



UNIVERSITI PUTRA MALAYSIA

***AUTOMATED MODEL-BASED TEST CASE GENERATION USING UML
ACTIVITY DIAGRAM***

MD ABDUL MONIM

FSKTM 2018 43



AUTOMATED MODEL-BASED TEST CASE GENERATION USING UML ACTIVITY DIAGRAM

MD ABDUL MONIM

MASTER OF COMPUTER SCIENCE

UNIVERSITI PUTRA MALAYSIA

2018



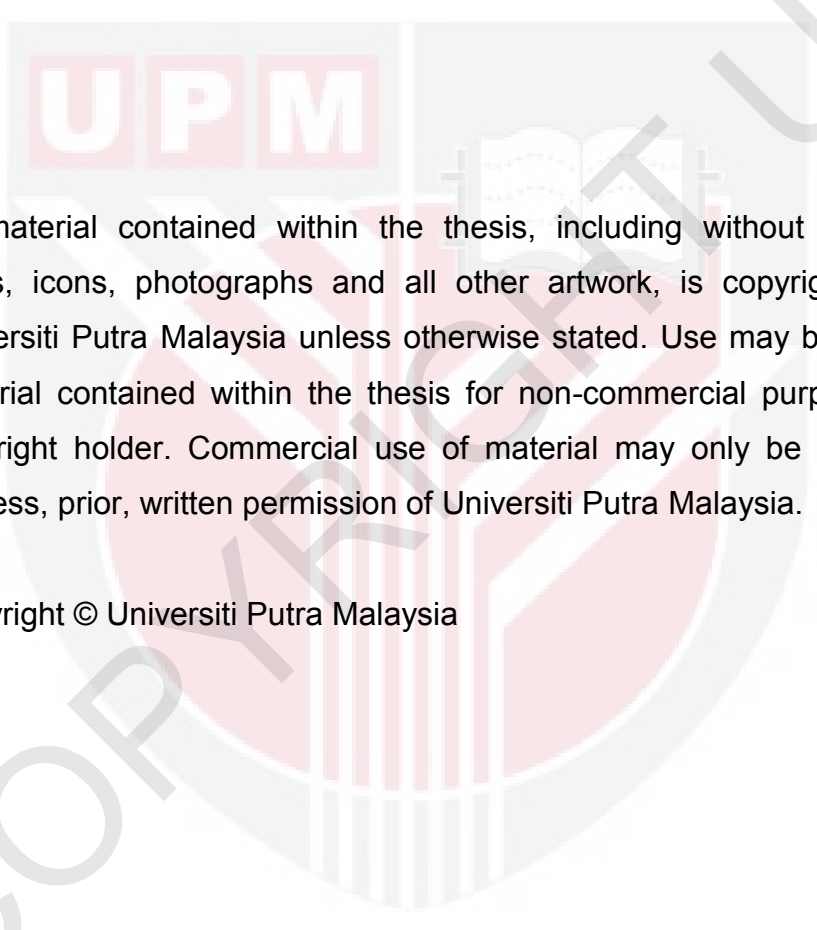
AUTOMATED MODEL-BASED TEST CASE GENERATION USING UML ACTIVITY DIAGRAM

By

MD ABDUL MONIM

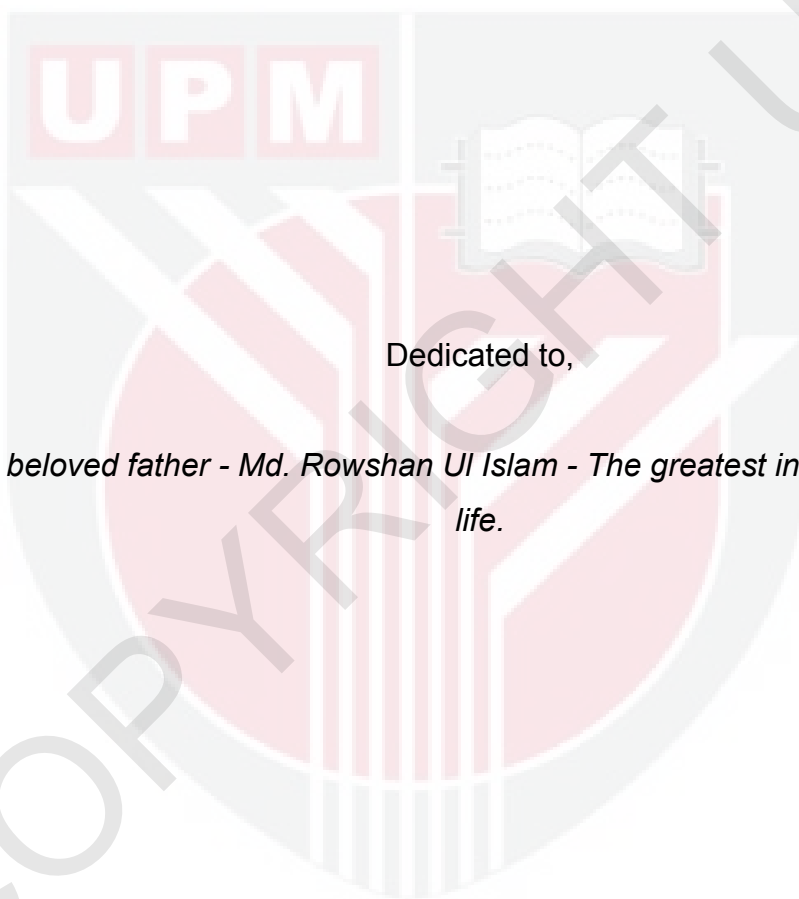
Thesis Submitted to the Faculty of Computer Science and Information Technology,
Universiti Putra Malaysia, in Fulfillment of the
Requirements for the Degree of Master of Computer Science

January 2018

The logo of Universiti Putra Malaysia (UPM) is a shield-shaped emblem. It features a red shield with a white border. Inside the shield, there is a stylized white 'U' and 'P' that form a central motif. Above the shield, the letters 'UPM' are written in white on a red rectangular background. The entire logo is centered on the page.

All material contained within the thesis, including without limitation text, logos, icons, photographs and all other artwork, is copyright material of Universiti Putra Malaysia unless otherwise stated. Use may be made of any material contained within the thesis for non-commercial purposes from the copyright holder. Commercial use of material may only be made with the express, prior, written permission of Universiti Putra Malaysia.

Copyright © Universiti Putra Malaysia



Dedicated to,

My beloved father - Md. Rowshan Ul Islam - The greatest inspiration in my life.

ABSTRACT

Software or application testing is a process of executing a program with the goal of finding defect to make better system. In software testing phase, writing test cases is one of the important activities. Manually writing test cases approach is lengthy of time period and need more effort to accomplish the process. On the other hand, automated test case generation technique is the way to solve this issue and Model-Based Test case generation approach would be the appropriate for this automation process. Usually, a model is required in Model-Based Testing approach to generate the test cases. This study aimed is to develop an automated tool that can be able to generate test cases from UML activity diagram. This study has successfully developed the automated test case generating tool based on Model-Based Testing approach that could generate test cases automatically using UML activity diagram as an input. In addition, the proposed model and technique has explained detail in this thesis. Furthermore, an expert review based on Post Study System Usability Questionnaire (PSSUQ) method was also conducted to evaluate the tool and also verify the study requirement and objectives. A group of experienced software testing expert was involved in this evaluation phase. As result, the evaluation was satisfied all the objectives. Based on this study, some future recommendations have been proposed also for the extension of the developed tool's quality and functionality to make a better tool.

