www.ijecs.in
*International Journal Of Engineering And Computer Science ISSN:2319-7242*
*Volume 4 Issue 6 June 2015, Page No. 12774-12780*

# Review of model based approach for automating the test case generation for Object Oriented Systems.

***Rajvir Singh[1], Preeti[2]***

[1]Assistant Professor, CSE Department,
Deenbandhu Chhotu Ram University, Murthal.
Rajvirsingh.cse@dcrustm.org

[2]Student, M.Tech(CSE),
Deenbandhu Chhotu Ram University, Murthal.
Preeti.lohchab32@gmail.com

**ABSTRACT: Testing plays an important part in software development process to ensure the quality and reliability of the developed product. For Object-oriented systems, model based testing has recently become very popular. This approach uses models representing system behavior to generate the test cases. In this paper, we would focus on the work done by various researchers in the field of model based testing approach. We would review the recent trends and different model-based approaches that have been proposed by different researchers. Finally, we would describe what are the present challenges and what work needs to be done in near future in this area.**

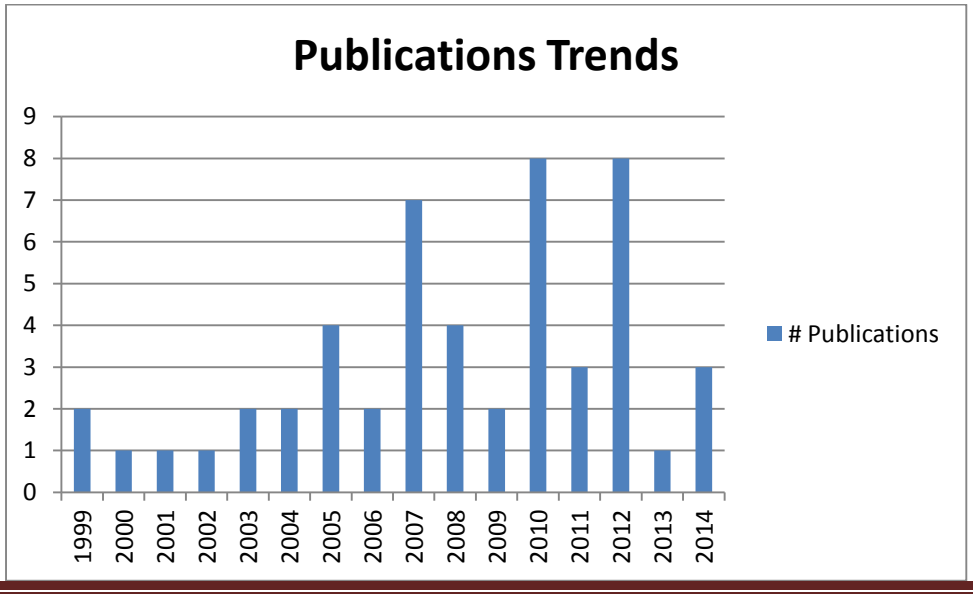
**KEYWORDS:** Object-oriented Software, Model Based Testing, Testing, Design based approach

## 1. INTRODUCTION

Software testing plays central role in ensuring the quality of delivered software product. But as the size and complexity of software increases it becomes increasingly difficult to test the product thoroughly. For Object oriented systems this task becomes more complex because it has to deal with new problems introduced by the Object-Oriented features such as *encapsulation, inheritance, polymorphism, and dynamic binding.* Interactions between objects may give rise to certain errors that could be hard to detect. Object-oriented environment for design and implementation of software brings about new issues in software testing. This is because the above important features of an object oriented program create several testing problems and bug hazards [1].

This paper presents a systematic literature review to analyze and report the findings in automated test case generation for object-oriented systems. Literature reviews provide a comprehensive view of the present research and allows one to find out the possible gaps in the present research.

The recent research trend in the field of automated test case generation for object-oriented systems is shown by the graph below.



As is clear from the graph, research in this field has attracted a lot of researchers recently. This number has grown over the years.

Rest of this paper is organized as follows: section 2 gives a background of the object-oriented test case generation process. Section 3 describes the review method that has been adopted for this paper; section 4 discusses the search result. Section 5 deals with the literature survey; section 6 covers the future work and section 7 includes the conclusion.

