Jeong-Jin Kang
Edward J. Rothwell
Nguyen Hang Dinh
Nguyen Thanh Thuy
Gyoo Seok Choi
Nguyen Ha Nam

Advanced and Applied Convergence Letters AACL 07

Advanced and Applied Convergence

2nd International Joint Conference, IJCC 2016
Hanoi, Vietnam, January 18-22 2016
Revised Selected Papers

IIBC
IPACT

An Automatic Visualization Mechanism of extracting Diverse Unit Level for Software Complexity

Junsun Hwang[1], Woo Sung Jang[2], R. Youngchul Kim[3]

[1,2,3]*SE Lab, Dept. of CIC(Computer Information Communication), Hongik University*
[1]*hwang@selab.hongik.ac.kr,* [2]*jang@selab.hongik.ac.kr,* [3]*bob@hongik.ac.kr*

Abstract

Nowadays the importance of software emphasized, it is difficult for our domestic smaller firms to perform the proper software quality control under their developing environment due to invisibility and increasing complexity of software. In this paper, to solve this problem, we propose the method how to visualize overall architecture of source code in order to make software quality control easier. The previous research applied and extended NIPA's method of software visualization, but the developer did not extract his/her desired module in a direct selecting manner when extracting the component applied the coupling. Therefore, the more the software complexity increases, the more the source code architecture complicates. So this coupling analysis between components is very difficult. To solve this problem, we show how to select the only module desired by the developer, or to select several modules with diverse component levels and extract each coupling for an easier analysis of the source code architecture.

Keywords: *The coupling graph; SW Visualization; the Size level of components; coupling; SW Quality;*

1. Introduction

In the global market, the market size of software rapidly grows and is 30% of the entire IT industry to $ 1 trillion. As the IT convergence is in progress, the roles and functions of software are becoming increasingly important in the final product. The importance of software is emphasized, but our domestic smaller firms have some difficulties in proper control of software quality under their developing environment due to invisibility and increasing complexity of software [1].

As referred to this paper, NIPA's method of software visualization is a method of visualizing source code of software in order to solve the aforementioned problems [1]. In this paper, the developer is to extract the coupling by applying and extending to the previous research [2] in which the developer extracts the Call Graph between the module selected in diverse component levels. The previous research [3], which is extracting the coupling, has some problems of not easily figuring out the coupling between modules, because no module is directly selected and all modules are extracted which incurred complexity of source code architecture in case of complicated software. Also it failed to extract the module to define the diverse components, and is not easy to figure out the coupling between modules at the diverse component level. To solve this problem, the developer chooses the desired level of various components on the module of a web page and extracts the coupling between the selected modules, and then easily analyzes the source code architecture.

The paper is organized as follows: Chapter 2 describes an extracting method on coupling through software visualization techniques, Chapter 3 shows the developer how to extract coupling between modules selected from diverse component level, Chapter 4 refers to conclusion and future research.

2. Related Works

J.J. Kang et al. (Eds.): IJCC 2016, AACL 07, pp. 155~159, 2016

The method of NIPA's software visualization is to visualize source code and its process for the high quality of software [1]. Source code is intended for visualization to overcome the software invisibility, Tool-Chain should be composed of open source tool of Source Navigator [4], Graphviz Dot Script [5], and SQLite [6].

Step 1 – Insert and analyze the target system in source navigator [3].

Step 2 – Extract SNDB file stored analyzed information in binary value and insert SNDB file in DBdump.exe for conversion to text [3].

Step 3 – Insert text information converted through DBdump.exe in SQLite DB table [3].

Step 4 – Generate Dot script to extract coupling using query statement with getting coupling score [3].

Step 5 – Generate the graph with inserting Dot script in Dot program. This graph is the source code architecture as applied coupling [3].

In software engineering, we suggest to minimize coupling and maximize cohesion to make the high quality of the software [7]. Coupling means interdependency of modules. The lower coupling is, the more it is independent: and it is easy to reuse and maintain. But the higher coupling is, the less it is independent: and it is difficult to reuse and maintain. The type of Coupling is all six such as Data, Stamp, Control, External, Common, Content. The closer it approaches data coupling, the more it is becoming independent for higher software quality, and the closer it approaches content coupling, the less it is becoming independent for lower software quality. Therefore, the developer is recommended refactoring to come into Data Coupling, if there is Content Coupling [3].

