- Les diagrammes SysML
- Présentation de DARwIn-OP
- Avant de commencer...
- Diagramme de contexte
- Diagramme des cas d'utilisation
- Diagramme d'exigences
- Diagramme de séquence
- Diagramme de définition de bloc
- Diagramme de bloc interne
- Diagramme de machine d'état
- Diagramme d'activité
- Diagramme paramétrique



Diagramme de séquence (1)

- Extrait de la documentation de DARwIn-OP :

Status	Robot Action
 The eye LED will turn on	Power is on
 The head LED will turn on green	The PC inside DARwIn-OP operating system is booting

Diagramme de séquence (2)

	Demo program is loaded and DARwIn-OP is ready to operate
	Demo program is loaded and DARwIn-OP is ready to operate

Diagramme de séquence (3) : Vue « Système » ou « Boîte noire »

- Il permet de décrire le scénario d'un cas d'utilisation.

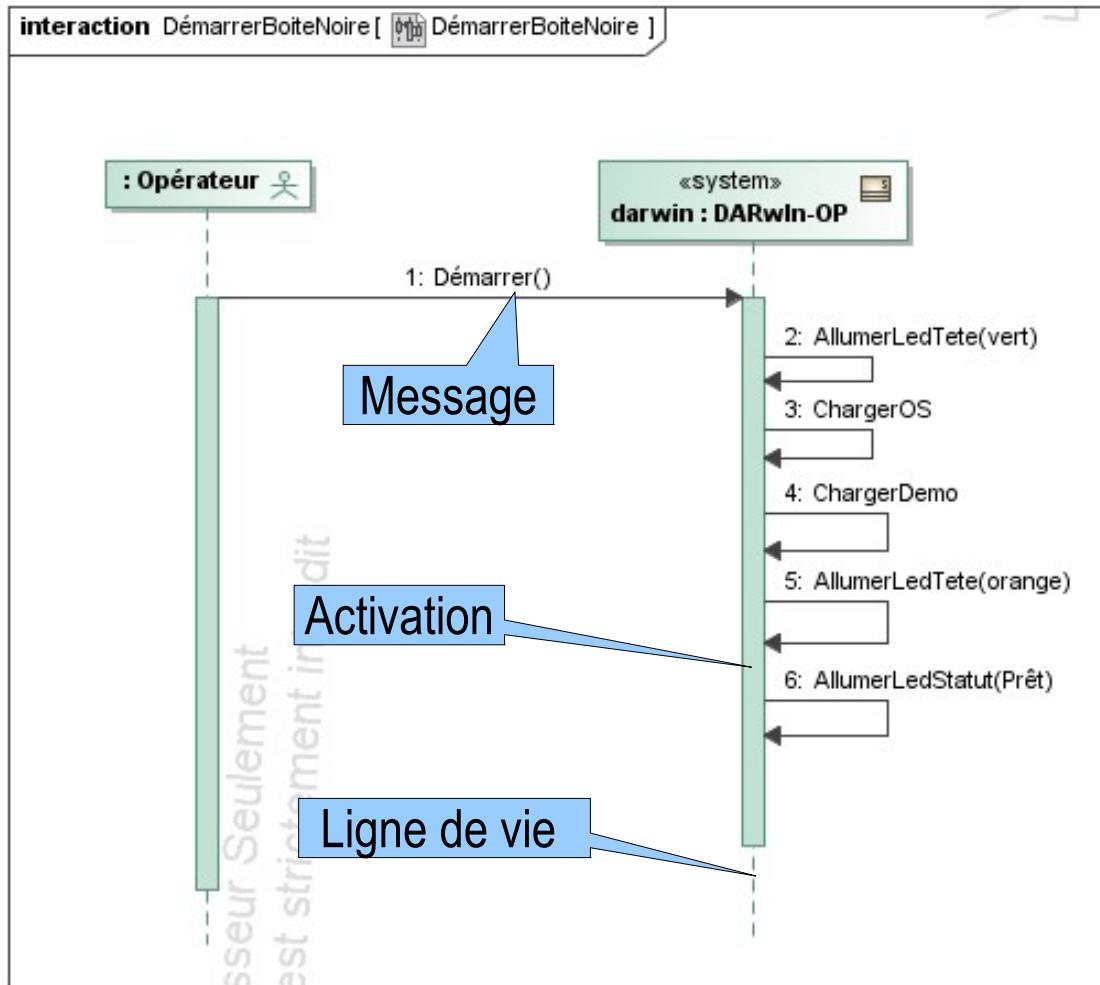


Diagramme de séquence (4) : Vue « Boite noire » avec interface

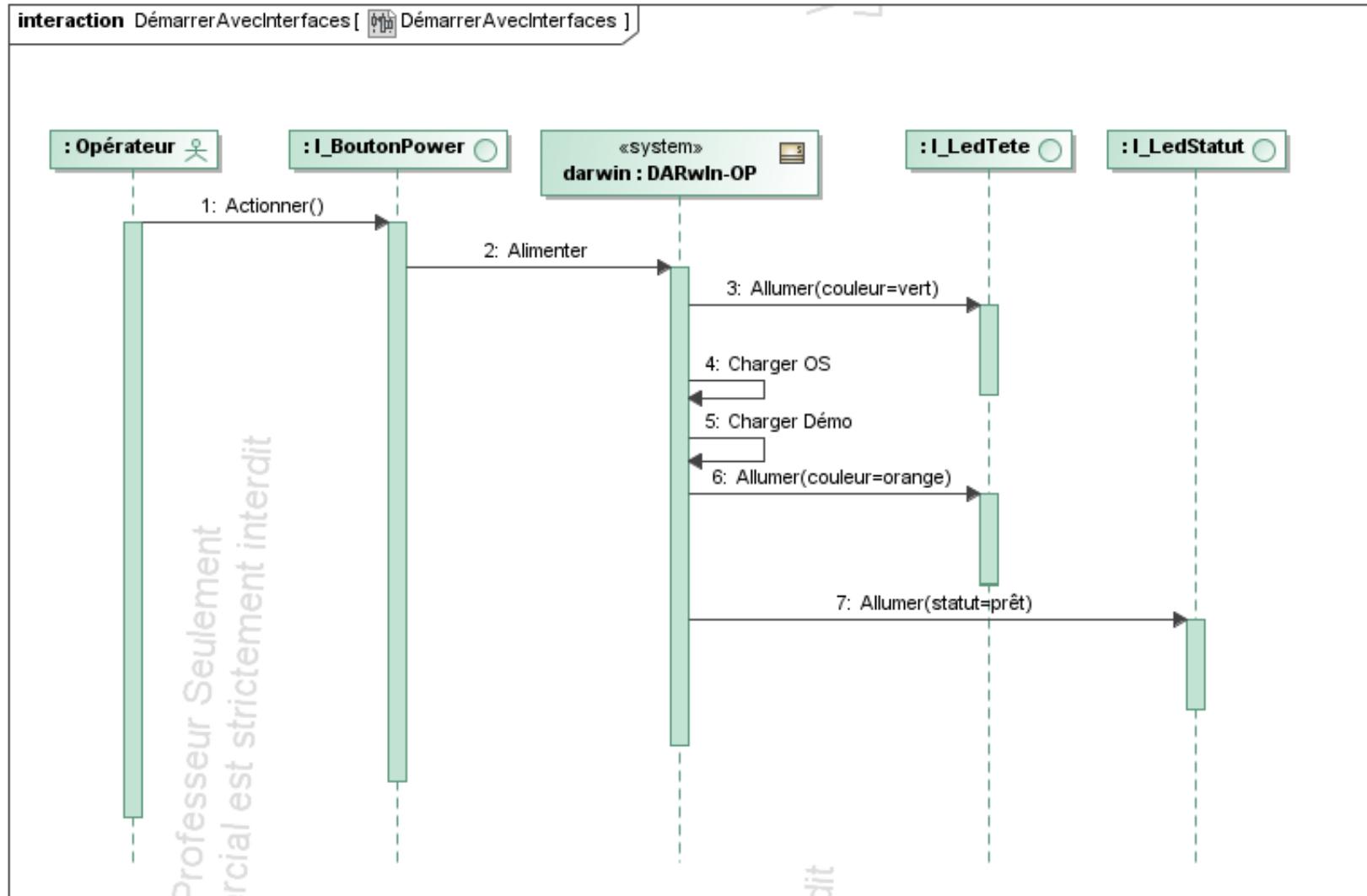


Diagramme de séquence (5)

- Un diagramme de séquence décrit chronologiquement un scénario d'un cas d'utilisation afin de présenter temporellement le fonctionnement du système, en montrant les échanges d'information entre les **instances de bloc** et les acteurs.
- Éléments utilisés :
 - **Ligne de vie** : Ligne verticale pointillée représentant l'existence d'un élément (acteur ou système)
 - **Message** : Flèche (pleine ou pointillée) représentant un échange unidirectionnel d'information
 - **Activation** : Rectangle (centré sur la ligne de vie) montrant les périodes d'activation d'un élément
 - **Fragment** : Interactivité liée à des conditions spécifiques (répétition, simultanéité, conditions à vérifier...)
 - Note : Commentaire pour documenter ou faciliter la compréhension du fonctionnement (facultatif)

