International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-8 Issue-10, August 2019

Test Case Generation for UML Behavioral Diagram by Traversal Algorithm

Rashmi Gupta, Vivek Jaglan

Abstract*: Software testing play crucial role in the software development as it consumes lot of time and resources. However testing process needs to be more efficiently done because overall software quality relies upon good testing approach. The present research focus on generation of test cases from UML diagrams. The combination graph is made by using activity and sequence diagrams. These diagrams proves to be more efficient as activity diagram gives the dynamic behavior of the model and sequence diagram is used to understand detailed functionality of the system. In this paper, a combined approach using Breadth first and depth first search is proposed which will generate expected test cases. The comparative study is done for test case generation using BFS and DFS algorithm and the result proves that the DFS traversal algorithm provides more accurate result for path coverage.

Keywords* : Test case generation, traversal algorithm, UML diagrams, software testing.

I. INTRODUCTION

The time, budget and quality are the key parameters for the software testing. The 50% of the total expenditure is used by software testing. Testing is not solely for debugging but also for quality assurance, verification and validation. There are three main aspects in software testing cycle. These are test case generation (TCG), test execution and test evaluation. It is easier to implement the last two parts. But the first part is a very challenging task for the software tester. It requires deep knowledge of software testing process. The scrutiny of Unified modeling language (UML) model is a very hard task because the whole information is scattered across various model views. The complexity of the problem is reduced by taking the UML models with enhancement in product size. The communication of objects in a sequence of messages is described by the sequence diagram. It also shows the life spans of objects respective to messages. The test cases are generated using sequence and activity diagram. After that sequence and activity diagram is transformed into corresponding graphs which are known as sequence diagram graph (SDG) and activity diagram graph (ADG). Then System Testing Graph (STG) is created by merging the two graphs. The diverse range of faults is covered

by generated test suite. The all possibilities are covered by combined UML diagrams. The paper is systematically arranged as follows. The prior art on creation of test cases using various UML diagrams is presented in section 2. The sequence and activity diagram for ATM machine and their transformation into graph is described in section 3. The case study of ATM machine test case generation is explained using STG in section 4. Finally, the conclusion and possible extension of the work is described.

Manuscript published on 30 August 2019.
***Correspondence Author(s)**
Rashmi Gupta*, CSE department, Amity University Haryana, Gurugram, India. Email: goyal.rashmi18@gmail.com
Vivek Jaglan, CSE department, Amity University Haryana, Gurugram, India. Email: jaglanvivek@gmail.com

II. LITERATURE REVIEW

This section highlights the works related to production of test cases using UML diagrams.
Rajiv mall et al. [4] presented a process to produce test cases by applying combination of use case and sequence diagram. They transform diagrams into their respective graphs. Then graphs are merged together to make the system graph. But, it was not clearly shown that how graphs are merged together. The optimization is not done for generated test cases.
Abinash Tripathy et al. [3] give an idea to produce test cases. They amalgamate UML AD and SD for card validation . Ranjita Kumari Swain et al. [5] have produced the test cases using AD. In their method, activity diagram is initially transformed into ADG. The depth first traversal technique is used for traversing the paths. All activity paths are produced by using proposed algorithm. Finally, path coverage criterion is used for TCG. Swagatika Dalai et al. [9] described a method to produce test cases for object oriented systems using integrated UML Models. Then traversing is completed to produce test cases. But optimization of test cases is still not completed.

III. PROPOSED APPROACH

Our proposed technique, first translates the activity and sequence diagrams into their respective graphs which is called as activity diagram graph (ADG) and sequence diagram graph (SDG). The subsequently, System Testing Graph (STG) will be made by integrating both activity and sequence graph. Then STG is traversed using BFS and DFS algorithm separately and resultant test cases are stored. The case study of an ATM system is presented by using this approach. .Subsequently the generated test cases are compared according to their success and failure rate.

A. TRANSFORMATION OF AD TO ADG

In this part, initially an AD is made for the problem. AD is utilized to describe the dynamic flow of processes of a system. After that AD is reformed into activity graph (AG) which is formed by information used from the diagram such as node id, activity name, target and status. Every activity in graph is recognized by a specific ID. Activity name indicates the name of every activity.

