Advanced and Applied Convergence Letters AACL 03

Advanced and Applied Convergence & Advanced Culture Technology

2nd International Symposium, ISAAC 2014
in conjunction with ICACT 2014
Jeju, Korea, November 2014
Revised Selected Papers

IIBC

IPACT

Table of Contents

A Case Study on Metamodel of Cause-Effect Graph Based on Model Transformation for Mobile Software Testing

Hyun Seung Son*, R. Young Chul Kim*

*SE Lab, Dept. of CIC(Computer and Information Communication), Hongik University, Sejong Campus, 339-701, Korea
e-mail : {son, bob}@selab.hongik.ac.kr

Abstract

The model transformation is a technique to transfer a model to other model or a model to a text. This technique may be used to automatically generate the code or the test case in model driven environment. On the existing method for test case generation, it implements a tool with algorithm's approach. This approach is a problem that he/she must modify the whole program against input model changed. In contrast, a model transformation looses the relationship against the input model, and easily modifies them. Also, it can implement more flexible way than the previous algorithms. With above advantages of model transformation, we apply to a cause-effect graph for automatic test case generation. The method for test case generation consists of 1) transforming decision table from cause-effect graph, and 2) transforming test case from decision table. And to apply this approach needs to use metamodel and transformation rule. In this paper, we propose a design of metamodel of cause-effect graph to apply model transformation.

Keywords: *Model Transformation, Cause-Effect Graph, Test case Generation, Mobile Testing, Metamodel.*

1. Introduction

The previous method for automatic test case generation writes programing language such as C, C++, Java, and etc. So a testing tool written by programing language is difficult to modify the program code when the input model is changed. Therefore, the existing method has limited in input models changed frequently.

But the model transformation [1-2] loose the relationship between the input models. In order to adapt model transformation, it must be required with elements of metamodel, engine, and rule language [3-6].

Our proposed approach automatically generates test case via decision table from cause-effect graph [7]. This method also requires the design of metamodel and writing rule for model transformation. Our previous research applied the model transformation techniques how to implement from cause-effect graph for automatic test case generation. In this paper, we are focused on designing a metamodel of cause-effect graph. The proposed test case generation tool is based on model transformation. But the metamodel of cause-effect graph not exist. We have to construct the structure with each name of elements of cause-effect graph. In the near future, our metamodel is used to combine each UML diagram with cause-effect graph based on model transformation. It can be easily extended with just re/writing the new rules.

The paper is organized as follows. Chapter 2 addresses a basic concept of the model transformation. Chapter 3 describes a metamodel of cause-effect graph and a case study. Chapter 4 gives conclusion and future works.

2. Related work

Figure 1 is the basic concept of model transformation. The figure is shown about simple scenario of

J.J. Kang et al. (Eds.): ISAAC 2014 & ICACT 2014, AACL 03, pp. 5~7, 2014

transformation to output model from input model [2]. Both input and output model conform to metamodel. The metamodel is to generally define abstract syntax modeling notation. The model transformation performs the written language with reference to metamodel. While the transformation language is written through rule, the rule writes similarities and differences between the two models in a natural language form, and the transformation language consists of commands like a program language that is performed in transformation engine. Therefore, model transformation to perform must require metamodel, transformation language, and transformation engine.

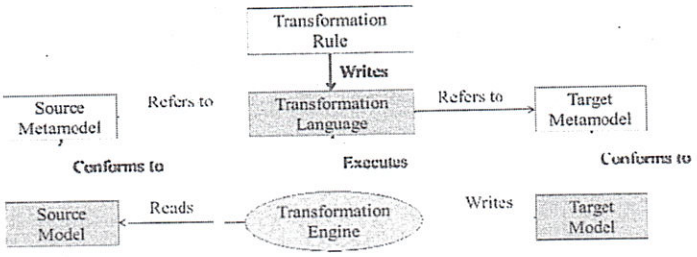


Figure 1. The basic concept of model transformation

3. The design of metamodel of cause-effect graph

The cause-effect graph is a method to express the relationship between cause and effect about some requirements. It is possible to represent logical relationship like 'and', 'or', 'not'. Figure 2 is an example of cause-effect graph. This example represents an effect that is true when two causes satisfy the logical expression of 'and'.

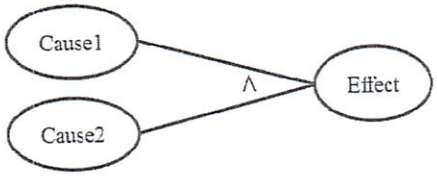


Figure 2. The example of cause-effect graph

Because the existing cause-effect graph does not using model transformation, the metamodel of cause-effect graph is not defined. Therefore, we need to design a metamodel with the basic name of cause-effect graph as shown figure 3.

