International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 04 Issue: 05 | May -2017 www.irjet.net p-ISSN: 2395-0072

A Systematic Review and Comparative study of existing testing techniques for Aspect-oriented software systems

Jyoti[1], Susheela Hooda[2]

[1]Student, Department of Computer Science, B.S.A.I.T.M, Faridabad, Haryana
[2] Assistant Professor, Department of Computer Science, B.S.A.I.T.M, Faridabad, Haryana

---

Abstract - *Software testing is a process of validating and verifying that a software product works as expected. It is indeed a very crucial process rather than a single process. To develop quality software, various approaches can be used like module-oriented, object-oriented, aspect-oriented and component based. Various issues for object oriented are considered but AOPs have never been properly analysed. Therefore, in this paper a collection of various references drawn from journals, conference and workshop proceedings, thesis, and technical reports on the subject of testing aspect-oriented software using various manual and automation(using meta-heuristics) techniques is presented. Each reference is accompanied by a summary of important finding. A tabular form representation analysing each paper in various fields is presented. In this paper , a very critical review of literature to help the researched is done supported by bibliography.*

***Key Words*: Software quality assurance; genetic algorithm; fuzzy logic; meta-heuristic approach; Aspect-oriented programming (AOP), AOP testing.**

1.INTRODUCTION

A program consists of supporting functions and the main program's business logic. To isolate supporting functions from the main program's business logic, Aspect-oriented Programming (AOP) is used. To increase modularity, is the main aim of AOP. Each cross cutting concern must encapsulate at one place. AOP helps in achieving this as all implementations have same cross cutting expressions. This helps in achieving minimal or no code scattering.

Main aim is to provide sufficiently good solution to an optimization_problem, for which a heuristic (partial search_algorithm) id chosen that provides good solution for this. Metaheuristics is higher level heuristics. Finding by trial and error is called heuristics. And meta means higher level. Thus, metaheuristics are better than simple heuristics. Metaheuristics are useful for a variety of problems.
Testing techniques are divided into manual and automation testing techniques. Test cases can be found manually or can be prioritized using meta-heuristic techniques like genetic algorithm and fuzzy logic. Till date various researches have been applied on object oriented programming. Not much has been covered in context of aspect-oriented programming.

The motivation of this paper is to analyse the various existing testing techniques of object-oriented systems and aspect-oriented software systems critically on the basis of the proposed criteria. This critical analysis will find appropriate areas of research where new results can be obtained. It would be easy to provide answers to various research problems faced by the various researchers who are involved in aspect-oriented software testing. New research areas would be found after proper analysis of researches which have been done and which still need to be covered.

This paper is divided into 5 different sections. Section 1 introduces the need of AOP and testing and research area to be covered. Section 2 presents the literature survey . Section 3 identifies the basis of comparison of the research papers. Section 4 defines the various comparison tables of the cited reference papers. At last, conclusion and future prospects of aspect-oriented testing are covered in section 5.

2. LITERATURE SURVEY

To have a proper understanding of aspect oriented, current scenario of research in this field is to be identified. In order to analyze various existing testing techniques available for aspect-oriented software systems, available literature was extensively studied. Applied various search techniques from sources like digital libraries of IEEE, ACM, Springer Link etc. During the study, various journal, technical reports and conference papers were referred from these sources.

Following criteria are used for comparison: explaining AOP with UML, meta-heuristic search algorithms, comparing AOP with OOP, problems in applying meta-heuristics with AOP.

First, let's identify the papers related to aspect oriented programming to get a basic insight into what aspect oriented programming is. Also, what techniques are being applied in testing aspect oriented programs.

To test aspect oriented programming, a method involving data flow diagrams [1] is being involved. Identification of various types of DUAs and proposal of a set of dataflow test criteria that require executing these DUAs is being done in this paper. For identifying the DUA's in an AspectJ program, a tool called DCT-AJ is implemented. It computes the coverage from a test suite. This paper is quite helpful for making the

data flow graph in aspect oriented systems. Similarly, for gathering information on the structure of an AOP, an Aspect Oriented Dependence Flow Graph (AODFG) [5] as an intermediate representation model is proposed. Once data flow graphs are drawn for an AOP, data flow based unit testing [6] can be done. This paper tells using control flow graphs to compute def-use pairs of an aspect or class being tested and using such information to guide the selection of tests for the aspect or class. After unit testing, structural testing becomes important. Thus, a structural testing approach for Aspect-Oriented Programs Based on Data and Control Flow [8] is being referred. An effective test technique is presented for AOP.