ABSTRAK

Pengujian perisian atau aplikasi adalah proses melaksanakan program dengan matlamat mencari kecacatan untuk membuat sistem yang lebih baik. Dalam kes fasa pengujian perisian, menulis kes ujian adalah salah satu aktiviti penting. Secara manual, pendekatan menulis kes ujian memakan masa yang panjang dan memerlukan lebih banyak usaha untuk mencapai proses tersebut. Sebaliknya, teknik penjanaan kes ujian automatik adalah cara untuk menyelesaikan masalah ini dan pendekatan Generasi Pengujian Kes Berasaskan Model adalah sesuai untuk proses automasi ini. Biasanya model ini memerlukan pendekatan Pengujian Berasaskan Model untuk menghasilkan kes ujian. Oleh itu, rajah aktiviti UML adalah model yang dipilih untuk projek ini. Matlamat kajian ini adalah untuk membangunkan alat automatik yang boleh menghasilkan kes ujian dari rajah aktiviti UML. Kajian ini telah berjaya membangunkan alat penjanaan kes ujian automatik berdasarkan pendekatan Pengujian Berasaskan Model yang boleh menjana kes ujian secara automatik menggunakan rajah aktiviti UML sebagai input. Di samping itu, model dan teknik yang dicadangkan telah dijelaskan secara terperinci dalam tesis ini. Tambahan pula, analisis pakar berdasarkan Kaedah Usulan Sistem Kajian Pos juga dijalankan untuk menilai alat dan juga mengesahkan keperluan dan objektif kajian. Sekumpulan pakar ujian perisian yang berpengalaman terlibat dalam fasa penilaian ini. Hasilnya, penilaian telah memenuhi semua objektif. Berdasarkan kajian ini, beberapa cadangan masa depan telah dicadangkan juga untuk lanjutan fungsi alat yang dibangunkan untuk membuat alat yang lebih baik.

ACKNOWLEDGEMENT

First of all I would like to thank to almighty Allah who helps us all the time, which is the most effective and efficient tools for all kind of achievements.

Here I would like to express my feelings to those who have offered supporting me to complete this project.

I would like to convey my thankfulness and appreciation to my supervisor Dr. Rozi Nor Haizan binti Nor, a Senior lecturer at Department of Software Engineering and Information System, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia for her precious contribution, unvarying supervision, instinctive advice, admirable support and continual persistence for the completion of this project. I am very much gratifying to her and feel very arrogant to have worked with her because it was not probable for me to complete this work without her inspirational enthusiasm and encouragement.

I would also like to thank the Faculty of Computer Science and Information Technology and the University who has generously providing me with all the facilities and equipments necessary for the success of this work.

Finally I would like to convey my special thanks to my parents whom have always given me tremendous support to realize my dreams. Without their love and encouragement, I would not had achieve this far.

APPROVAL

This thesis entitled “**Automated Model-Based Test Case Generation Using UML Activity Diagram**” was submitted to the Senate of Universiti Putra Malaysia and has been accepted as fulfillment of the requirement for the degree of Master of Computer Science.



Name of Supervisor

Dr. Rozi Nor Haizan binti Nor

Faculty of Computer Science and Information Technology

Universiti Putra Malaysia

DECLARATION

I hereby confirm that:

- This thesis is my original work;
- Quotations, illustrations and citations have been duly referenced;
- This thesis has not been submitted previously or concurrently for any other degree at any other institutions;
- Intellectual property from the thesis and copyright of thesis are fully-owned by Universiti Putra Malaysia, as according to the Universiti Putra Malaysia (Research) Rules 2012;
- Written permission must be obtained from supervisor and the office of Deputy Vice-Chancellor (Research and Innovation) before thesis is published (in the form of written, printed or in electronic form) including books, journals, modules, proceedings, popular writings, seminar papers, manuscripts, posters, reports, lecture notes, learning modules or any other materials as stated in the Universiti Putra Malaysia (Research) Rules 2012;
- There is no plagiarism or data falsification/fabrication in the thesis, and scholarly integrity is upheld as according to the Universiti Putra Malaysia (Graduate Studies) Rules 2003 (Revision 2012-2013) and the Universiti Putra Malaysia (Research) Rules 2012. The thesis has undergone plagiarism detection software.

