International Journal of Recent Advances in Multidisciplinary Topics
Volume 2, Issue 2, February 2021
https://www.ijramt.com | ISSN (Online): 2582-7839

Mining Combinatorial Test Design Elements from Design Artifact

Chanamallayya Hiremath[1], Shivani Rejintal[2], Vachanshree Chabbi[3], Raghavendra Eeti[4*]
[1,2,3,4]Department of Computer Science and Engineering, Dayananda Sagar College of Engineering, Bengaluru, India

***Abstract*: Objects such as UML operation diagrams and sequence diagrams provide details about the parameters, values and constraints of the underlying CTDM. Our aim of research is to create a tool that allows test designers to come up with the CTDM. This paper provides an approach to extracting CTDM-related knowledge from sequence diagrams, such as parameters and values.**

***Keywords*: Combinatorial testing, Sequence diagram, Test design model.**

1. Introduction

This paper introduces a new method for extracting CTDM Knowledge from sequence diagrams related. Our Main Key. In this article, the contribution involves presenting a rule-based proposal Method for defining from the sequence the model elements. The accompanying analyzer tool (UML) diagram Analyzer & Modeler sequence diagram) that extracts the Elements of Architecture. These laws have been independently applied, Checked on individual sequence diagrams and findings dependent on the general interpretation of the specifications.

2. Methodology

We will be designing and materializing a web technology based UI for the project. We will provide a versatile tool which allows the users to build their own UML diagrams and generate the combinatorial test design parameters table for the UML diagram.

A. UI

The UI will be designed and developed using Facebook's Open source UI library for Javascript, React.

B. Data structure

The UML diagram will have to be stored in a structured format for ease of use. The data will be stored in JavaScript Object Notation or JSON.

3. Literature Survey

Zoltan et al. [27] researched and presented a comprehensive image of 13 UML 2.0 sequence diagram semantics and how they vary. The Object Management Group (OMG) specification [31] provides a basic understanding of how the semantics of sequence diagrams work. In fact, the speaker discusses the use of formal semantics to describe the sequence diagrams. Several published papers discuss the generation of test cases from sequence diagrams for general software testing as appropriate. Samuel et al. [7] produce the Message Dependence Graph (MDG), an intermediate method for further study from sequence diagrams.

Slices from the MDG are generated for each state on the sequence diagram using the dynamic slicing method of edge labeling. Test cases are created based on these slices for cluster level research. Nayak et al. [2] introduce an automated approach to synthesize the test data with the details rooted in sequence diagram, class diagram and OCL constraints and map it to an intermediate shape called Structured Composite Graph (SCG). Test specifications are then generated from SCG and test data is in turn generated using a constraint solving method for each specification. From Cartaxo et al.

Nie et al. are undertaking a comprehensive, widespread CT survey. [16] which encompasses all facets of CT, from test modeling to CT implementations. Krishnan et.al. offers heuristics for finding the variables, stages and constraints. [6]. The foundation of the system of group partition (CPM) proposed by Ostrand et al. [28] is to segment the space of feedback into divisions and choices. The classification Tree Method (CTM) was used by Grochtmann and Grimm [8] to segregate the input domain into classifications and groups and model it further in the tree structure. Borazjany et al. [17] recommend a technique for input space modeling using two steps, Input Structure Modeling (ISM) and Modeling Input Parameter (IPM). Grindal & Offutt et al., both [20] introduces a CT modeling approach known as the Input Parameter Model Model (IPM). Segall et al. [3] mention the problems of correctness, completeness and redundancy that often arise and thus direct the testers.

4. Conclusion

The project is still in the implementation and development phase. Once the proposed system is realized, it will be able to provide a tool to create simple UML diagrams and generate their test parameters from it.

*Corresponding author: raghavendra.eeti@tuta.io

Acknowledgement

Prof. Preeti Satish of Dayananda Sagar College of Engineering will be guiding us through the project.

References

[1] Ashalata Nayak and DebashishSamanta, “Automatic Test Data Synthesis using UML Sequence Diagrams,” in *Journal of Object Technology*, vol. 09, no. 2, pp. 75-104, March-April 2010.

[2] Itai Segall, R. Tzoref-Brill, and A. Zlotnick, “Simplified modeling of combinatorial test spaces,” IEEE 5th International Conference on Software Testing, Verification and Validation (ICTS), 2012, pp. 573- 579.

[3] Sergiy A. Vilkomir, Khalid A. Abdelfattah and Sudha Gummadi, “MIST: Modeling input space for testing tool,” Proceddings of the 13th IASTED International Conference Software Engineering and Applications (SEA 2009), Cambridge, MA, USA, 2009, pp. 210–217.

[4] Sergiy A. Vilkomir, W. Thomas Swain and Jesse H. Poore, “Software input space modeling with constraints among parameters,” 33rd Annual IEEE International Computer Software and Applications Conference, Seattle, Washington, USA, 2009, Vol. 1, pp. 136–141.

[5] D. M. Cohen, S. R. Dalal, J. Parelius and G. C. Patton, “The combinatorial design approach to automatic test generation,” IEEE software, vol. 13, No 5, pp. 83–89, September 1996.

[6] R. Krishnan, S. M. Krishna and P. S. Nandhan, “Combinatorial testing: Learnings from our experience,” ACM SIGSOFT Software Engineering Notes, 2007, vol. 3, pp. 1–8.

[7] P.Samuel and R.Mall, “A novel test case design technique using dynamic slicing of UML sequence diagrams,” in e-Informatica, 2008, 2(1) , pp. 71-92.

[8] M.Grochtmann and K.Grimm, “Classification Trees for Partition Testing,” Journal of Software Testing, Verification, and Reliability, 3(2):63–82, 1993.

[9] Mario Brcic and Damir Kalpic, “Combinatorial testing in software projects,” MIPRO, 2012 Proceedings of the 35th International Convention, 2012, pp. 1508–1513.

[10] “UML basics: The sequence diagram.” [Online]. Available: http://www.ibm.com/developerworks/rational/library/3101.html.

[11] Itai Segall, R. Tzoref-Brill, and A. Zlotnick, “Common patterns in combinatorial models,” IEEE 5th International Conference on Software Testing, Verification and Validation (ICTS), 2012, pp. 624-629.

[12] Grady Booch, James Rumbaugh and Ivar Jacobson, “The unified modeling language,” Reference Manual, Addison Wesley, 2001.