AN EVALUATION ON THE COMPREHENSIBILITY OF UML ACTIVITY AND STATE CHART DIAGRAMS WITH REGARD TO MANUAL TEST CASE GENERATION

HAITHAM RAED IBRAHIM

SUPERVISOR
DR. NOR LAILY BINTI HASHIM

MASTER OF SCIENCE (INFORMATION TECHNOLOGY)
UNIVERSITI UTARA MALAYSIA
2017

AN EVALUATION ON THE COMPREHENSIBILITY OF UML ACTIVITY AND STATE CHART DIAGRAMS WITH REGARD TO MANUAL TEST CASE GENERATION

A dissertation submitted to Dean of Awang Had Salleh Graduate School
in Partial Fulfillment of the required for
Master of Science (Information Technology)
University Utara Malaysia

By

Haitham Raed Ibrahim

Permission to Use

In presenting this dissertation in partial fulfillment of the requirements for a postgraduate degree from Universiti Utara Malaysia, I agree that the Universiti Library may make it freely available for inspection. I further agree that permission for the copying of this dissertation in any manner, in whole or in part, for the scholarly purpose may be granted by my supervisor(s) or, in their absence, by the Dean of Awang Had Salleh Graduate School of Arts and Sciences. It is understood that any copying or publication or use of this dissertation or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to Universiti Utara Malaysia for any scholarly use which may be made of any material from my dissertation.

Requests for permission to copy or to make other use of materials in this project dissertation, in whole or in part, should be addressed to:

Dean of Awang Had Salleh Graduate School of Arts and Sciences

UUM College of Arts and Sciences

Universiti Utara Malaysia

06010 UUM Sintok

Abstrak

Gambar rajah aktiviti dan *statechart* adalah gambar rajah UML yang paling kerap digunakan untuk menguji sistem berdasarkan spesifikasinya. Salah satu ciri penting gambar rajah UML adalah boleh difahami. Analisis kandungan kajian terdahulu menekankan kekurangan penilaian pakar mengenai kefahaman gambar rajah aktiviti dan *statechart* berkaitan dengan penjanaan kes ujian. Oleh itu, objektif utama kajian ini adalah bagi menilai kefahaman pakar penguji perisian ke atas gambar rajah aktiviti dan *statechart* UML dalam penjanaan kes ujian. Pertama, analisis kandungan telah dilakukan untuk mengenal pasti kriteria boleh difahami. Kriteria tersebut adalah berdasarkan kesukaran dan keyakinan subjektif. Seterusnya, satu set soalan penilaian direka berdasarkan analisis kandungan yang telah dilakukan. Kemudian, kes ujian dijana secara manual daripada gambar rajah aktiviti dan *statechart* satu kajian kes yang telah disesuaikan. Temu bual telah dijalankan dengan lima pakar untuk mengesahkan soalan penilaian yang dibentuk. Pakar tersebut menilai kefahaman ke atas gambar rajah aktiviti dan *statechart* dengan menggunakan soalan-soalan penilaian tersebut. Hasil kajian ini memberikan butiran khusus mengenai ciri yang berbeza daripada gambar rajah aktiviti dan *statechart*. Selain itu, ia mencadangkan bahawa gambar rajah aktiviti adalah lebih difahami daripada gambar rajah *statechart* dalam aspek penjanaan kes ujian. Hasil kajian ini diharapkan dapat memudahkan para penguji perisian untuk memilih satu daripada beberapa jenis gambar rajah pengujian yang sedia ada.

Abstract

The activity and state chart diagrams are the most frequently used UML diagrams for testing a system based on its specification. One of the key important qualities of the UML diagrams is their comprehensibility. The content analysis of previous studies highlighted the lack of experts' evaluation of the comprehensibility of activity and state chart diagrams with regard to test case generation. Thus, the main objective of this study is to evaluate the comprehensibility of the UML activity and state chart diagrams for test case generation. First, a content analysis was performed to identify the comprehensibility criteria. The criteria are perceived difficulty and subjective confidence. Next, a set of evaluation questions was designed based on the content analysis. Then, test cases were generated from activity and state chart diagrams manually of an adapted case study. An interview was conducted with five experts to validate the evaluation questions. The experts evaluated the comprehensibility of the activity and state chart diagrams by using the evaluation questions. The result of the study provided specific details of the different characteristics of activity and state chart diagrams. Further, it suggested that the activity diagram is more comprehensible than the state chart diagram in the aspect of test case generation. The finding of this study could assist software testers in choosing the appropriate UML diagrams for software testing.

