Software Refactoring at the Class Level using Clustering Techniques

Abdulaziz Alkhalid[a], Mohammad Alshayeb[b*], Sabri A. Mahmoud[b]
[a] Department of Systems and Computer Engineering
Carleton University
1125, Colonel by Drive,
Ottawa, ON, Canada, K1S 5B6
akhalid@sce.carleton.ca

[b] Information and Computer Science Department
King Fahd University of Petroleum and Minerals
Dhahran 31261, Saudi Arabia
{alshayeb, smasaad} @kfupm.edu.sa

Software becomes more and more complex as it adapts new requirements, is enhanced or is modified. Thus, the quality of the software decreases. Therefore, there is a need to reduce the software's complexity and improve its quality. Refactoring reduces software complexity and improves quality by restructuring the code into a more readable form that improves its internal structure without changing its external functionality. However, it is a challenging task and requires effort from the software designer. In this paper, we propose a method for identifying ill-structured software at the class level that provides heuristic refactoring advice to software designers in order to create balance between coupling and cohesion using pattern recognition techniques. To identify the ill-structured code we use three clustering techniques, namely, the Single Linkage algorithm (SLINK), the Complete Linkage algorithm (CLINK) and the Weighted Pair-Group Method using Arithmetic averages (WPGMA). In addition to these clustering techniques, we also use the Adaptive K-Nearest Neighbor (A-KNN) algorithm and compare its performance with the other clustering techniques. The results show that software structuring at the class level using A-KNN is superior to SLINK, CLINK and WPGMA in terms of performance and computational complexity.

Keywords: *Software refactoring; code restructuring ; clustering; coupling; cohesion;*

* Corresponding author