## 2. BACKGROUND:

According to IEEE testing is,

> "The process of exercising or evaluating system or system components by manual or automated means to verify that it satisfies specified requirements".

In other words, "Testing is the process of executing a program with the intent of finding errors".

Software Testing involves three processes: Test Case Generation, Test Case Execution and Test Case Evaluation. Manual testing is time-consuming, labor-intensive and error-prone. Therefore, it is required to automate the testing effort. Automation, which automates a part of testing process, reduces the human effort in finding bugs and errors.

There are mainly three types of testing approaches:

**Model based testing:**
Models are the intermediate artifacts between requirement specification and final code. Models preserve the essential information from the requirement, and are the basis for implementation.
A model of software is a depiction of its behavior where behavior can be described in terms of the input sequences accepted by the system, the set of actions, conditions, the flow of data through the application's modules and routines. For example, control flow, data flow, and program dependency graphs. Model based testing uses these models as bases for generating test cases for the system under test. It combines both black and white box testing features and is also called gray box approach.

**Specification based testing:**
In this approach test cases are derived directly from the specification or from some other kind of model of what the system should do. It is essentially a black box approach.

**Code based testing:**
It ensures that each and every statement in the program is executed at least once during the test. It is a white box approach.

Code based testing is not an entirely satisfactory approach to ensure thorough testing of modern software products. Code based testing has two important disadvantages. First, certain aspects of behavior of a system are difficult to extract from code but are easily obtained from design models. e.g., all different sequences in which messages may be interchanged among classes during the use of software are very difficult to extract from the code. Another prominent disadvantage of code based testing is, it is very difficult to automate and code based testing overwhelmingly depends on manual test case design, [2].

Now-a-days model based testing methodology has gained popularity for testing the object oriented system and has become an obvious choice in software industries. It has following advantages over other two testing approaches:

1) Traditional software testing techniques consider only static view of code which is not sufficient for testing dynamic behavior of object-oriented systems, [3].
2) Use of code to test an object-oriented system is complex and tedious task. In contrast, models help software testers to understand systems in a better way and find test information only after simple processing of models compared to code, [4].
3) Model-based test case generation can be planned at an early stage of the software development life cycle, allowing carrying out coding and testing in parallel [4].

Because of these advantages most of the researchers have focused on the model based test case generation approach. Therefore, this paper focuses on the work done in the field of model based test case generation for object-oriented systems.

## 3. Review Method:

This paper is based on systematic literature review which addresses the following research questions:

1. How much work has been done in the field of automated test case generation for object-oriented systems?
2. How much of the work identified in first question focuses on design based test case generation approach?
3. What are the limitations of current research?

### 3.1 Search Process

The research process was a manual search of different journal papers, surveys and conferences related to automated test case generation since 1999. The selected journals and conferences are shown in table 1. Each of these journals deals with the issue of automating the test case generation process.

**Table 1**

| Source | Acronym |
|---|---|
| Institute of electrical and electronics engineers | IEEE |
| Journal of object technology | JOT |
| International journal of software engineering | IJSE |
| Indian journal of computer science and engineering | IJCSE |
| International Journal of Advanced Computer Science and Applications | IJACSA |
| International Journal of Software Engineering & Applications | IJSEA |
| International Journal of Inventive Engineering and Sciences | IJIES |

3.2 Inclusion and Exclusion criteria

Articles published from 1999 to 2014 were included for the research. As a first step, irrelevant papers were excluded manually based on titles. Papers dealing with software testing and test case generation in general were also excluded. All the journals dealing with automated test case generation for object-oriented system were included. Only studies written in English were considered. Papers with revised versions were included and their initial versions were discarded.

## 4. Search Result:

After applying the search criteria described in the previous section, 32 articles have been identified as relevant to our topic of research. Out of these 32 articles 20 focus on the model based approach, 4 on specification based approach, 1 on code based testing and remaining 7 articles use some other approaches for test case generation. These 32 articles along with brief information about them have been listed in the table 2.

