Sujet de Stage Master recherche
Vérification formelle de diagrammes UML

Christine Choppy et Kais Klai

LIPN, CNRS UMR 7030, Université Paris 13
{christine.choppy,kais.klai}@lipn.univ-paris13.fr

1 Contexte scientifique et problématique

L'évolution des systèmes distribués se caractérise par une complexité croissante et un rôle toujours plus critique. Les logiciels orchestrant de tels systèmes doivent réagir correctement et en particulier face aux situations critiques. La vérification des propriétés de ces systèmes est reconnue comme un problème difficile et se heurte à plusieurs problèmes d'ordre pratique et théorique. Un premier obstacle apparaît au niveau de la définition du langage de spécification utilisé. Si celui-ci est trop expressif, alors on ne peut pas, mathématiquement, l'analyser de manière automatique. Un second obstacle est l'explosion combinatoire des états possibles des systèmes. En effet, malgré les avancées spectaculaires de la technologie des ordinateurs, il arrive que l'on soit incapable d'analyser intégralement des systèmes par manque d'espace ou de temps. Le défi est donc de proposer des modèles qui soient suffisamment expressifs tout en restant raisonnablement analysables. Une fois l'étape de modélisation terminée, on souhaite vérifier que le système satisfait un certain nombre de propriétés. L'objectif est donc de développer des outils de vérification efficaces permettant de combattre le problème de l'explosion combinatoire de l'espace d'états du système rencontré lors de la vérification du modèle (model-checking).

En pratique, le compromis de trouver un modèle qui soit à la fois suffisamment expressif et raisonnablement analysable n'est pas facile à trouver. Les modèles de spécification utilisés dans l'industrie sont en général informels et présentent trop d'ambiguïté pour être vérifiés automatiquement. Les modèles formels ne sont pas assez expressifs et/ou théoriquement difficiles à maîtriser par un non spécialiste (un ingénieur). Des travaux existants ont essayé de réduire cet écart en proposant des associations de modèles informels (ou semi-formels) vers des modèles formels qui supportent le model-checking, effectuées de manière transparente pour l'ingénieur (voir par exemple [4,2,5,7]). Ce stage s'intègre dans cette thématique et propose l'utilisation de techniques de vérification formelle autour de la méthodologie UML (Unified Modeling Language)[6]. UML est un langage de modélisation graphique, semi-formel et largemenr répandu dans le monde industriel, il est adopté désormais comme un standard industriel. La sémantique d'UML n'étant pas complètement et formellement définie, il n'est pas possible de vérifier de manière exhaustive la sémantique des modèles.

Nous nous proposons dans ce stage de proposer une vérification formelle d'une partie de la vue dynamique de UML : les diagrammes d'états (statecharts). Les diagrammes d'états peuvent être utilisés pour spécifier le comportement de plusieurs éléments des modèles décrits en UML2.0 tels que les instances d'une classe UML.

2 Objectif du stage

L'objectif de ce stage est d'exploiter des techniques de vérification formelles afin de vérifier les diagrammes d'état issus d'une modélisation en UML2.0. Trois parties majeures seront alors étudiées et éventuellement implémentées :

1. Choisir et motiver un langage ou un modèle formel [1,3] (Réseaux de Petri, automates finis, systèmes de transition,...) cible, supportant une vérification formelle, vers lequel une traduction d'un diagramme d'état UML sera effectuée.

   Il sera demandé d'étudier la faisabilité ainsi que la pertinence d'une telle traduction.

2. Choisir une logique formelle [1,3,6] (LTL, OCL, ...) qui permettrait d'exprimer des propriétés du diagramme d'état UML suffisamment expressive et automatiquement vérifiable sur le modèle cible choisi en 1.
3. Concevoir et analyser un outil de vérification de modèle (model-checker) de diagrammes d'état UML.

Ce programme ambitieux sera encadré et conduit de manière à adapter les objectifs pour un stage de master. Il pourrait conduire à un travail de thèse.

3 Profil souhaité

Le candidat recherché devra avoir des connaissances de base sur le modèle UML, et sur la vérification de systèmes (model-checking). Il devra également maîtriser un language de programmation.

Références

1. Béatrice Bérard, Michel Bidoit, Alain Finkel, François Laroussinie, Antoine Petit, Laure Petrucci, and Philippe Schnoebelen. *Systems and Software Verification. Model-Checking Techniques and Tools.* Springer, 2001.
2. Simona Bernardi, Susanna Donatelli, and José Merseguer. From UML sequence diagrams and statecharts to analysable Petri net models. In *WOSP '02 : Proceedings of the 3rd international workshop on Software and performance*, pages 35–45, New York, NY, USA, 2002. ACM.
3. S. Haddad, F. Kordon, and L. Petrucci. *Méthodes formelles pour les systèmes répartis et coopératifs.* Hermes, October 2006.
4. T. Holvoet and P. Verbaeten. Petri charts : an alternative technique for hierarchical net construction. In *Proceedings of IEEE Conference on System, Man, and Cybernetics*, 1995.
5. Zhaoxia Hu and Sol M. Shatz. Mapping UML diagrams to a Petri net notation for system simulation. In *SEKE*, pages 213–219, 2004.
6. UML Revision Task Force. *OMG UML Specification.* see : http ://www.uml.org.
7. Shuzhen Yao and Sol M. Shatz. Consistency checking of UML dynamic models based on Petri net techniques. *CIC*, pages 289–297, 2006.