Various testing techniques [2] are identified in aspect-oriented software systems. To develop new and better testing techniques, it becomes important to analyze existing aspect-oriented testing techniques. The paper reviewed [2] identifies all these existing techniques for AOP.

What is Aspect oriented testing? [3] A very well explained answer to this question is given in the paper presented by *Ghani et al*. Taking reference from around 81 references drawn from journals, conference and workshop proceedings, thesis, and technical reports on the subject of testing aspect-oriented software, a proper study is presented.

After understanding the concept of aspect oriented testing and various testing techniques, an approach for systematic testing [4] of aspect oriented programs is required. *Alexander et al* [4] explained the systematic testing of AOP by identifying the key issues relevant to systematic testing of AOP. Another Systematic Aspect-Oriented Refactoring and Testing Strategy [9] is presented by *Deursen et al* [9]. Aspect-oriented solution when integrated with the original system, some issues are there. This paper considers those.

After the basic information on Aspect oriented programming and its testing, quantifying the effects of Aspect-oriented programming [7] is done. The positive and negative effects of AOP on typical maintenance activities of a Web information system are being analyzed. To find out the effects of software aspectization [10], a measuring method is proposed by *Ceccato et al* [10] which helps to find the advantages and disadvantages of using the AOP approach. Considering the effects of AOP, how beneficial is AOP for modularity is considered next as it is said that aspects are used for gaining modularity. Finding the truth of AOP and its impact on modularity [11] as proposed by *Adam Przybyłek* [11].

Delamare et al [12] proposed a test-driven approach for the development and validation of the PCD (point cut descriptor). Using aspect oriented for advanced separation of concerns is being proposed by *Singh et al* [13]. This paper proposes several techniques for crosscutting concern identification.

Next concern that came in after formulating aspect oriented programming is finding how much easy or hard it is to test the AOP code. Comparing it with OOP, *Ceccato et al* [14] proposed testing crosscutting functionalities as aspects in successive steps. A report identifying the difficulties in testing Aspect-Oriented Programming [15] was referred. It discusses the difficulties of structure-based and fault-based testing of AO programs. Test requirement coverage in AOP is not that straightforward. On identifying these difficulties in testing AOP, a solution was proposed by *Zhao et al* [16] which describes an AspectJ program testing method based on a fault model which was made using the dependency model and interaction model.

Once unit and structural testing of Aspect oriented is over, it is time for regression testing. *Zhao et al* [17] presented a regression test selection technique for AspectJ programs. Some test cases to execute the changed code for the new version are to be selected from the original test suite. The technique consisting of various types of control flow graphs does the same. But considering the impact the aspects can have on the main class, *Xu* [18] defined a new test selection criterion for AOPs to achieve higher precision. To cover model based testing in aspect oriented system soft wares, a very recent (2016) paper by *S. Hooda et al* [39] is considered which reviews around 94 papers and narrows them down to around 25 for easy reference of researchers.

Second, after identifying the work done related to aspect oriented programming and its testing, next comes identifying the work done in studying an Aspect oriented programming with UML Activity diagrams. Having knowledge of UML activity diagram is equally important for testing AOP coverage efficiently.

Cross cutting concerns must be handled at an earlier software phases. Thus, Modeling and Integrating Aspects with UML Activity Diagrams [19] becomes a necessary step. *Cui et al* [19] proposed an aspect-oriented modeling and integration approach at the design level. In order to refine more concert, executable code-based test suites [20], aspect flow graph is required and a transition tree as proposed by *Xu et al* [20]. A proper definition and comparison of various related path criteria is required. Thus, *Rapps et al* [21] defined a Data Flow Analysis Techniques for Test Data Selection.

Third, for optimizing the test cases and reducing the effort, Meta heuristic techniques have to be considered. Heuristics means to find or to discover by trial and error. And Meta means higher level and metaheuristics generally perform better than simple heuristics. Meta-heuristic algorithms have the ability to obtain the optimal solution in a very large search space of candidate solutions. Meta-heuristics consists of applying Artificial intelligence like Genetic algorithm, Fuzzy logic to a problem. Thus, now papers introducing Meta heuristic techniques are referred.