Diagramme de séquence (5) : Les fragments

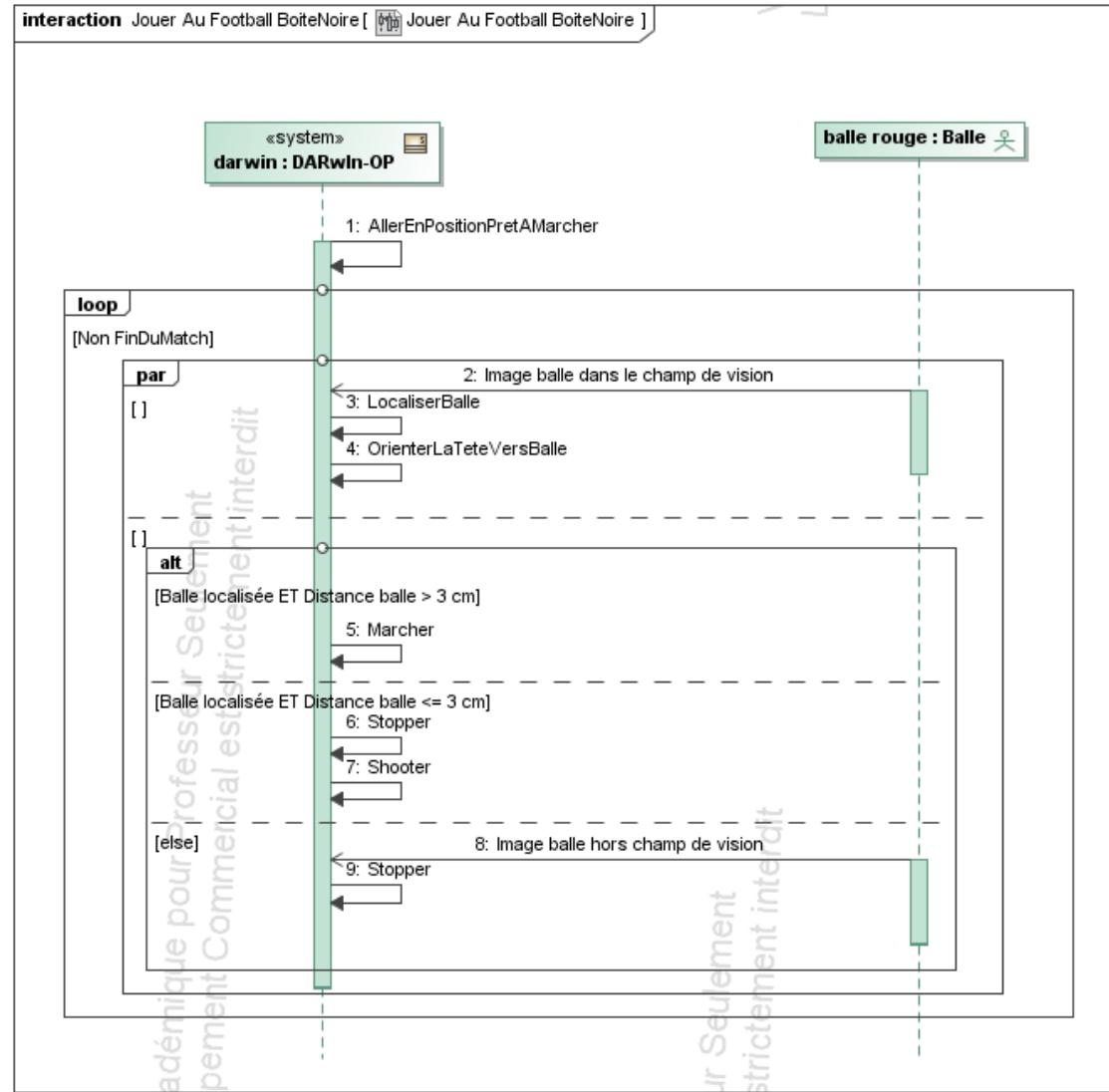


Diagramme de séquence (6) : Les fragments

- Fragments courants (opérateurs d'interactivité tracés dans des cadres labélisés) :
 - **Ref nom** : référence à un autre diagramme de séquence (lorsqu'on ne veut pas détailler localement)
 - **Opt [condition]** : fragment optionnel dépendant d'une unique condition identifiée
 - **Alt [conditions]** : fragments alternatifs dépendants d'au moins deux conditions (l'action "else" est exécutée par défaut)
 - **Par [actions]** : actions parallèles (multiples) pouvant avoir lieu ensemble et sans ordre particulier
 - **Loop min...max [escape]** : action répétée d'un nombre mini à un nombre maxi de fois, possédant une sortie possible sur une condition identifiée
 - **Break [condition]** : abandon d'action (passage à la suite chronologique) sur une condition identifiée

Diagramme de séquence (7) : Les messages

- Nature des messages (chacun est accompagné d'un commentaire numéroté) :
 - → **Synchrone** : l'émetteur est bloqué (attend un retour), pendant que le destinataire est actif
 - → **Asynchrone** : l'émetteur n'attend pas de retour et son activité n'est pas perturbée
 - < -- - **Retour** : réponse à un message synchrone
 -  **Réflexif** : interne à un élément marquant un événement particulier

Sommaire



- Introduction
- Les diagrammes SysML
- Présentation de DARwIn-OP
- Avant de commencer...
- Diagramme de contexte
- Diagramme des cas d'utilisation
- Diagramme d'exigences
- Diagramme de séquence
- Diagramme de définition de bloc
- Diagramme de bloc interne
- Diagramme de machine d'état
- Diagramme d'activité
- Diagramme paramétrique

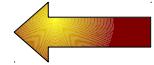


Diagramme de définition de bloc (1)

- Extrait de la documentation de la structure interne de DARwIn :

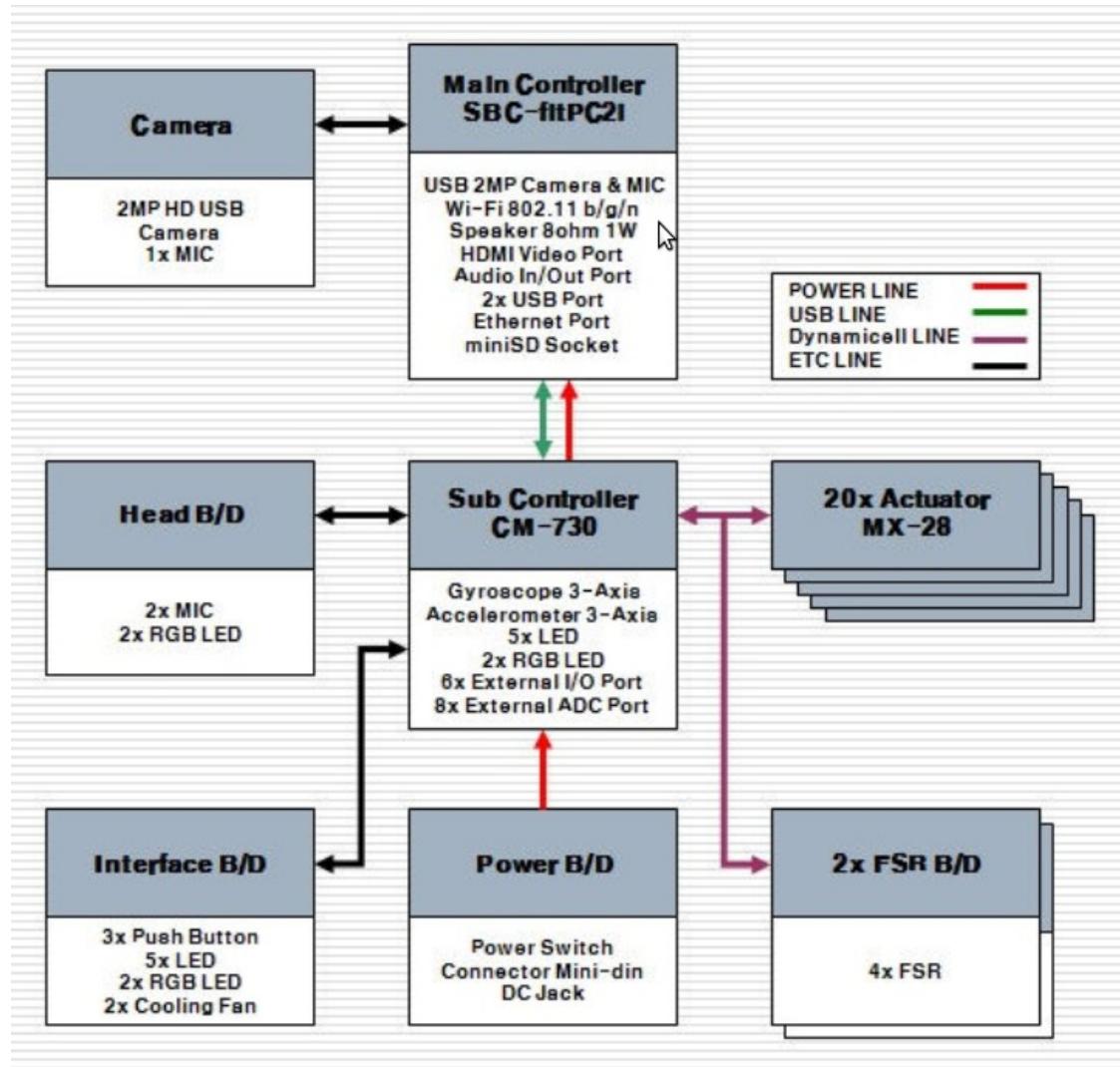


Diagramme de définition de bloc (2)

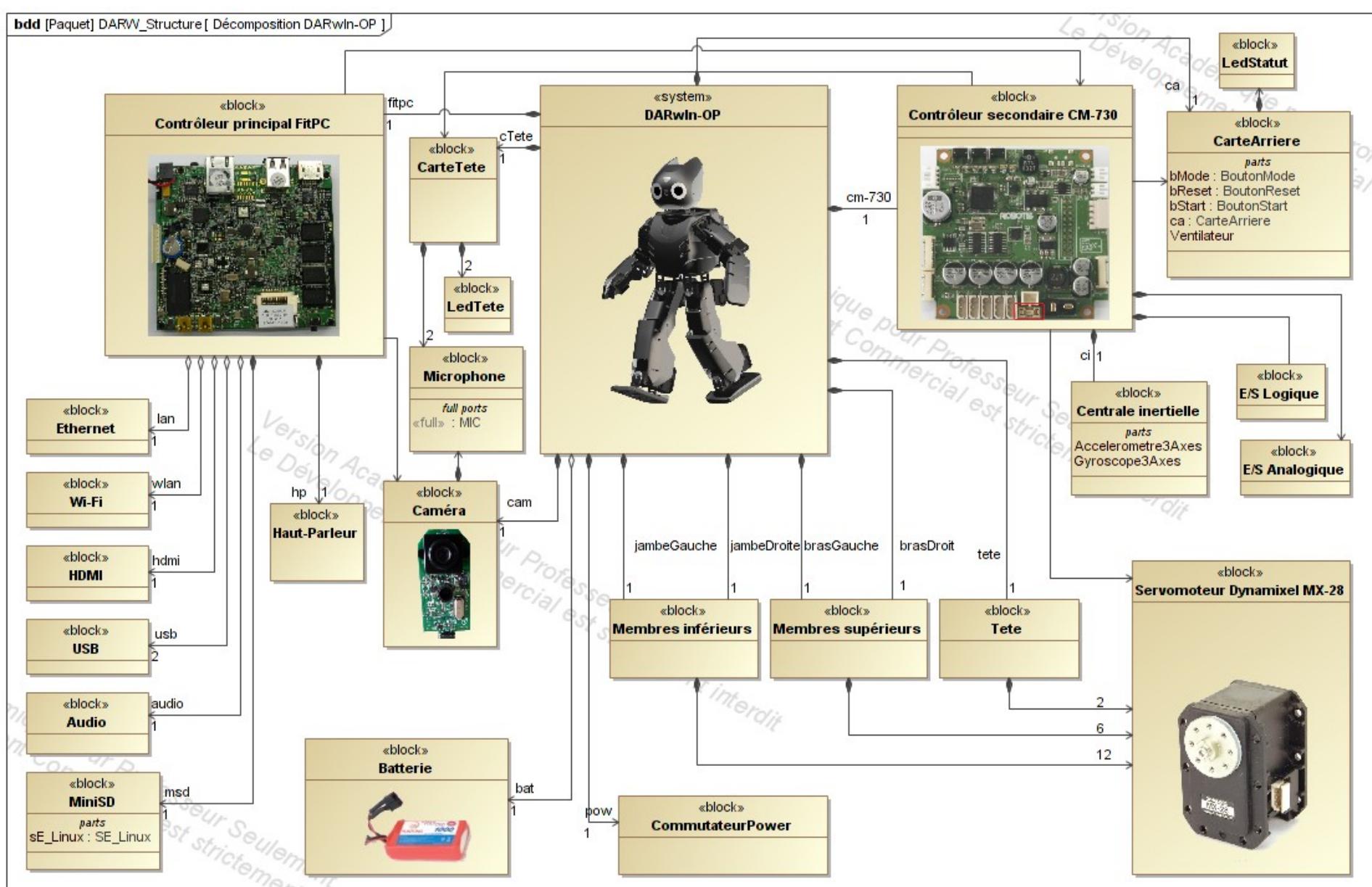


Diagramme de définition de bloc (3)