**Table 2**

| S.N | Title | main journal | Journal/ Conference | Authors | Year | Pages | Main Tool(s) | Algorithm | Basic Approach used | Type of Testing | Metrics used for effectiveness measure | Remarks |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | The State-based Testing of Object-Oriented Programs | IEEE | | C.D. Turner, D.J. Robson | 1993 | 302-310 | | | emphasis on interaction between features and object's state | | | complementary to other functional and structural approaches to validation |
| 2 | Automatic Test Case Generation from Requirements Specifications for Real-time Embedded Systems | IEEE | | S.J. Cunning, J.W. Rozenblit | 1999 | v 784-v 789 | | | | | | |
| 3 | Test Case Generation from UML State Diagrams | IEEE | | Y.J. Kim, H.S. Hong, S.M. Cho, D.H. Bae, S.D. Cha | 1999 | 17 | | na | based on control flow and data flow | class level testing | | specification based approach, class level testing |
| 4 | A Test Class Framework for Generating Test Cases from Z Specifications | IEEE | | MIAO Huaikou, LIU Ling | 2000 | 164-171 | developed a TCGS tool | na | uses concept of test classes and test class framework | | | test cases are derived from z specifications |
| 5 | Interclass Testing of Object Oriented Software | | Technical Report GIT-CC-02-28 | Vincenzo Martena,Alessandro Orso, Mauro Pezzue | 2002 | 28 | | | based on data flow analysis | interclass testing | | tackles the problem of state-dependent faults |
| 6 | Generating Test Cases from UML Activity Diagram based on Gray-Box Method | IEEE | 11th Asia-Pacific Software Engineering Conference (APSEC'04) | Wang Linzhang, Yuan Jiesong, Yu Xiaofeng, Hu Jun, Li Xuandong, Zheng Guoliang | 2004 | 8 | developed a prototype tool UMLTFG | | uses UML activity diagrams | gray box testing | | |
| 7 | Automated Generation of Test Programs from Closed Specifications of Classes and Test Cases | IEEE | 26th International Conference on Software Engineering | Wee Kheng Leow, Siau Cheng Khoo, Yi Sun | 2004 | 10 | | | based on the generation of a test program | | | specification based testing |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8 | A Survey on Automatic Test Case Generation | | | M.Prasanna, S.N Sivanandam, R.Venkatesan, R Sundarrajan | 2005 | 6 | | | focuses on various testing approaches used for automatic test case generation | | | |
| 9 | Model Based Object-oriented Software Testing | | | Santosh Kumar Swain, Subhendu Kumar Pani, Durga Prasad Mohapatra | 2005 | 30-36 | | | focuses on various model based testing techniques for automatic test case generation | | | model based testing |
| 10 | Automatic Test Generation A Use Case Driven Approach | IEEE | | Cle mentine Nebut, Franck Fleurey, Yves Le Traon, Jean-Marc Je´ze´ quel | 2005 | 140-155 | | uses a custom algo for designing use case transition system, test objectives | uses a use case driven approach for embedded systems | | statement coverage | model based testing |
| 11 | UML Sequence Diagram Based Testing Using Slicing | IEEE | IEEE Indicon 2005 Conference | Philip Samuel, Rajib Mall and Sandeep Sahoo | 2005 | 176-178 | | | uses UML sequence diagrams and dynamic slicing technique for testing | | slice test coverage | model based testing |
| 12 | An Approach to Integration Testing of Object-Oriented Programs | IEEE | Seventh International Conference on Quality Software | Zhe (Jessie) Li, Tom Maibaum | 2007 | 6 | developed a prototype tool | | uses the concept of coordination Contract | integration testing | | |
| 13 | Automatic Test Case Generation from UML Models | IEEE | 10th International Conference on Information Technology | Monalisa Sarma,Rajib Mall | 2007 | 196-201 | | | uses the combination of sequence diagrams and use case diagrams | | use case dependency coverage and all sequence diagram message path sequence | model based testing |
| 14 | Automatic Test Case Generation from UML Sequence Diagrams | IEEE | 15th International Conference on Advanced Computing | Monalisa Sarma,Rajib Mall | 2007 | 60-65 | | | uses sequence diagrams,use case template and class diagrams to develop test vectors | system testing | sequence diagram message path sequence coverage | model based testing |
| 15 | Automatic testing of object-oriented software | | | Bertrand Meyer, Ilinca Ciupa, Andreas Leitner, Lisa (Ling) Liu | 2007 | 17 | a test framework AUTOTEST is used | | based on automation of test case and test oracles generation | unit(class testing) | | automatically retains information for regression testing |
| 16 | Kiasan/KUnit: Automatic Test Case Generation and Analysis Feedback for Open Object-oriented Systems | IEEE | Academic and Industrial Conference - Practice And Research Techniques | Xianghua Deng, Robby, John Hatcliff | 2007 | 3,12 | | | | | | |