Meta-heuristic techniques are very much used in software testing. *Pandey et al* [22] proposed a technique called search based software testing. This paper gives an insight into the recent trends of the applications of search based techniques to generate Applications of meta-heuristic techniques in the field of software testing phases. Meta-heuristic techniques can also be applied in generation of test data [23]. The aim is to generate the optimum set of test data. Next meta-heuristic techniques are needed to generate search–based test data for branch coverage [24]. This paper introduces multi–objective branch. While studying meta-heuristics, some evolutionary test environment has been identified for automatic structural testing [25]. The environment has been developed that performs fully automatic test data generation for most structural test methods.

The fourth, after getting knowledge of meta-heuristics, a basis of object oriented was reviewed to find the applications of meta-heuristics in object oriented programming. Nothing much has been identified in the field of meta-heuristics in aspect oriented. So, a general comparison of AOP vs OOP is also being referred in order to apply meta-heuristics in aspect oriented.

Dr. Suresh [26] presented a method of Software quality assurance for object-oriented systems using meta-heuristic search techniques. This paper tells how much capable meta-heuristic search techniques are in software fault classification for Apache Integration Framework. Also, regression testing of object oriented systems [27] has been reviewed. A hybrid technique is formed which identifies the changes that are not visible in design models. Finally, comparison of AOP with OOP is done and identifying whether AOP is harder or easier than OOP [14]. An incremental testing process is proposed in this paper, which tests crosscutting functionalities as aspects in successive steps. Model based testing approach for object oriented systems is being considered by *Arilo et al* [38]. By identifying four hundred and six papers, this paper narrowed down it to seventy-eight papers for reference.

On identifying the meta-heuristic techniques and their usage in object-oriented programs, the problem was identified that not much has been done for implementation of meta-heuristics in aspect oriented programming. A proper review regarding the same has also never been done. So, next thing is identifying the papers in this field. The strategy goes by finding papers written for genetic algorithm, fuzzy algorithm or their combination. Fifth, let's identify the work done in the field of using genetic algorithm and mutation testing in aspect oriented programs.

Genetic algorithm has been applied to increase the efficiency of a Data Flow-based Test Data Generation Approach [28]. How effective it is to use a genetic algorithm for the same has been identified here. Another major application of genetic programming is in effort estimation [29]. *Ferrucci et al* [29] found the impact different fitness functions can have on effort estimation. To consider the indirect impact of aspects, *Delamare et al* [37], proposed an approach. When many methods are indirectly impacted by aspects, this approach can reduce the testing efforts by relying on genetic algorithm.

Fuzzy logic has been used in aspect oriented for proposing a model on coupling measures [33]. In object oriented and module oriented systems, quality metrics are used for quality features but very less has been done for aspect-oriented systems. Various factors like Number of dependencies, responsibility and Instability have been related with coupling for making the comparison. Another application of fuzzy logic approach has been identified in measuring the complexity of a generic aspect-oriented system [34]. A generalized framework for finding the complexity of AO systems has been defined. That it takes into account three AOP languages, which are AspectJ, CaesarJ and Hyper/J.

Fuzzy clustering is very much useful in various applications. One of them is using it in test-case prioritization. *Nida et al* [35] identified an approach to rank the test cases as per their preference degrees. Software testing optimization can also be achieved by fuzzy clustering [36]. This is very much useful in reducing the time spent in testing.

3. COMPARISON CRITERIA

Comparing each paper based on the following comparison techniques:

1. Papers explaining aspect oriented programming.
2. Papers explaining aspects with UML diagrams.
3. Understanding meta-heuristic search techniques.
4. Object oriented systems and their comparison with AOP.
5. Applying genetic algorithm and mutation testing.
6. AOP using fuzzy logic.
7. Fuzzy clustering and its applications.

These criterias are considered for comparing the various research papers referred. These criterias are important to identify the work done yet in this field and what is not done and what can be done. This paper draws conclusion on what further research could be undertaken in this field.

4. COMPARATIVE TABLE

Table -1: Analysis of cited papers by criteria

s.no.	Criteria	Reference
A	Aspect oriented programming and testing.	1-18, 39
B	Aspects with UML Activity diagrams	19,20,21

C	Meta heuristic search algorithms	22,23,24,25
D	OOP vs AOP	14,26,27, 38
E	Genetic algorithm and mutation testing	28,29,30,31,32, 37
F	AOP testing using fuzzy logic	33,34
G	Fuzzy clustering	35,36

Table 1 compares the cited papers on the basis of the comparison criteria considered. Very relevant information is provided in this table for researchers as to which research paper to consider for the given information.
Table 2 gives a brief analysis of the cited papers by year of publications.