- Les blocs sont utilisés pour spécifier les **hiérarchies** et les **interconnexions** au sein du système.
- Le BDD traduit la **structure arborescente** du système composé de **sous-systèmes**.
- Les blocs représentent des types (modèles) de composants matériels ou logiciels qui constituent le système.
- L'idée est de rendre les blocs **réutilisables** dans d'autres contextes (notion d'héritage).
- La **multiplicité** permet de préciser le nombre d'instances de bloc (« parts ») présentes dans le système.

Sommaire



- Introduction
- Les diagrammes SysML
- Présentation de DARwIn-OP
- Avant de commencer...
- Diagramme de contexte
- Diagramme des cas d'utilisation
- Diagramme d'exigences
- Diagramme de séquence
- Diagramme de définition de bloc
- Diagramme de bloc interne
- Diagramme de machine d'état
- Diagramme d'activité
- Diagramme paramétrique



Diagramme de bloc interne (1) : DARwIn

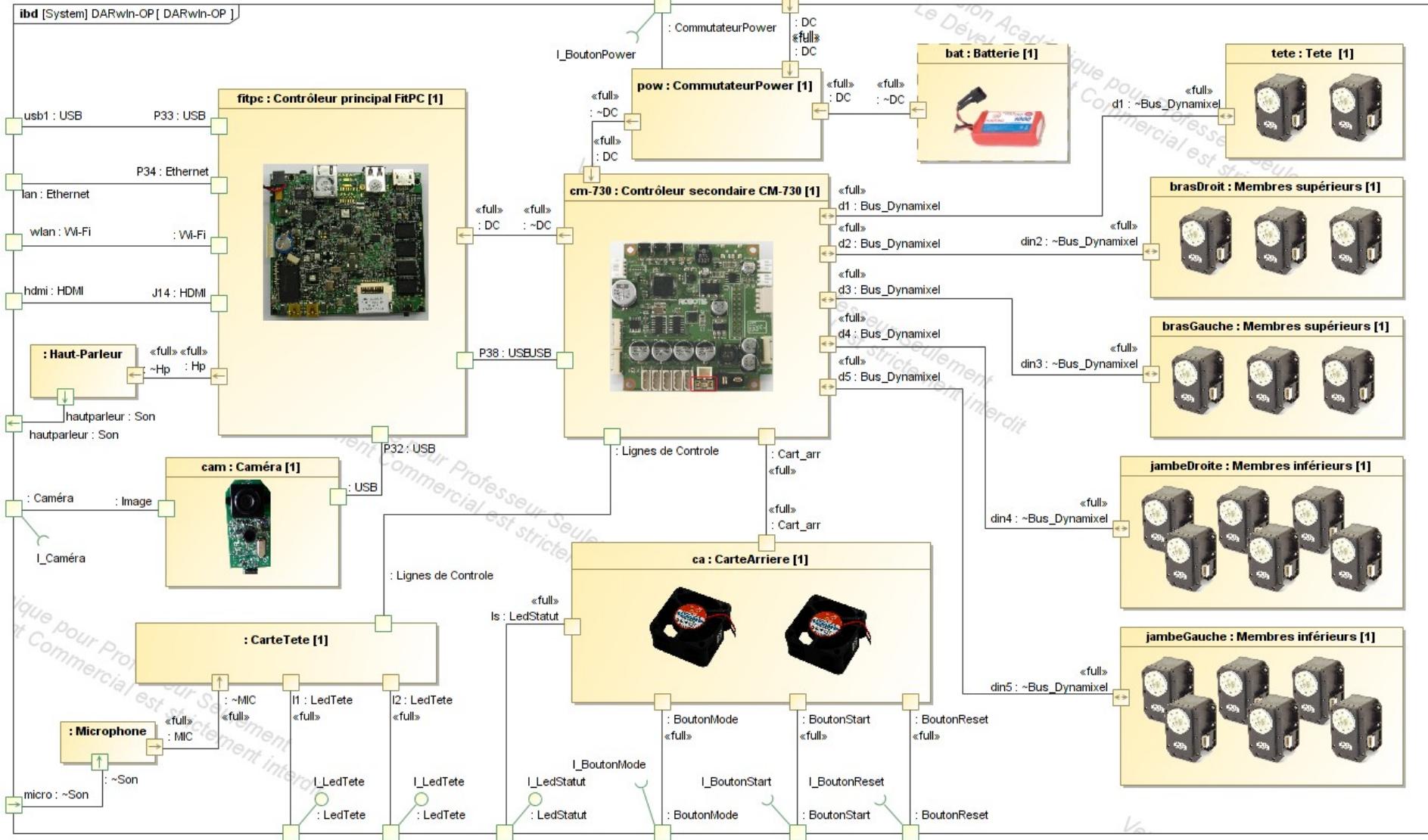


Diagramme de bloc interne (2)

- Décrit l'organisation **interne** d'un bloc (ou d'un ensemble de blocs) en montrant de façon détaillée les liens entre leurs constituants (liens internes) et avec les autres blocs (liens externes) ainsi que la nature de leur échange (Information, Matière, Énergie) .
- Représentation graphique : Un rectangle par élément
 - nom l'instance : nom du bloc (type) [multiplicité]
 - instance issue d'une composition, le rectangle est en trait continu
 - instance issue d'une agrégation, les rectangles sont en traits pointillés

- La modélisation des **interfaces** entre les blocs est un des **points critiques** de la modélisation SysML.
 - ex. A380 : 530 km de câble, 100 000 liaisons et 40 300 connecteurs...
- Différents types d'interface :
 - Électrique,
 - Mécanique,
 - Logicielle (Information),
 - Homme-Machine, etc.
- Une interface spécifiant un flux d'information doit prendre en compte l'aspect **logique** (contenu) et l'aspect **physique** (signaux électriques, bit, octets, etc.) de cette information.
- Le **port** permet de modéliser les interfaces
- Le port : point d'interaction avec le bloc

Diagramme de bloc interne (4)

Ports complets (full ports) : SysML 1.3

- permet de représenter une partie intégrante du bloc sur la frontière du bloc principal (main block boundary)
- un port complet est typé par un bloc; il peut ainsi combiner les flux d'éléments en entrée/sortie (information, matière, énergie) et l'exécution d'opérations.
- les ports complets peuvent être « **conjugés** » ayant pour effet d'inverser la direction des éléments
- un port complet représente un bloc physique (connecteur électrique, assemblage mécanique)
- A utiliser lorsque qu'on modélise une partie actuelle d'un système (mode « Boîte Blanche »)

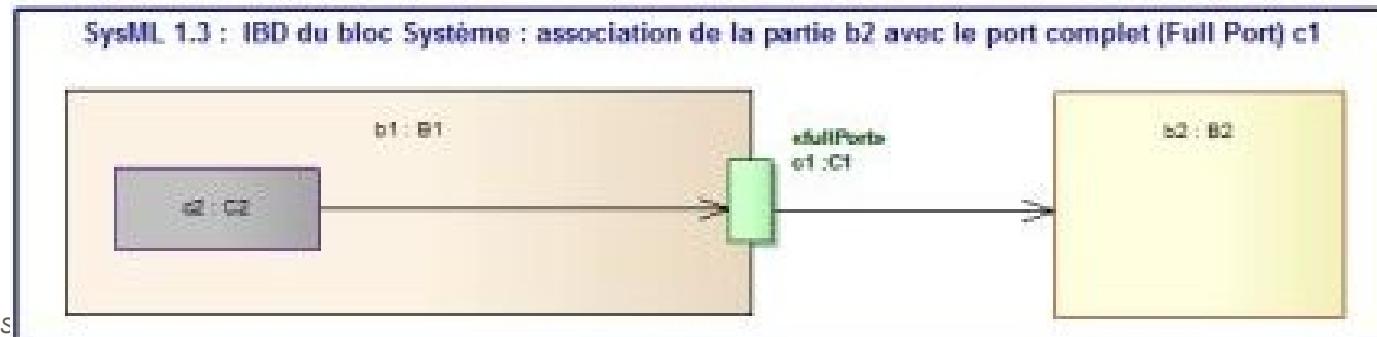


Diagramme de bloc interne (5)