| 17 | A Novel Approach to Generate Test Cases from UML Activity Diagrams | JOT | | Debasish Kundu, Debasis Samanta | 2009 | 65-83 | | uses a custom algo based on DFS and BFS | uses UML 2.0 syntax to generate test cases from activity diagrams of a single use case | | activity path coverage | model based testing |
| 18 | Research on Method of Object-Oriented Test Cases Generation Based on UML and LTS | IEEE | The 1st International Conference on Information Science and Engineering (ICISE2009) | Fanping Zeng, Zhide Chen, Qing Cao, Lianglang Mao | 2009 | 5055-5058 | | | uses UML state diagram model that represent state transition to generate test cases | | | model based testing |
| 19 | A Test Case Generation Technique and Process | | | Nicha Kosindrdecha, Jirapun Daengdej | 2010 | 8 | | | based on requirement prioritizationmethod and uses use cases for test case generation | | no of test cases, domain specific requirement coverage, total time | model based testing |
| 20 | Test Case Generation Based on State and Activity Models | JOT | | Santosh Kumar Swain, Durga Prasad Mohapatra, Rajib Mall | 2010 | 27 | | custom algo based on DFS | uses a combination of state and activity models | integration testing | state-activity coverage, transition coverage, activity path coverage | model based testing |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 21 | Test Case Generation Based on Use case and Sequence Diagram | IJSE | | Santosh Kumar Swain, Durga Prasad Mohapatra, Rajib Mall | 2010 21-52 | | custom algo | uses a combination of use case and sequence diagram | integration and system testing | full predicate coverage | model based testing |
| 22 | GenRed: A Tool for Generating and Reducing Object-Oriented Test Cases | IEEE | IEEE 34th Annual Computer Software and Applications | Hojun Jaygarl, Kai-Shin Lu, Carl K. Chang | 2010 127-136 | custom tool GenRed is used | custom algo | based on the enhancement of RANDOOP tool | | | |
| 23 | Automated Test cases generation for Object Oriented Software | IJCSE | | A.V.K.Shanth i, DR.G.Mohank umar | 2011 543-546 | | evolutionary genetic algo, DFS | uses the UML class diagram and concept of data mining to generate optimal test cases | | | model based testing |
| 24 | EvoSuite: Automatic Test Suite Generation for Object oriented Software | | | Gordon Fraser, Andrea Arcuri | 2011 | custom tool EvoSuite is 4 developed | | based on the insertion of assertions in the code under test | | code coverage and branch coverage | code based testing |
| 25 | Test Case Generation For Concurrent Object-Oriented Systems Using Combinational Uml Models | IJACSA | | Swagatika Dalai,Arup Abhinna Acharya, Durga Prasad Mohapatra | 2011 97-102 | | | uses combinational UML model(Sequence diagram and Activity diagram) for generating test cases for concurrent systems | | | model based testing |
| 26 | Generation and Optimization of Test cases for Object-Oriented Software Using State Chart Diagram | | | Ranjita Kumari Swain, Prafulla Kumar Behera, Durga Prasad Mohapatra | 2012 407-424 | | uses custom algos | uses state chart diagrams for generation and optimization of test cases | | state coverage, action coverage, transition path coverage and condition coverage | model based testing |
| 27 | Minimal TestCase Generation for Object-Oriented Software with State Charts | IJSEA | | Ranjita Kumari Swain, Prafulla Kumar Behera, Durga Prasad Mohapatra | 2012 39-59 | | uses custom algos | uses state chart diagrams for generation and optimization of test cases | | state coverage, action coverage, transition path coverage and condition coverage | model based testing |
| 28 | Behavior based Automated Test Case Generation for Object Oriented Systems | IJSEA | | Rohit Verma, Rajesh Bhatia | 2012 49-60 | custom tool is developed | | uses combination of class diagram, sequence diagram and state chart diagram | | piecewise coverage, transition coverage, round trip path coverage | model based testing |
| 29 | Semi-Automatic Search-based Test Generation | IEEE | Fifth International Conference on Software Testing, | Yury Pavlov, Gordon Fraser | 2012 777-784 | EVOSUIT E tool is used | | integrates user-feedback into the genetic algo applied to generate test cases | unit(class testing) | | |
| 30 | Test Case Generation for Classes in Objects-Oriented Programming Using Grammatical Evolution | | | Jirawat Chaiareerat, Peraphon Sophatsathit, Chidchanok Lursinsap | 2012 251-257 | | | uses Grammatical Evolution technique for user specified grammar to generate test cases | unit(class testing) | branch coverage | |
| 31 | Object Oriented Test Case Generation Technique using Genetic Algorithms | IJSEA | | V.Mary Sumalatha, G.S.V.P.Raju | 2013 20-26 | | | based on the application of genetic algo on sequence diagram | | | model based testing |
| 32 | Review of Automatic Test Case Generation from UML Diagram using Evolutionary Algorithm | IJIES | | Kirandeep Kaur, Vinay Chopra | 2014 17-20 | | | uses multi objective genetic algo for test case generation from UML sequence diagram | | | model based testing |

