Ingénierie des modèles appliquée à la cybernétique

Etudes et Expérimentations

Pascal ANDRE & Etudiants ALMA-IR/MIAGE

AeLoS / LS2N – UMR CNRS 6241
{Firstname.Lastname}@univ-nantes.fr

AeLoS Seminars

Outline

Outline

Contexte

Le logiciel prend une part croissante dans la production automatisée, que ce soit au niveau de l'industrie, de la mécanisation mais aussi du service avec l'assistance de robots et d'intelligence articifielle. Le lancement de programmes comme "Industrie du futur" ont permis de faire converger les efforts en cybernétique. On parle d'Industrie 4.0 pour ces innovations relatives à l'internet des objets, la robotique, l'intelligence articifielle ou le big data.

Le projet proposé ici est bien plus modeste, il consiste à mettre en place une approche basée sur les modèles (Ingénierie des modèles) [Brambilla, 2012] pour construire du logiciel sûr de contrôle d'automates.

Contexte du travail

Ingénierie des modèles pour la cybernétique

- modélisation
- vérification (preuves et tests)
- transformation de modèles, raffinelment
- séparation des préoccupations

Thématiques :

- Service based Component (SbC) engineering (Kmelia)
- Cyber-Physical Production Systems (CPPS)
- Communication protocols and Services
- IoT...

IR :

- Automatiser la production logicielle de contrôleurs d'automates
- Automatiser la production logicielle de protocoles de communications

Motivations

Dans ce projet exploratoire, nous avons pour objectif final de réaliser un outil (une chaîne logicielle) qui permette de modéliser des automates et générer des programmes de commande pour les piloter en tenant compte des actions à réaliser, des événements provenant du contexte et d'un changement, potentiellement à la volée, d'ordre de pilotage.

La mise en œuvre se fait sur un environnement réel par l'intermédiaire du digital twin.

Application

Lego Mindstorm Education - LeJoS - Android

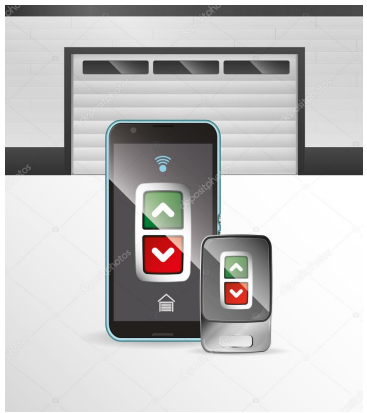
Cas d'étude

Application

Lego Mindstorm Education - LeJoS - Android

Cas d'étude

- DomoDoor - domotique - contrôle de porte



source : http://https://fr.depositphotos.com/122422584/stock-illustration-automatic-garage-door.html

http://www.portail-automatique-reunion-974.fr/motorisation/commander-son-portail-depuis-smartphone-ou-tablette/

https://fr.liftmaster.com/for-homes/myq-connected-home

Application

Lego Mindstorm Education - LeJoS - Android

Cas d'étude

- DomoDoor - domotique - contrôle de porte
- DLC - contrôle automatique de la lumière d'une salle

source : https://blog.materielelectrique.com/controle-eclairage-domotique/

Application

Lego Mindstorm Education - LeJoS - Android

Cas d'étude

- DomoDoor - domotique - contrôle de porte
- DLC - contrôle automatique de la lumière d'une salle
- Riley Rover - automatisme