Acknowledgement

In the name of God, the Most Gracious, the Most Merciful.

All praises and thanks to the Almighty, Allah (SWT) for helping me to finish this study. Allah gives me the opportunity, strength and the ability to complete my Master degree after a long, continuous work. No volume of words is enough to express my gratitude towards my guide, Dr. Nor Laily Binti Hashim; without her knowledge and assistance plus her recommendations, this study would not have been successful. She has helped me to explore the topic in an organised manner and provided me with all the ideas on how to work towards a research-oriented endeavour.

It would not be possible for me to complete the study without the support and encouragement from my family and friends. First and foremost, my gratitude goes to my wife Maha for supporting and providing great inspiration for me to finish my master's study. May Allah bless her and my two lovely kids Ahmed and Dania. Secondly, my father and mother and their prayers for me, my aunt Nadia Putrus, my father-in-law and mother-in-law who motivated me and gave me their endless support. Finally, to my dearest brother's soul (Ahmed). To my great friends, especially Dr. Nassir Farhan; thanks for standing beside me and giving me support for all the periods of my study.

Special thanks to all who have helped or contributed to making this study a success.

Table of Contents

Last of Tables

List of Figures

List of Appendices

List of Abbreviations

MBT	Model Base Testing
UML	Unified Modelling Language
DFS	Depth First Search
LR	Literature Review
GA	Genting Algorithm
OOAD	Object-Oriented Analysis and Design
AOAD	Aspect-Oriented Analysis and Design
IFD	Interaction Flow Diagram
ADT	Activity Dependency Table
AFG	Activity Flow Graph
TFG	Testing Flow Graph
ECFG	Extended Control Flow Graph
EFSM	Extended Finite State Machine
TeGeMiOOSc	Test Generation and Minimization for OO software with State Charts
NDT	Node Description Table

CHAPTER ONE
INTRODUCTION

1.1 Overview

This chapter provides an introduction to the study which begins with the background of the study, followed by the discussion of the problem. Subsequently, research questions are provided and used to construct the objectives. Finally, this chapter presents the scope as well as the significance of the research. This chapter is concluded with the summary of the main issue of this study.

1.2 Introduction

The software systems that exist throughout the world and their designs are rapidly developing and becoming more complex, a trend which very likely will continue in the near future (Meena, 2013). This development of complex software systems is a fault-prone process and these incur a great loss of time and money if neglected (Mailewa, Herath, & Herath, 2015). In this regard, Manaseer, Manaseer, Alshraideh, Abuhashish and Adwan (2015) and Jain, Jain and Dhankar (2014) remarked that software testing is the most widely used approach to ensure software quality that assists software faults detection.

On the same note, Bansal (2014) and Vashishtha, Singla and Singh (2014) stated that software testing typically consumes about 50% of the development effort, cost, and time to achieve a higher level of quality. Consequently, to reduce test challenges,

The contents of the thesis is for internal user only

REFERENCES

Afzal, W., & Torkar, R. (2008). Incorporating metrics in an organizational test strategy. In *Software Testing Verification and Validation Workshop, 2008. ICSTW'08. IEEE International Conference on* (pp. 304–315).

Agarwal, R., De, P., & Sinha, A. P. (1999). Comprehending object and process models: An empirical study. *IEEE Transactions on Software Engineering*, *25*(4), 541–556.

Al Dallal, J., & Sorenson, P. (2006). Generating class based test cases for interface classes of object-oriented black box frameworks. *Transactions on Engineering, Computing and Technology*, *16*, 90–95.

Alderson, J. C., & Banerjee, J. (1996). How might impact study instruments be validated. *Unpublished Manuscript Commissioned by UCLES*.

Ali, M. A., Shaik, K., & Kumar, S. (2014). Test Case Generation using UML State Diagram and OCL Expression. *International Journal of Computer Applications*, *95*(12)1-6.

Anda, B., Sjoberg, D., & Jorgensen, M. (2001). Quality and understandability of use case models. In *European Conference on Object-Oriented Programming* (pp. 402–428).

Aranda, J., Ernst, N., Horkoff, J., & Easterbrook, S. (2007). A framework for empirical evaluation of model comprehensibility. In *Proceedings of the International Workshop on Modeling in Software Engineering* (p. 7-12).

