Sujet de master 1 ■ Architectures logicielles ALMA ■ 2017–2018

Automatiser la production logicielle de protocoles de communications

Encadrant principal : Pascal Andre,
courriel : `Pascal.Andre@univ-nantes.fr`
tél. : 02 76 64 58 50

Equipe AeLoS, LS2N

1 Le 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) [**?**] pour construire du logiciel sûr de contrôle de protocoles de communication mobiles vers des automates. Ce projet est en lien avec un autre projet de TER.

2 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. Ici nous nous focalisons sur la communication.

L'objectif est de mettre en place une chaîne de production de contrôleurs de communication 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 [**?**]. 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....). Dabns ce projets nous insistons plus sur les propriétés liées à la communication.

Qu'ils soient décrits en UML 2 ou dans d'autres langages de description d'architectures, les modèles considérés sont suffisamment détaillés pour être rendus exécutables [1]. Cela permet, en

1. Les transformations de modèles deviennent pertinentes si les modèles contiennent suffisamment d'informations.

complément de plusieurs techniques de vérification basées sur de la preuve de théorème ou du *model checking*, d'envisager de tester ces modèles.

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 [**?**], Kmelia [**?**], UML [**?**], AADL [**?**]), 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 [**?**] 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.

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) [**?**] et des lignes de production logicielles (Software Product Lines) [**?**, **?**]. 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 [**?**, **?**]. L'équipe **AeLoS** travaille sur la vérification et la validation de logiciels au niveau modèles.

Nous ciblons des modèles à composants et services avec un comportement plus ou mmoins 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 ce projet exploratoire, 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 oeuvre seront implantés en utilisant des legos MINDSTORMS®.

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.

3 Travail à faire - contributions attendues

Dans une vision agile, on commencera par mettre en œuvre une solution concrète opérationnelle de l'automatisme pour monter progressivement en abstraction et placer les différentes briques de l'AGL (Atelier Génie Logiciel) en fonction des besoins et de l'augmentation du périmètre des exigences : programmation des communications entre automates, digital twin et synchronisation, simulation, modélisation, vérification, etc.

A terme on souhaite implanter une application mobile pour le pilotage en temps réel de l'automatisme et fournir des ordres à la volée avec des protocoles de communication sans fil (bluetooth et/ou wifi).

En paralèlle, une partie bibliographie et positionnement scientifique seront demandés pour établir une petite synthèse de l'état du projet avec une réflexion sur les solutions envisagées, notamment pour la preuve de propriétés formelles sur les modèles de communication des systèmes développés [**?**].

Du point de vue du développement logiciel, et dans une vision agile, le groupe d'étudiant(e)s progressera de manière itérative par lotissement de la mise en œuvre de la ligne de production logicielle en augmentant à chaque itération le périmètre de complexité du système : automate autonome, communication avec une interface de contrôle, communication avec un autre automate, mode dégradé, sûreté de fonctionnement, adaptation dynamique, (ordres à la volée)...

A terme ou souhaite étudier les problèmes de sécurité liées auc=x communications.

4 Projet

Groupe de 3 étudiants parcours ALMA motivés, rigoureux, intéressés par la thématique Automatisme et Génie logiciel.

Equipe d'accueil : AeLoS

Langages et environnement : Java, LeJOS, NXT, Transformation de modèles, réseaux sans fils, vérification...

Gestion de projet agile

Références

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

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

[3] Sanford Friedenthal, Alan Moore, and Rick Steiner. *A Practical Guide to SysML : Systems Modeling Language*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2008.

[4] Pascal André, Gilles Ardourel, Christian Attiogbé, and Arnaud Lanoix. Using assertions to enhance the correctness of kmelia components and their assemblies. *ENTCS*, 263 :5 – 30, 2010. Proceedings of FACS 2009.

[5] Pascal André and Alain Vailly. *Développement de logiciel avec UML2 et OCL ; cours et exercices corrigés*, volume 6 of *Collection Technosup*. Editions Ellipses, 2013. ISBN 9782729883539.

[6] Peter H. Feiler and David P. Gluch. *Model-Based Engineering with AADL : An Introduction to the SAE Architecture Analysis & Design Language*. Addison-Wesley Professional, 1st edition, 2012.

[7] C. Atkinson. *Component-based Product Line Engineering with UML*. Addison-Wesley object technology series. Addison-Wesley, 2002.

[8] Awais Rashid, Jean-Claude Royer, and Andreas Rummler. *Aspect-Oriented, Model-Driven Software Product Lines : The AMPLE Way*. Cambridge University Press, New York, NY, USA, 2011.

[9] Graeme Shanks, Elizabeth Tansley, and Ron Weber. Using ontology to validate conceptual models. *Commun. ACM*, 46(10) :85–89, October 2003.

[10] Martin Gogolla, Jarn Bohling, and Mark Richters. Validating uml and ocl models in use by automatic snapshot generation. *Software and Systems Modeling*, 4(4) :386–398, 2005.

[11] Pascal André, Jean-Marie Mottu, and Gerson Sunyé. Costotest : A tool for building and running test harness for service-based component models (demo). In *Proceedings of the 25th International Symposium on Software Testing and Analysis*, ISSTA 2016, pages 437–440, New York, NY, USA, 2016. ACM.

- `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/`