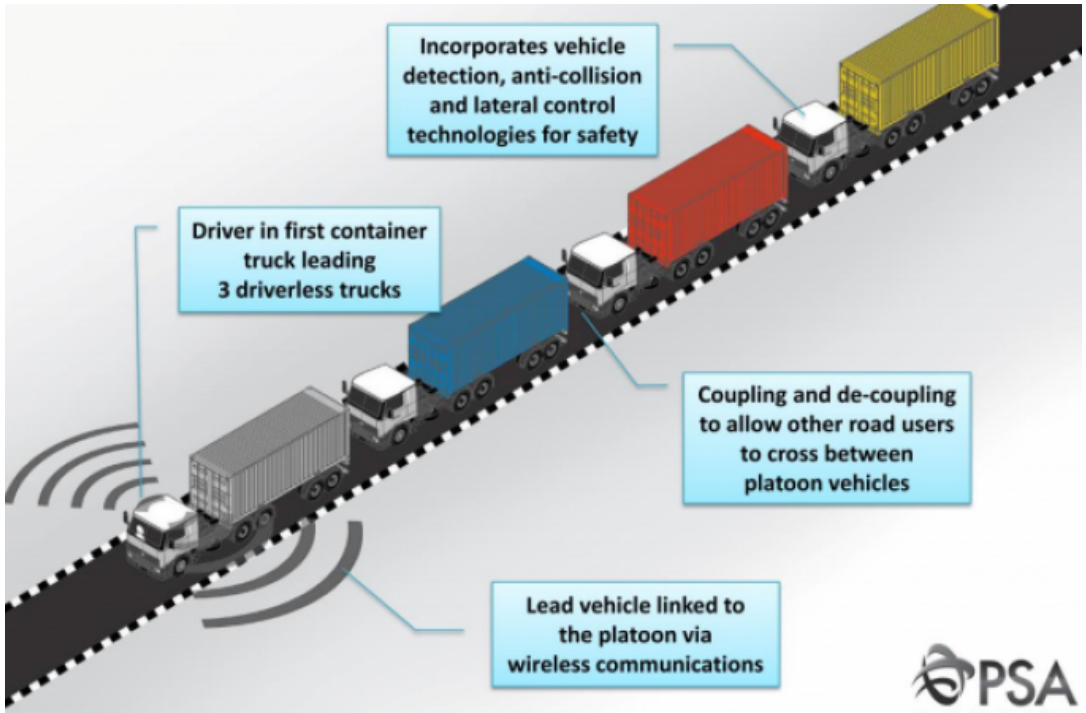
source : https://blog.materielelectrique.com/controle-eclairage-domotique/

Application

Lego Mindstorm Education - LeJoS - Android

Cas d'étude

- DomoDoor - domotique - contrôle de porte
- DLC - contrôle automatique de la lumière d'une salle
- Riley Rover - automatisme
- Platoon - automatisme



source : https://theloadstar.co.uk/psa-roadtest-autonomous-truck-platooning-singapore/

Application

Lego Mindstorm Education - LeJoS - Android

Cas d'étude
Domotique

source : https://dyw7ncnq1en5l.cloudfront.net/optim/article/1767/domotique-illus1_2.jpg

Outline

Objectifs thématiques

L'objectif est de mettre en place une chaîne de production de contrôleurs de ces automates, soit individuellement soit en systèmes d'agents interagissant de telle sorte que différentes contraintes de fonctionnement, de sûreté de fonctionnement et de performance soient pris en compte [Rierson, 2013]. Par exemple, on trouvera des contraintes fonctionnelles et non-fonctionnelles telles que :

- L'automate respecte des contraintes de sûreté de fonctionnement, par exemple un véhicule ne sort pas de sa zone d'intervention.
- L'automate dispose de propriétés de vivacité, par exemple une porte ne reste pas bloqué du fait de son déplacement ou du fait d'un manque d'énergie...)
- L'automate réalise son traitement dans un temps acceptable (efficacité) avec une utilisation raisonnée de ses ressources.
- etc.

Certaines propriétés sont générales (absence de blocage, réinitialisabilité...), d'autres sont liées à l'environnement ou au système lui-même (énergie, dangerosité, qualité de service....).

Qu'ils soient décrits en UML 2 ou dans d'autres langages de description d'architectures, les modèles considérés sont suffisamment détaillés pour être vérifiés et rendus exécutables.

Outline

Approche 1/2

Du point de vue logiciel, on se placera dans une approche composants et services. Cette visioion modulaire facilite l'interopérabilité des systèmes et des niveaux d'abstractions (entre un processus métier et une séquence d'ordre de commande). On considèrera au moins deux niveaux :

- le niveau modélisation et simulation où sont représentés les fonctionnements individuels et collectifs, décrites et analysées les contraintes sous forme d'une image numérique (digital twin). On pourra utiliser un ou plusieurs langages de modélisation (SysML [Friedenthal et al., 2008], Kmelia [André et al., 2010], UML [André and Vailly, 2013], AADL [Feiler and Gluch, 2012]), des outils de vérification associés, des outils de simulation, etc. L'ensemble pourra être mis en œuvre dans COSTO, le support d'outil du modèle à composante Kmelia [André et al., 2010] ou un autre environnement.
- le niveau opérationnel où sont mis en œuvre les commandes sur les dispositifs physiques. On utilise pour cela des outils de communication vers les automates (programmable logic controller -PLC), les robots ou les machines.