Table -1: Analysis of cited papers by year

Year	**Publications**
1982	1
2001	1
2003	1
2004	2
2005	1
2006	4
2007	3
2008	1
2009	2
2010	4
2011	2
2012	3
2013	5
2014	2
2015	5
2016	2
Total	39

Table -3: Comparison of all cited papers

5. CONCLUSION

In this paper, a broad survey of aspect oriented programming and its testing using meta-heuristic techniques is covered. Topics covered are aspect oriented programming and its testing, UML diagram, understanding meta-heuristic search algorithms, comparing OOP and AOP, genetic algorithm and mutation testing, AOP testing using fuzzy logic, fuzzy clustering.

What could be concluded is that much work has been found in the field of applying meta-heuristic techniques to object oriented programming. But very less work has been done in applying meta-heuristic techniques to aspect oriented programming. So researchers can look forward in this field.

All the topics mentioned above are cited with proper references mentioning their year of publishing for easy understanding of researchers . This information could be very useful and beneficial for researcher or practitioner , who is relatively new, in gathering a lot of information in this field.

Author/title	Cat.	Testing level	Software Domain	Behavior Models	Tools support	to Advantage	Extension
F. Wedyan and S. Ghosh, 2010 [1]	A	Data flow testing	AOP	a tool called DCT-AJ	yes	Tool measuring dataflow coverage is implemented	DCT-AJ can be extended to handle inheritance, polymorphism, aliasing etc.
A. Singhal, A.Bansal, and A. Kumar,2013 [2]	A	Various testing techniques	AOP	Review	ND	Ready reference for all existing testing techniques in AOP	Can be extended to refine the advantages and disadvantages of each paper.
A. A. Abdul Ghani *et al*, 2013 [3]	A	Various testing techniques	AOP	Bibliography	ND	Provides avenues of exploration for researcher in AOP testing	Can be extended to refine the advantages and disadvantages of each paper.
Roger T.A Alexander *et al*, 2004 [4]	A	Systematic Testing	AOP	fault model	yes	Derived testing criteria from candidate fault model	Given model needs to be extended and refined.
S. Ahmad *et al*, 2014 [5]	A	Structural testing	AOP	Dependency flow graph (AODFG)	yes	Data structure is made to provide dependence information.	Needs to be applied on complex programs.
Jianjun Zhao and J. Zhao,2003	A	Unit testing	AOP	Data flow based	yes	Data flow based unit testing approach is implemented.	Test extended aspect or class.
U. Kulesza *et al*,2006 [7]	A	Maintenance activities	AOP vs OOP	Compared AO and OO implementations	ND	Discussed maintainability attributes of web-based information system	Side effects of using AOP to be discussed more.
Liping Xiong,2013 [8]	A	Structural Testing	AOP	Data and control flow	yes	BDUC expression which represents the data flow and control flow between all units under test.	Automatic generation of BDUC expressions and usefulness for larger programs.
A. Deursen *et al*, 2005 [9]	A	Systematic Testing	AOP	Refactoring and testing strategy (JHotDraw)	yes	An aspect-oriented fault model and adequacy criteria is made	An automated tool for testing and refactoring is required.

Author/title	Cat.	Testing level	Software Domain	Behavior Models	Tools to support	Advantage	Extension
Mariano Ceccato *et al*,2004 [10]	A	ND	AOP	defining metrics suite for AOP software	ayes	Measuring different kinds of coupling relationship and a new metric for cross cutting technique.	
A. Przybyłek,2011 [11]	A	ND	AOP	Software Modularity	yes	Using the CBO and LCOM metrics to AOP	More factors for comparison to be covered.
R. Delamare *et al*, 2009 [12]	A	ND	AOP	Test driven approach	yes	Point cut Descriptors using Advice Tracer	Needs to detect other types of faults that could not be injected in the systems
Narender Singh *et al*, 2011 [13]	A	Early stages of software development	AOP	AORE to focus on crosscutting concerns.	yes	Discussed many AORE approaches to deal with crosscutting concerns	Development of complete methodology to e done.
Mariano Ceccato *et al*, 2015 [14]	A,D	Incremental testing	AOP	An incremental testing process is considered.	yes	Allows testing the base code and the cross-cutting functionalities, implemented as aspects, in successive steps.	Detailed definition and implementation of the AOP incremental testing to be covered.
F. C. Ferrari *et al*., 2015[15]	A	Structural and Mutation Testing	AOP	PWIW operators (Pointcut Weakening by Insertion of Wildcards)	yes	Implemented for larger systems as well.	Creating adequate test suites for such large systems is required.
C. Zhao *et al*,2006 [16]	A	Fault-Based Testing	AOP	AspectJ programs testing method	yes	Fault model, dependency model and interaction model direct the testing process.	Empirical studies to justify the model to be added more.
J. Zhao *et al*,2006 [17]	A	Regression Testing	AOP	Control flow graphs	No	Can be applied to modified individual aspects or classes as well as the whole programs.	Regression testing tool can be discussed.