The adjacency list is made by upcoming activity which is provided by activity target. The status of decision component after passing is known as activity status.

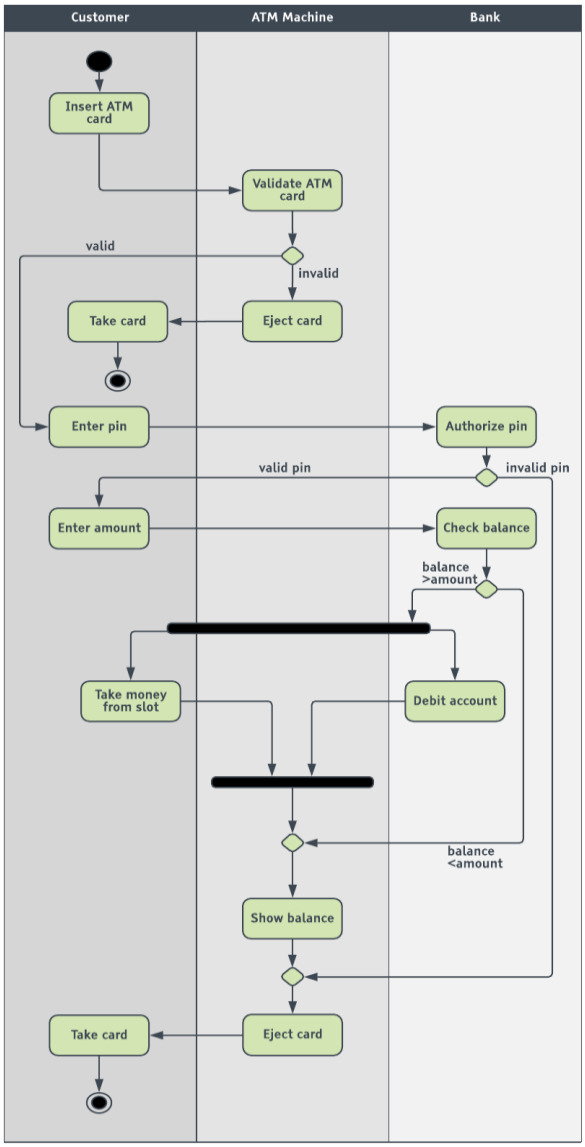

Fig 1. Activity Diagram of ATM

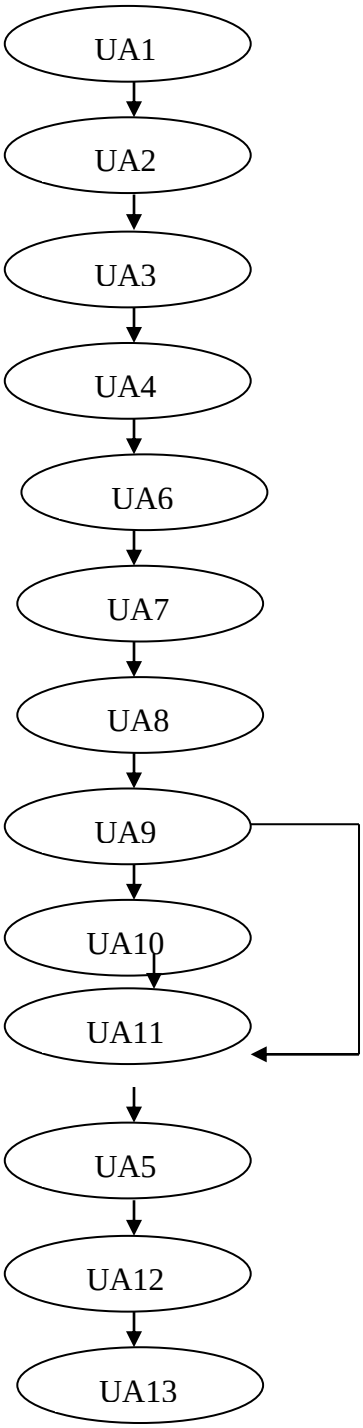

Fig 2. Activity Diagram Graph

B. TRANSFORMATION OF SD INTO SDG

Initially SD is made for the problem. Sequence diagram (SD) utilized the object interactions in sequence manner. After that SD is reformed into sequence graph (SG) which is formed by information used from the diagram such as node id, message name, number, target and operand status. Every message in graph is recognized by a specific ID. Message name indicates the name of every message. The adjacency list is made by upcoming message which is provided by message target.

