ISSN 2287-3252

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 :

KISTI NTIS Division & GSDC Center, LG Hitachi Co., Ltd, ALLforLAND Co., Ltd, KYUNGBONG Co., Ltd, SOFTITECH Co., Ltd, TAEJIN Infortech Co., Ltd, Geo Matics Co., Ltd, MIJU C&D Co., Ltd, R2soft Co., Ltd, Hanbit Academy, Inc., Korea IT Consulting Co., Ltd, Open Link System Co., Ltd, SungWon-IT Co., Ltd, SJ info&communications Co., Ltd, Neighbor system Co., Ltd, Able IT Co., Ltd, INFORMADE Co., Ltd, SelimTSG Co. Ltd, KORNEC Co., Ltd, MTData Co., Ltd, Mobile Law Co., Ltd, DELTASYSTEM Co., Ltd, Daewoo Information Systems Co., Ltd, LG CNS Co., Ltd, ICTWAY Co., Ltd, GFT Co., Ltd, QbizOn Co., Ltd, NANUS Information Co., Ltd, Hankyung I-NET Co., Ltd, VITZROSYS Co., Ltd, Duplex Co., Ltd, Maverick Systems Co., Ltd, Bizmerce Co., Ltd, DAWON ICT Co., Ltd, INNOZIUM Co., Ltd, MetaBiz Co., Ltd, Human Information Co., Ltd, AtechIns Co., Ltd, Comtec System Co., Ltd

KCS KOREA CONVERGENCE SOCIETY
http://www.kcons.or.kr

Session 5-D (SW Visualization (SE Center)) Chair Seung Yeob Yu(Namseoul Univ.)

● 14:30~15:50 Tuesday June 30, 2015

01. W-07-06_Computer Simulation on HPDC Process by Filling and Solidification Analysis / 360
Tae-Hoon Yoon(Namseoul Univ., Korea), Hong-Kyu Kwon(Namseoul Univ., Korea)

02. W-13-09_Extracting Software Architecture based on Reverse Engineering / 362
Woo Sung Jang(Hongik Univ., Korea), Chae Yun SEO(Hongik Univ., Korea), R. Young Chul Kim(Hongik Univ., Korea), Woo Yeol Kim(Daegu National Univ. of Education, Korea), Young Soo Kim(NIPA, Korea)

03. W-13-10_Internal Code Visualization for Analyzing Code Complexity / 364
So Young Moon(Hongik Univ., Korea), Sang Eun Lee(NIPA, Korea), R. Youngchul Kim(Hongik Univ., Korea)

04. W-13-11_Replacing Source Navigator with Abstract Syntax Tree Metamodel (ASTM) on the open source oriented tool chains for SW Visualization / 366
Hyun Seung Son(Hongik Univ., Korea), So Young Moon(Hongik Univ., Korea), R. Young Chul Kim(Hongik Univ., Korea), Sang Eun Lee(NIPA, Korea)

05. W-13-12_Requirement Tracking Visualization for Validating Requirement Satisfaction / 368
Bokyung Park(Hongik Univ., Korea), Haeun Kwon(Hongik Univ., Korea), Young Soo Kim(NIPA, Korea), R. Young Chul Kim(Hongik Univ., Korea)

06. W-13-13_Mobile Based Testing with Code Visualization / 370
Keunsang Yi(Hongik Univ., Korea), Hyeoseok Yang(Hongik Univ., Korea), R. Young Chul Kim(Hongik Univ., Korea)

07. W-33-06_Content Analysis of Green Advertisements in Korea / 372
Mi-Jeong Kim(Hanyang Univ., Korea), Sangpil Han(Hanyang Univ., Korea)

08. W-33-09_Online Public Opinion Dissonance between Korean and Chinese Netizens: its Causes, Functions and Solutions / 374
JiHye Lee(Namseoul Univ., Korea), SeungYeobYu(Namseoul Univ., Korea)