Bansal, A. (2014). A Comparative Study of Software Testing Techniques. *International Journal of Computer Science and Mobile Computing.*, *3*, 579–584.

Board, Q. (2013). Certified Tester Expert Level Modules Overview. *International Software Testing Qualifications Board*, *Version 1*.

Boghdady, P. N., Badr, N. L., Hashem, M., & Tolba, M. F. (2011a). A proposed test case generation technique based on activity diagrams. *International Journal of Engineering & Technology IJET-IJENS, 11*(3)92-97.

Boghdady, P. N., Badr, N. L., Hashem, M., & Tolba, M. F. (2011b). Test case generation and test data extraction techniques. *Inter. J. Electr. Comput. Sci*, *11*(3), 87–94.

Booch, G. (2005). *The unified modeling language user guide*. Pearson Education: India.

Budgen, D., Burn, A. J., Brereton, O. P., Kitchenham, B. A., & Pretorius, R. (2011). Empirical evidence about the UML: a systematic literature review. *Software: Practice and Experience*, *41*(4), 363–392.

Byckling, P., Gerdt, P., Kuzniarz, L., & Sajaniemi, J. (2006). Increasing comprehensibility of object models: Making the roles of attributes explicit in UML diagrams. *Nordic Journal of Computing*, *13*(3), 149-161.

Chandu, P. (2015). An Analytical Way to Improvise Test Execution and Review of Software Metrics for The Software Quality. *Journal of Theoretical and Applied Information Technology*, *73*(1)1-6.

Chism, N. V. N., Douglas, E., & Hilson Jr, W. J. (2008). Qualitative research basics: A guide for engineering educators. *Rigorous Research in Engineering Education NSF DUE-0341127*.

Choudhary, D., & Kumar, V. (2011). Software testing. *Journal of Computational Simulation and Modeling*, *1*(1), 1-5.

Condori-Fernandez, N., Daneva, M., Sikkel, K., & Herrmann, A. (2011). Practical relevance of experiments in comprehensibility of requirements specifications. In *Workshop on Empirical Requirements Engineering (EmpiRE 2011)* (pp. 21–28).

Cox, K., Phalp, K., & Shepperd, M. (2001). Comparing use case writing guidelines. In *7th International Workshop on Requirements Engineering: Foundation for Software Quality, Interlaken, Switzerland* (Vol. 45, p. 101-112).

Creswell. (2012). *Educational Research Planning, Conducting, and Evaluating Quantitative and Qualitative Research*. Sage.

Crowder, J. A., Carbone, J. N., & Demijohn, R. (2016). Systems Engineering Tools and Practices. In *Multidisciplinary Systems Engineering* (pp. 89–103). Springer.

Cruz-Lemus, J. A., Genero, M., Caivano, D., Abrahão, S., Insfrán, E., & Cars'\i, J. A. (2011). Assessing the influence of stereotypes on the comprehension of UML sequence diagrams: A family of experiments. *Information and Software Technology*, *53*(12), 1391–1403.

Delanote, D., Van Baelen, S., Joosen, W., & Berbers, Y. (2008). An automatic test data generation from UML state diagram using genetic algorithm. *2008 13th IEEE International Conference on Engineering of Complex Computer Systems*.

Doungsa-ard, C., Dahal, K. P., Hossain, M. A., & Suwannasart, T. (2007). An automatic test data generation from UML state diagram using genetic algorithm. *The University of Bradford Institutional Repository*, (http://bradscholars.brad.ac.uk).

Dubey, S. K., & Sharma, D. (2015). Software Quality Appraisal Using Multi-Criteria Decision Approach. *I.J. Information Engineering and Electronic Business, 1530-1362/04*.

Eriksson, H.-E., & Penker, M. (2000). Business modeling with UML. *Business Patterns at Work, John Wiley & Sons: New York, USA*.

Felderer, M., Büchler, M., Johns, M., Brucker, A. D., Breu, R., & Pretschner, A. (2015). Security Testing: A Survey. *Advances in Computers*.

Felderer, M., & Herrmann, A. (2015). Manual test case derivation from UML activity diagrams and state machines: A controlled experiment. *Information and Software Technology*, *61*, *1–15*, *61*, 1–15.

Figl, K., & Laue, R. (2011). Cognitive complexity in business process modeling. In *International Conference on Advanced Information Systems Engineering* (pp. 452–466).