Approche 2/2

On considèrera au moins deux niveaux :

- le niveau modélisation et simulation ...
- le niveau opérationnel ...

Des niveaux intermédiaires peuvent être mis en œuvre pour faciliter la mise en place de la chaîne de production de code en s'inspirant de l'approche dirigé par les modèles (MDD) [Brambilla, 2012] et des lignes de production logicielles (Software Product Lines) [Atkinson, 2002, Rashid et al., 2011].

Dans le développement dirigé par les modèles (MDD), il est essentiel de s'assurer de la correction des modèles avant de commencer le processus de transformations et de génération de code. On diminue ainsi le coût élevé de la détection tardive d'erreurs [Shanks et al., 2003, Gogolla et al., 2005]. $\Longrightarrow$ CostoTest

Hypothèses

Nous ciblons des modèles à composants et services avec un comportement plus ou moins complexe et détaillé, comprenant des données et des communications (les données ne sont pas limitées à des paramètres, les services ne sont pas limités à des opérations). Le niveau de détail des spécifications est suffisant pour être exécutable, soit directement, soit par l'attribution des opérations concrètes pour une notation plus abstrait.

Dans ces projets exploratoires, nous avons donc pour objectif de proposer une méthodologie et d'expérimenter concrètement les propositions dans une approche agile. Les logiciels de contrôles mis en œuvre seront implantés en utilisant des legos MINDSTORMS® et des applications Android [Garin, 2012].

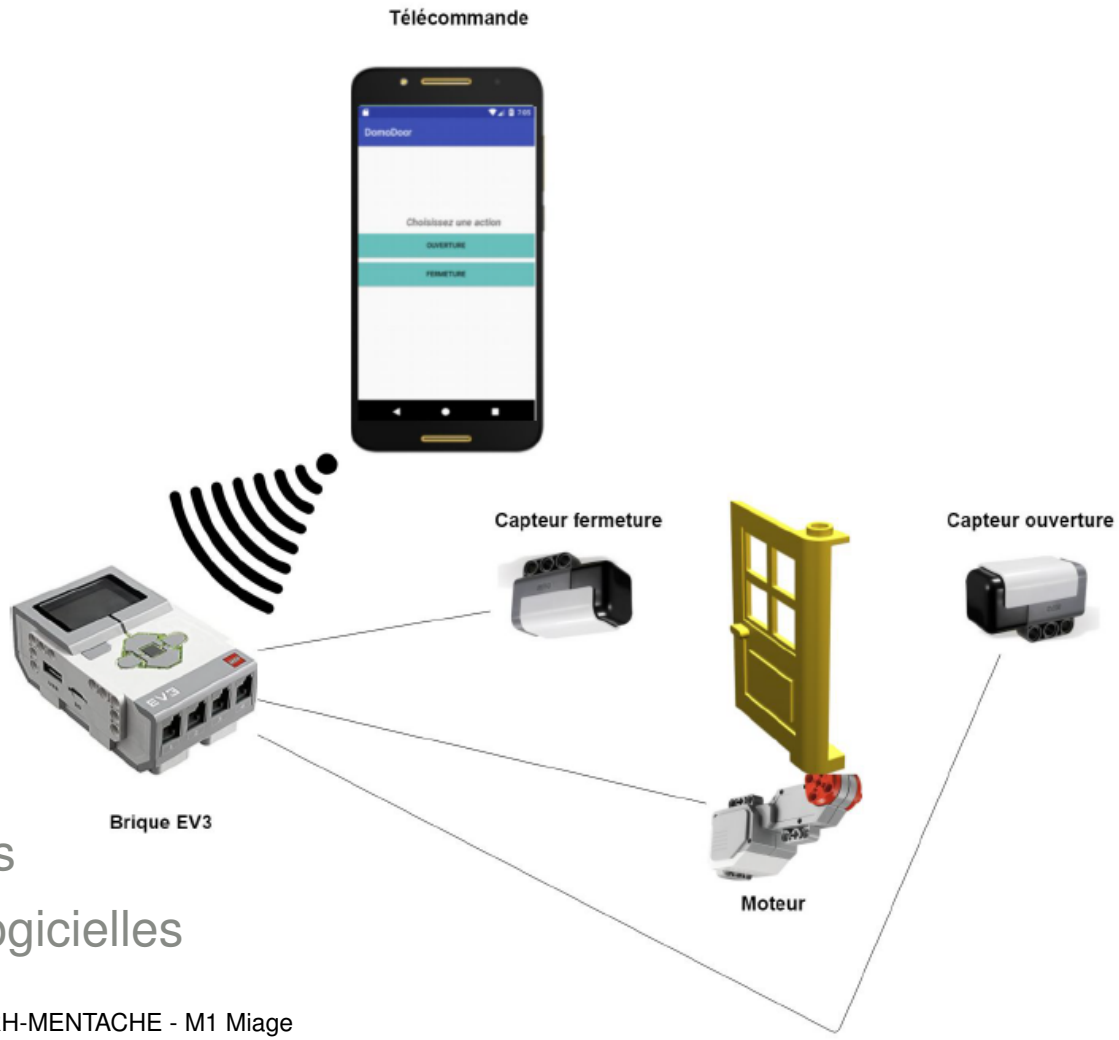
Les technologies identifiées pour réaliser notre outil de ligne de production logicielle reposent sur l'ingénierie des modèles et notamment sur la transformation de modèles, les outils de modélisation et vérification formelles.