The 5th International Conference on Convergence Technology 2015
June 29 - July 2, 2015, Chateraise Gateaux Kingdom Sapporo Hotel, Hokkaido, Japan

Replacing Source Navigator with Abstract Syntax Tree Metamodel (ASTM) on the open source oriented tool chains for SW Visualization

[1]Hyun Seung Son, [2]So Young Moon, *[3]R. Young Chul Kim, [4]Sang Eun Lee
[1,2,*Corresponding Author3] *Hongik University, Korea, {son, msy, bob*}@selab.hongik.ac.kr*
[4] *NIPA, Korea, selee@nipa.kr*

Abstract In the previous approaches, we constructed a tool chains based on open source such as source navigator, SQLite, DOT, which can visualize source code to check code complexity. But we can't customize to get more data from the too-chain. At this time, we replace source navigator with ASTM on it. Then we suggest a whole procedure for SW visualization with the ASTM. Actually AST has a role to analyze the expressions of functions and classes, and also the definition and declaration of variables through static analysis of the program code with the AST. But the existing ASTs are not compatible with other AST due on the specific parser. For this reason, we implemented OMG's standard *Abstract Syntax Tree Metamodel* (ASTM), which defined metamodel of the AST within any compiler. That is, we can represent diverse programing languages with just an ASTM.

Keywords: *Abstract Syntax Tree (AST), Meta-model, Visualization, Reverse Engineering*

1. Introduction

Most companies and ventures develop the software code without any design due on time and cost. They just release SW product quickly, but may spend more cost at the maintenance stage. This approach may be the low quality of SW product. Therefore, the companies need to show inside of the developing code for the SW visualization. The visualization is able to trace requirements from a program code through reverse engineering [1]. For the SW visualization, diverse tools are required such as *Source Navigator* [2], *Graphviz* [3], and a parser. The parser generates *Abstract Syntax Tree* (AST) during compiling the program code. But the existing ASTs are not compatible with other AST due on the specific parser. Figure 1 shows the previous tool chain mechanism for Nipa's SW visualization. But it is hard to customize Source Navigation because of Open source in A* in figure 1. We are deeply considering to make our own parser for whatever we analyze.

Therefore, Industry companies defines OMG's standard named *Abstract Syntax Tree Metamodel (ASTM)* [4], which is metamodel of abstract syntax tree with the existing compiler. The main purpose of the ASTM easily exchanges the metadata repository between the software in such as software modernization, platforms, and distributed heterogeneous environment. The ASTM consists of the defined elements to represent the AST from the existing programming languages such as *C, C++, C#, Java, Ada, VB/.Net, COBOL, FORTRAN, Jovial,* and so on.

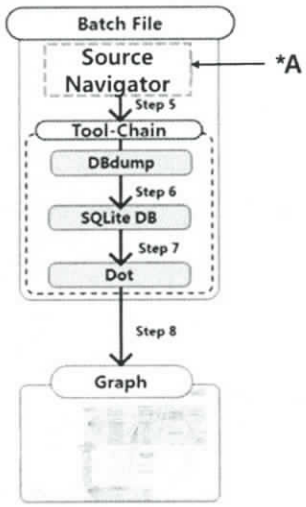

Figure 1. The previous tool chain mechanism [1]

But OMG's ASTM has defined and complicated with 193 elements of metamodel, but just specifications without any implementation.

2. SW Visualization Mechanism

For SW visualization mechanism, it is required that 1) the parser generates the abstract syntax tree and 2) the visualizer needs to generate a graph. The figure 2 shows B* area of a whole structure for SW visualization dislike A* part in figure 1. The parser generates ASTM from a program code such as *C, C++,* or *Java.* The visualizer generates the graph from the

ASTM. Through this process, we can extract requirements via design from the program code. We will use the existing parsers as *C/C++ Development Tooling* (CDT) [5] and *Java Development Tools* (JDT) [6]. The CDT is a tool in *Eclipse platform* to develop *C/C++* application. It supports to create the project, to build the program, to edit the *C/C++* code, to analyze the static code, and to debug & refactor functions. The JDT is a tool to develop Java application. It supports the same function like the CDT.