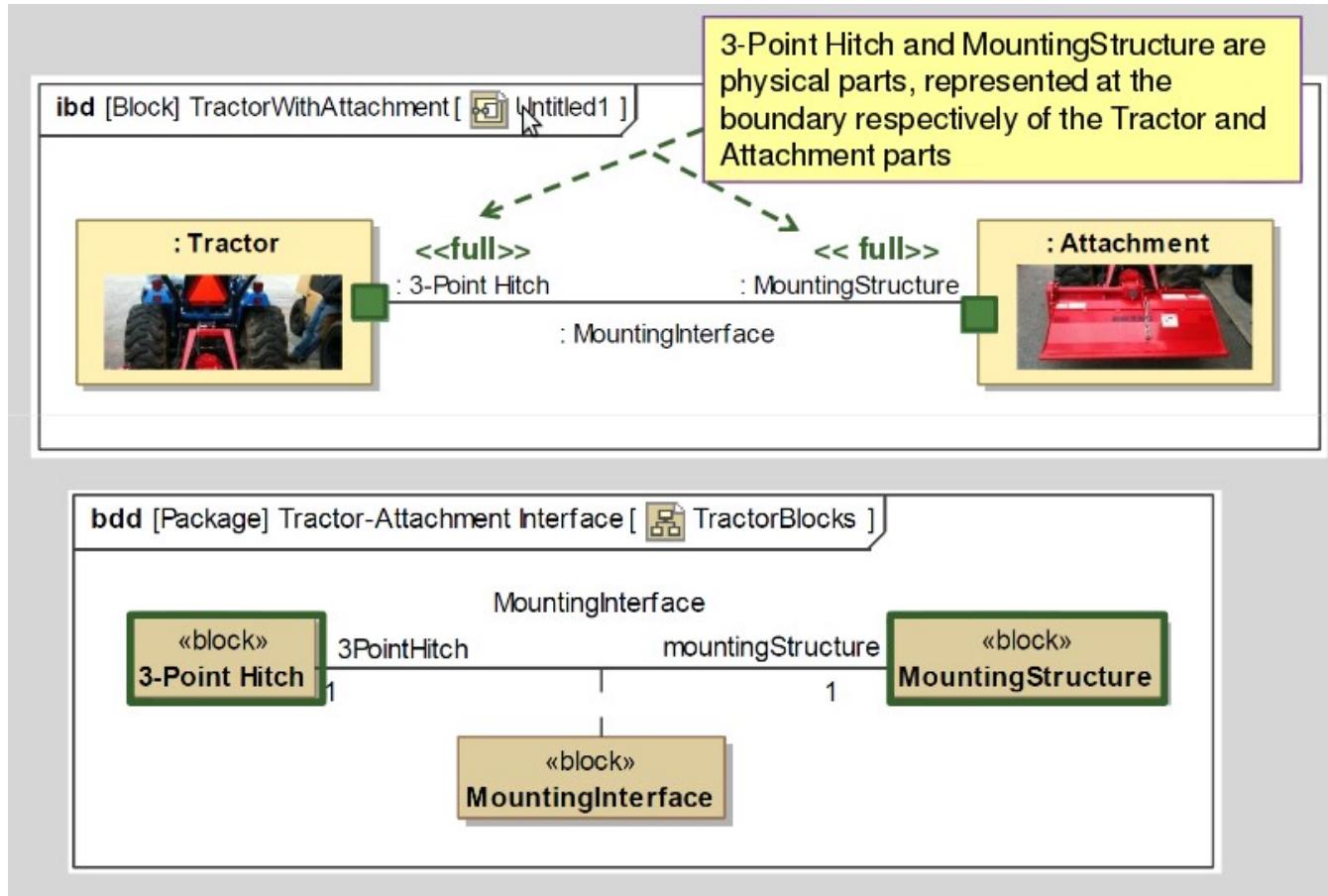


Diagramme de bloc interne (6)

Ports proxy : SysML 1.3

- Permet d'exposer certaines opérations et caractéristiques d'un bloc (ou bloc interne) à l'extérieur d'un bloc. A utiliser lorsque le bloc est vu comme une « Boite Noire » .
- Sert d'intermédiaire aux fonctions du bloc principal ou de ses parties intégrantes
- Un port proxy ne porte pas de comportement, ni ne constitue une partie du bloc principal.
- Les flux d'éléments ou l'exécution d'opérations sur le port proxy sont transmis directement vers le bloc principal ou une partie intégrante.
- Un port proxy est typé par un bloc d'interface (**block interface**) pour spécifier les fonctions disponibles (éléments, opérations), alors que les ports complets comme indiqué précédemment sont typés par des blocs

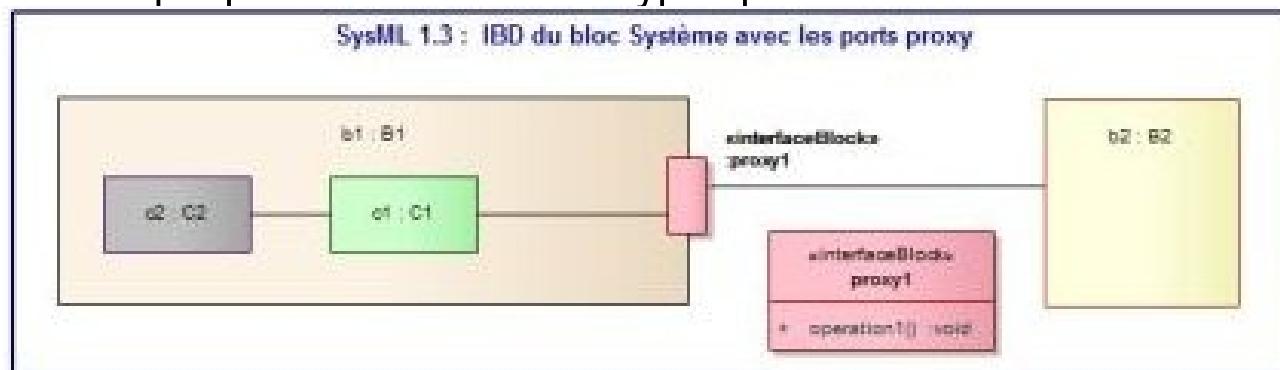


Diagramme de bloc interne (7)

- Ports et flux imbriqués (nested ports & flows) : SysML permet de définir des ports imbriqués ; pour cela le bloc utilisé comme type du port possède lui-même des ports

