Contrôler la visibilité des aspects avec Aspectboxes

Alexandre Bergel

RMoD Team, INRIA - Lille Nord Europe - USTL - CNRS UMR 8022, Lille, France
http://www.bergel.eu

Résumé. La composition et l'intéraction des aspects est un domaine de recherche très actif. Bien que plusieurs solutions existent, telles que l'agencement des aspects et des advices, les approches proposées par des langages à aspects supposent qu'une connaissance générale des aspects soit nécessaire pour pouvoir les composer, et même ceci ne permet pas d'éviter les interactions implicites résultant d'une composition.
Cet article présente les aspectboxes, un mécanisme de visibilité pour aspects. L'unité élémentaire de visibilité est un aspectbox. Un aspectbox encapsule des définitions d'aspects. Un aspectbox peut être utilisé par d'autres aspectboxes pour aider la construction incrémentale de logiciel à base d'aspects. Une classe peut utiliser un aspectbox dans le but de bénéficier des aspects définis.

1 Introduction

La programmation par aspects définit des constructions linguistiques permettant une meilleure modularité complétant les mécanismes traditionnels d'héritage dans les langages de programmation à objets tels que Java et C++. Un aspect définit un comportement transversal à des hiérarchies de classes (Kiczales et al., 2001) ou de composition de fonctions (Dutchyn et al., 2006; Gal et al., 2001).

Un des challenges de la programmation orientée par aspects est la composition d'aspects (Douence et al., 2004; Klaeren et al., 2000; Nagy et al., 2005; Brichau et al., 2002; Tanter, 2008; Havinga et al., 2006; Marot et Wuyts, 2008). L'ordonnancement d'aspects, de points de jointure et d'advices, est couramment utilisé (Kiczales et al., 2001; Tanter, 2006). Bien que la plupart des langages à aspects offrent de tels mécanismes de composition, une mise en pratique d'un grand nombre d'aspects s'avère être une tâche complexe (Lopez-Herrejon et al., 2006). Une des principales difficultés à composer des aspects réside dans l'identification et le contrôle des interactions implicites résultant de cette composition.

En considérant un aspect comme une extension d'un système de base, apporter de multiples extensions à un même système peut engendrer des interactions involontaires, même si ces extensions ont été développées de façon indépendante. Nous pensons que la cause de cette situation est le manque d'un mécanisme de visibilité pour la programmation par aspects.

Associer une visibilité aux aspects a essentiellement deux caractéristiques :

– Lorsqu'ils ont des visibilités différentes, les aspects n'ont pas à être composés puisqu'ils ne sont pas visibles les uns des autres ;