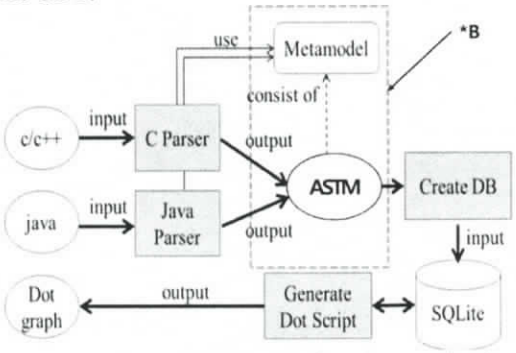

Figure 2. The new SW visualization mechanism

3. A Case study with new mechanism

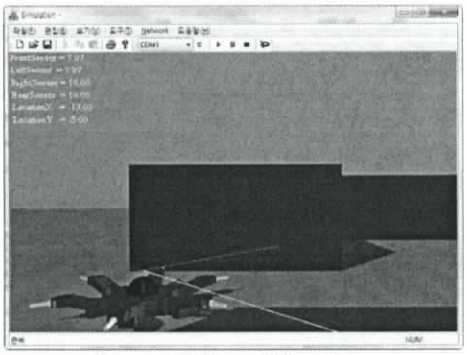

Figure 3. Target Simulator

We use target simulator (500 KLOC) into New SW visualization mechanism. Figure 3 shows target simulator for robot Modeling& simulation development tool. We apply this program to analyze how to check code complexity. After inserting this code in Figure 2, we get a graph to have a relationship among classes and modules like figure 4. We also get the calling & called graph to recognize some complexity parts of source code. Therefore, we can easily do fixing bad code or refactoring for improving code. Figure 4 shows the analyzed result of the program code, that is, robot Modeling & & simulation.

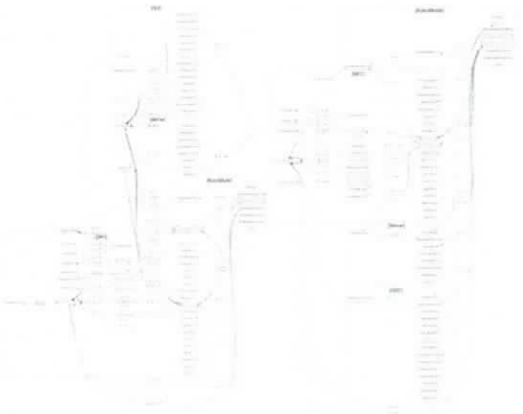

Figure 4. The analyzed result of the program code

4. Conclusions

In this previous mechanism, we can't customize to get more data from the too-chain. At this time, we replace source navigator with ASTM on it. The *Abstract Syntax Tree Metamodel* (ASTM) is useful to convert from the diverse program codes to *Abstract Syntax Tree* (AST). Then we suggest a whole procedure for SW visualization with the ASTM.

In the future, we will apply the existing parser such as CDT and JDT. Using them, we will develop new tools for SW visualization.

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). This work was supported by the National Research Foundation of KOREA (NRF) and Center for Women In Science, Engineering and Technology (WISET).

References

[1] Sang-Eun Lee, et al., "SW development quality management manual (SW Visualization)", National IT Industry Promotion Agency (NIPA), Dec. 12, 2013.
[2] Source Navigator, http://sourcenav.sourceforge.net/
[3] Graphviz, http://www.graphviz.org/
[4] OMG, "Architecture-driven Modernization: Abstract Syntax Tree Metamodel (ASTM) Version 1.0", OMG Document Number: formal/2011-01-05
[5] CDT, http://www.eclipse.org/cdt/
[6] JDT, http://www.eclipse.org/jdt/