Outline

Lego 1 : DomDoor (IR 1)

Cas + exigences

Lego Mindstorm Education - LeJoS - Android

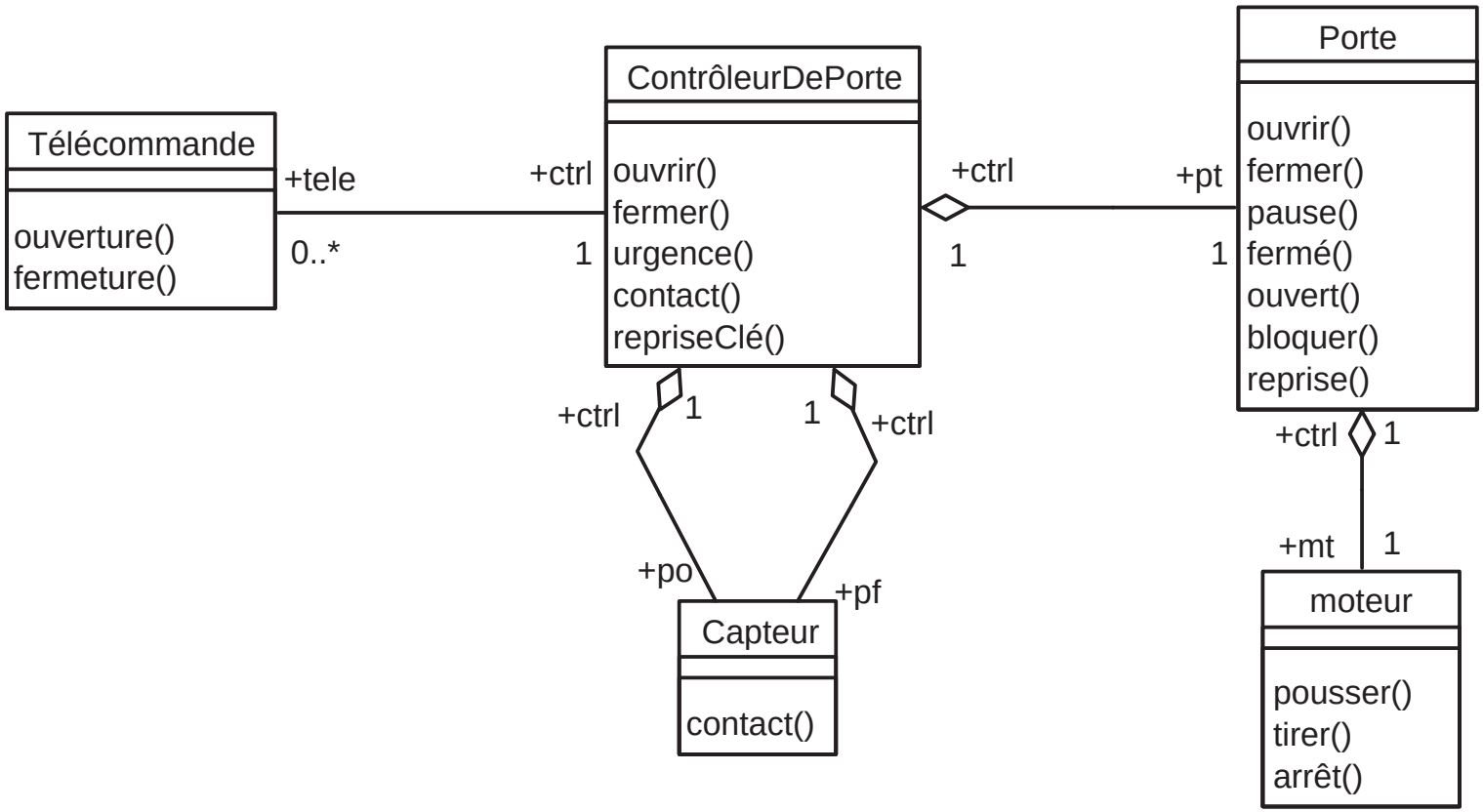


- Etude de cas