Retrieval Number: J11900881019/19©BEIESP
DOI: 10.35940/ijitee.J1190.0881019
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

Message number is the order of every message that is considers for rearranging the message. The operand status is used to point the alternative message in.

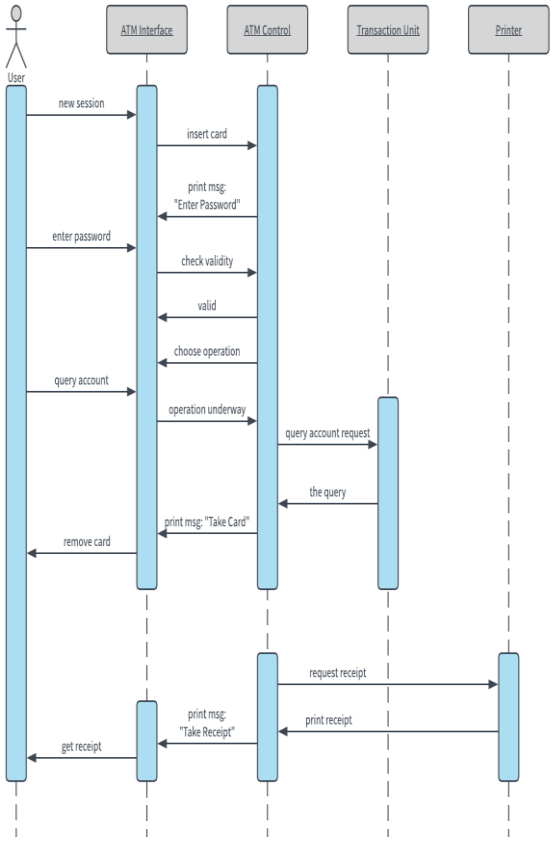

Fig 3. Sequence diagram of ATM System

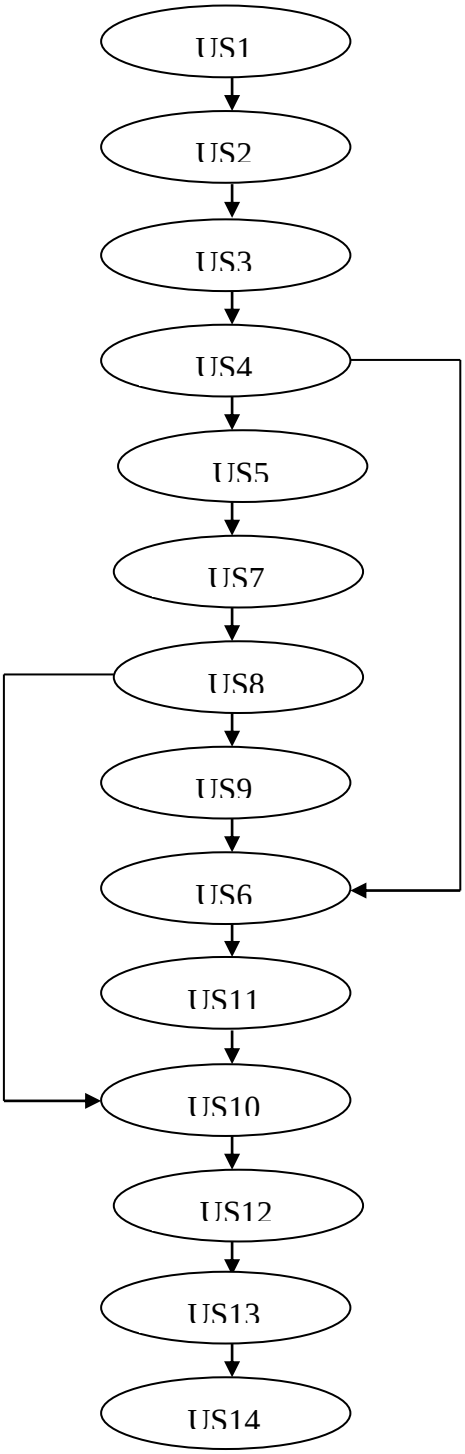

Fig 4. Sequence Diagram Graph

C. INTEGRATED SYSTEM TESTING GRAPH (STG)

By combining the AG and SG, STG is formed. In system testing graph, identical nodes from both activity and sequence graphs are marked and combined to form system nodes in system testing graph. As these nodes are more significant because they are having the properties of both activity and sequence graph. The system testing graph will be formed as mentioned in the next diagram.