Signature: _____ Date: _____

Name and Matric No.: _____

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
	ABSTRACT.....	iii
	ABSTRAK.....	iv
	ACKNOWLEDGEMENT.....	v
	APPROVAL.....	vi
	DECLARATION.....	vii
	TABLE OF CONTENTS.....	viii
	LIST OF FIGURES.....	xi
	LIST OF TABLES.....	xii
	ACRONYMS AND AVVREVIATIONS.....	xiii
1	INTRODUCTION.....	1
	1.1 Overview.....	1
	1.2 Problem Statement.....	2
	1.3 Objective.....	3
	1.4 Scope.....	4
	1.5 Structure of Thesis.....	4
2	LITERATURE REVIEW.....	7
	2.1 Overview of test case generation and testing techniques.....	7
	2.2 Model-Based Testing.....	8
	2.3 Automated Test Case Generation.....	10
	2.4 Comparison of some existing Model-Based Testing tools.....	10
	2.5 Related Works.....	13
	2.6 Summary.....	18
3	RESEARCH METHODOLOGY.....	19
	3.1 Overview.....	19
	3.2 Research Methodology of the project.....	19

	3.3 Summary.....	25
4	PROPOSED SYSTEM: AUTOMATED TEST CASE GENERATING TOOL.....	26
	4.1 Overview.....	26
	4.2 Proposed Model.....	26
	4.3 Model-Based Testing Approach.....	27
	4.3.1 Modeling.....	28
	4.3.2 Test Case Generation.....	32
	4.4 Summary.....	36
5	DESIGN AND IMPLEMENTATION.....	37
	5.1 Overview.....	37
	5.2 Functionality of the Proposed Tool.....	37
	5.3 Design of the Proposed Tool.....	38
	5.3.1 Unified Modeling Language (UML) Design.....	39
	5.4 Implementation of the Proposed System.....	45
	5.4.1 System Environment.....	46
	5.4.2 Hardware Environment.....	46
	5.4.3 System Development.....	46
	5.5 Summary.....	55
6	TESTING AND EVALUATION.....	56
	6.1 Overview.....	56
	6.2 Functional Testing.....	56
	6.2.1 Unit Testing.....	57
	6.2.2 Integration Testing.....	60
	6.3 Evaluation.....	62
	6.3.1 Usability Testing.....	62
	6.3.2 Evaluation Result (PSSUQ Result).....	64
	6.4 Summary.....	64
7	CONCLUSION AND RECOMMENDATIONS FOR FUTURE WORK.....	66
	7.1 Overview.....	66

7.2	Conclusion.....	66
7.3	Contribution.....	67
7.4	Limitation.....	67
7.5	Recommendation for Future Works.....	68
REFERENCES		69
APPENDIX A.....		72
APPENDIX B.....		76
APPENDIX C.....		77



LIST OF FIGURES

Figure	Page
Figure 2.1: A typical Model-Based (MBT) testing process.....	9
Figure 4.1: Proposed model for test case generation.....	27
Figure 4.2: Model-Based Testing approach.....	28
Figure 4.3: Overall steps of test case generation.....	36
Figure 5.1: UML Use-Case diagram of proposed tool.....	40
Figure 5.2: UML Activity Diagram of the proposed tool.....	42
Figure 5.3: UML Sequence diagram of the proposed tool.....	43
Figure 5.4: UML Class diagram of the proposed tool.....	45
Figure 5.5: Eclipse Modeling Tools with Papyrus plug-in.....	47
Figure 5.6: Structure of a (.uml) file of a UML activity diagram.....	48
Figure 5.7: The main interface of the tool.....	49
Figure 5.8: The interface after click on “Upload Diagram” button.....	50
Figure 5.9: The interface after click on cancel instead of open a file.	51
Figure 5.10: The interface after open a valid (.uml) file.....	51
Figure 5.11: The interface after click on the “Generate Test Cases” button.....	53
Figure 5.12: The interface when test cases generated successfully.	54
Figure 5.13: Sample output of generated test cases.....	55
Figure 6.1: PSSUQ result chart.....	64
Figure B1: Level zero DFD of Proposed Tool.....	76
Figure C1: Gantt chart of the overall project.....	77

