ICCT 2015

"The 5th International Conference on Convergence Technology 2015"

Vol. 5 No. 1

Date: June 29 – July 2, 2015
Place: Chateraise Gateaux Kingdom Sapporo Hotel, Hokkaido, Japan
Co–organized by:
- Korea Convergence Society
- Korea Institute of Science and Technology Information
- The Korean Association for Comparative Government
- The Society of Digital Policy & Management
- Convergence Society for SMB
- Konyang Univ. Well–Dying LAB
- Korea Mobile Enterprise Promotion Association
- DAEHAN Society of Industrial Management

Sponsored by:

KOREA CONVERGENCE SOCIETY
http://www.kcons.or.kr
Session 5-C (SW Visualization (SE Center))

- 13:00-14:20

Tuesday June 30, 2015

1. W-13-01_Hybrid Visual Security Analysis to Prove Vulnerability for Safety / 262
   Seok Mo Kim(Dankook Univ., Korea), Sang Eun Lee(Software Engineering Center, Korea), Su Nam Jeon(Software Engineering Center, Korea), Young B. Park(Dankook Univ., Korea)

2. W-13-02_Compatibility Enhancing Agent for Managing Software Toolchain / 264
   Eun Seung Lee(Dankook Univ., Korea), Young Soo Kim(Software Engineering Center, Korea), Byungho Park(Hongik Univ., Korea), Young B. Park(Dankook Univ., Korea)

3. W-13-03_A Visualized Blocking Method against a Hidden Malware in the Image / 266
   Byungho Park(Ministry of National Defence, Korea), R. Young Chul Kim(Hongik Univ., Korea), Young B. Park(Dankook Univ., Korea), Daeho Shin(Korea e-Government Exp. Associations, Korea), Young Soo Kim(NIPA, Korea), Sang Eun Lee(NIPA, Korea)

4. W-13-04_A Guideline for Realization on extracting automatic size maturity level of diverse component via Source Codes / 268
   Jun Sun Hwang(Hongik Univ., Korea), R. Youngchul Kim(Hongik Univ., Korea), Sang Eun Lee(NIPA, Korea)

   Jun Cho(Gangneung-Wonju National Univ., Korea), Kihyun Kim(Gangneung-Wonju National Univ., Korea), Jinhyung Park(Gangneung-Wonju National Univ., Korea), Sungjin Cho(Gangneung-Wonju National Univ., Korea), Byungkook Jeon(Gangneung-Wonju National Univ., Korea), Sungkuk Cho(Gangneung-Wonju National Univ., Korea)

   Kihyun Kim(Gangneung-Wonju National Univ., Korea), Jun Cho(Gangneung-Wonju National Univ., Korea), Jinhyung Park(Gangneung-Wonju National Univ., Korea), Sungjin Cho(Gangneung-Wonju National Univ., Korea), Byungkook Jeon(Gangneung-Wonju National Univ., Korea), Sungkuk Cho(Gangneung-Wonju National Univ., Korea)

7. W-13-07_Extracting Designs via Code on Reverse Engineering / 274
   Haeun Kwon(Hongik Univ., Korea), Bokyung Park(Hongik Univ., Korea), R. Youngchul Kim(Hongik Univ., Korea), Sang Eun Lee(NIPA, Korea)

8. W-13-08_Extracting performance factors against performance degradation through Code Visualization / 276
   Jeon-Hee Kang(Hongik Univ., Korea), R. Young Chul Kim(Hongik Univ., Korea), Sang Eun Lee(NIPA, Korea), Su Nam Jeon(NIPA, Korea)
A Guideline for Realization on extracting automatic size maturity level of diverse component via Source Codes

JunSun Hwang, *R. Youngchul Kim, +SangEun Lee

SE Lab, Dept. of CIC(Computer Information Communication), Hongik University,
Sejong City, Korea, {hwang, bob}@selab.hongik.ac.kr

National IT Industry Promotion Agency, Korea, selee@nipa.kr

Abstract

The previous research cannot extract module (class, component, package) whatever the developer wants [1]. Therefore, in one case of SW complexity, developer is very difficult to read the graph with relationship among the calling & called modules. To solve this problem, we propose a method to create the graph with what he/she selects any modules, which does easily analyze the components within the inner code. In this paper, it shows a guideline how to realize this method.

Keywords: The calling & called graph, Nipa’s Visualization, the Size level of components

1. Introduction

SW Visualization has a goal of enhancement for SW quality against SW non-visualization [2]. This approach easily chooses to any modules, and represent the relationship among them. The previous approach didn’t handle directly to choose any module. Therefore it did just show to the whole calling & called modules of the original code, which very difficult to read the very complex graph [1]. In addition, they didn’t extract any size maturity level of components without defining diverse components. To solve this problem, we define the process mechanism like figure 1. With this process, we easily analyze the graph extracted with modules.

2. The Extraction Method to relationship among selected modules extract models

2.1 Step 1 - Upload to Visual SVN Server

For Configuration management, it should save source code into Visual SVN Server [2].

To do it, it uploads the visualizing target system to SVN Repository after connecting Eclipse with SVN. Figure 1 shows Process mechanism.

2.2 Step 2 - Save to Jenkins workspace

It saves project file which is uploaded in SVN to Jenkins workspace generated through automatically building Jenkins [2].

2.3 Step 3 - select Modules to extract

It outputs directory to push a search button like Figure 2 after inputting the path of target system stored in Jenkins workspace on web page.

After selecting modules in order to extract from the outputted directory, it pushes a button to create a diagram like figure 2. Then it executes a batch file in BAT File Location. The selected modules is stored into dm_list.txt of the server.

2.4 Step 4 - Analyze the target system

To execute batch file, we put target system located in Jenkins workspace into Source Navigator (SN), and then analyze them. The analyzed information is stored in SNDB file as the binary one.

2.5 Step 5 - Parsing SNDB file
4. Conclusion

Developer is very difficult to read the graph with relationship among the calling & called modules. To solve this problem, we propose a method to create the graph with what he/she selects any modules, which does easily analyze the components within the inner code. In this paper, it shows a guideline how to realize this method. SW Visualization has a goal of enhancement for SW quality against SW non-visualization [2]. This approach easily chooses to any modules, and represent the relationship among them. With this process, we easily analyze the graph extracted with modules.

Acknowledgments 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 Research and Development Service through the Telecommunications Technology Association (TTA) funded by the National IT Industry Promotion Agency (NIPA).

References