Author/title	Cat.	Testing level	Software Domain	Behavior Models	Tools to support	Advantage	Extension
G. Xu, 2006 [18]	A	Regression Testing	AOP	A three-phase technique	yes	Achieves safety and higher precision	Possible effectiveness to be covered.
Z. Cui *et al*, 2009 [19]	B	Model Testing	AOP	UML Activity diagrams	yes	Light weight extension to standard activity diagrams .	Integrated model against system requirements to be covered.
Weifeng Xu *et al*, 2006 [20]	B	Model Testing	AOP	A hybrid testing model	yes	Manageable, code based and executable test suites.	Extend to system architecture and abstract level.
Sandra Rapps *et al*, 1982 [21]	B	Test Data Selection	ND	The data flow criteria	yes	Bridge the gap of the requirement	Can find the relative cost.
Abhishek Pandey *et al*, 2014 [22]	C	Search based testing	ND	Meta Heuristic Search Algorithms	yes	Wonderful demonstration using small programs.	Can be scaled to large programs.
Srivastava *et al*, 2008 [23]	C	Test Data Generation	ND	GA and ACO	yes	Both have reasonable amount of success in generating test data for software testing	Not much.
Lakhotia *et al*,2007 [24]	C	Test Data Generation	ND	A Multi–Objective Approach	yes	Consuming as much dynamic memory as possible at the same time	Evaluation on larger real world programs is required.
Wegener *et al*, 2001[25]	C	Structural Testing	ND	Evolutionary test environment	yes	Overall testing procedure has been considerably accelerated	Further improvements of the evolutionary structural test.
Y. Suresh, 2015 [26]	D	Software quality assurance	OOP	neural network models with meta-heuristic search based algorithms	yes	Better classification accuracy rate	Other optimization techniques can be added.
Vincent *et al,* 2013[27]	D	Regression Testing	OOP	Hybrid Technique	yes	Reduction of the cost of regression testing	Other OO software systems to be analysed.
Mahajan *et al*, 2012 [28]	E	Test Data Generation	ND	GA based approach	yes	Ga better than random approach.	100% coverage to be achieved.

Author/title	Cat.	Testing level	Software Domain	Behavior Models	Tools to support	Advantage	Extension
Ferrucci et al, 2010 [29]	E	Effort Estimation	ND	Different summary measures(GP)	yes	Fitness function significantly influenced the accuracy of estimations using GP.	More datasets required to verify the results.
Wedyan et al, 2012 [30]	E	Mutation Testing	AOP	Approach for eliminating equivalent mutants	yes	A fault model that solves the problems with previous fault models.	Large scale empirical study on several larger subject programs to be done.
Leme et al, 2015 [31]	E	Mutation Testing	AOP	Proteum/AJv2 tool	yes	Integrated environment for testing Java and AspectJ applications	Cost reduction of mutation testing for AOP to be covered.
Ferrari et al, 2010 [32]	E	Mutation Testing	AOP	Proteum/AJ	yes	Automates the mutation testing of AspectJ programs	Larger aspect-oriented systems to be covered.
Chishti et al, 2016 [33]	F	Coupling Measures.	AOP	Rule based fuzzy model Algorithms	yes	identified the factors which could affect the coupling	Cover other issues like flexibility reliability. Maintainability etc.
Kumar et al, 2010 [34]	F	Measure Complexity	AOP	Fuzzy Logic	yes	Fuzzy model defined to measure code complexity.	Developing a model to get total complexity.
Gökçe et al,2015 [35]	G	Model-based Testing	ND	ACL and FCM Algorithm.	yes	Better model-based test case prioritization approaches	Can use multivariate data analysis techniques.
Kumar et al, 2013[36]	G	Software testing optimization	ND	Fuzzy Logic toolbox of MatLab	yes	Fuzzy is used for clustering to get more efficient and accurate results.	Path/condition coverage may still be tested.
Delamare et al, 2012 [37]	E	Integration Testing	AOP	genetic algorithm for AspectJ programs	yes	Fine grained integration test orders produced.	Several large size systems to be covered.
Dias et al, 2007[38]	D	Survey	OOP	MBT review	ND	Tabular form review of various references done.	Expansion and recommendations are required.
S Hooda et al, 2016 [39]	A	Review	AOP	MST review for AOSS	ND	Improve understanding of MBT for future researchers in AOSS.	Minimize the faults.

