Prevalent Criteria's in Regression Test Case Selection Techniques: An Exploratory Study

Priyanka Dhareula
Computer Science Department
Himachal Pradesh University
Shimla (H.P), India
priyankarana.id@gmail.com

Anita Ganpati
Computer Science Department
Himachal Pradesh University
Shimla (H.P), India
anitaganpati@gmail.com

Abstract— **Regression testing is done after needful changes, ensuring that changes are working as required and does not produce unexpected results for a system under test. Note worthy difficulty in regression testing is selection of significant subgroup of test cases. This paper has analyzed techniques of regression test selection (RTS) for test case optimization in various domains. The study identified most prevalent criteria's used by various researchers. This study analyzed two broad groups of techniques under which test cases are optimized i.e. code-based and requirement-based techniques. Further most prevalent criteria's were identified and techniques were grouped under them. The study is also focused on the level of test granularity used by different researchers. Two main granularity levels were identified for code based testing i.e. fine granularity and coarse granularity. From this study it is also concluded that no such technique could be generalized because they are proposed for different domains of interest.**

Keywords: Regression testing, maintenance, test case selection, selection techniques.

I. INTRODUCTION

Software maintenance is a process that incurs forty to seventy percent of total expense of developing entire software. Software maintenance enriches existing functionalities, does error correction, and omits outdated means. As the software is modified the application of maintenance i.e. regression testing is done to retest changes made and to examine the impact of alterations to the software. A process ensuring, that no new error are present in the earlier tested code is regression testing. If the test case results in an error it means that the test was successful and there is a need to correct the errors. Correction of code results in changed configuration. Regression testing is pursued ensuring that changes do not lead to unintended behavior or superfluous errors in the code [18].

A large amount of studies have flourished in support of regression testing. Three generic domains are identified for test case optimization for performing regression testing. These are selection, prioritization and test case minimization. Test case minimization tries to discover and then remove the obsolete or superfluous test cases from original test suite. Test case selection is the best solution to the problematic situation of selecting a subgroup of test cases to be used to test the altered parts of the code. Lastly, prioritization of test cases deals with the identification of perfect ordering of test cases that exploits early fault discovery. Out of these three domains of test case optimizations, this paper chooses test case selection mainly due to its flexibility during test case optimization.

This paper identified two main broad groups under which the studies are performed i.e. code-based techniques and requirement-based techniques.

Effectiveness of RTS techniques can be determined by its practicality in reducing the cost of testing software that undertakes

978-1-4673-7910-6/15/$31.00 ©2015 IEEE

repeated modifications. Regression test selection entails two main stages. Early phase labels unmodified shares of code that are affected by the modifications. Next subgroup of test cases is formed from early test suite that can effectively test unaffected parts of the code. In addition to this some more test cases are augmented that can test the new modules added to the code. This would assist in analyzing the impact of newly augmented module to the previously existing module.

Respectively technique attempts to condense the cardinalities of a test suite. Complications in RTS are extensively explored in recent decades; innumerable techniques have surfaced with the newer criteria's. This paper identified the most prevalent criteria's used in RTS. This study also analyzed the percentage of level of test granularity used by the researchers. The paper also identified that no such technique can be generalized as they are proposed for different domains such as agent based system [16], component based systems [12], procedural oriented software [9], object-oriented software's [1,19] and web-based applications [22] using different approaches.

2. LITERATURE REVIEW

Suri and Singal [4] proposed a time-constrained Ant Colony optimization technique (ACO). The expense of running the test case is determined by its execution time. Test cases are prioritized in demand to attain minimum expense of execution and to uncover maximum faults. Srivastava et al. [16] contributed a RTS technique for agent oriented software system using message flow graph (MFG). Modification traversing test cases were selected on the basis of execution traces of test cases. Buchgeher et al. [10] reports the experience of authors while development of tool-based approach, which would support manual regression test case selection.

Pasala et al. [2] contributed a dynamic approach for selection of test cases. Their technique is centered on the availability of either source code or version change information. Class and method dependencies information are the main components contributing to their approach. Abandah and Alsmadi [11] assessed the efficiency of metrics for call graph in assessing the defects in software products. Bertolino et al. [3] presented the XACML smart coverage selection approach, built on a proposed XACML policy coverage criterion. XACML policy specifies the constraints and conditions that a subject needs to comply with for accessing a resource and doing an action in a given environment.