3. Extracting method of Coupling between modules in diverse component level

The developer can select the type of coupling in a web page by the checkbox as ① in Figure 1. By using php as ② in Figure 1, if data coupling is selected, add a hexadecimal number 0x20, and if stamp coupling is

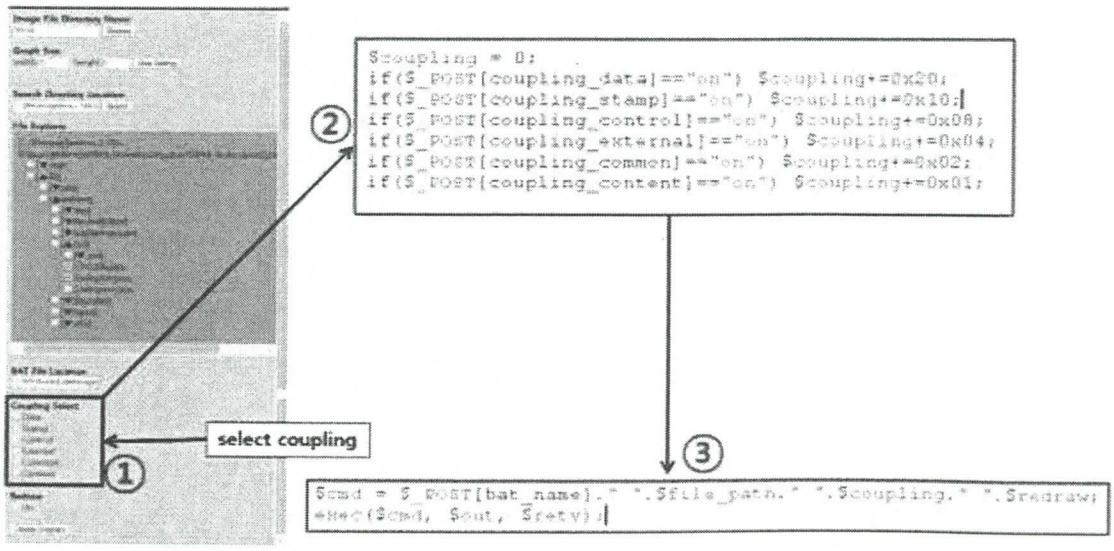

Figure 1. Process to convey coupling variable selected in Web Page

selected, add 0x10 to the variable coupling. If the control coupling is selected, add 0x08, and if the external

coupling is selected, add 0x04, and if the common coupling is selected, add 0x02, and if the content coupling is selected, add to 0x01.

This added variable coupling as ③ in the Figure 1 is conveyed as a parameter of the batch file. A parameter of the batch file is being sent to the parameters of Tool-Chain, and then this processes the data.

Next, a new query statement was prepared without using the regular DB formula. It was used two types of query to define the module in the diverse component level. The first case is the module in a class unit. If the module is selected in a class unit, input the NAME column data of type table in CLASS_NAME column of MODULE_MAPPING table, and input 'null' in PACKAGE_NAME TYPE column, as shown in the insert statement of Figure 2. And input 'file' in MODULE_TYPE column so as to state that the module is class in the java file.

```
module_mapping_creation = "insert into MODULE_MAPPING "
 +"select distinct '"+name+ "' as MODULE_NAME, TYPE.NAME as CLASS_NAME, 'null' as PACKAGE_NAME, 'file' as MODULE_TYPE, '"
 +architecture_name +"' as ARCHITECTURE_NAME, '" + project_name + "' as PROJECT_NAME from TYPE "
 + "where TYPE.PROJECT_NAME = '" + project_name +"' "
 + "and TYPE.ANNOTATE='class' "
 + "and not TYPE.ANNOTATE='method' "
 + "and TYPE.NAME = '"+name+"' "
 + "and TYPE.PACKAGE_PATH='"+module_path+"'";
```

Figure 2. 'Insert' statement if the module is in a class unit

The second case is the module in a package unit. If the module is selected in a package unit, input the java name NAME that belongs in the package in CLASS_NAME column of MODULE_MAPPING table, and input the name of the package which is selected in PACKAGE_NAME, as shown in the insert statement of Figure 3. And input 'package' in MODULE_TYPE column so as to state that the module is java package.

```
module_mapping_creation = "insert into MODULE_MAPPING "
 +"select distinct '"+name+ "' as MODULE_NAME, '"+file_name+"' as CLASS_NAME, '"+package_name+"' as PACKAGE_NAME, 'package' as MODULE_TYPE, '"
 +architecture_name +"' as ARCHITECTURE_NAME, '" + project_name + "' as PROJECT_NAME from TYPE "
 + "where TYPE.PROJECT_NAME = '" + project_name +"' "
 + "and TYPE.ANNOTATE='class' "
 + "and not TYPE.ANNOTATE='method' "
 + "and TYPE.NAME = '"+file_name+"' "
 + "and TYPE.PACKAGE_PATH='"+module_path+"'";
```

Figure 3. 'Insert' statement if the module is in a package unit

Next, the changed part is to extract the coupling scores between modules. A query was changed to the four cases in order to extract the scores of coupling between modules, which had different component levels.

```
query = "select Caller_class, Callee_class, sum(data),sum(stamp),sum(control),sum(external),sum(common),sum(content) "
 + "from Coupling as CP "
 + "where CP.Caller_package='null' "
 + "and CP.Callee_package='null' "
 + "group by Caller_class, Callee_class";
```