Garg, P. (2015). Role of Testing Strategies in Improving Quality of Software. *International Journal of Research*, *2*(12), 800–805.

Gravino, C., Scanniello, G., & Tortora, G. (2008). An empirical investigation on dynamic modeling in requirements engineering. In *International Conference on Model Driven Engineering Languages and Systems* (pp. 615–629).

Gulia, P., & Chugh, J. (2015). Comparative Analysis of Traditional and Object-Oriented Software Testing. *ACM SIGSOFT Software Engineering Notes*, *40*(2), 1–4.

Gupta, J. (2014). *An Investigation of Test Cases Generation from Activity Diagram*. Thesis. Thapar University Patiala.

Gupta, N., Yadav, V., & Singh, M. (2016). A Review on Automated Regression Testing. *International Journal of Engineering Technology Science and Research*, *3*, 46–50.

Habib, Z. (2009). *The critical success factors in implementation of software process improvement efforts: CSFs, motivators & obstacles*. University of Gothenburg.

Hadar, I., & Hazzan, O. (2004). On the contribution of UML diagrams to software system comprehension. *Journal of Object Technology*, *3*(1), 143–156.

Harel, D., & Politi, M. (1998). *Modeling reactive systems with statecharts: the STATEMATE approach*. McGraw-Hill, Inc.

Hashim, N. L., & Salman, Y. D. (2011). An Improved Algorithm With regard to test case generation From Uml Activity Diagram Using Activity Path. In *Proceedings of the 3rd International Conference on Computing and Informatics, ICOCI*.

Heinecke, A., Bruckmann, T., Griebe, T., & Gruhn, V. (2010). Generating test plans for acceptance tests from uml activity diagrams. In *Engineering of Computer Based Systems (ECBS), 2010 17th IEEE International Conference and Workshops on* (pp. 57–66).

Ingle, S. E., & Mahamune, M. R. (2015). An UML Based software Automatic Test Case Generation: Survey. *International Research Journal of Engineering and Technology*, *2*, 971–973.

Jain, A., Jain, M., & Dhankar, S. (2014). A Comparison of RANOREX and QTP Automated Testing Tools and their impact on Software Testing. *Nternational Journal of Engineering, Management & Sciences (IJEMS) ISSN-2348*, *1*(1), 8–12.

Jena, A. K., Swain, S. K., & Mohapatra, D. P. (2014a). ? In *Issues and Challenges in Intelligent Computing Techniques (ICICT), 2014 International Conference on* (pp. 621–629).

Jena, Swain, & Mohapatra. (2014b). A novel approach for test case generation from UML *International Conference on* activity diagram. In *(ICICT),* (pp. 621–629).

Kansomkeat, S., Offutt, J., Abdurazik, A., & Baldini, A. (2008). A comparative evaluation of tests generated from different UML diagrams. *International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2008. SNPD'08. Ninth ACIS* (pp. 867–872).

Kansomkeat, S., & Rivepiboon, W. (2003). Automated-generating test case using UML statechart diagrams. In *Proceedings of the 2003 annual research conference of the South African institute of computer scientists and information technologists on Enablement through technology* (pp. 296–300).

Kaur, G., & Bajaj, A. (2015). A Notation of Unified Modeling Language in various Areas. *Asian Journal of Research in Social Sciences and Humanities*, *5*(3), 195–205.

Khandai, M., Acharya, A. A., & Mohapatra, D. P. (2011a). A novel approach of test case generation for concurrent systems using UML Sequence Diagram. In *Electronics Computer Technology (ICECT), 2011 3rd International Conference on* (Vol. 1, pp. 157–161).

Khandai, M., Acharya, A. A., & Mohapatra, D. P. (2011b). Test case generation for concurrent system using UML combinational diagram. *International Journal of Computer Science and Information Technologies, IJCSIT*, *2*-11-18.

Khurana, N., Chhillar, R. S., & Chhillar, U. (2016). A Novel Technique for Generation and Optimization of Test Cases Using Use Case, Sequence, Activity Diagram and Genetic Algorithm. *Journal of Software*, *11*, 242–250.

Kim, H., Kang, S., Baik, J., & Ko, I. (2007). Test cases generation from UML activity diagrams. In *Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007. SNPD 2007. Eighth ACIS International Conference on* (Vol. 3, pp. 556–561).