Tao et al. [6] proposed step-by-step procedure to study the cost effectiveness with a hierarchical cost model. Zhang et al. [13] presented a technique for test cases optimization when no test case coverage information is available. Koju et al. [21] in their technique involves two parts, first they create control flow graph, and secondly dangerous edges are recognized from designed control flow graph on the basis of which test cases are selected for virtual machine based programs.

Vokolos and Frankl [9] textual differencing work is developed on the notion of comparing old version of source to new version of the program. Yoo and Harman [20] presented multi-objective test case selection approach. They studied history of fault-detection, code coverage and execution expense. They gave a formulation of two and three objectives. The main aim of their paper was to determine that which objectives are more important for regression testing. Zhang et al. [5] offered a clustering execution profile technique. Cluster analysis groups the program executions that are similar in feature. Like this system behavior is understood and the test cases are selected accordingly to optimize the test suite effectively.

Beydeda and Gruhn [19] presented a hybrid approach. The classes containing methods are displayed by two control flow graphs. One of them is method specification graph and the other is method implementation graph. Nanda et al. [1] performs test selection by taking into account the variations to configuration of databases and file apart from the basic code-centric approaches. Traceability is created between the test cases and external entities. Test case selection is performed by identifying changes made to the non – code components of the system.

Bhuyan et al. [15] proposed a SOA testing perspective model (STMP). The model supports multiple agent testing. Models under three different perspectives have been proposed. These are Service Developer Perspective, Service Tester and Service provider perspective. UML use case diagram supports to breakdown the requirement into small stories that are easily understood. In their optimization process test cases relating to activity nodes are generated. Their process identifies the modifications made to existing services by identifying the changes or modifications in activity node version of activity diagram of that service.

Sundmark et al. [7] regression testing method is replay-based.. They evaluated two problem areas. Firstly the selection of test case and secondly is of test case reproducibility. Engstrom et al. [8] emphasized on regression test case selection technique based on details of history of test cases and then test cases are prioritized. The paper also highlights and is investigating manual regression testing process currently being used. Tsai et al. [22] offered a model based adaptive technique. They use Simplified Coverage Relationship Model (S-CRM) and test case potency. Test case potency is its possibility to detect a fault. The paper proposed two coverage probability algorithms to rank test cases.

Stochel and Sztando [14] proposed a test case prioritization and selection technique. Historic data and expert knowledge were together used as Inputs to their model. Their approach can be used in software project where feasibility to split the system into smaller functional areas exists. Rana and Ganpati [17] presented a requirement based hybrid approach where the relationship of input and output was studied.

3. OBJECTIVES

The main objective behind this study is to identify most prevalent criteria's used by researchers for regression test case selection. However the main objectives of the study are:

1. To get through understanding of progression made in regression selection techniques.
2. To summarized prevalent criteria's used by various technique studied in this paper.
3. To identify the test granularity used by different techniques.
4. To determine if any specific technique can be generalized.