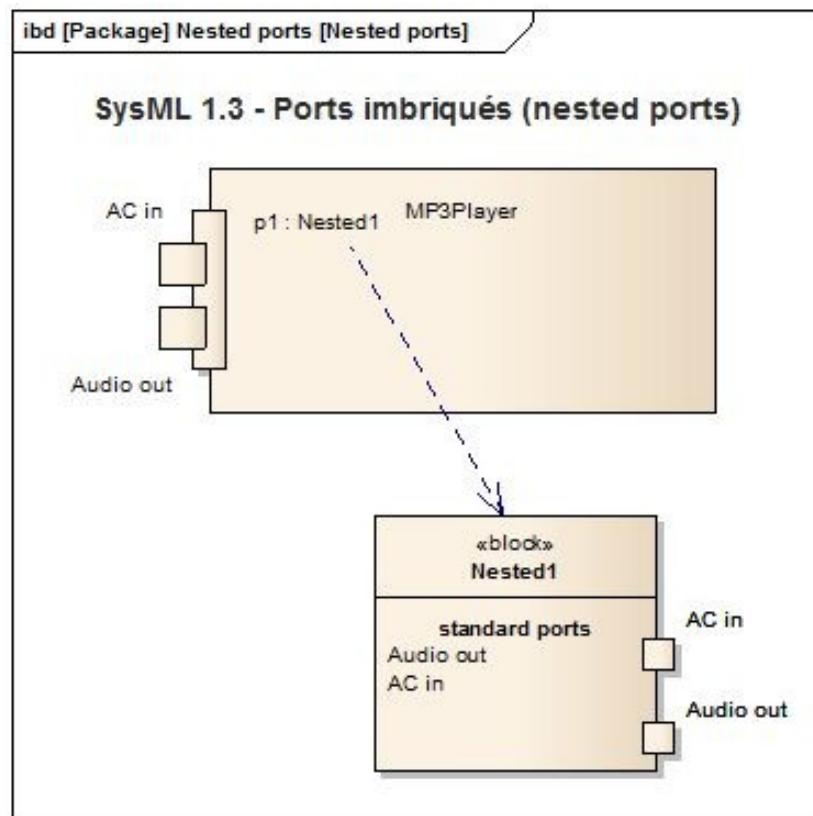


Diagramme de bloc interne (8) : Allocation logique et physique

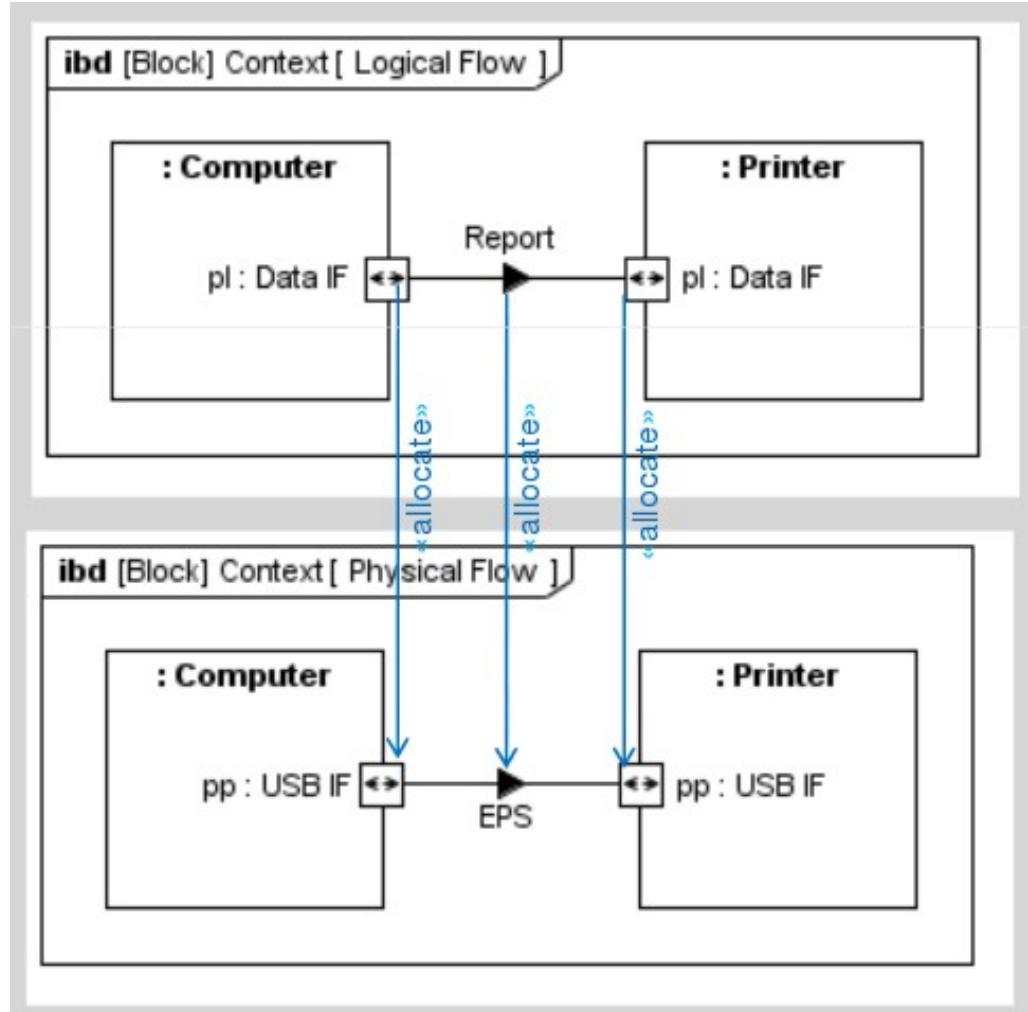


Diagramme de bloc interne (9) : Servomoteur

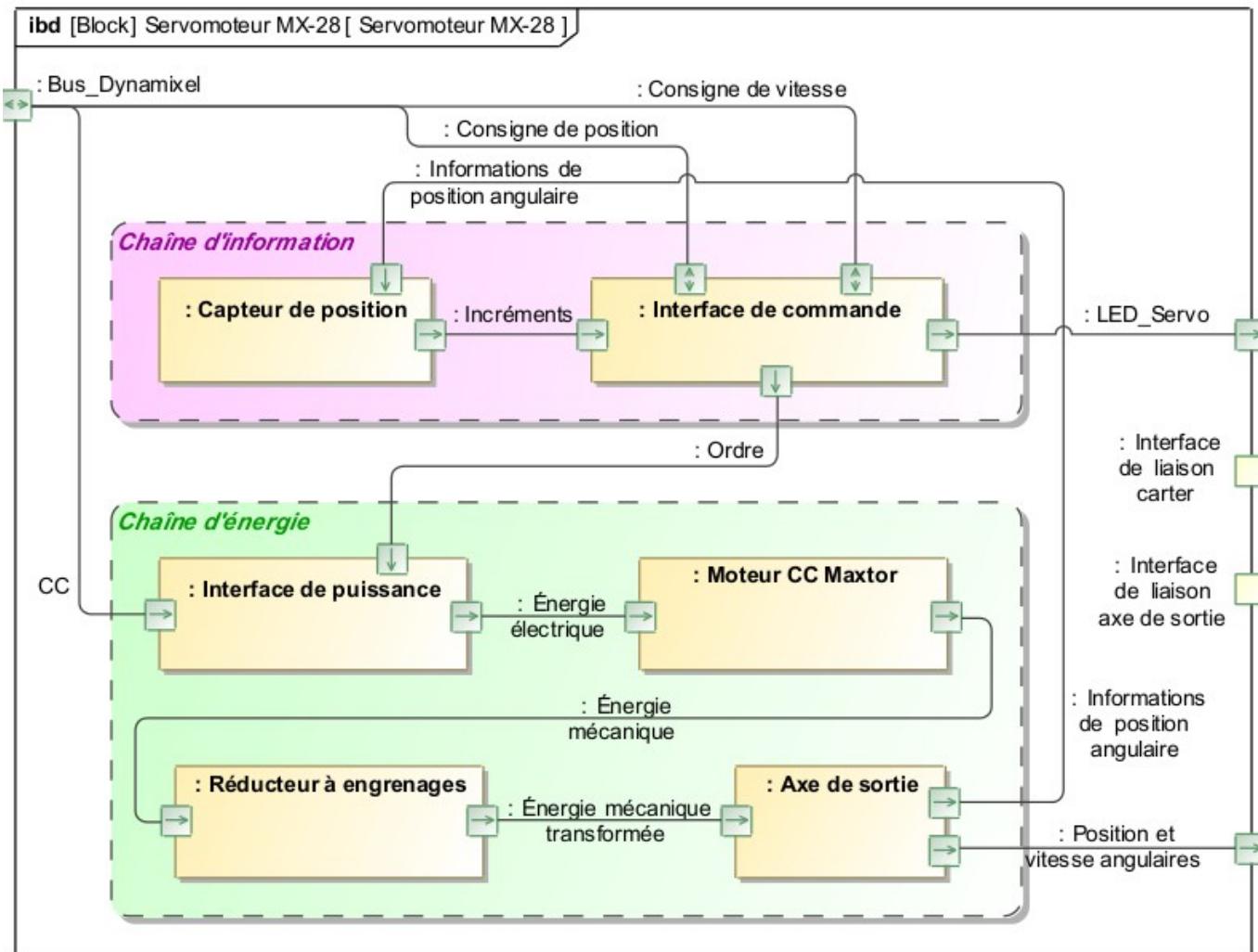


Diagramme de séquence (complément) : Vue « Boite Blanche »

- Une fois les blocs internes identifiés, on peut compléter l'analyse avec des diagrammes de séquence vue « Boite Blanche » où l'on voit apparaître les instances de blocs internes.

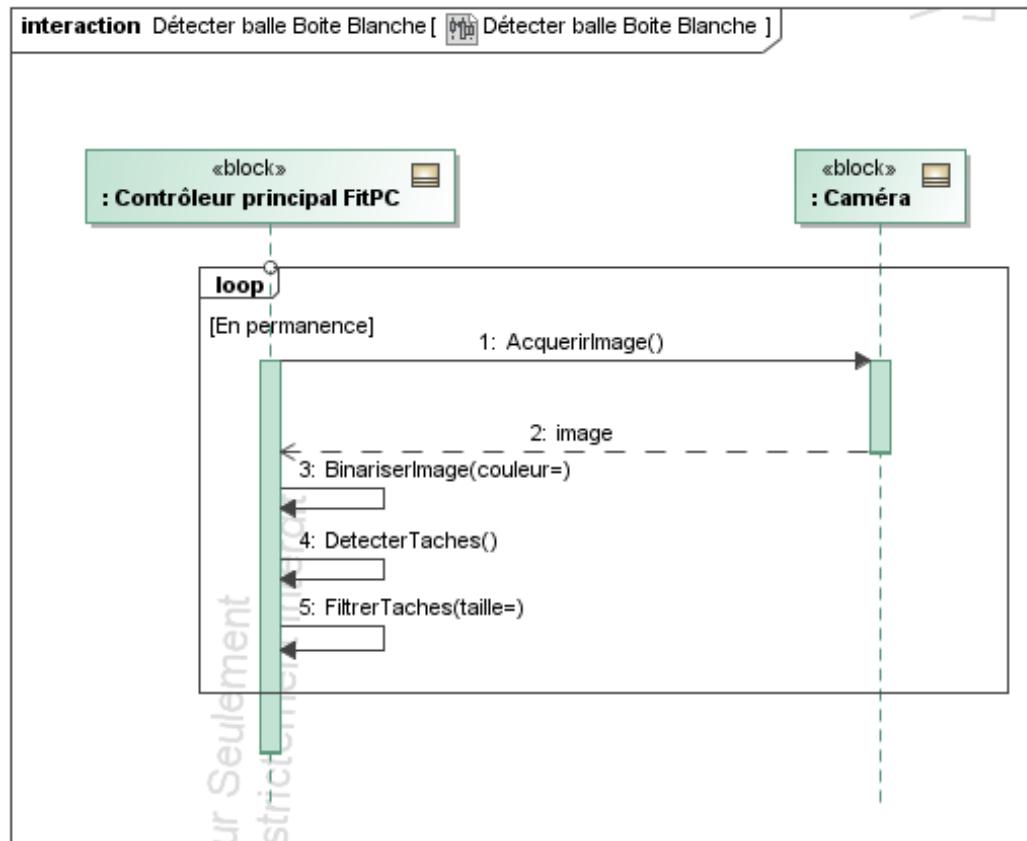
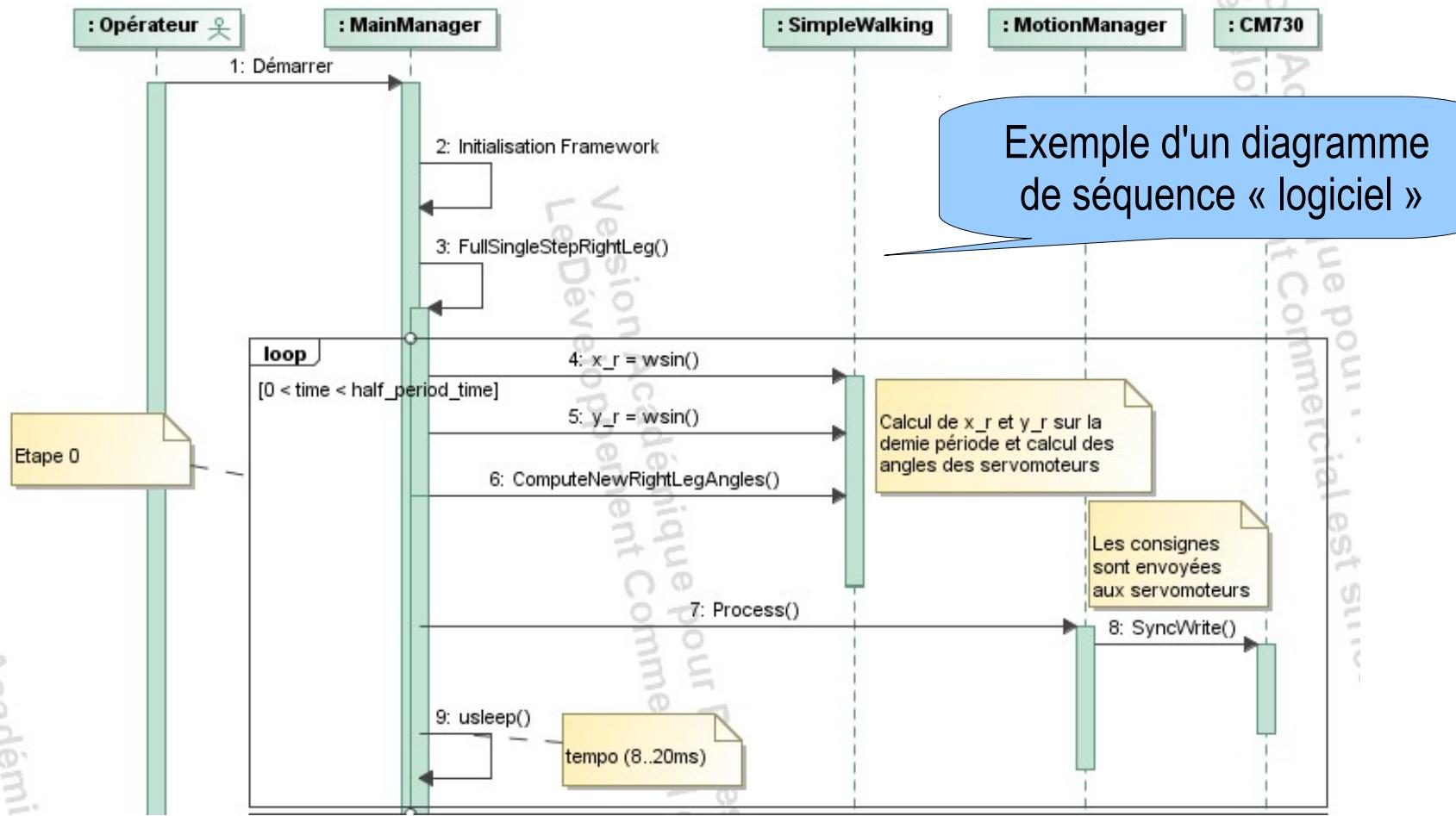