Kingston, J., & Macintosh, A. (2000). Knowledge management through multi-perspective modelling: representing and distributing organizational memory. *Knowledge-Based Systems*, *13*(2), 121–131.

Konka, B. B. (2012). *A case study on Software Testing Methods and Tools*. University of Gothenburg.

Koriat, A. (2011). Subjective confidence in perceptual judgments: a test of the self-consistency model. *Journal of Experimental Psychology: General*, *140*(1), 117.

Kosindrdecha, N., & Daengdeg, J. (2010). A test generation method based on state diagram. *Journal of Theoretical and Applied Information Technology*, 28–44.

Kouider, S., De Gardelle, V., Sackur, J., & Dupoux, E. (2010). How rich is consciousness? The partial awareness hypothesis. *Trends in Cognitive Sciences*, *14*(7), 301–307.

Kramer, A., & Legeard, B. (2016). *Model-Based Testing Essentials-Guide to the ISTQB Certified Model-Based Tester-Foundation Level*. John Wiley & Sons.

Kundu, D., & Samanta, D. (2009). A Novel Approach to Generate Test Cases from UML Activity Diagrams. *Journal of Object Technology*, *8*(3), 65–83.

Kundu, D., Sarma, M., Samanta, D., & Mall, R. (2009). System testing for object-oriented systems with test case prioritization. *Software Testing, Verification and Reliability*, *19*(4), 297–333.

Kuzniarz, L., Staron, M., & Wohlin, C. (2004). An empirical study on using stereotypes to improve understanding of UML models. In *Program Comprehension, 2004. Proceedings. 12th IEEE International Workshop on* (pp. 14–23).

Lewins, A., & Silver, C. (2007). Using Software for Qualitative Data Analysis: A Step-by-Step Guide. Thousand Oaks, Calif.: Sage.

Lewis, W. E. (2016). *Software testing and continuous quality improvement*. CRC press.

Li, L., Li, X., He, T., & Xiong, J. (2013). Extenics-based test case generation for UML activity diagram. *Procedia Computer Science*, *17*, 1186–1193.

Liebel, M. T. G., & Tichy, M. (2015). Comparing Comprehensibility of Modelling Languages for Specifying Behavioural Requirements. In *First International Workshop on Human Factors in Modeling (HuFaMo 2015). CEUR-WS* (pp. 17–24).

Linzhang, W., Jiesong, Y., Xiaofeng, Y., Jun, H., Xuandong, L., & Guoliang, Z. (2004). Generating test cases from UML activity diagram based on gray-box method. *Software Engineering Conference, 2004. 11th Asia-Pacific*, 284–291.

Lucantonio, V. (2015). Enanching the consistency between requirements and test cases through the definition of a Controlled Natural Language. *Digital Vetenskapliga Arkivet.*

Macintosh, A., Coleman, S., & Schneeberger, A. (2009). eParticipation: The research gaps. In *Electronic participation* (pp. 1–11). Springer.

Mailewa, A. B. (2015). Reducing Software Testing Time with Combinatorial Testing and Test Automation. *St. Cloud State University the Repository at St. Cloud State.*

Mailewa, A., Herath, J., & Herath, S. (2015). A Survey of Effective and Efficient Software Testing. *Micsymposium.org*. Retrieved from http://www.micsymposium.org

Manaseer, S., Manaseer, W., Alshraideh, M., Abuhashish, N., & Adwan, O. (2015). Automatic Test Data Generation for Java Card Applications Using Genetic Algorithm. *Journal of Software Engineering and Applications*, *8*(12), 603-609.

McQuillan, J. A., & Power, J. F. (2005). A survey of UML-based coverage criteria for software testing. *Department of Computer Science. NUI Maynooth, Co. Kildare, Ireland*.

Meena, D. K. (2013). *Test Case Generation From UML Interaction Overview Diagram and Sequence Diagram*. National Institute of Technology Rourkela.

Mendonca, M. G., Maldonado, J. C., De Oliveira, M. C. F., Carver, J., Fabbri, S. C. P. F., Shull, F., … Basili, V. R. (2008). A framework for software engineering experimental replications. In *Engineering of Complex Computer Systems, 2008. ICECCS 2008. 13th IEEE International Conference on* (pp. 203–212).

Miles, M. B., & Huberman, A. M. (1994). *Qualitative data analysis: An expanded sourcebook*. Sage.

Mingers, J. (2001). Combining IS research methods: towards a pluralist methodology. *Information Systems Research*, *12*(3), 240–259.