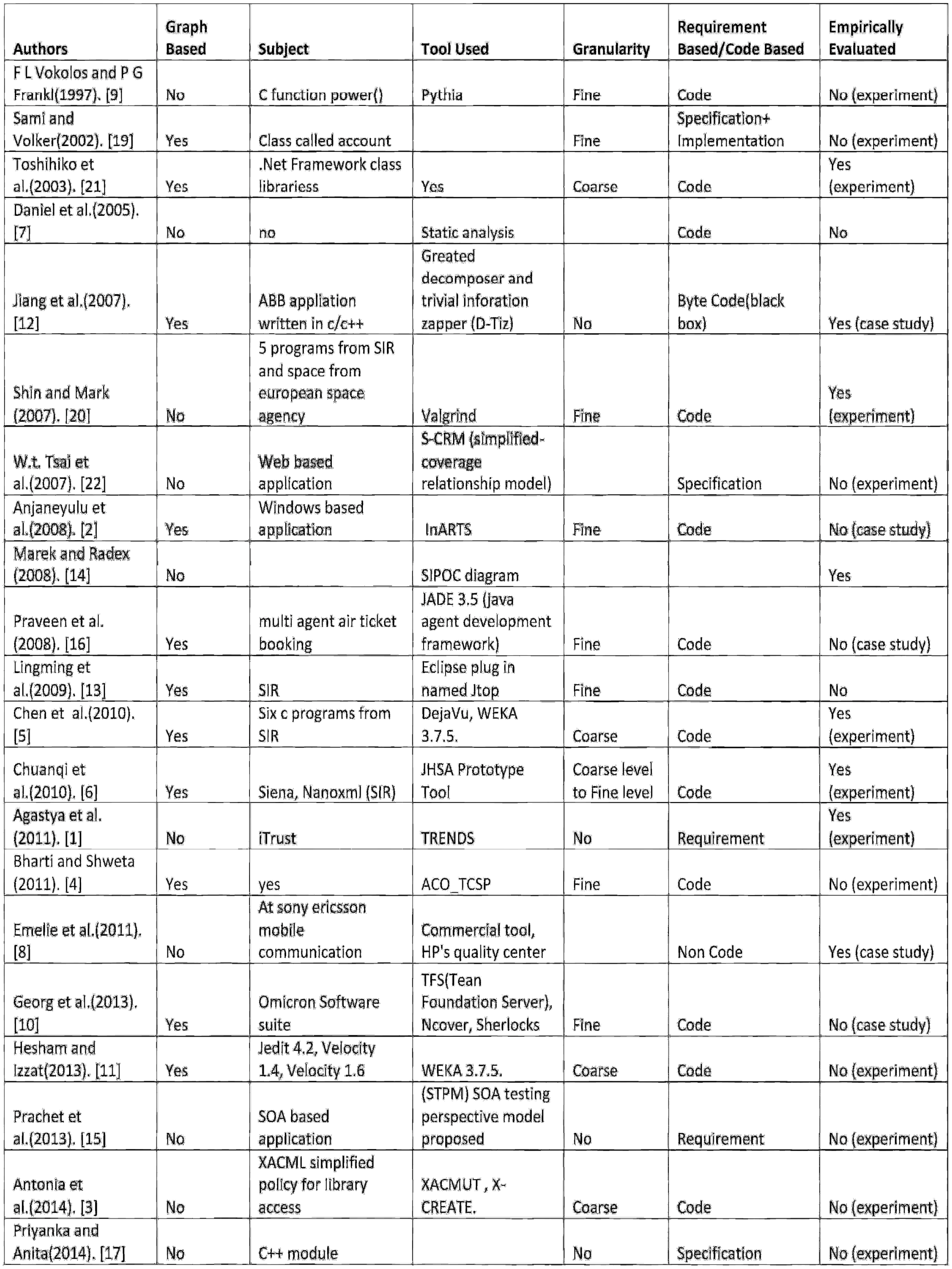
4. RESEARCH METHODOLOGY

This paper aims at summarizing the most prevalent criteria's used by researchers in regression test selection techniques. A comprehensive study of the work by various scholars lays the foundation for research methodology in this paper. For this study various databases like ACM Digital Library, IEEE Explore, Springer and online journals were searched. These databases wrap relevant journals, conferences and workshop proceedings for software engineering. This paper selected research papers pertaining to the arena of regression test case selection.

5. RESULTS AND ANALYSIS

The paper identified most prevalent criteria's for performing test case selection in regression testing. As it is evident from the TABLE I that two main branches studied for regression test case selection are the code based techniques and requirement based techniques. This study identified thirteen and nine prevalent criteria's for code based and requirement based approaches respectively.