## 5. Literature Survey:

As is clear from the above list most of the work in the field of automated test case generation for object-oriented systems focus on the model based approach (mainly UML diagram). So, focus of this review paper has been on the model based approach for test case generation.

UML has emerged as an industrial standard for modeling software systems [6]. UML is a visual modeling language that can be used to specify, visualize, construct, and document the artifacts of a software system [6].

A number of researchers have used Models for testing the object oriented systems. This approach has increasingly become popular. C.D. Turner and D.J. Robson used the concept of FSA (Finite State Automata) for generation of test cases and validation of object-oriented programs. It emphasizes on the validation of interaction between the features of a class, [5].

Wang Linzhang, Yuan Jiesong, Yu Xiaofeng, Hu Jun, Li Xuandong and Zheng Guoliang, in their paper [7], derive test scenarios directly from the activity diagram modeling an operation. Then, all the information for test case generation, i.e. input/output sequence and parameters, the constraint conditions and expected object method sequence, is extracted from each test scenario. At last, the possible values of all the input/output parameters could be generated by applying category-partition method, and test suite could be systematically generated to find the inconsistency between the implementation and the design.

Ranjita Kumari Swain, Prafulla Kumar Behera and Durga Prasad Mohapatra, in [8], proposed a test data generation scheme using state chart diagram which optimizes test coverage by minimizing time and cost.

Debasish Kundu and Debasis Samanta used UML Activity Diagrams to generate test cases in [4]. They used UML 2.0 syntax for generating test cases from activity diagrams with use case scope. They considered a coverage criterion called activity path coverage criterion with the aim to cover faults like synchronization faults, faults in a loop.

Santosh Kumar Swain, Durga Prasad Mohapatra and Rajib Mall, in [9], proposed a novel technique by combining state and activity models of the system to construct an intermediate representation which they named as state-activity diagram. This technique is very effective in detecting integration faults.

Santosh Kumar Swain, Subhendu Kumar Pani, Durga Prasad Mohapatra,in [11], focus on various model based techniques for automatic object-oriented software testing.

A use case driven approach has been used by Cle´mentine Nebut, Franck Fleurey, Yves Le Traon, Jean-Marc Je´ze´ quel for automated test case generation for embedded systems in [12].
Philip Samuel, Rajib Mall and Sandeep Sahoo have used UML sequence diagrams and dynamic slicing technique in their paper published in 2005. It uses slice coverage as the test coverage criteria, [13].

Monalisa Sarma,Rajib Mall in 'Automatic Test Case Generation from UML Models' use the combination of sequence diagrams and use case diagrams for test case generation and uses sequence diagram message path coverage and use case dependency coverage as metrics for measuring the effectiveness of generated test cases, [14].