LIST OF TABLES

Table	Page
Table 2.1: Different types of MBT tool for test case generation.....	11
Table 2.2 Summary table of some related works.....	15
Table 5.1: Use-Case Details.....	41
Table 6.1: Test cases of Reading a UML activity diagram module....	57
Table 6.2: Test cases of Convert the UML activity diagram into Activity Graph module.....	58
Table 6.3: Test cases of Traversing the Activity Graph by using DFS algorithm to identify test sequences module.....	59
Table 6.4: Test cases of Generate the test cases from test sequences module.....	60
Table 6.5: Test cases for Integration testing.....	61
Table 6.6: Respondent details.....	63

ACRONYMS AND ABBREVIATIONS

ATCGT	Automated Test Case Generating Tool
AD	Activity Diagram
AG	Activity Graph
BPMN	Business Process Model and Notation
DFS	Depth First Search
DOM	Document Object Model
DFD	Data Flow Diagram
EFSM	Extended Finite State Machine
FSM	Finite State Machine
GUI	Graphical User Interface
IDE	Integrated Development Environment
MBT	Model-Based Testing
OCL	Object Constraint Language
OMG	Object Management Group
OS	Operating System
PSSUQ	Post Study System Usability Questionnaire
QA	Quality Assurance
SUT	System Under Test
TC	Test Cases
UI	User Interface
UML	Unified Modeling Language
USB	Universal Serial Bus

XMI

XML Metadata Interchange

XML

Extensible Markup Language



CHAPTER 1

INTRODUCTION

1.1 Overview

Today we are living in the software economy era, where every business is now software related. While this is exciting, but developing software is also have an array of strenuous challenges. Software and Applications are developed to enhancing our daily life. Peoples are using software and web application in their daily life activities. Software is called good when its performance and efficiency shows some good sign. Making a good quality full software is the main goal for all the developer and they follow a standard rule to gain that quality. A software system or a web application had to go through into a development life cycle and testing is an important phase of this software/application development life cycle. “Due to extra pressure to finish projects on time project managers are likely to reduce the testing activities” (Galín, D., 2004). The budget allocated for software testing has increased recently and the company managers are afraid that it is getting out of control. “They cite the reliance on manual testing as their biggest challenge, so the amount of test automation is increasing” (Sogeti, C., 2015-16). Software testing can be done by manually or automatically. In manual testing, the testing requires doing all the process by manually includes input,

analysis, and writing and managing the test cases. Manual testing deals with human interaction with all the process from beginning to end of the process and it is a time-consuming process. A human can get tired of doing all the process continuously. On the other hand, In automated testing most of the testing processes are automated. Generating test cases, executing the test cases and producing the test result are done by automatically. Though, one of the software testing fundamental is hundred percent automation is not possible in the field of software testing. Some task still need the intervention of human. According to (Sogeti, C., 2015-16) the average amount of test case automation has increased from 29% in 2014 to 45% in 2015. In automated testing process need some of the tools to support that automation and there are a huge amount of tools are available in the market. Different kind of tools was developed for different types of programming language and methods. The scientists and academics have focused more attention on this automated testing field to developing different techniques, methods, approaches, and tools.

1.2 Problem Statement

If the software testing stage can be finished early in software development life cycle then the total development process will be shortened and the software product is possible to deliver early. So, the main concern is reducing the testing time by applying any technique or approach and test

case generation is mostly related part to consume the length. Manual test case writing approach is lengthy of time period and need more effort to accomplish the process. Manually have to write all test cases from the requirement and then execute the test cases are also done by manually. Writing test cases from the requirement is a very long process, boring and error-prone. Hence, automated test case generation is the way to solve this issue.