Diagramme de séquence (complément) : Vue « Boîte Blanche »



Sommaire



- Introduction
- Les diagrammes SysML
- Présentation de DARwIn-OP
- Avant de commencer...
- Diagramme de contexte
- Diagramme des cas d'utilisation
- Diagramme d'exigences
- Diagramme de séquence
- Diagramme de définition de bloc
- Diagramme de bloc interne
- Diagramme de machine d'état
- Diagramme d'activité
- Diagramme paramétrique



Diagramme de machine d'état (1)

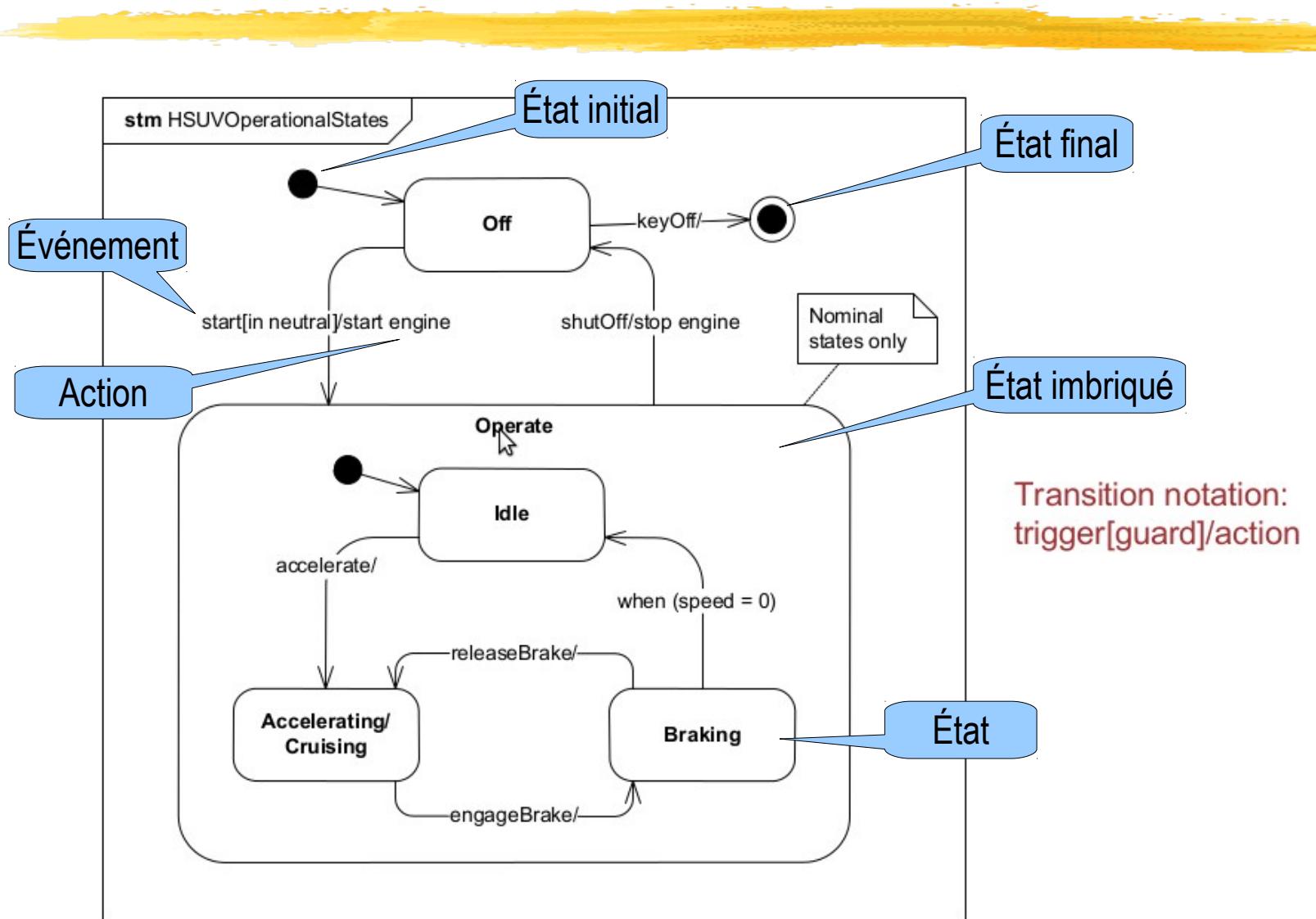
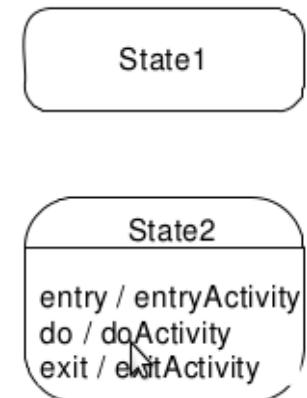


Diagramme de machine d'état (2)

- Utilisé habituellement pour représenter le **cycle de vie** d'un bloc
- Changement d'état sur **événement** asynchrone (**trigger**) et/ou sur **condition** (**guard**)
 - Transition entre deux états avec déclencheur (trigger), condition de garde (guard) et action associée : déclencheur [condition de garde] / action
- État avec actions :
 - en entrée (**entry**) : exécutées à chaque entrée dans l'état
 - en sortie (**exit**) : exécutées à chaque sortie de l'état
 - permanente (**do**) : exécutées le temps que dure l'état
- Un état peut inclure des états **imbriqués** ou **parallèles**
- Possibilité d'émettre/recevoir des signaux pour communiquer avec d'autres blocs pendant les transitions.



Sommaire



- Introduction
- Les diagrammes SysML
- Présentation de DARwIn-OP
- Avant de commencer...
- Diagramme de contexte
- Diagramme des cas d'utilisation
- Diagramme d'exigences
- Diagramme de séquence
- Diagramme de définition de bloc
- Diagramme de bloc interne
- Diagramme de machine d'état
- Diagramme d'activité
- Diagramme paramétrique



Diagramme d'activité (1)

- Permet la modélisation du comportement dynamique d'un système.
- Proche du Grafcet (plus riche)
- Plusieurs niveaux d'utilisation :
 - **Macroscopique** : comportements des éléments du contexte du système
 - **Microscopique** : comportement d'un bloc ~ proche du codage de la fonction.

Diagramme d'activité (2) : exemple d'un système de surveillance

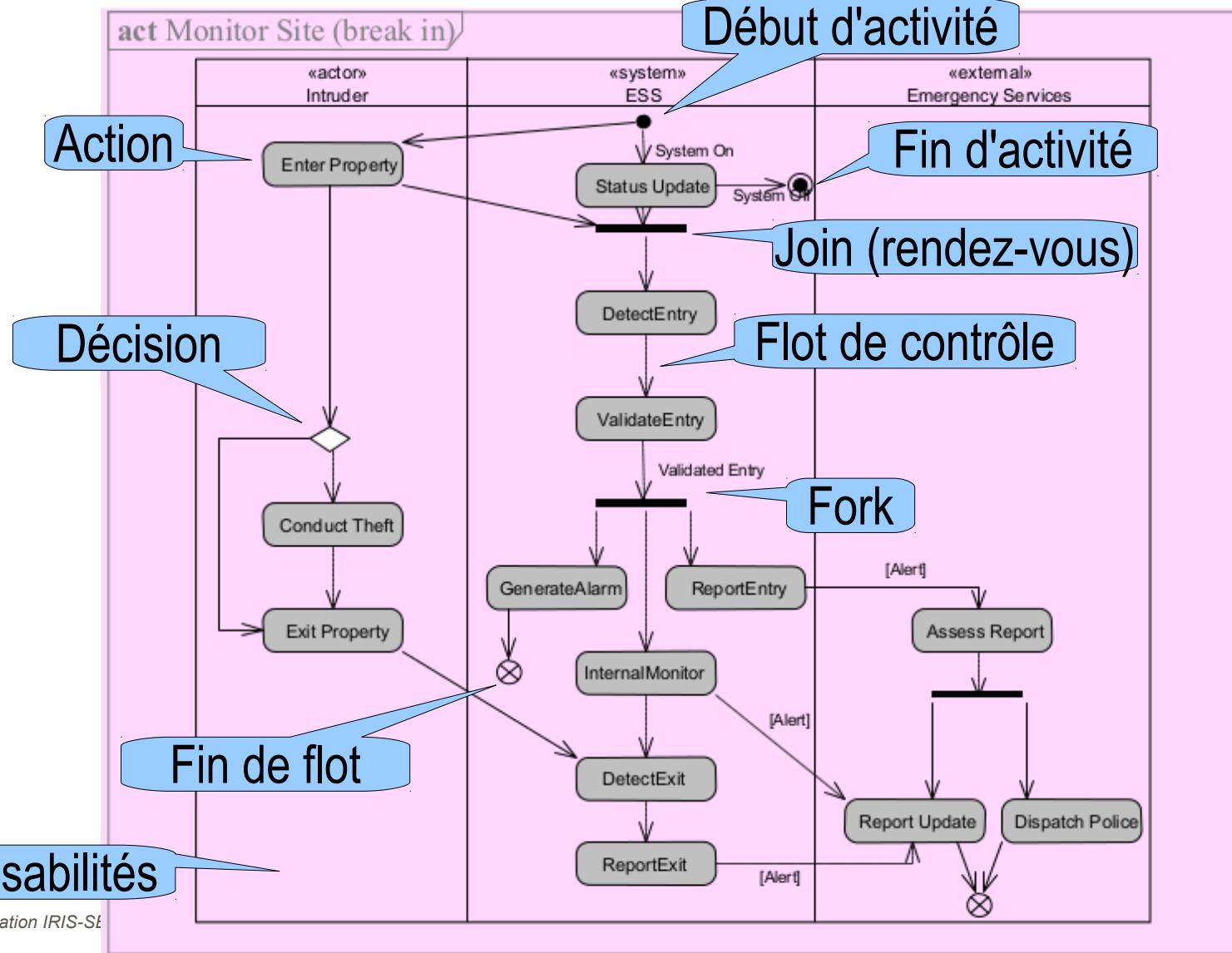
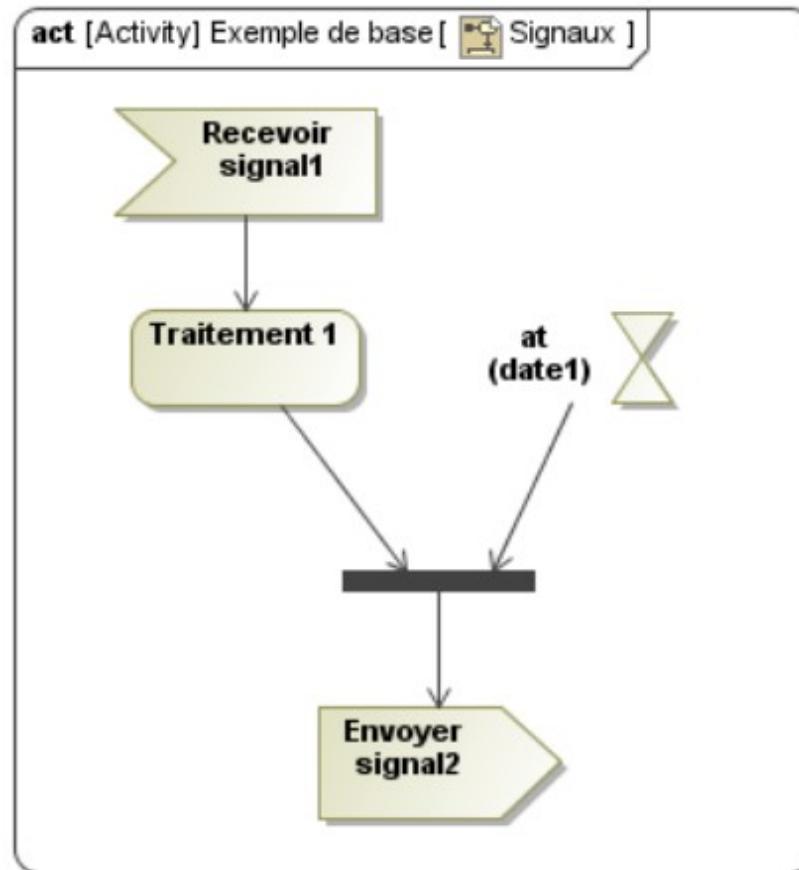