Sequence diagrams, use case template and class diagrams have been used by Monalisa Sarma and Rajib Mall for system testing of the software, in [15]. This approach ensures sequence diagram message path sequence coverage.

Fanping Zeng, Zhide Chen, Qing Cao and Liangliang Mao used UML state diagram model that represent state transition to generate test cases, in [16]. A variation of this approach has also been proposed which uses state chart diagrams for generation and optimization of test cases, [21].

A requirement prioritization method approach has been proposed by Nicha Kosindrdecha and Jirapun Daengdej which is based on use case diagram which ensures domain specific requirement coverage, in [17].

A.V.K.Shanthi and DR.G.Mohankumar applied the concept of data mining to generate optimal test cases from UML class diagram, in [18].

Combinational UML models (sequence diagram and Activity diagram,[19] and class diagram, sequence diagram and state chart diagram, [20] ) have also been used by researchers for automating the test case generation for object-oriented systems.

Some researchers have also applied genetic algorithm to UML sequence diagrams for test case generation, [22], [23]. A multi objective genetic algorithm has been used by Kirandeep Kaur and Vinay Chopra for generating test cases from UML sequence diagram, [23].

5.1 Limitations of Design based approach

Work of these researchers and various other researches done in this field show that model based testing provides better test coverage especially for behavioral aspects which are difficult to identify in the code. Another advantage of this approach is that whenever a code change occurs to fix a coding error, the test cases are not affected as the changed code still confirms to the model.
But this approach requires the testers to be familiar with the models and its underlying mathematics and theories. They need to be aware of the tools and programming languages necessary for performing various tasks. Moreover, can never displace code based testing, since models constructed during the development process lack several details of implementation that are required to generate test cases[4] .

## 6. Future Work:

The real work that remains for the near future is fitting specific models (finite state machines, grammars or language-based models) to specific application domains [2]. We must form an understanding of how we are testing and be able to sufficiently communicate that understanding so that testing insight can be encapsulated as a model for any and all to benefit from. Modifications can be done in existing models or new models can also be designed which are more generic and can host a wide variety of applications and provide an optimal test suite for testing.

## 7. Conclusion:

Model Based Testing approach provides the test cases early in the development of software development life cycle. It increases productivity and it doesn't require changes in the test cases whenever changes are made to

the code to correct the coding error. But it lacks certain implementation level details that are available for code based testing. At present there are not any guidelines or measures that can be used to weigh one model against the other. So, careful analysis of test requirement should be done before selecting a relevant model. A number of researchers have proposed different model based approaches for test case generation. But still a lot of work needs to be done to find the optimal test suite for applications under test.

**References:**


[1] J. Philipps, A. Pretschner, O. Slotosch,E. Aiglstorfer, S. Kriebel, K. Scholl, Model based test case generation for smart cards, in:Proc. 8th Intl. Workshop on Formal Meth. For Industrial Critical Syst., 2003, pp. 168–192.

[2] Santosh Kumar Swain, Subhendu Kumar Pani, Durga Prasad Mohapatra, Model based Object Oriented Software Testing, JATIT.

[3] R. V. Binder. *Testing Object-Oriented Systems Models, Patterns, and Tools*. Addison Wesley, Reading, Massachusetts, October 1999.

[4] Debasish Kundu, Debasis Samanta. A Novel Approach to Generate Test Cases from UML Activity Diagrams. Chair of Software Engineering, 2008.

[5] C.D. Turner, DJ. Robson. The State-based Testing of Object-Oriented Programs

[6] G. Booch, J. Rumbaugh, and I. Jacobson. The Unified Modeling Language Reference Manual. Addison-Wesley, Reading, Massachusetts, 1999.

[7] Wang Linzhang, Yuan Jiesong, Yu Xiaofeng, Hu Jun, Li Xuandong and Zheng Guoliang. Generating Test Cases from UML Activity Diagram based on Gray-Box Method

[8] Ranjita Kumari Swain, Prafulla Kumar Behera, Durga Prasad Mohapatra. Generation and Optimization of Test cases for Object-Oriented Software Using State Chart Diagram.

[9] Santosh Kumar Swain, Durga Prasad Mohapatra, Rajib Mall. Test case generation based on state and activity models.