TABLE I: Regression Test Case Selection Techniques

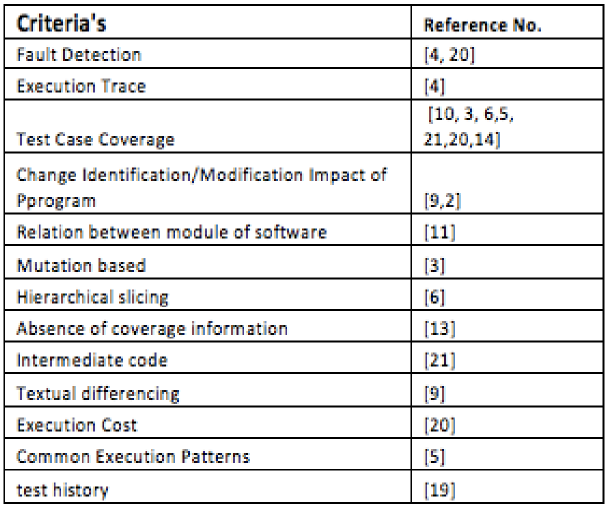


Authors	Graph Based	Subject	Tool Used	Granularity	Requirement Based/Code Based	Empirically Evaluated
F L Vokolos and P G Frankl(1997). [9]	No	C function power()	Pythia	Fine	Code	No (experiment)
Sami and Volker(2002). [19]	Yes	Class called account		Fine	Specification+ Implementation	No (experiment)
Toshihiko et al.(2003). [21]	Yes	.Net Framework class librariess	Yes	Coarse	Code	Yes (experiment)
Daniel et al.(2005). [7]	No	no	Static analysis		Code	No
Jiang et al.(2007). [12]	Yes	ABB appliation written in c/c++	Greated decomposer and trivial inforation zapper (D-Tiz)	No	Byte Code(black box)	Yes (case study)
Shin and Mark (2007). [20]	No	5 programs from SIR and space from european space agency	Valgrind	Fine	Code	Yes (experiment)
W.t. Tsai et al.(2007). [22]	No	Web based application	S-CRM (simplified-coverage relationship model)		Specification	No (experiment)
Anjaneyulu et al.(2008). [2]	Yes	Windows based application	InARTS	Fine	Code	No (case study)
Marek and Radex (2008). [14]	No		SIPOC diagram			Yes
Praveen et al. (2008). [16]	Yes	multi agent air ticket booking	JADE 3.5 (java agent development framework)	Fine	Code	No (case study)
Lingming et al.(2009). [13]	Yes	SIR	Eclipse plug in named Jtop	Fine	Code	No
Chen et al.(2010). [5]	Yes	Six c programs from SIR	DejaVu, WEKA 3.7.5.	Coarse	Code	Yes (experiment)
Chuanqi et al.(2010). [6]	Yes	Siena, Nanoxml (SIR)	JHSA Prototype Tool	Coarse level to Fine level	Code	Yes (experiment)
Agastya et al. (2011). [1]	No	iTrust	TRENDS	No	Requirement	Yes (experiment)
Bharti and Shweta (2011). [4]	Yes	yes	ACO_TCSP	Fine	Code	No (experiment)
Emelie et al.(2011). [8]	No	At sony ericsson mobile communication	Commercial tool, HP's quality center		Non Code	Yes (case study)
Georg et al.(2013). [10]	Yes	Omicron Software suite	TFS(Tean Foundation Server), Ncover, Sherlocks	Fine	Code	No (case study)
Hesham and Izzat(2013). [11]	Yes	Jedit 4.2, Velocity 1.4, Velocity 1.6	WEKA 3.7.5.	Coarse	Code	No (experiment)
Prachet et al.(2013). [15]	No	SOA based application	(STPM) SOA testing perspective model proposed	No	Requirement	No (experiment)
Antonia et al.(2014). [3]	No	XACML simplified policy for library access	XACMUT , X-CREATE.	Coarse	Code	No (experiment)
Priyanka and Anita(2014). [17]	No	C++ module		No	Specification	No (experiment)

5.1 CODE BASED CRITERIA'S

This paper analyzed various single and multi-criterion code based technique. TABLE II summarizes most prevalent criteria's employed by code-based regression test selection techniques.

TABLE II: Code Based Criterions



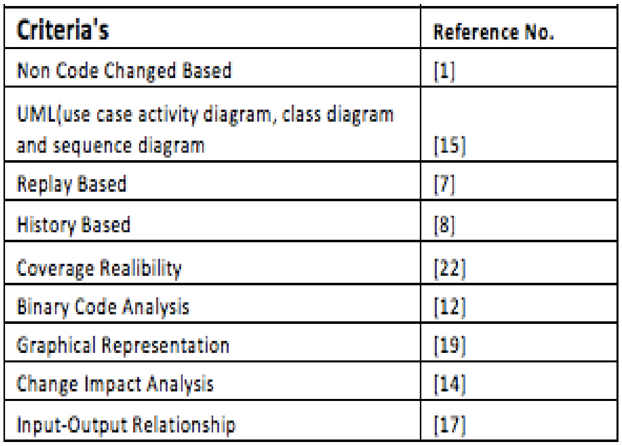
Criteria's	Reference No.
Fault Detection	[4, 20]
Execution Trace	[4]
Test Case Coverage	[10, 3, 6,5, 21,20,14]
Change Identification/Modification Impact of Pprogram	[9,2]
Relation between module of software	[11]
Mutation based	[3]
Hierarchical slicing	[6]
Absence of coverage information	[13]
Intermediate code	[21]
Textual differencing	[9]
Execution Cost	[20]
Common Execution Patterns	[5]
test history	[19]

As it is evident from TABLE II that techniques [3, 4, 5, 6, 9, 20, 21] are multi-criterion. It has been revealed in TABLE II that test case coverage is the most chief criterion used by various researchers. 50% of the code based technique follows test case that give maximum coverage [3,5,6,10,14,20,21] followed by fault detecting test cases [4,20] and change identifying test cases [2,9]. Test granularity is defined as the level of detail with which the tests address software. Two types of granularities were reported in various studies as shown in Table 1. Two types of granularities are fine granularity and coarse granularity. Under the code based technique 64.28% techniques reported fine granularity and 35.71% techniques reported coarse granularity. The studies showed a tradeoff between the effectiveness of a technique and the cost. Fine granularity result in effective test case selection but at the same time it could be time consuming and costly. Whereas coarse granularity based technique results in optimized set of test case with in given cost constrains (cost is determined in terms of time), the effectiveness of test cases selected still remains a questions.