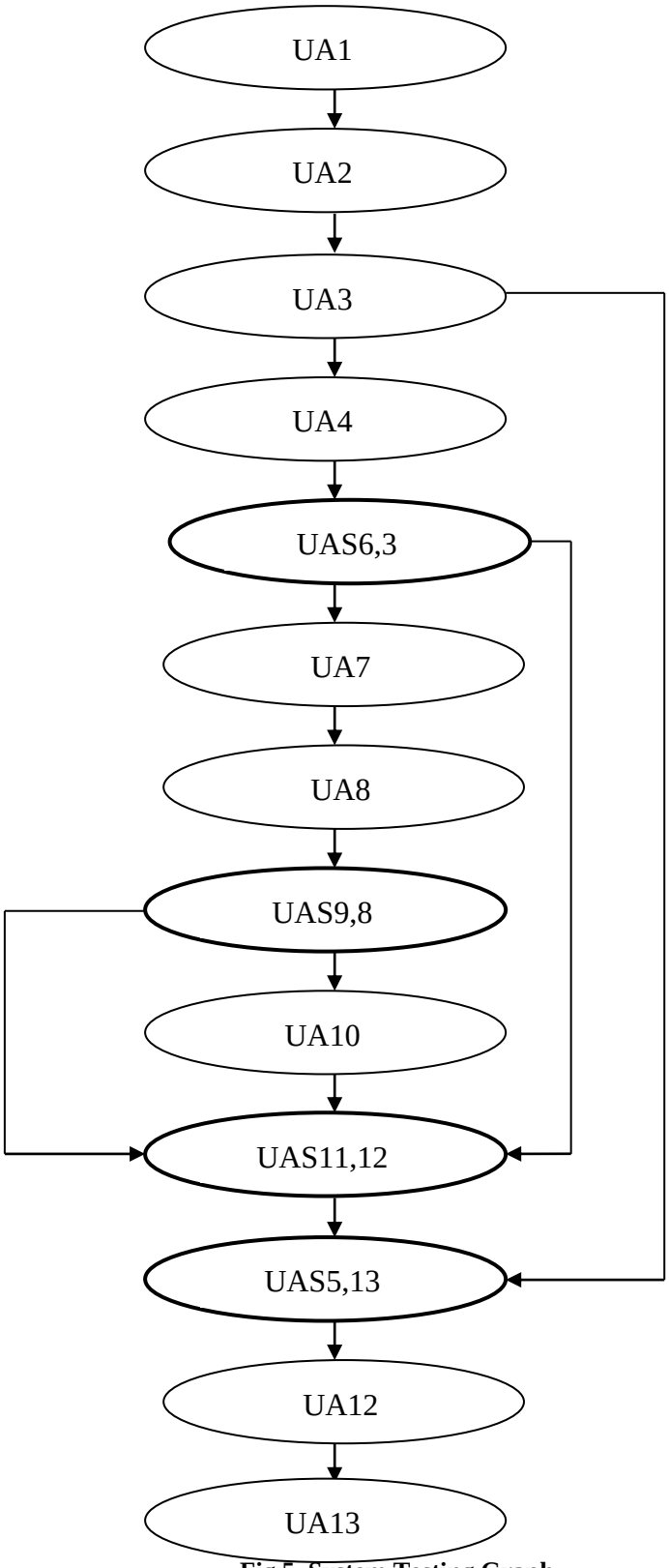

Fig 5. System Testing Graph

IV. Test Case Generation (TCG)

After formation of the ADG, SDG and STG, subsequently the test cases are produced for each of the graph. The experimental results of TCG from AG, SG, and STG are presented in this section for ATM case which depicts in figure 1 and 2. Test cases are produced by proposed algorithm in a simple path flow which provides excellent comparison among test cases for various graphs. DFS and BFS algorithms are used to traverse the graphs. The outcome of DFS algorithm reveals that entire nodes in a graph are merged into single test case instead of branches exists in the graph. In testing process, a branch is moving towards generation of new test cases.

A. ACTIVITY GRAPH

Creation of Test cases using DFS algorithm:
a. Success case: UA1 => UA2 => UA3 => UA4 => UA6 => UA7 => UA8 => UA9 =>UA10 =>UA11=>UA5 =>UA12 =>UA13
Detail: Start => Insert ATM Card => Validate ATM Card => Enter Pin=> Authorize pin => Enter Amount => Check Balance => Debit amount => Show balance =>Take Card => Eject Card => Finish.
Creation of Test cases using BFS algorithm:
a. Failed case: UA1 => UA2 => UA3 => UA4 => UA5 => UA6 => UA7 => UA8 =>UA9 =>UA11=>UA5 =>UA12 =>UA13.
Detail: Start => Insert ATM Card => Validate ATM Card =>Take Card => Enter Pin=> Authorize pin => Enter Amount => Check Balance => Debit amount => Show balance =>Take Card => Eject Card =>Finish

B. SEQUENCE GRAPH

Creation of Test cases using DFS algorithm:
US1 => US2 => US3 => US4 => US5 => US7 => US8 => US9 =>US6=>US10=> US11 =>US12 =>US13.
Detail: ATM (New Session)=>Insert Card=> Enter Pin => check validity => choose operation => query request => check Balance => accept request =>Print message => Remove card => Request receipt => Print receipt => get receipt =>exit.
Creation of Test cases using BFS algorithm: US1 => US2 => US3 => US4 => US6 => US5 => US7 => US8 => US10=> US9 =>US6=> US11 =>US12 =>US13.