Figure 4. 'Select' statement in case a module reference and the modules that are referenced are all 'class'

The first is to seek a coupling if a module reference and the module that is referenced are all 'class'. To do this, input the condition in case that all a package reference and the package that is referenced are 'null', in coupling table which the information on coupling between modules was entered. And a query statement is being prepared in order to group a class reference and the class that is referenced, by grouping Caller_class and Callee_class. A query statement is shown in Figure 4.

```
query2 = "select Caller_package, Callee_class, sum(data),sum(stamp),sum(control),sum(external),sum(common),sum(content) "
 + "from Coupling as CP "
 + "where not CP.Caller_package='null' "
 + "and CP.Callee_package='null' "
 + "group by Caller_package, Callee_class";
```

Figure 5. 'Select' statement in case of a module reference in a package unit

The second is to seek a coupling if a module reference is in a package unit. To do this, input the condition in case that a package reference is not 'null' and the package that is referenced is 'null', in coupling table. And a query statement is being prepared in order to group a package reference and the class that is referenced, by grouping Caller_class and Callee_class. A query statement is shown in Figure 5.

```
query3 = "select Caller_class, Callee_package, sum(data),sum(stamp),sum(control),sum(external),sum(common),sum(content) "
 + "from Coupling as CP "
 + "where CP.Caller_package='null' "
 + "and not CP.Callee_package='null' "
 + "group by Caller_class, Callee_package";
```

Figure 6. 'Select' statement in case of the module that is referenced in a package unit

The third is to seek a coupling if the module that is referenced is in a package unit. To do this, input the condition in case that a package reference is 'null' and the package that is referenced is not 'null', in coupling table. And a query statement is being prepared in order to group a class reference and the package that is referenced, by grouping Caller_class and Callee_class. A query statement is shown in Figure 6.

```
query4 = "select Caller_package, Callee_package, sum(data),sum(stamp),sum(control),sum(external),sum(common),sum(content) "
 + "from Coupling as CP "
 + "where not CP.Caller_package='null' "
 + "and not CP.Callee_package='null' "
 + "and not CP.Caller_package=CP.Callee_package "
 + "group by Caller_package, Callee_package";
```

Figure 7. 'Select' statement in case a module reference and the module that is referenced are all 'package'

The fourth is to seek a coupling if a module reference and the module that is referenced are all 'package'. To do this, input the condition in case that all a package reference and the package that is referenced are not 'null', in coupling table. And also input the condition in case that a package reference and the package that is referenced are different. And a query statement is being prepared in order to group a package reference and the package that is referenced. A query statement is shown in Figure 7.

4. Conclusion

We propose to perform an easier Quality Control of Software by restoring the source code architecture via software visualization. In this paper, we solve the problem that the developer faced with a difficulty in analysis, because the previous source code architecture applied coupling was expressed in a complicated manner in case of restoring. It becomes more helpful for the developer to select the module what he/she wants from diverse component level, and to have an easier analysis and refactoring of source code architecture. Future research will be studied as a method of automatically extracting module in the subsystem.

Acknowledgement

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2013R1A1A2011601) and the Human Resource Training Program for Regional Innovation and Creativity through the Ministry of Education and National Research Foundation of Korea (NRF-2015H1C1A1035548).

Reference

[1] NIPA, *SW Quality Management Manual(SW Visualization)*, 2013.
[2] JunSun Hwang, R. Youngchul Kim, SangEun Lee, "A Guideline for Realization on extracting automatic size maturity level of diverse component via Source Codes," The 5th International Conference on Convergence Technology(ICCT) 2015, Vol. 5, No. 1, pp. 268–269, June 2015.
[3] Geon-Hee Kang, R. Young Chul Kim, Geun Sang Yi, Young Soo Kim, Yong. B. Park, Hyun Seung Son, "A Practical Study on Code Static Analysis through Open Source based Tool Chains," KIISE Transactions on Computing Practices, Vol. 21, No. 2, pp. 148–153, February 2015.
[4] SN Group, Source Navigator User Guide, http://sourcenav.sourceforge.net/
[5] GV team, GraphViz User's Guide, http://www.graphviz.org/Documentation.php
[6] SQLite., About SQLite, http://www.sqlite.org/about.html
[7] Roger S, Software Engineering: A Practitioner's Approach, 7th Ed., Pressman, 2009.

ISSN 2288-6060

Advanced and Applied Convergence Letters

The AACL series is committed to the publication of proceedings of Advanced and Applied Convergence. Its objective is to publish original researches in various areas of Smart Convergence. This will provide good chances for academia and industry professionals as well as practitioners to share their ideas, problems and solutions relating to the multifaceted aspects.

Research papers were strictly peer-reviewed by program committees to make sure that the papers accepted were high quality and relevant to the current and future issues and trends in Advanced and Applied Smart Convergence.

The scope of AACL includes the entire area of advanced and applied convergence from the current and future trends. The language of publication is English.