Diagramme d'activité (2) : complément



Sommaire



- Introduction
- Les diagrammes SysML
- Présentation de DARwIn-OP
- Avant de commencer...
- Diagramme de contexte
- Diagramme des cas d'utilisation
- Diagramme d'exigences
- Diagramme de séquence
- Diagramme de définition de bloc
- Diagramme de bloc interne
- Diagramme de machine d'état
- Diagramme d'activité
- Diagramme paramétrique

Diagramme paramétrique (1)

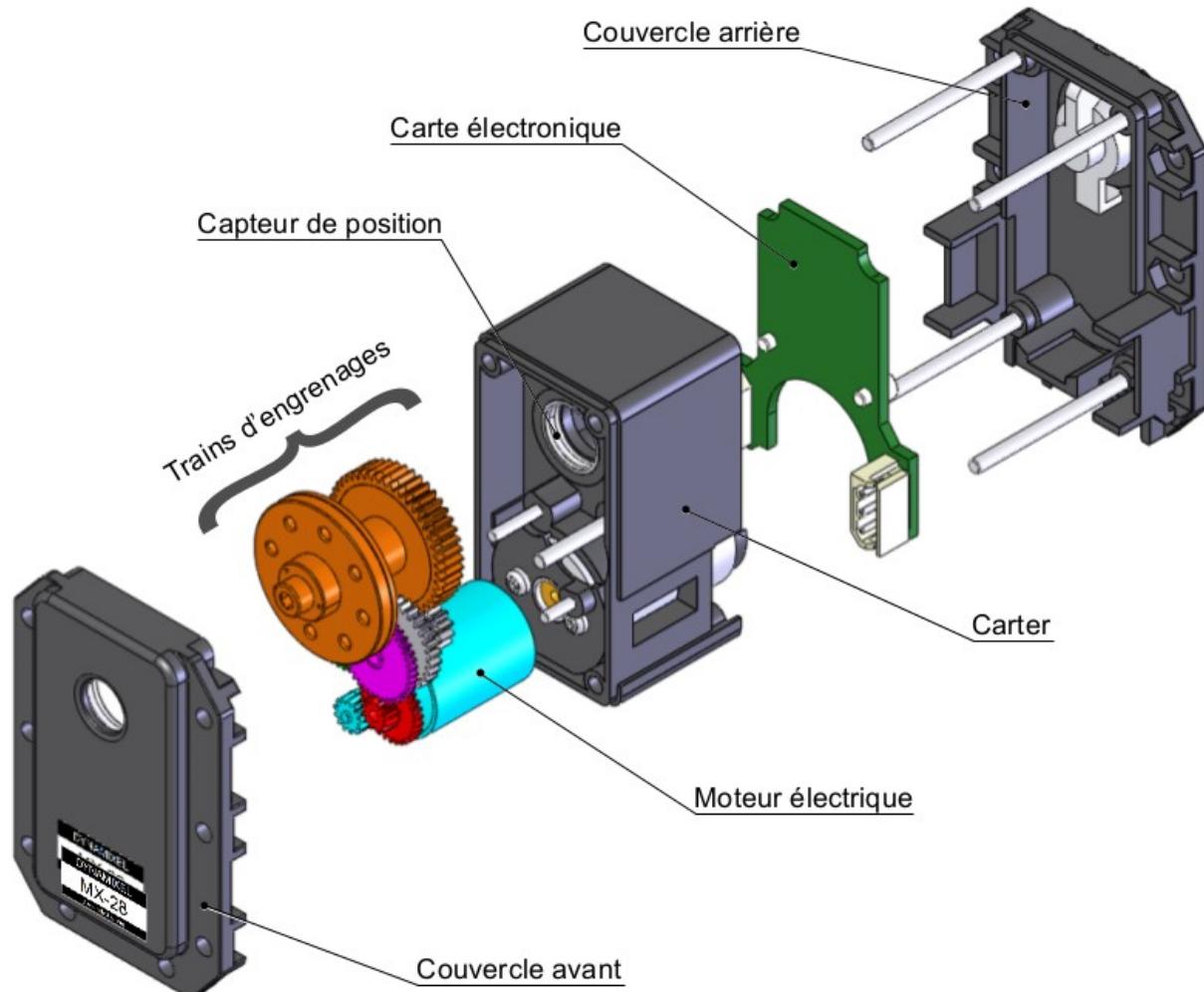


Diagramme paramétrique (2)

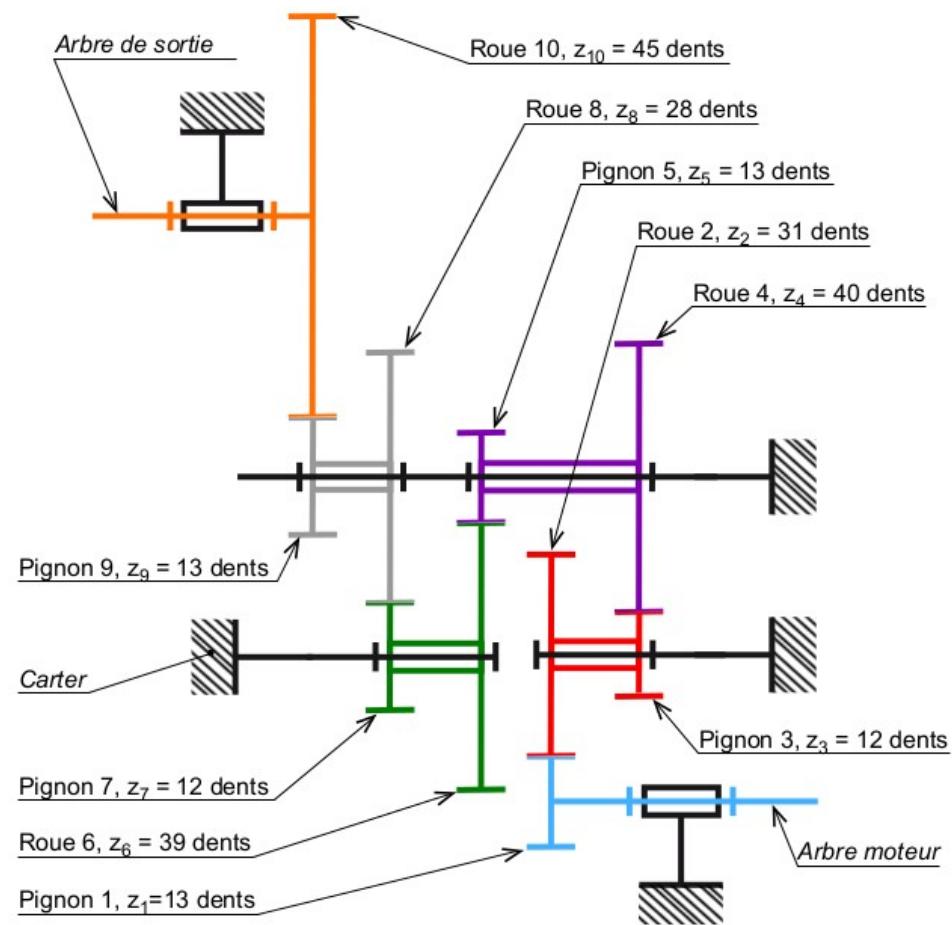
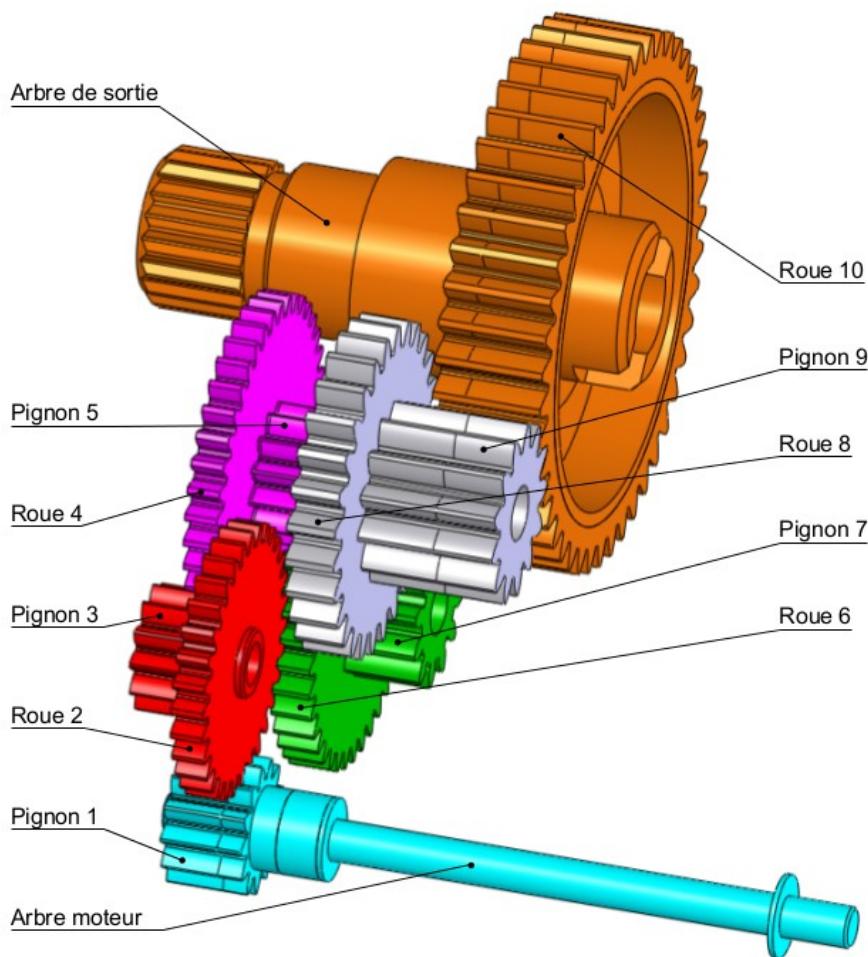


