MDT: Tool Support for Measuring Logical Coupling

Suntae Kim[1], Bingu Shim[2]

[1]Dept. of Computer Engineering, Kangwon National University,
Sam-Cheok Si, Kangwon Do, South Korea
{stkim}@kangwon.ac.kr
[2]Software Engineering Expert Group, Seoul, South Korea,
{lanore}@swexpertgroup.com

1 Introduction

Coupling is a key concept that represents the extend of inter-relationships between software entities[7]. It is used to comprehend a software system and investigate refactoring points to improve modifiability of the system[3]. Logical coupling suggested by *Gall* denotes a concept to quantify the level of relationships between software entities in the evolutionary perspective by utilizing revision history in the software repository[4]. In order to measure logical coupling, it is essential to provide an appropriate measurement method, and develop an automatic tool to support the method.

Some researchers have attempted to address this issue (see [2][4][8]). The suggested approaches for measuring logical coupling between the fine-grained software entities such as methods and attributes of a class. However, most of the approaches are only based on frequency of co-changes from version archives so that the accuracy of the logical coupling is relatively low.

In our previous work [6], we proposed multi-dimensional approach to measuring logical coupling from revision history. In the approach, time, volume and expertise factors for each transaction are computed at first. Then, logical coupling between software entities(e.g., classes and methods) are derived from calculating support and confidence. In this paper, we present tool support called MDT(Multi-Dimensional logical coupling measurement Tool) for our approach. The tool enables identifying transactions from the revision history, computing three factors and logical coupling between software entities. It shows logical coupling entities and the extent of coupling in the tool when a user selects a specific software entity. In this paper, we presented MDT and its software architecture.

2 Approach Overview for Measuring Logical Coupling

This section summarizes our proposed approach to measuring logical coupling based on time, volume and expertise factors of transactions[6]. It is composed of three steps. As a first step, it starts with identifying transactions from the

ICCA 2013, ASTL Vol. 24, pp. 24 - 27, 2013

VCS(Version Control System)[1] by grouping revisions that exist in the time-window. Transaction is a set of resource revisions occurred by a developer within designated time-windows. We adopted the sliding time-windows proposed by Zimmermann *et al.*[8] that identifies consecutive revisions in a sliding window as a transaction.

As a second step, time, volume and expertise factors are computed for each transaction. *Transaction time* factor is a significance factor that reflects a time which the transaction is occurred. As the transaction time is recent, the factor becomes high. *Transaction volume* factor is a significance factor for volume, indicating that the entities in a small size of a transaction are strongly believed as more tightly coupled. *Expertise* factor shows the level of ownership of the code. As the developer has more ownership for the entities in the transaction, the transaction takes more high value.

As the last step, the logical coupling is calculated based on the three factors by applying the data mining techniques. It is calculated with *support* indicating the number of transactions containing two entities A and B together, and *confidence* presenting the proportion of existence of entity B within all transactions containing the entity A. In order to compute support and confidence for an entity e, transactions containing e and all entities in the transaction are extracted as candidate logically coupled entities with e. In this step, transactions containing factors less than 0.1 are filtered out. Then, *support* and *confidence* are computed according to their definition. We can say that as values of support and confidence are high, the entities are highly logically coupled with the entity e.

3 Tool Support: MDT

We have developed tool support for the proposed approach, named MDT(Multi-Dimensional logical coupling measurement Tool) as shown in Fig. 1. MDT is developed as an Eclipse plug-in that retrieves revision history from code repository, annotates three factors, and calculates logical coupling between software entities according to the aforementioned process. The coupling information is shown in *Logical Coupling* view when a user selects a specific software entity in the *Project Explore* view.

MDT is composed of several components as shown in Fig. 2. *Transaction Identifier*, *Factor Attach* and *Logical Coupling Calculator* components are in charge of covering major steps for measuring logical coupling. *CVS Repository Adapter* connects to CVS code repository and provides revision information by parsing CVS specific log data. Also, *Repository Adapter* stores the information in a local data base to provide raw data to compute logical coupling without opening new network connections so that it dramatically decrease computing time. We used HSQL(Hyper SQL Database)[5] to improve performance because it offers local in-memory database engine and it does not need to install so that it is easy to distribute with MDT. *Logical Coupling Calculator* provides an external interface to other applications for logical coupling information between software