- Exigences logicielles

source : DCD2-BEJJAJI-ZOUGARH-MENTACHE - M1 Miage

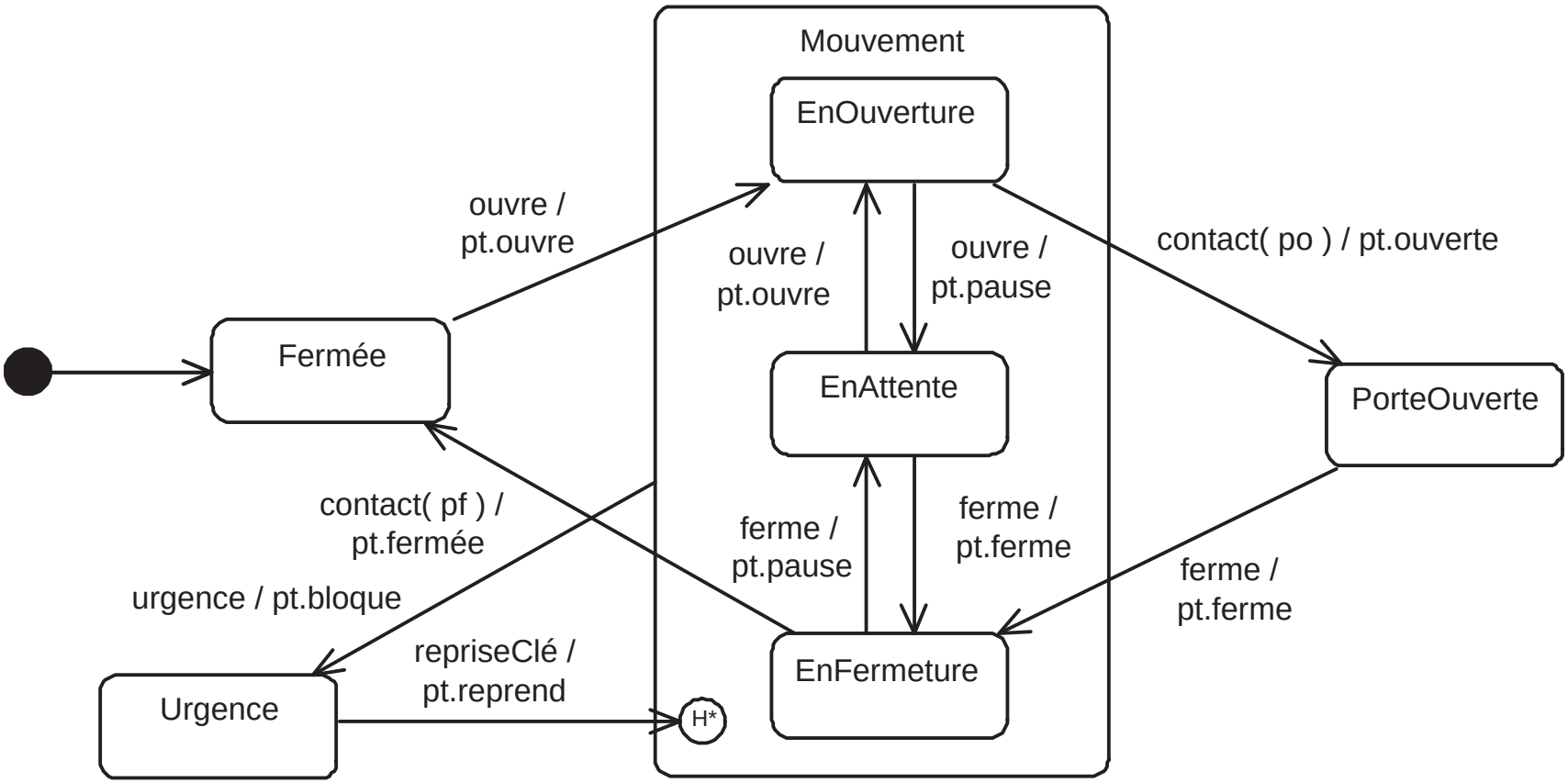
Lego 1 : DomDoor (IR 1)