Detail: ATM (New Session)=>Insert Card=> Enter Pin => check validity => choose operation => query request => check Balance => Request receipt =>accept request =>Print message => Remove card => Print receipt => get receipt =>exit.
The flow of user's activity from the starting to the finis is described by the result of test cases. The general layout of the system is shown by the generated test cases. The generated test cases utilized a common word that makes the system easier to read by non-expert. The whole process in a system cannot exhibit by the test cases. The information gathered from AD is onerous to process due to use of non standard common rules.
These diagrams show the methods in details. This information provides the help to tester in filling the application during the process of testing.

C. STG GRAPH

Creation of Test cases using DFS algorithm:
a. Success case: UA1 => UA2 => UA3 => UA4 => UAS6,3 => UA7 => UA8 => UAS9,8 =>UA10

=>UAS11,12=>UAS5,13=>UA12 =>UA13
Detail: Start => Insert ATM Card => Validate ATM Card => Enter Pin=> Authorize pin => Enter Amount => Check Balance => Debit amount => Show balance =>Take Card => Eject Card =>Finish.

Creation of Test cases using BFS algorithm:
a. Failed case: UA1 => UA2 => UA3 => UA4 => UA5 => UAS6,3=> UA7 => UA8 => UAS9,8 =>UA11 US12=> UAS5,13=>UA12 => UA13.
Detail: Start => Insert ATM Card => Validate ATM Card =>Take Card => Enter Pin=> Authorize pin => Enter Amount => Check Balance => Debit amount => Show balance =>Take Card => Eject Card =>Finish

STG is found by mixing the ADG and SDG which generates the more detailed number of test cases. The general layout of the system which the user can do from the starting to end is exposed by the generated test cases. Sometimes the generated test cases may also consist of redundant information. In this case study, 'success/failed notification' has been shown twice.