REFERENCES

[1] F. Wedyan and S. Ghosh, "A Dataflow Testing Approach for Aspect-Oriented Programs," in 2010 IEEE 12th International Symposium on High Assurance Systems Engineering, 2010.

[2] A. Singhal, A. Bansal, and A. Kumar, "A critical review of various testing techniques in aspect-oriented software systems," ACM SIGSOFT Software Engineering Notes, vol. 38, no. 4, p. 1, 2013.

[3] A. A. Abdul Ghani, A. A. A. Ghani, and R. M. Parizi, "Aspect-Oriented Program Testing: An Annotated Bibliography," J. Softw. Maint. Evol.: Res. Pract., vol. 8, no. 6, 2013.

[4] Roger T. Alexander, James M. Bieman, Anneliese A. Andrews "Towards the Systematic Testing of Aspect-Oriented Programs," 2004.

[5] S. Ahmad, A. A. A. Ghani, and F. M. Sani, "Dependence Flow Graph for Analysis of Aspect- Oriented Programs," International Journal of Software Engineering & Applications, vol. 5, no. 6, pp. 124–144, 2014.

[6] Jianjun Zhao and J. Zhao, "Data-flow-based unit testing of aspect-oriented programs," in Proceedings 27th Annual International Computer Software and Applications Conference. COMPAC 2003.

[7] U. Kulesza, C. Sant'Anna, A. Garcia, R. Coelho, A. Staa, and C. Lucena, "Quantifying the Effects of Aspect-Oriented Programming: A Maintenance Study," in 2006 22nd IEEE International Conference on Software Maintenance, 2006.

[8] Liping Xiong, L. Xiong, and J. Li, "A structural testing approach for Aspect-Oriented Programs based on data and control flow," in 2013 IEEE 4th International Conference on Software Engineering and Service Science, 2013.

[9] A. Deursen, A. Marius Marin, and L. M. F. Moonen, A Systematic Aspect-oriented Refactoring and Testing Strategy, and Its Application to JHotDraw. 2005.

[10] Mariano Ceccato and Paolo Tonella, Measuring the Effects of Software Aspectization. 2004.

[11] A. Przybyłek, "Where the Truth Lies: AOP and Its Impact on Software Modularity," in Lecture Notes in Computer Science, 2011, pp. 447–461.

[12] R. Delamare, B. Baudry, S. Ghosh, and Y. Le Traon, "A Test-Driven Approach to Developing Pointcut Descriptors in AspectJ," in 2009 International Conference on Software Testing Verification and Validation, 2009.

[13] Narender Singh and Nasib Singh Gill, Aspect-Oriented Requirements Engineering for Advanced Separation of Concerns: A Review. 2011.

[14] Mariano Ceccato, Paolo Tonella and Filippo Ricca , Is AOP code easier or harder to test than OOP code?.2015.

[15] F. C. Ferrari et al., "Testing of aspect-oriented programs: difficulties and lessons learned based on theoretical and practical experience," J. Braz. Comput. Soc., vol. 21, no. 1, 2015.

[16] C. Zhao and R. T. Alexander, "Testing AspectJ programs using fault-based testing," in Proceedings of the 3rd workshop on Testing aspect-oriented programs - WTAOP '07, 2007.

[17] J. Zhao, T. Xie, and N. Li, "Towards regression test selection for AspectJ programs," in Proceedings of the 2nd workshop on Testing aspect-oriented programs - WTAOP '06, 2006.

[18] G. Xu, "A regression tests selection technique for aspect-oriented programs," in Proceedings of the 2nd workshop on Testing aspect-oriented programs - WTAOP '06, 2006.

