

Extraction and Evaluation of Software Components from Object-Oriented Artifacts

Amit Rathee and Jitender Kumar Chhabra

Computer Engg. Dept., National Institute of Technology, Kurukshetra-136119, Haryana, India

E-mail: amit1983_rathee@rediffmail.com, jitenderchhabra@gmail.com

Thesis summary

Keywords: CBSD, reusability, component extraction, software artifacts, component, JavaBeans, frequent usage patterns

Received: March 7, 2021

A doctoral thesis is summarized in this paper that focuses on strengthening the Component-Based Software Development (CBSD) approach by proposing an efficient approach for extracting and evaluating reusable software components from an Object Oriented (OO) software by utilizing its various artifacts. The carried out research work mainly consists of two main steps: (1) extracting a possible set of components by utilizing optimal software artifacts and clustering techniques; (2) identifying reusable components by evaluating the quality of different components using the proposed reusability metric suite. The carried out research work significantly helps in identifying and extracting the reusable components for the CBSD environment and the proposed metric suite helps in evaluating the quality of all components.

Povzetek: Predstavljen je povzetek doktorata na temo obdelovanja programskih komponent pri objektnem programiranju.

1 Introduction

With a fast-paced changing world, software functionalities demand continuous modification. Software reuse principles significantly help in faster development within allotted budget and CBSD is commonly used for it. The key composing unit (aka reusable unit) in the CBSD environment is called a component and it hides the complexity of its implementation behind its provides and requires interface. Such components possess larger granularity as compared to classes/ objects in object-oriented languages. Hence reusable components should be identified from existing OO legacy software systems and stored in a component library in order to use them for future development. This motivates the researchers to identify the affecting factors and develop some efficient techniques for extracting high quality components and quantifying their overall quality.

2 Methodology

The thesis deals with extracting and evaluating reusable components by utilizing soft computing techniques, efficient selection of software artifacts, and bio-inspired algorithms. It consists of five main steps as depicted in Figure-1.

1. The first step aims at analyzing dependency relations among different software elements (classes and/ or interfaces) based on optimal dependency information extracted by utilizing different software artifacts

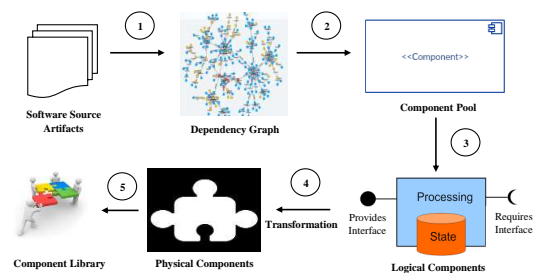


Figure 1: Proposed Research Methodology.

[1, 7, 8]. Based on the study, it was determined that combined use of structural, conceptual, and change-history based (called evolutionary) relations helps in estimating optimal dependency relations with 60% or more weight factor value assigned to evolutionary relations [2, 9]. Further, the authors determined that frequent usage patterns help in measuring more accurate structural dependency relations [3].

2. In the second step, different software elements and their dependencies are modeled as a graph and are clustered by grouping one or more strongly-connected elements into a single cluster such that each cluster is minimally connected with rest of the clusters. Different clustering algorithms are studied and best clustering algorithm is further used [5].
3. In the third step, different obtained clusters are ana-

lyzed to identify interfaces (provides and requires) of the component [4], called as the logical component.

4. The logical components are transformed into the corresponding reusable physical component by following recommendations of a Java Beans component model [6].
5. The authors further propose a set of reusability metric suite for measuring the reusability of a component and use it in the fifth step to identify high quality components for the CBSD environment based on cohesion, coupling, customizability, self-completeness, and interface complexity parameters.

3 Results

Dependency relations of software elements as well as clustering algorithms are analysed using precision, recall, F-measure and modularization metric (TurboMQ). Reusable software components are empirically identified and evaluated using some well-known IR metrics and TurboMQ metric. The results for the proposed metric suite are collected and evaluated over three different categories of software, specifically designed to have different levels of reusability. Moreover, human expertise is also considered for cross verifying the obtained reusability scores.

4 Conclusion

The thesis proposes an efficient novel approach for extracting reusable software components from existing legacy softwares. The thesis proposes a new efficient frequent usage pattern structural dependency measure approach and uses it in combination with other conceptual and evolutionary dependency relations for optimal measurement of the dependency especially useful from component point-of-view. It also proposes a novel metric suite for measuring the reusability of a software component designed as per the specifications of Java Beans. The carried out research work is able to effectively quantify various dependencies and overall quality of software components and can be used by IT companies to develop reusable component repositories.

References

- [1] Imad Eddine Araar and Hassina Seridi. Software features extraction from object-oriented source code using an overlapping clustering approach. *Informatica*, 40(2), 2016.
- [2] Amit Rathee and Jitender Kumar Chhabra. Clustering for software modularization by using structural, conceptual and evolutionary features. *Journal of Universal Computer Science*, 24(12):1731–1757, 2018a. doi: 10.1007/978-981-10-5780-9_9.
- [3] Amit Rathee and Jitender Kumar Chhabra. Improving cohesion of a software system by performing usage pattern based clustering. *Procedia Computer Science*, 125:740–746, 2018b. doi: <https://doi.org/10.1016/j.procs.2017.12.095>.
- [4] Amit Rathee and Jitender Kumar Chhabra. Reusability in multimedia softwares using structural and lexical dependencies. *Multimedia Tools and Applications*, 78(14):20065–20086, 2019a. doi: <https://doi.org/10.1007/s11042-019-7382-1>.
- [5] Amit Rathee and Jitender Kumar Chhabra. A multi-objective search based approach to identify reusable software components. *Journal of Computer Languages*, 52:26–43, 2019b. doi: <https://doi.org/10.1016/j.cola.2019.01.006>.
- [6] Amit Rathee and Jitender Kumar Chhabra. Mining reusable software components from object-oriented source code using discrete pso and modeling them as java beans. *Information Systems Frontiers*, 22(6):1519–1537, 2020. doi: <https://doi.org/10.1007/s10796-019-09948-4>.
- [7] Cherait, Hanene, and Nora Bounour. "History-based approach for detecting modularity defects in aspect oriented software." *Informatica* 39.2 (2015).
- [8] Amit Rathee, and Jitender Kumar Chhabra. "Software modularization by estimating structural and conceptual relations among classes and using hierarchical clustering." *International Conference on Advanced Informatics for Computing Research*. Springer, Singapore, 2017. doi: 10.1007/978-981-10-5780-9_9
- [9] Amit Rathee and Jitender Kumar Chhabra. "Sensitivity Analysis of Evolutionary Algorithm for Software Reusability." *MENDEL*. Vol. 25. No. 1. 2019. doi: <https://doi.org/10.13164/mendel.2019.1.031>