Diagramme paramétrique (3)

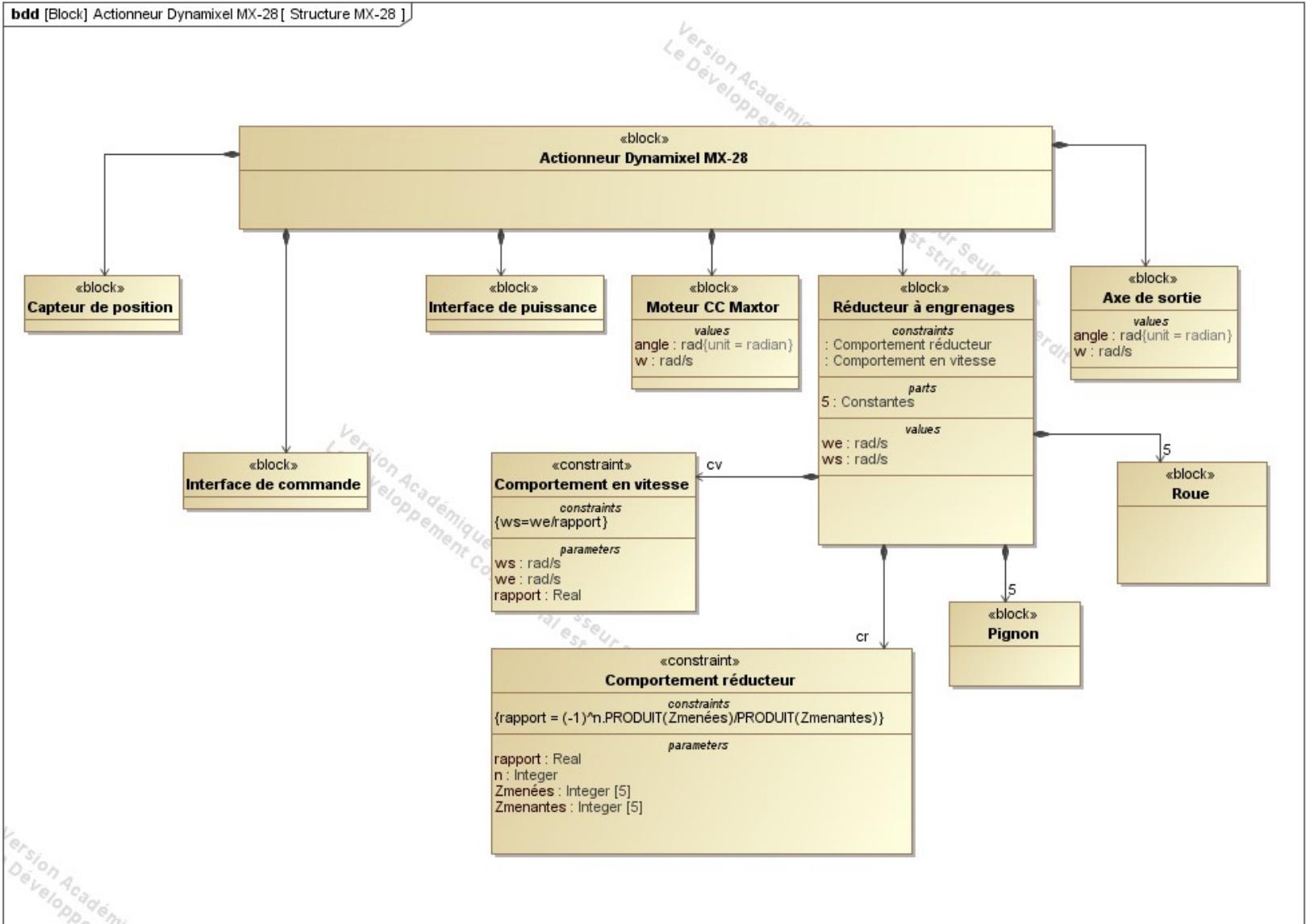


Diagramme paramétrique (4)

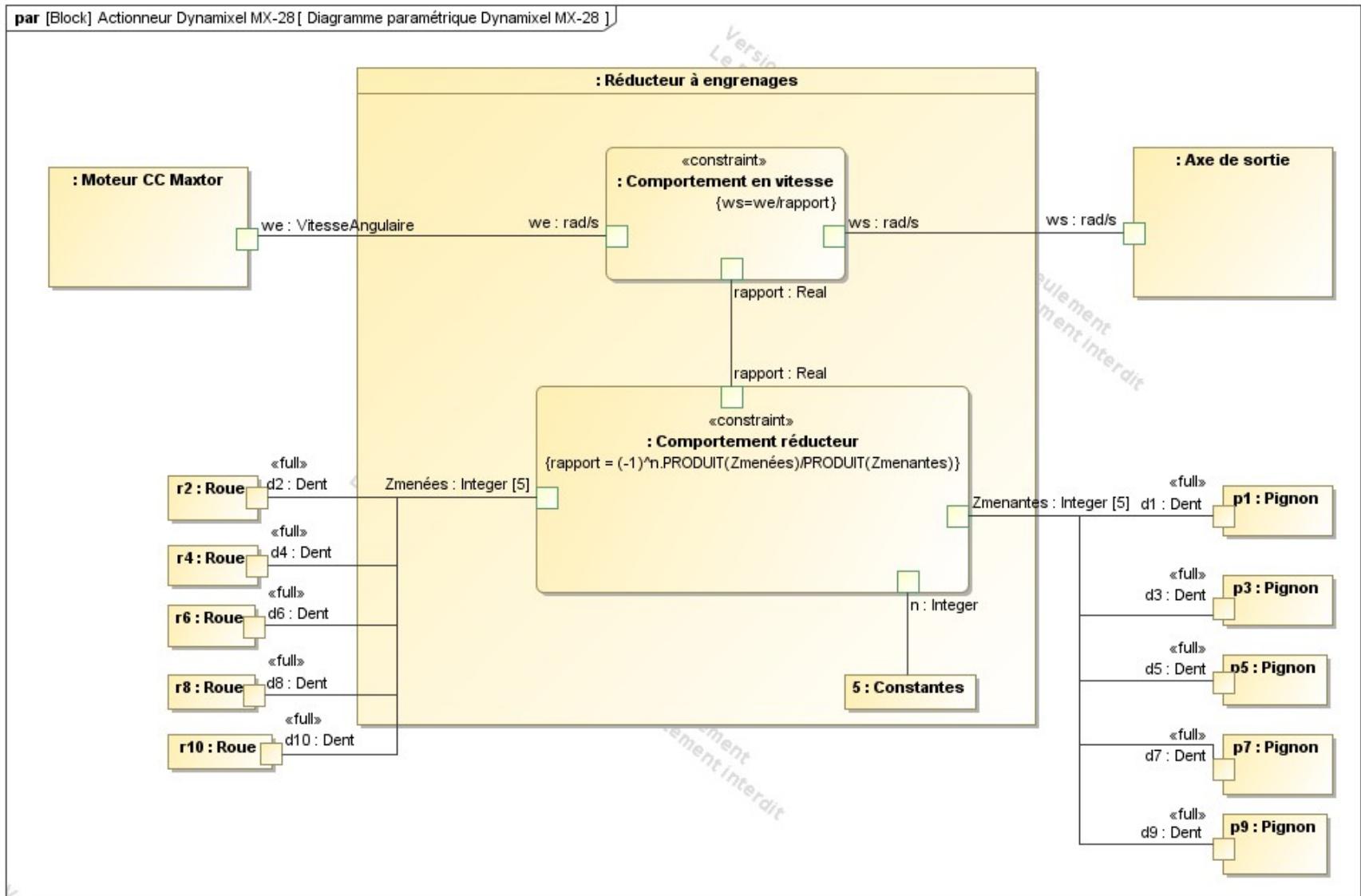


Diagramme paramétrique (3)

- Le **diagramme paramétrique** est un diagramme de structure exprimant les relations entre les grandeurs caractéristiques du système.
- **Étape 1** : définition des relations physiques à l'aide de blocs stéréotypés « constraint » qui sont des parties (parts) des blocs qu'ils paramètrent. Les relations s'exprime en langage « libre » ou en **OCL**
- **Étape 2** : élaboration du diagramme paramétrique en instanciant les blocs et les blocs de contraintes et en établissant les relations entre les instances.

- Article « SysML, un langage modèle »
 - Jean-Pierre Lamy - Revue « Technologie » - Avril 2012
- Livre « SysML par l'exemple »
 - Pascal Roques
- Sites officiels :
 - <http://www.sysml.org>
 - <http://www.uml.org>
- Guide pour modéliser : <http://www.sysmod.de>
- Wikipedia