5.2 REQUIREMENT BASED CRITETIA'S

Due to the non-availability of source code to the testers it becomes very difficult to perform regression testing. The solution to this problem would be to perform testing on the basis of requirements. The requirements of a system can be defined in terms of input –output relationship [17] as tabulated in TABLE III.

TABLE III: Requirement Based Criterions



Criteria's	Reference No.
Non Code Changed Based	[1]
UML(use case activity diagram, class diagram and sequence diagram	[15]
Replay Based	[7]
History Based	[8]
Coverage Realibility	[22]
Binary Code Analysis	[12]
Graphical Representation	[19]
Change Impact Analysis	[14]
Input-Output Relationship	[17]

Non-code change based approach can relate to file configuration and databases [1]. It has been tabulated in Table III that nine criterions adds to the non- code based techniques/ requirement based techniques. Requirement based testing becomes imperative when there is non-availability of source code.

Graph models have been extensively used in both code-based techniques and requirement based techniques. This research analyzed several representations of graph models for regression test selection techniques. This paper analyzed 21 techniques out of which 52.38% techniques were graph based, as it is evident from Table 1.

From Table 1 it is evident that no specific technique in this study was found to be general as all technique correlate to different domains.

6. CONCLUSIONS AND FUTURE WORK

This paper has analyzed twenty one techniques for regression test selection by different authors. There are two main classes under which techniques are studied, code-based and requirement-based. Further this paper identified most prevalent criteria's used by researchers for regression test selection techniques. Accordingly the techniques were grouped on the basis of their criteria use. This study identified that selection of test cases is predominantly centered on the coverage of test case followed by test case fault detection and change identification capability. Nevertheless test case effectiveness can also be determined with it level of granularity. Two type of test granularity were identified under code-based regression testing i.e. fine granularity and coarse granularity. The studies showed a tradeoff between the effectiveness of a technique and the cost. Fine granularity result in effective test case selection but at the same time it could be time consuming and costly.

Whereas coarse granularity based technique results in optimized set of test case with in given time and cost constrains, the effectiveness of test cases selected still remains a questions. Though the results of various researchers have shown that there are tradeoffs between the effectiveness and time/ cost constraints. It is also concluded from this study that no specific technique can be generalized as they deliberate using distinctive platform and they belong to specific dominion. Our future work will propose a multi-criterion technique for regression test case selection.

REFERENCES

[1] A. Nanda; S. Mani; S. Sinha; M. J. Harrold; and A. Orso, “Regression Testing in the Presence of Non-code Changes”, Proc. of Software Testing, Verification and Validation (ICST), 4th Int. Conf., IEEE, 21-25 March 2011, pp.21 – 30.

[2] A. Pasala; Y.L. H. L. Y. Fung; F. Akladios and R. P. Gorthi, “An approach to select regression tests to validate .NET applications upon deployment of components upgrades”, Proc. of 1st Bangalore Annual Compute Conf., ACM, 2008, Article No. 6.

[3] A. Bertolino; Y. L. Traon; F. Lonetti; E. Marchetti and T. Mouelhi, “Coverage- based Test Cases Selection for XACML Policies”. Proc. of Software Testing, Verification and Validation Workshops (ICSTW), 7th Int. Conf., IEEE, 31 March - 4 April 2014, pp. 12-21.

[4] B. Suri and S. Singal, “Implementing Ant Colony Optimization for Test Case Selection and Prioritization”, Int. J. on Computer Science and Engineering (IJCSE). ISSN: 0975-3397, May 2011, Vol. 3 No. 5.

[5] C. Zhang; Z. Chen; Z. Zhao; S. Yan; J. Zhang and B. Xu, “An Improved Regression Test Selection Technique by Clustering Execution Profiles”, Proc. of Quality Software (QSIC), 10th Int. Conf., IEEE, 14-15 July 2010, pp. 171-179.