1.3 Objective

The main objective of this project is to propose an automated test case generation techniques for developing a tool that can generate test case automatically. The objective of this project is,

- i) To adopt an approach for test case generation technique.
- ii) To propose a model to automate the test case generating techniques for generate test case earlier in development life cycle.
- iii) To develop an automated test case generating tool that can generate test case automatically in the requirement stage.
- iv) To evaluate the proposed tool in order to measure the usability of the tool.

1.4 Scope

- i) The test case generation process based only behavior of the system or application.
- ii) This Model-Based Testing approach has applied only for developing the model and generating the test cases, not for test execution.
- iii) Formal Model-Based Testing process has used in this project.
- iv) Only offline Model-Based Testing approach have used.

1.5 Structure of the Thesis

An overview of the whole thesis organization is described in this section. The thesis contains seven chapter starts from Chapter 1 Introduction. This chapter consists of an overview of software testing, problem statement, objective, scope, and structure of the thesis.

Chapter 2 Literature Review, describes briefly the background of the project techniques and methods. This chapter contains an overview of test case generation and testing technique, Model-Based Testing, automated test case generation, comparison of different existing Model-Based Testing tools, related works that include a table of summary of some related previous works and a summary.

Chapter 3 Research Methodology, elaborates the overall method of conducted research. An overview, research methodology of the project that includes all the phases that have followed throughout the conducted research such, literature review phase, analysis phase, system development phase, evaluation of the proposed system phase, project conclusion and documentation phase, and a summary.

Chapter 4 Proposed System: Automated test case generation tool, a brief description of the techniques and methods that have used in the proposed system for developing the tool. This chapter consists of an overview, proposed model, a broad description of Model-Based Testing approach including modeling, UML activity diagram, steps of applied techniques to generate test cases, test case generation and a summary of the chapter.

Chapter 5 Design and Implementation, provides all the detail design of the proposed system and implementation process with screenshots of interface of the tool. The chapter consists of an overview, functionality of the proposed tool, design of the proposed tool that includes different types of UML structural and behavioral diagrams, implementation of proposed tool that includes description of system environment and system development with screen images of the proposed tool.

Chapter 6 Testing and Evaluation, explains all the testing process of the proposed tool and also the details evaluation methods. This chapter consists of an overview, functional testing includes unit testing and integration testing, evaluation includes usability testing and evaluation result, and a summary of the chapter.

Chapter 7 Conclusion and Recommendations for future work, this final chapter summarizes the overall project work. The chapter starts from an overview, conclusion, contribution, limitation, and ends with the recommendation for future work.

REFERENCES

- Anbunathan, R., & Basu, A. (2017). Executable Test Generation from UML Activity Diagram Using Genetic Algorithm. *International Journal of Computer Science and Information Technology & Security (IJCSITS)*.
- Chouhan, C., Shrivastava, V., Sodhi, S. P., Soni, P. (2013). Test Case Generation on the Origin of Activity Diagram for Navigational Mobiles. *International Journal of Advanced Computational Engineering and Networking*.
- Fruhling, A., & Lee, S. (2005). Assessing the Reliability, Validity and Adaptability of PSSUQ. In *Americas Conference on Information Systems (AMCIS)*.
- Galin, D. (2004). *Software Quality Assurance From theory to implementation*, Pearson Education.
- Hooda, I., & Chhillar, R. (2014). A Review: Study of Test Case Generation Techniques. *International Journal of Computer Applications*, 107(16), 33-37. doi:10.5120/18839-0375
- Jain, S. P., Lalwani, K. S., Mahajan, N. K., Gadekar, B. J. (2014). Automatic Test Case Generation Using UML Models. *International Journal of Advanced Computational Engineering and Networking*.
- Jupudy, I., Saraf, N., Manjula, R. (2016). Comparative Analysis of Model Based and Formal Based Software Testing Methods. *International Journal of Advanced Research in Computer Science and Software Engineering*, Volume 6, Issue 3, ISSN: 2277 128X.
- Kaur, P., & Gupta, G. (2013). Automated Model-Based Test Path Generation from UML Diagrams via Graph Coverage Techniques. *International Journal of Computer Science and Mobile Computing*.
- Kaushik, S., & Tyagi, K. (2016). Critical Review on Test Case Generation Systems and Techniques. *International Journal of Computer Applications*, 133(7), 24-29. doi:10.5120/ijca2016907916.