Modèle - DC



Lego 1 : DomDoor (IR 1)

Modèle - DET



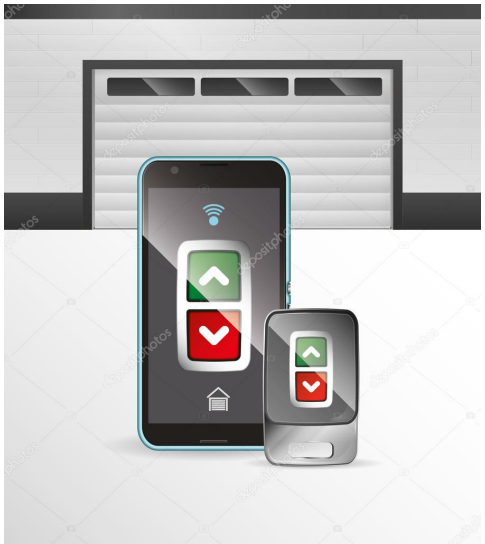
Lego 1 : DomDoor (IR 1)

Explication, démonstration

Lego Mindstorm Education - LeJoS - Android

- Maquette
- Transformations de modèles

Démonstrations



Lego 2 : Riley Rover (Miage, IR2)

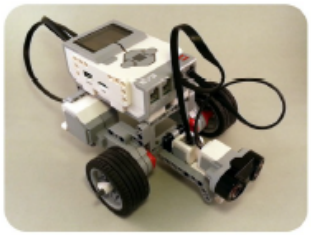