[6] C. Tao; B. Li; X. Sun and Y. Zhou, “A Hierarchical Model for Regression Test Selection and Cost Analysis of Java Programs”. Proc. of Software Engineering Conf. (APSEC), 17th Asia Pacific, IEEE, 30 Nov – 3 Dec 2010, pp. 290-299.

[7] D. Sundmark; A. Pettersson and H. Thane, ” Regression Testing of Multi-Tasking Real-Time Systems: A Problem Statement”. Newslett. ACM SIGBED Review - Special issue: IEEE RTAS 2005 work-in-progress. Vol. 2/Issue. 2, April 2005, pp. 31-34

[8] E. Engstrom; P. Runeson and A. Ljung, “Improving Regression Testing Transparency and Efficiency with History-Based Prioritization – an Industrial Case Study”. Proc. of Software Testing, Verification and Validation (ICST), 4th Int. Conf., IEEE, 21-25 March 2011, pp. 367-376.

[9] F.I. Vokolos and P.G. frankl, “Pythia: a regression test selection tool based on textual differencing”. Proc. of ENCRESS '97 IFIP TC5 WG5.4, 3rd Int. conf. on Reliability, quality and safety of software-intensive systems, ACM, 1997, pp. 3-21.

[10] G. Buchgeher; C. Ernstbrunner; R. Ramler and M. Lusser, “Towards Tool-Support for Test Case Selection in Manual Regression Testing”, Proc. of Software Testing, Verification and Validation Workshops (ICSTW), 6th Int. Conf., IEEE, 18-22 March 2013, pp. 74-79.

[11] H. Abandah and I. Alsmadi, “Call Graph Based Metrics To Evaluate Software Design Quality”, Int. J. of Software Engineering and Applications, January 2013, Vol. 7/No. 1.

[12] J. Zheng; L. Williams; B. Robinson and K. Smiley, “Regression Test Selection for Black-box Dynamic Link Library Components”, Proc. of Incorporating COTS Software into Software Systems: Tools and Techniques, 2nd Int. Workshop, IEEE, 20-26 May 2007, pp. 9.

[13] L. Zhang; J. Zhou; D. Hao; L. Zhang and H. Mei, “Jtop: Managing JUnit Test Cases in Absence of Coverage Information”. Proc. of Automated Software Engineering, 24th Int. Conf., IEEE, 16-20 November 2009, pp. 677-679.

[14] M.G. Stochel and R. Sztando, “Testing Optimization for Mission - Critical, Complex, Distributed Systems”, Proc. of Computer Software and Applications, 32nd Annu. Int. Conf., IEEE, 28 July – 1 Aug 2008, pp. 847-852.

[15] P. Bhuyan; A. Kumar and D. P. Mohapatra, “SOA Testing Perspective Model for Regression Testing”, Proc. of Engineering (NUiCONE), Nirma University Int. Conf., IEEE, 28-30 November 2013, pp. 1-5.

[16] P. R. Srivastava; V. A. Karthik; V. S. Reddy and G. Ragurama. “Regression Testing Technique for Agent Oriented Software”. Proc. of Information Technology, Int. Conf., IEEE, 17-20 Dec. 2008, pp. 221-225.

[17] P. Rana and A. Ganpati, “An Algorithm for Optimal Test Case Selection for Specification Based Regression Testing”; Proc. of 9th Nat. Conf. on SACTA, ISBN 978-81-928-3802-1, 19 April 2014, pp. 284-290.

[18] R. Pressman, “Software Engineering: A Practitioner’s Approach”, New York, McGraw-Hill, 2002.

[19] S. Beydeda and V. Gruhn, “Class specification implementation graphs and their application in regression testing”, Proc. of Computer Software and Applications, 26th Annu. Int. Conf., IEEE, 2002, pp. 835-840.

[20] S. Yoo and M. Harman, “Pareto Efficient Multi Objective Test Case Selection”. Proc. of Int. Symp. on Software testing and analysis, ACM, 2007, pp. 140-150.

[21] Toshihiko Koju; S. Takada and N. Doi, “Regression Test Selection Based on Intermediate Code for Virtual Machine”, Proc. of Software Maintenance, Int. Conf., IEEE, 2003, 22-26 Sept 2003, pp. 420-429.

[22] W.T. Tsai; X. Zhou; R. A. Paul; Y. Chen and X. Bai, “A Coverage Relationship Model for Test Case Selection and Ranking for Multi-version Software”, Published in High Assurance Systems Engineering Symp., IEEE, 2007, 14-16 Nov. 2007, pp. 105-112.