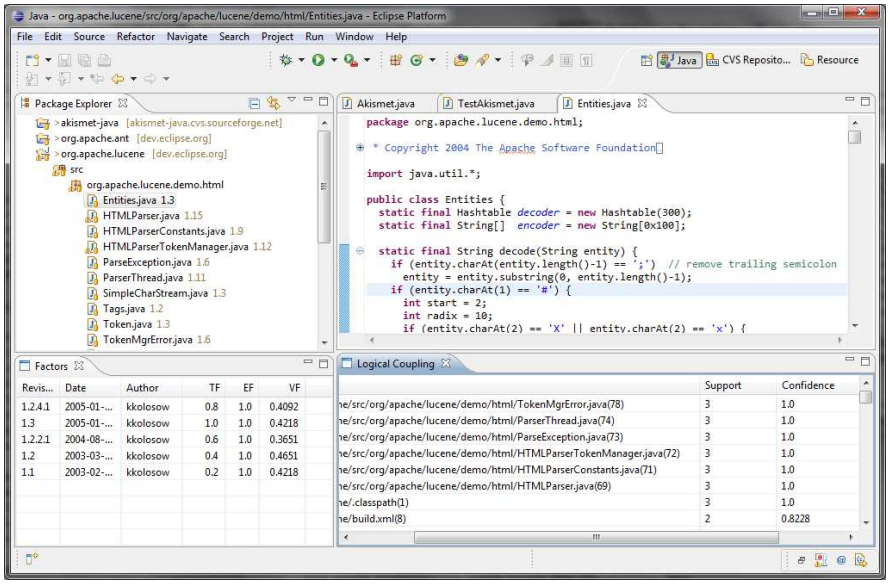

Fig. 1: MDT - Screen Capture

entities. By utilizing the information, we are planning to build *Aspect Identifier* as well as *Refactoring Point Recommender*.

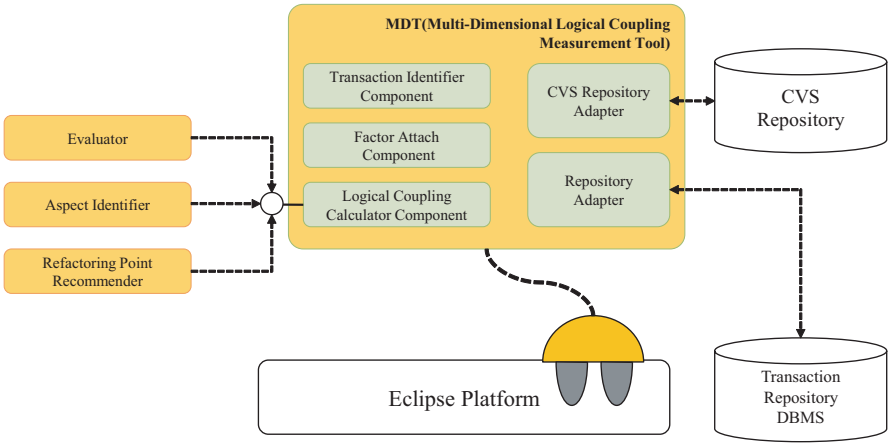

Fig. 2: Software Architecture of MDT

4 Conclusion and Future Work

In this paper, we have presented tool support named MDT for multi-dimensional approach to measuring logical coupling between software entities. MDT has been developed as an Eclipse plug-in. Due to the space limit, validation of the proposed approach and tool application could not be described. As future work, we have a plan to apply MDT into several open source java projects and show feasibility of the proposed approach.

References

1. Concurrent Versions System. http://savannah.nongnu.org/projects/cvs.
2. eROSE, Reengineering of Software Evolution in Eclipse. http://www.st.cs.uni-saarland.de/softevo/erose/download.php.
3. M. Fowler, K. Beck, J. Brant, W. Opdyke, and D. Roberts. *Refactoring: Improving the Design of Existing Code*. Addison Wesley, 1999.
4. H. Gall, K. Hajek, and M. Jazayeri. Detection of Logical Coupling Based on Product Release History. In *Proceedings of International Conference on Software Maintenance*, pages 190–197, Bethesda, Maryland, 1998.
5. HyperSQL. http://www.hsqldb.org/.
6. S. Kim, B. Shim, and K. Jin. Logical Coupling in Multi-Dimensional Aspects. In *Proceedings of International Confernce on Information Technology and Computer Science*, page To be published, Beijing, China, 2013.
7. I. Sommerville. *Software Engineering 8th Edition*. Addison Wesley, 2007.
8. T. Zimmermann, P. Weigerber, and A. Zeller S. Diehl. Mining Version Histories to Guide Software Changes. *IEEE Transactions on Software Engineering*, 31(6):429–445, 2005.