[19] Z. Cui, L. Wang, X. Li, and D. Xu, "Modeling and integrating aspects with UML activity diagrams," in Proceedings of the 2009 ACM symposium on Applied Computing - SAC '09, 2009.

[20] Weifeng Xu, Dianxiang Xu, Vivek Goel and Ken Nygard, Aspect Flow Graph for testing aspect-oriented programs. 2006.

[21] Sandra Rapps and Elaine J. Weyuker, Data Flow Analysis Techniques for Test Data Selection. 1982.

[22] Abhishek Pandey, Dr. Soumya Banerjee and Dr. G. Sahoo, Applications of Meta Heuristic Search Algorithms in Software Testing. 2014.

[23] P. R. Srivastava, V. Ramachandran, M. Kumar, G. Talukder, V. Tiwari, and P. Sharma, "Generation of test data using meta heuristic approach," in TENCON 2008 - 2008 IEEE Region 10 Conference, 2008.

[24] K. Lakhotia, M. Harman, and P. McMinn, "A multi-objective approach to search-based test data generation," in Proceedings of the 9th annual conference on Genetic and evolutionary computation - GECCO '07, 2007.

[25] J. Wegener, A. Baresel, and H. Sthamer, "Evolutionary test environment for automatic structural testing," Information and Software Technology, vol. 43, no. 14, pp. 841–854, 2001.

[26] Y. Suresh, "Software quality assurance for object-oriented systems using meta-heuristic search techniques," in 2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), 2015.

[27] Pierre-Luc Vincent, Linda Badri and Mourad Badri, Regression Testing of Object-Oriented Software.2013.

[28] M. Mahajan, S. Kumar, and R. Porwal, "Applying genetic algorithm to increase the efficiency of a data flow-based test data generation approach," ACM SIGSOFT Software Engineering Notes, vol. 37, no. 5, p. 1, 2012.

[29] F. Ferrucci, C. Gravino, R. Oliveto, and F. Sarro, "Genetic Programming for Effort Estimation: An Analysis of the Impact of Different Fitness Functions," in 2nd International Symposium on Search Based Software Engineering, 2010.

[30] F. Wedyan and S. Ghosh, "On generating mutants for AspectJ programs," Information and Software Technology, vol. 54, no. 8, pp. 900–914, 2012.

[31] Filipe Gomes Leme , Fabiano Cutigi Ferrari ,José Carlos Maldonado, Awais Rashid , Multi-Level Mutation Testing of Java and AspectJ Programs Supported by the Proteum/AJv2 Tool.2015.

[32] F. C. Ferrari, E. Y. Nakagawa, A. Rashid, and J. C. Maldonado, "Automating the mutation testing of aspect-oriented Java programs," in Proceedings of the 5th Workshop on Automation of Software Test - AST '10, 2010.

[33] F. Chishti and A. Singhal, "'Proposed model on coupling measures in aspect oriented software development using fuzzy logic,'" in 2016 2nd International Conference on Advances in Computing, Communication, & Automation (ICACCA) (Fall), 2016.

[34] R. Kumar, P. S. Grover, and A. Kumar, "A Fuzzy Logic Approach to Measure Complexity of Generic Aspect-Oriented Systems," The Journal of Object Technology, vol. 9, no. 3, p. 59, 2010.

[35] N. Gökçe, F. BELLİ, M. EMİNLİ, and B. T. DİNÇER, "Model-based test case prioritization using cluster analysis: a soft-computing approach," TURKISH JOURNAL OF ELECTRICAL ENGINEERING & COMPUTER SCIENCES, vol. 23, pp. 623–640, 2015.

[36] G. Kumar and P. K. Bhatia, "Software testing optimization through test suite reduction using fuzzy clustering," CSI Transactions on ICT, vol. 1, no. 3, pp. 253–260, 2013.

[37] R. Delamare and N. A. Kraft, "A Genetic Algorithm for Computing Class Integration Test Orders for Aspect-Oriented Systems," in 2012 IEEE Fifth International Conference on Software Testing, Verification and Validation, 2012.

[38] A. C. Dias Neto, R. Subramanyan, M. Vieira, and G. H. Travassos, "A survey on model-based testing approaches," 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 - WEASELTech '07, 2007.

[39] Susheela Hooda, Dr. Sandeep Dalal, Kamna Solanki , A systematic review of model-based testing in Aspect-Oriented Software Systems.2016.