[10] Santosh Kumar Swain, Durga Prasad Mohapatra, Rajib Mall. Test case generation based on Use case and sequence diagram.

[11] Santosh Kumar Swain, Subhendu Kumar Pani, Durga Prasad Mohapatra. Model Based Object-oriented Software Testing, 2005.

[12] Cle´mentine Nebut, Franck Fleurey, Yves Le Traon, Jean-Marc Je´ze´ quell. Automatic Test Generation: A Use Case Driven Approach, 2005.

[13] Philip Samuel, Rajib Mall and Sandeep Sahoo. UML Sequence Diagram Based Testing Using Slicing, 2005.

[14] Monalisa Sarma,Rajib Mall. Automatic Test Case Generation from UML Models, 2007.

[15] Monalisa Sarma,Rajib Mall. Automatic Test Case Generation from UML Sequence diagrams, 2007.

[16] Fanping Zeng, Zhide Chen, Qing Cao, Liangliang Mao. Research on Method of Object-Oriented Test Cases Generation Based on UML and LTS, 2009.

[17] Nicha Kosindrdecha, Jirapun Daengdej. A Test Case Generation Technique and Process, 2010.

[18] A.V.K.Shanthi, DR.G.Mohankumar. Automated Test cases generation for Object Oriented Software,2011.

[19] Swagatika Dalai,Arup Abhinna Acharya, Durga Prasad Mohapatra. Test Case Generation for Concurrent Object-Oriented Systems Using Combinational Uml Models, 2011.

[20] Rohin Verma, Rajesh Bhatia. Behavior based Automated Test Case Generation for Object Oriented Systems, 2012.

[21] Ranjita Kumari Swain, Prafulla Kumar Behera, Durga Prasad Mohapatra. Minimal TestCase Generation for Object-Oriented Software with State Charts, 2012.

[22] V.Mary Sumalatha, G.S.V.P.Raju. Object Oriented Test Case Generation Technique using Genetic Algorithms, 2013.

[23] Kirandeep Kaur, Vinay Chopra. Review of Automatic Test Case Generation from UML Diagram using Evolutionary Algorithm, 2014.

[24] S.J. Cunning, J.W. Rozenblit. Automatic Test Case Generation from Requirements Specifications for Real-time Embedded Systems, 1999.

[25] Y.J. Kim, H.S. Hong, S.M. Cho, D.H. Bae, S.D. Cha. Test Case Generation from UML State Diagrams, 1999.

[26] MIA0 Huaikou, LIU Ling, A Test Class Framework for Generating Test Cases from Z Specifications, 2000.

[27] Wee Kheng Leow, Siau Cheng Khoo, Yi Sun. Automated Generation of Test Programs From Closed Specifications of Classes and Test Cases, 2004.

[28] M.Prasanna, S.N. Sivanandam, R.Venkatesan, R.Sundarrajan. A Survey on Automatic Test Case Generation, 2005.

[29] Zhe (Jessie) Li, Tom Maibaum. An Approach to Integration Testing of Object-Oriented Programs, 2007.

[30] Bertrand Meyer, Ilinca Ciupa, Andreas Leitner, Lisa (Ling) Liu. Automatic testing of object-oriented software,2007.

[31] Xianghua Deng, Robby, John Hatcliff. Kiasan/KUnit: Automatic Test Case Generation and Analysis Feedback for Open Object-oriented Systems, 2007.

[32] Hojun Jaygarl, Kai-Shin Lu, Carl K. Chang. GenRed: A Tool for Generating and Reducing Object-Oriented Test Cases, 2010.

[33] Gordon Fraser, Andrea Arcuri. EvoSuite: Automatic Test Suite Generation for Object-oriented Software, 2011.

[34] Yury Pavlov, Gordon Fraser. Semi-Automatic Search-based Test Generation, 2012.

[35] Jirawat Chaiareerat, Peraphon Sophatsathit, Chidchanok Lursinsap. Test Case Generation for Classes in Objects-Oriented Programming Using Grammatical Evolution, 2012.

[36] Rajvir Singh. Test Case Generation for Object-Oriented Systems: A Review, 2014.


Note: There is no conflict of interest between authors.