Cas + exigences

Ultrasonic Sensor

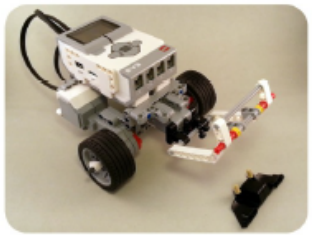
Colour Sensor

Gyro Sensor

Gripper



Colour / Ultrasonic combination

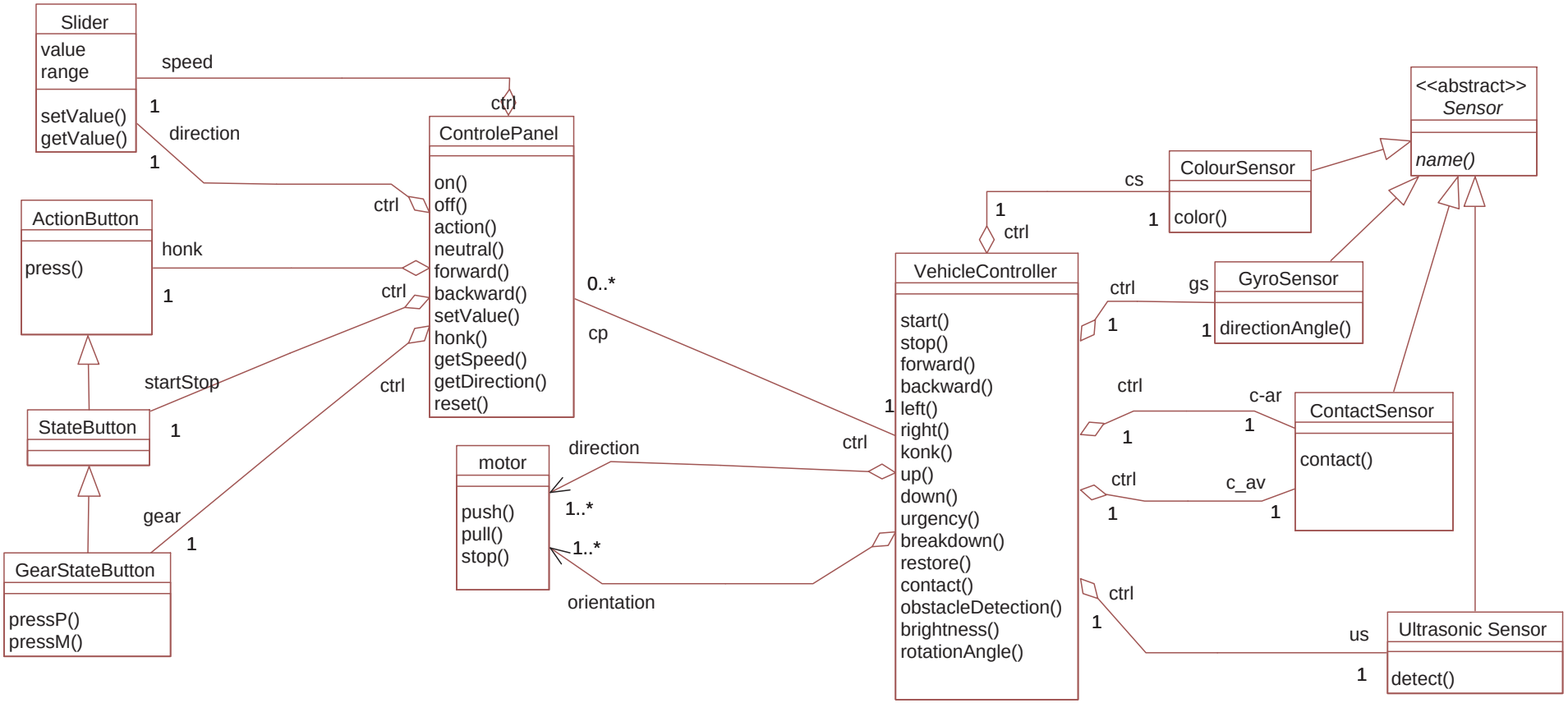


Cargo Delivery

- Etude de cas
- Exigences logicielles

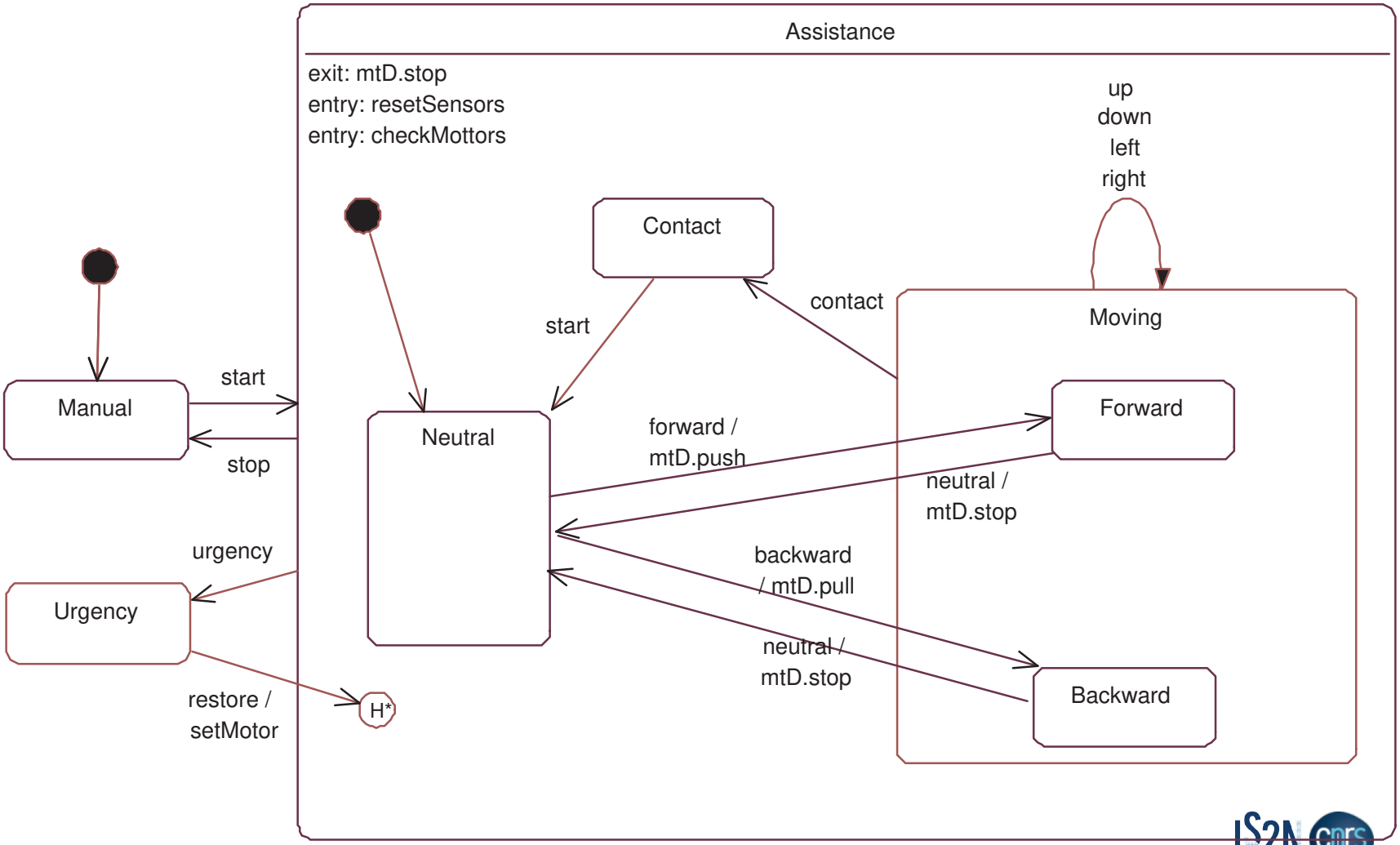
Lego 2 : Riley Rover (Miage, IR2)