- Liang, Y. (2003). From use cases to classes: a way of building object model with UML. *Information and Software Technology*, 45(2), 83-93. doi:10.1016/s0950-5849(02)00164-7
- Nikiforova, O., Putintsev, S., & Ahilčenoka, D. (2016). Analysis of Sequence Diagram Layout in Advanced UML Modelling Tools. *Applied Computer Systems*, 19(1). doi:10.1515/acss-2016-0005
- Patil, S. S., Jadhav, P. A. (2017). Functional Test Case Generation based on Model Driven Testing using FSM and UML Activity Diagram. *International Journal of Advanced Research in Computer Science*.
- Pressman, R. S. (2001). *Software Engineering: A Practitioner's Approach*, 7th Edition, McGraw-Hill Press.
- Priya, S. S., & Sheba, P. D. (2013). Test Case Generation From UML Models—A Survey. In *International Conference on Information Systems and Computing (ICISC)*.
- Rumbaugh, J., Jacobson, I., Booch, G. (2005). *The Unified Modeling Language Reference Manual (2nd edition)*, Addison-Wesley, Reading, MA.
- Rushby, J. (2007). Automated Test Generation and Verified Software. In *Conference on Verified Software: Theories, Tools, and Experiments (VSTTE), Zurich, Switzerland*.
- Shah, S., Shahzad, R., Bukhari, S., & Humayun, M. (2016). Automated Test Case Generation Using UML Class & Sequence Diagram. *British Journal of Applied Science & Technology*, 15(3), 1-12. doi:10.9734/bjast/2016/24860
- Sharma, H. K., Singh, S. K., Ahlawat, P. (2014). Model-Based Testing: The New Revolution in Software Testing. *Database Systems Journal*.
- Sogeti, C. (2015-16). *World Quality Report*. Participants: 1560 executives. Retrieve from <https://www.capgemini.com/thought-leadership/world-quality-report-2015-16>, 80 pp.

- Suhag, V., & Bhatia, R. (2014). Model based Test Cases Generation for Web Applications. *International Journal of Computer Applications*, 92(3), 23-31. doi:10.5120/15991-4948
- Sumalatha, V. M. & Raju, G. S. V. P. (2012). UML Based Automated Test Case Generation Technique Using Activity-Sequence Diagram. *The International Journal of Computer Science & Applications (TIJCSA)*, Volume 1, No. 9, ISSN – 2278-1080.
- Swain, S. K., Pani, S. K., Mohapatra, D. P. (2010). Model Based Object-Oriented Software Testing. *Journal of Theoretical and Applied Information Technology*.
- Teixeira, F. A., & Silva, G. B. (2017). EasyTest: An Approach for Automatic Test Cases Generation from UML Activity Diagrams. *Advances in Intelligent Systems and Computing Information Technology - New Generations*, 411-417. doi:10.1007/978-3-319-54978-1_54
- Utting, M., Pretschner, A., & Legeard, B. (2011). A taxonomy of model-based testing approaches. *Software Testing, Verification and Reliability*, 22(5), 297-312. doi:10.1002/stvr.456
- Wang, Y. & Zheng, M. (2011). Test Case Generation from UML Models. *Research gate*.