Mingsong, C., Xiaokang, Q., & Xuandong, L. (2006). Automatic test case generation for UML activity diagrams. In *Proceedings of the 2006 international workshop on Automation of software test* (pp. 2–8).

Mohanty, S., Acharya, A. A., & Mohapatra, D. P. (2011). A model based prioritization technique for component based software retesting using uml state chart diagram. In *Electronics Computer Technology (ICECT), 2011 3rd International Conference on* (Vol. 2, pp. 364–368).

Mussa, M., Ouchani, S., Al Sammane, W., & Hamou-Lhadj, A. (2009). A survey of model-driven testing techniques. In *Quality Software, 2009. QSIC'09. 9th International Conference on* (pp. 167–172).

Nayak, A., & Samanta, D. (2011). Synthesis of test scenarios using UML activity diagrams. *Software & Systems Modeling*, *10*(1), 63–89.

Nikfard, P., bin Ibrahim, S., Rohani, B. D., bin Selamat, H., & Naz'ri, M. (2013). An Evaluation for Model Testability approaches. *International Journal Of Computers & Technology*, *9*(1), 938–947.

Oluwagbemi, O., & Asmuni, H. (2015). An Approach for Automatic Generation of Test Cases from UML Diagrams. *International Journal of Software Engineering and Its Applications*, *9*(8), 87–106.

Otero, M. C., & Dolado, J. J. (2004). Evaluation of the comprehension of the dynamic modeling in UML. *Information and Software Technology*, *46*(1), 35–53.

Pahwa, N., & Solanki, K. (2014). UML based test case generation methods: A review. *International Journal of Computer Applications*, *95*(20)311-316.

Pandey, N., & Mohapatra, D. P. (2012). Test Case Generation from UML Interaction Diagrams. In *International Conference on Computing* (p. 17-24).

Patel, P. E., & Patil, N. N. (2013). Testcases Formation using UML Activity Diagram. In *Communication Systems and Network Technologies (CSNT), 2013 International Conference on* (pp. 884–889).

Patil, K., & Ganeshwade, M. M. (2014). Generating Automated Test Cases using Model Based Testing. *International Journal of Enhanced Research in Science Technology & Engineering*, *3*, 18–24.

Razali, R., Snook, C. F., & Poppleton, M. R. (2007). Comprehensibility of uml-based formal model: A series of controlled experiments. In *Proceedings of the 1st ACM international workshop on empirical assessment of software engineering languages and technologies: Held in conjunction with the 22nd IEEE/ACM international conference on automated software engineering (ASE) 2007* (pp. 25–30).

Razali, R., Snook, C., Poppleton, M., Garratt, P., & Walters, R. (2007). Usability assessment of a UML-based formal modelling method. In *19th Annual Psychology of Programming Workshop (PPIG'07)* (pp. 56–71).

Reza, H., Ogaard, K., & Malge, A. (2008). A model based testing technique to test web applications using statecharts. In *Fifth International Conference on Information Technology: New Generations* (pp. 183–188).

Ribiero, N. F., & Yarnal, C. M. (2010). The Perceived Difficulty Assessment Questionnaire (PDAQ): Methodology and Applications for Leisure Educators and Practitioners. *Schole*, *25*.

Robson, C., & McCartan, K. (2016). *Real world research*. John Wiley & Sons.

Rumbaugh, J., Jacobson, I., & Booch, G. (2004). *Unified Modeling Language Reference Manual*. Pearson Higher Education.

Salman, Y. D., & Hashim, N. L. (2014). An Improved Method of Obtaining Basic Path Testing for Test Case Based On UML State Chart. *Science International*, *26*(4)1-8.

Salman, Y. D., & Hashim, N. L. (2016). Automatic Test Case Generation from UML State Chart Diagram: A Survey. In *Advanced Computer and Communication Engineering Technology* (pp. 123–134). Springer.

Samuel, P., Mall, R., & Bothra, A. K. (2008). Automatic test case generation using unified modeling language (UML) state diagrams. *IET Software*, *2*(2), 79–93.

Scanniello, G., Gravino, C., Risi, M., Tortora, G., & Dodero, G. (2015). Documenting design-pattern instances: a family of experiments on source-code comprehensibility. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *24*(3), 14.

Schweighofer, T., & Hericko, M. (2014). Approaches for Test Case Generation from UML Diagrams. In *SQAMIA* (pp. 91–98).