Modèle - DC



Lego 2 : Riley Rover (Miage, IR2)

Modèle - DET



Lego 2 : Riley Rover (Miage, IR2)

Explication, démonstration

- Maquette
- Transformations de modèles

- Miage : application mobile + maquette
- AIMA : communications

Démonstrations

Outline

Discussion

Outline

Documentation

- https://www.lego.com/fr-fr/mindstorms
- https://lejos.sourceforge.io/nxj.php
- https://hal.archives-ouvertes.fr/hal-01147205v1
- https://hal.archives-ouvertes.fr/hal-01628303
- http://costo.univ-nantes.fr/

Références I

André, P., Ardourel, G., Attiogbé, C., and Lanoix, A. (2010).
Using assertions to enhance the correctness of kmelia components and their assemblies.
ENTCS, 263 :5 – 30.
Proceedings of FACS 2009.

André, P. and Vailly, A. (2013).
Développement de logiciel avec UML2 et OCL ; cours et exercices corrigés, volume 6 of *Collection Technosup*.
Editions Ellipses.
ISBN 9782729883539.

Atkinson, C. (2002).
Component-based Product Line Engineering with UML.
Addison-Wesley object technology series. Addison-Wesley.

Brambilla, M. (2012).
Model-driven Software Engineering (Mde).
Morgan & Claypool.

Feiler, P. H. and Gluch, D. P. (2012).
Model-Based Engineering with AADL : An Introduction to the SAE Architecture Analysis & Design Language.
Addison-Wesley Professional, 1st edition.

Références II

Friedenthal, S., Moore, A., and Steiner, R. (2008).
A Practical Guide to SysML : Systems Modeling Language.
Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Garin, F. (2012).
Android - 3e éd. : Apprenez à développer efficacement pour le leader des OS mobiles.
Etude, développement et intégration. Dunod.

Gogolla, M., Bohling, J., and Richters, M. (2005).
Validating uml and ocl models in use by automatic snapshot generation.
Software and Systems Modeling, 4(4) :386–398.

Rashid, A., Royer, J.-C., and Rummler, A. (2011).
Aspect-Oriented, Model-Driven Software Product Lines : The AMPLE Way.
Cambridge University Press, New York, NY, USA.

Rierson, L. (2013).
Developing Safety-Critical Software : A Practical Guide for Aviation Software and DO-178C Compliance.
Taylor & Francis.

Shanks, G., Tansley, E., and Weber, R. (2003).
Using ontology to validate conceptual models.
Commun. ACM, 46(10) :85–89.