Cluster Analysis of Object-Oriented Programs

A thesis presented in partial fulfilment of the requirements for the degree of

Master of Science

In

Computer Science

at Massey University, Palmerston North,

New Zealand.

Vyacheslav Yakovlev

2009

Abstract

In this thesis we present a novel approach to the analysis of dependency graphs of object-oriented programs, and we describe a tool that has been implemented for this purpose. A graph-theoretical clustering algorithm is used in order to compute the modular structure of programs. This can be used to assist software engineers to redraw component boundaries in software in order to improve the level of reuse and maintainability.

The analysis of the dependency graph of an object-oriented program is useful for assessing the quality of software design. The dependency graph can be extracted from a program using various different methods, including source code, byte code, and dynamic (behavioral) analysis. The nodes in the dependency graph are classes, members, packages and other artifacts, while the edges represent uses and extends relationships between those artifacts. Once the dependency graph has been extracted, it can be analysed in order to quantify certain characteristics of the respective program. Examples include the detection of circular dependencies and measurements of the responsibility or independence of units based on their relationships. Tools like JDepend[1] implementing these principles have become very popular in recent years.

Our work includes grouping types in dependency graphs using different clustering methods:

- Grouping into namespaces

[1] http://clarkware.com/software/JDepend.html

- Grouping into clusters using graph clustering algorithms
- Grouping into clusters using rules

The detected mismatches are candidates for refactoring.

We have developed a tool for processing dependency graphs clustering and producing results where users can outline possible design violations.

Acknowledgements

The very first thanks must be to the project supervisors Jens Dietrich and Catherine McCartin. Their input and guidance were essential for the success of this project.

We thank Kiwiplan Ltd for their involvement in the project. Kiwiplan is an innovative software development company specialising in providing a wide range of software solutions for the corrugating and packaging industry. Participation of Kiwiplan's developers brought commercial software development expertise and mentoring into the research work.

Special thanks belong to Manfred Duchrow from "Consulting & Software" in Laichingen, Germany. His Class Dependency Analyser (CDA) tool was particularly helpful for our software analysis. We thank Manfred Duchrow for presenting our research work in the software visualisation conference in September 2008.

We acknowledge Technology Fellowship (TIF) New Zealand along with Kiwiplan Ltd for sponsorship of our project and providing funding support that made this work happen.

We would like to thank Ewan Tempero from Auckland University for collecting, managing and providing to us a collection of open source Java software applications. The name of this collection is "Qualitas Corpus". This allowed us to have solid base for our analysis and evaluation of the results.

Contents

List of Figures

List of Tables