Sekaran, U., & Bougie, R. (2010). Research Design. *JW Ltd, Research Methods for Business-*, 110.

Sharma, S., & Vishawjyoti, M. (2013). Study And Analysis Of Automation Testing Techniques. *Journal of Global Research in Computer Science*, *3*(12), 36–43.

Shirole, M., & Kumar, R. (2013). UML behavioral model based test case generation: a survey. *ACM SIGSOFT Software Engineering Notes*, *38*(4), 1–13.

Shirole, M., Suthar, A., & Kumar, R. (2011). Generation of improved test cases from UML state diagram using genetic algorithm. In *Proceedings of the 4th India Software Engineering Conference* (pp. 125–134).

Shneiderman, B. (1992). Designing the user interface: strategies for effective human-computer interaction, 2nd edn. Addison. *Reading, MA*.

Shukla, D. (2014). Analyzing the Comprehensibility of Aspect-Oriented Modelling and Design of Software System. *International Journal of Computer Applications*, *95*(21).

Shukla, & Chandel. (2012). A Systematic Approach for Generate Test Cases Using UML Activity Diagrams. *International Journal of Research in Management & Technology (IJRMT), ISSN: 2249-9563*, *2*, 469–475.

Siau, K., & Cao, Q. (2001). Unified modeling language: A complexity analysis. *Journal of Database Management (JDM)*, *12*(1), 26–34.

Sinkovics, R. R., & Alfoldi, E. A. (2012). Progressive focusing and trustworthiness in qualitative research. *Management International Review*, *52*(6), 817–845.

Sommerville, I. (2010). Software testing. *Ian Sommerville 2004, Software Engineering, 7th Edition, Prechelt@inf.fu-Berlin.de*, *7*.

Swain, R. K., Behera, P. K., & Mohapatra, D. P. (2012). Minimal TestCase Generation for Object-Oriented Software with State Charts. *arXiv Preprint arXiv:1208.2265*.

Swain, S. K., Mohapatra, D. P., & Mall, R. (2010). Test Case Generation Based on State and Activity Models. *Journal of Object Technology*, *9*(5), 1–27.

Thanki, M. H. J., & Shinde, S. M. (2014). Test case generation using UML activity diagram-A survey, *3*, 292–300.

Tripathy, A., & Mitra, A. (2013). Test Case Generation Using Activity Diagram and Sequence Diagram. In *Proceedings of International Conference on Advances in Computing* (pp. 121–129).

Utting, M., & Legeard, B. (2010). *Practical model-based testing: a tools approach*. Morgan Kaufmann.

Vashishtha, V., Singla, T., & Singh, S. (2014). Software Testing. *International Journal of Research*, *1*(8), 1258–1264.

Verma, S., Yadav, K. P., & Tiwari, U. K. (2012). Software Testing Techniques for Finding Errors. *International Journal of Research Review in Engineering Science and Technology*, *2*, 433–439.

Wang, L. (2015). GUI test automation for Qt application. *Digital Vetenskapliga Arkivet.*

Wang, X., Jiang, X., & Shi, H. (2015). Prioritization of test scenarios using hybrid genetic algorithm based on UML activity diagram. In *Software Engineering and Service Science (ICSESS), 2015 6th IEEE International Conference on* (pp. 854–857).

Willig, C., & Stainton-Rogers, W. (2007). *The SAGE handbook of qualitative research in psychology*. Sage.

Xie, S., Kraemer, E., & Stirewalt, R. E. K. (2007). Empirical evaluation of a UML sequence diagram with adornments to support understanding of thread interactions. In *15th IEEE International Conference on Program Comprehension (ICPC'07)* (pp. 123–134).

Yang, N., Yu, H., Sun, H., & Qian, Z. (2010). Mapping uml activity diagrams to analyzable petri net models. In *2010 10th International Conference on Quality Software* (pp. 369–372).

Yu, L., Xu, X., Liu, C., & Sheng, B. (2012). Using grounded theory to understand testing engineers' soft skills of third-party software testing centers. In *Software Engineering and Service Science (ICSESS), 2012 IEEE 3rd International Conference on* (pp. 403–406).

Zhu, H., Hall, P. A. V, & May, J. H. R. (1997). Software unit test coverage and adequacy. *Acm Computing Surveys (Csur)*, *29*(4), 366–427.