Vers une nouvelle approche d'extraction de la logique métier d'une application orientée objet

Ismaïl Khriss*, Gino Chénard**

* Université du Québec à Rimouski
300, allée des Ursulines, C.P. 3300 Rimouski (Québec) G5L 3A1
ismail_khriss@uqar.qc.ca
** Université du Québec à Montréal
201, avenue du Président-Kennedy, Montréal (Québec) H2X 3Y7
chenard.gino@courrier.uqam.ca

Résumé. Les compagnies font face à d'énormes coûts pour maintenir leurs applications informatiques. Ces applications contiennent des connaissances corporatives importantes qui deviennent difficiles à récupérer après plusieurs années d'opération et d'évolution. Plusieurs approches ont été proposées afin d'extraire du code source des abstractions pour aider les développeurs à assimiler ces connaissances. Cependant, l'abstraction extraite par la plupart des approches combine la logique métier de l'application et son architecture. Nous proposons une nouvelle approche pour extraire le modèle d'une application orientée objet. Ce modèle est donné comme un diagramme de classes UML présentant les classes métier de l'application et leurs interrelations. Cette approche a été validée sur plusieurs systèmes écrits en Java et donne de bons résultats pour les systèmes bien structurés avec un bon style de programmation.

1 Introduction

Au cours de leur vie, les logiciels ont à évoluer. Cette évolution peut être nécessaire afin de corriger des erreurs ou pour l'adaptation à de nouvelles exigences ou technologies. Dans un contexte idéal, une application est bien documentée ce qui rend aisé sa compréhension. Malheureusement, la documentation est souvent incomplète, insuffisante ou non à jour. En l'absence d'une documentation adéquate, la maintenance logicielle devient difficile. Pour combler le manque de documentation, les développeurs passent alors un temps non négligeable à tenter de comprendre le code source du logiciel.

Une solution pour simplifier la tâche de compréhension du logiciel est de faciliter la navigation dans le code source. Des environnements de développement modernes tels que Visual Studio.Net et Eclipse offrent de telles fonctionnalités. Malheureusement, cette pratique présente des limites pour les logiciels de grande taille, car le développeur peut être enseveli sous la quantité d'information affichée. Il faut donc chercher à simplifier et à mieux cibler l'information affichée. Une abstraction intéressante pouvant être obtenue représente les règles métier d'une application. Peu d'approches de rétro-ingénierie tentent d'extraire cette abstraction. Il est difficile de différencier le code relié à la logique métier de celui relié à l'infrastructure de l'application.