V. CONCLUSION AND FUTURE WORK

Software testing is necessary to validate the requirements and to maintain software quality. The comparison of DFS and BFS algorithm to produce automatic test cases for AD and SD has been presented in this paper. The suitable test cases are produced from both algorithms. The test cases which are produced according to AD, SD, and STG graphs are given. The activity diagram based coverage criterion is considered for generation of test cases. But the test cases cannot display the whole process like lack of information in filling of application during the testing. So it becomes onerous to process further because of not using of standards words in the diagram. The test cases which are produced from SD only screen the system section. But the produced test cases display the whole information of business process inside a system. The test cases produced from the integrated graph shows the features of AD and SD. But consist of unnecessary information. The optimization of test cases by using genetic algorithm will be combined with the current work.
STG is traversed using both DFS and BFS algorithms and compared on the basis of their success and failure rate.
As is it is clear from the test cases generated success rate of test cases is higher in DFS algorithm as compared to BFS algorithm. Further there is scope for research in the same area by finding some more parameters in the algorithms.

REFERENCES

1. V.K. Shanthi a., Mohan Kumar G. A Novel Approach for Automated Test Path Generation using TABU Search Algorithm. International Journal of Computer Applications. 2012;48(13):28–34.
2. Srivastaval J, Dwivedi T. Software Testing Strategy Approach On Source Code Applying. 2015;6(3):25–31.
3. Tripathy A, Mitra A. Test case generation using activity diagram and sequence diagram. In: Proceedings of International Conference on Advances in Computing. Springer; 2013. p. 121–9.
4. Monalisa, S., Debashish, K., & Rajib, M. Automatic test case generation from UML sequence digrams. Proceedings of IEEE Conference on Software Maintenance. 2007
5. Ranjita, K. S., Vikas, P., & Prafulla, K. B. Generation of test cases using activity diagram. Interntional Journal of Computer Science and Informatics, 2013;3(2).
6. Meiliana, aIrwandhi S., aRicky S. Alianto, aDaniel, bFord LG . Automated Test Case Generation from UML Activity Diagram and Sequence Diagram using Depth First Search Algorithm in International Conference on Computer Science and Computational Intelligence 2017, ICCSCI, Elseveir 2017.
7. Faria JP, Paiva ACR, Yang Z. Test generation from UML sequence diagrams. Proceedings - 2012 8th International Conference on the Quality of Information and Communications Technology, QUATIC 2012. 2012;245–50.
8. Panthi V. Automatic Test Case Generation using Sequence Diagram. 2012;2(4):22–9.
9. Swagatika, D. , Arup Acharya and Durga Prasad Mohapatra .Test case generation for concurrent object oriented systems using combinational UML models., 2012
10. Hettab A, Chaoui A, Aldahoud A. Automatic Test Cases Generation From Uml Activity Diagrams Using Graph. 2013.
11. Sumalatha VM, others. Object Oriented Test Case Generation Technique using Genetic Algorithms. International Journal of Computer Applications. 2013;61(20).
12. Asad S, Shah A, Shahzad RK, Shafique S, Bukhari A, Humayun M. Automated Test Case Generation Using UML Class & Sequence Diagram. 2016;15(3):1–12.
13. Khurana N, Chillar RS A Novel Technique for Generation and Optimization of Test Cases Using Use Case, Sequence, Activity Diagram and Genetic Algorithm in journal of software .2015.

AUTHORS PROFILE

Rashmi Gupta is pursuing her Ph. D. from Amity University Haryana .She has total 8 years of teaching and research experience.. Her research area is Software Engineering, Soft computing and algorithms.

Dr. Vivek Jaglan is currently working as an Associate Professor, in CSE department in Amity School of Engineering and Technology, Amity University Haryana .He has total 13.5 years of teaching and research experience. He has published more than 40 papers in various International journals as well as in conferences of good repute. His research area is Artificial intelligence, agent technology, and robotics.