Sujet de master recherche ■ Architectures logicielles distribuées ■ 2006–2007

Vérification en B des propriétés fonctionnelles de composants Kmelia

Encadrant principal : Christian Attiogbé
courriel : `Christian.Attiogbé@lina.univ-nantes.fr`
tél. : 02 51 12 58 18

Co-encadrant(s) : Henri Habrias

Cadre du travail

Une des préoccupations de notre équipe est l'étude des mécanismes d'analyse, de spécification et de développement rigoureux de composants logiciels ayant des propriétés formellement exprimées. Dans ce cadre nous avons proposé un modèle à composants basé sur les services : le modèle Kmelia. Le modèle est assorti d'un langage de spécification ayant le même nom.
L'équipe développe à des fins d'expérimentation des travaux de recherche, un environnement nommé COSTO (*Component Study TOolbox*).

Dans l'approche Kmelia, un composant (abstrait) a un invariant ($InvC$) et offre plusieurs services. Chaque service (d'un composant) a une PREcondition ($preS$) et une POSTcondition ($postS$).

Dans le but de développer des composants sûrs, on souhaite garantir en les vérifiant, les propriétés des composants et des services. Plusieurs propriétés sont identifiées ; par exemple la composabilité (qui est la combinaison de l'interopérabilité statique et de l'interopérabilité dynamique), préservation de l'invariant, substituabilité, etc.

Nous avons étudié et développé des outils pour vérifier certaines propriétés, par exemple la composabilité. Un point méthodologique important consiste à réutiliser les résultats et les outils existants par ailleurs dans le domaine de la vérification formelle ; nous avons ainsi utilisé les Lotos/CADP et Mec pour analyser des composants et services spécifier en Kmelia.
De la même façon nous souhaitons utilisé la méthode B pour vérifier des propriétés qui peuvent l'être par la preuve de théorèmes : par exemple les propriétés fonctionnelles. Ainsi on peut par exemple, comme dans la méthode B, exiger que tout service préserve les propriétés du composant auquel il appartient (i.e. le déroulement du service ne remet pas en cause les propriétés garanties par le composant) ; une telle obligation de preuve serait : *tout service qui peut se dérouler (se termine, assure sa postcondition et) préserve les propriétés du composant.*

Ceci s'écrit en B, de façon un peu abrégée :

$$InvC \wedge preS \implies [service]InvC$$

Une autre obligation de preuve est :

$$InvC \wedge preS \implies [service]PostS$$

à condition que l'environnement structurel du composant remplisse sa part du *contrat de service.*

Objectif du stage

Etudier les *propriétés fonctionnelles* des composants et de leurs services dans le cadre du modèle Kmelia. Etudier l'expression et la transformation systématique en B de telles propriétés. Développer (en Java) un module qui s'intégrera dans la boîte à outils COSTO.

Travail à réaliser

S'imprégner rapidement du modèle Kmelia. Etudier, approfondir et développer proprement les hypothèses considérées ci-dessus (les obligations de preuve) :

- Etudier différentes façon d'expliciter les différentes propriétés fonctionnelles des composants et des services (on restera dans le cadre de la logique du premier ordre).
- Etudier et valider les différentes hypothèses : les $postS$ sont vraies ; les $POST$ garantissent l'invariant $InvC$; etc :

$$(postS \implies InvC)$$

$$(postS \implies InvC)$$

- Tout service qui peut se dérouler (se termine, assure sa postcondition et) préserve les propriétés du composant :

$$InvC \wedge preS \implies (postS \implies InvC)$$

$$InvC \wedge preS \implies (postS \wedge InvC)$$

- Exprimer de la même façon, d'autres propriétés fonctionnelles.
- A partir de spécifications Kmelia,
  - extraire les propriétés (fonctionnelles), pour engendrer les obligations de preuve ;
  - construire une machine abstraite B avec les informations necessaires pour engendrer les obligations de preuve souhaitées ;
  - fournir à l'utilisateur une machine B et des propriétés exploitables ;
  - étudier la possibilité d'utiliser directement le prouveur ou d'autres outils de démonstration ;
  - ...

Références

[1] Christian Attiogbé, Pascal André, and Gilles Ardourel. Checking Component Composability. In *5th International Symposium on Software Composition, SC'06*, volume 4089 of *LNCS*. Springer, 2006.

[2] Pascal André, Gilles Ardourel, and Christian Attiogbé. Vérification d'assemblage de composants logiciels Expérimentations avec MEC. In Michel Gourgand and Fouad Riane, editors, *6e conférence francophone de MOdélisation et SIMulation, MOSIM 2006*, pages 497–506, Rabat, Maroc, April 2006. Lavoisier.

[3] B. Meyer. The Grand Challenge of Trusted Components. In *Proceedings of 25th International Conference on Software Engineering*, pages 660–667. IEEE Computer Society, 2003.

[4] Robert Allen and David Garlan. A Formal Basis for Architectural Connection. *ACM Transactions on Software Engineering and Methodology*, 6(3) :213–249, July 1997.

[5] Jean-Raymond Abrial. *The B-Book Assigning Programs to Meanings*. Cambridge University Press, 1996. ISBN 0-521-49619-5.

[6] John Wordsworth. *Software Engineering with B*. Addison-Wesley, September 1996.