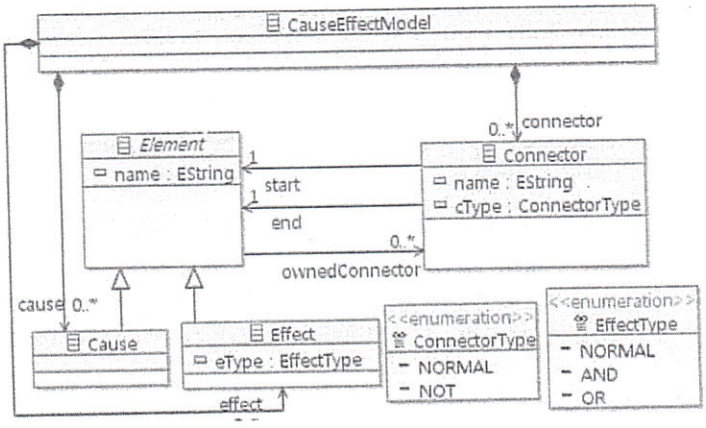
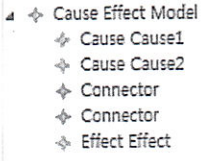


Figure 3. The metamodel of cause-effect graph

CauseEffectModel is a root node of cause-effect graph that includes all elements. The internal root node is

able to include Cause, Effect, and Connector. The Cause is causes, the Effect is effect, and the Connector is the connection of cause and effect and used to express the conditions.

In order to validate the designed metamodel, we draw the proposed metamodel using Ecore Tools [8], and generate the editor of cause-effect graph using Eclipse Modeling Framework (EMF) [9] in Eclipse. Figure 4 is the input result of an example of case-effect graph as shown figure 2. Figure 4(a) is an input result using editor. Figure 4(b) is the context of XML Metadata Interchange (XMI) file [10]. Through this result, we can validate the proposed metamodel.



(a) Editor

```
<?xml version="1.0" encoding="UTF-8"?>
<cedm:CauseEffectModel xmi:version="2.0"
xmlns:xmi="http://www.omg.org/XMI"
xmlns:cedm="http://cedm/1.0">
  <cause name="Cause1" ownedConnector="//@connector.0"/>
  <cause name="Cause2" ownedConnector="//@connector.1"/>
  <connector start="//@cause.0" end="//@effect.0"/>
  <connector start="//@cause.1" end="//@effect.0"/>
  <effect name="Effect" ownedConnector="//@connector.0
//@connector.1" eType="AND"/>
</cedm:CauseEffectModel>
```

(b) XMI file

Figure 4. The input result of cause-effect graph

4. Conclusions

In this paper, we design a required metamodel to apply cause-effect graph with model transformation for automatic test case generation. Because the metamodel of cause-effect graph not exist, we should construct the structure using name of elements of cause-effect graph. Also, we show a case study to fill the data of cause-effect graph in order to validate the defined metamodel. Further research should be conducted, which is not dealt in this study on rule to perform model transformation and design of other metamodel.

Acknowledgement

This work was supported by the IT R&D Program of MKE/KEIT [10035708, "The Development of CPS (Cyber-Physical Systems) Core Technologies for High Confidential Autonomic Control Software"] and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2013R1A1A2011601).

References

[1] OMG, MDA Guide Version 1.0, http://www.omg,org/mda, 2003.
[2] D. S. Frankel, *Model Driven Architecture: Applying MDA to Enterprise Computing*, Wiley, 2003.
[3] K. Czarnecki, S. Helsen, "Feature-Based Survey of Model Transformation Approaches," *IBM Systems Journal*, Vol. 45 No. 3, pp. 621-64, 2006.
[4] H.S. Son, J.S. Kim , R. Y.C. Kim, "SMTL Oriented Model Transformation Mechanism for Heterogeneous Smart Mobile Models," *International Journal of Software Engineering and Its Applications*, Vol. 7, No. 3, pp. 323-331, 2012.
[5] W.Y. Kim, H.S. Son, J.S. Kim, R. Y.C. Kim, "Adapting Model Transformation Approach for Android Smartphone Application," *Advanced Communication and Networking*, CCIS 199, pp. 421-429, 2011.
[6] W.Y. Kim, H.S. Son, J.S. Kim, R. Y.C. Kim, "Development of Windows Mobile Applications using Model Transformation Techniques," *Journal of KIISE : Comuting Practices and Letters*, Vol. 16, No. 11, 1091-1095, 2010.
[7] W.R. Elmendorf, *Cause-effect graphs in functional testing*, IBM Poughkeepsie Laboratory, 1973.
[8] Ecore Tools, http://eclipse.org/ecoretools/
[9] EMF, http://eclipse.org/emf/
[10] OMG. MOF 2.0/XMI Mapping, v2.1.1. OMG Available Specification, 2007.

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.