Prioritization in Automotive Software Testing: Systematic Literature Review

Ankush Dadwal, Hironori Washizaki, Yoshiaki Fukazawa
Department of Computer Science and Engineering
Waseda University
Tokyo, Japan
ankush.dadwal@toki.waseda.jp

Takahiro Iida, Masashi Mizoguchi, Kentaro Yoshimura
Control Platform Research Department
Center for Technology Innovation - Controls
Hitachi, Ltd. Research & Development Group
takahiro.iida.ac@hitachi.com

***Abstract*—Automotive Software Testing is a vital part of the automotive systems development process. Not identifying the critical safety issues and failures of such systems can have serious or even fatal consequences. As the number of embedded systems and technologies increases, testing all components becomes more challenging. Although testing is expensive, it is important to reduce bugs in an early stage to maintain safety and to avoid recalls. Hence, the testing time should be reduced without impacting the reliability. Several studies and surveys have prioritized Automotive Software Testing to increase its effectiveness. The main goals of this study are to identify: (i) the publication trends of prioritization in Automotive Software Testing, (ii) which methods are used to prioritize Automotive Software Testing, (iii) the distribution of studies based on the quality evaluation, and (iv) how existing research on prioritization helps optimize Automotive Software Testing.**

***Index Terms*—Automotive Software Testing, Prioritizing, Systematic Literature Review**

I. Introduction

Currently the automotive industry is undergoing a major transition. Automakers have been adding new functions and systems to meet the market's demand for an ever-growing amount of software-intensive functions. However, these new functions and systems have some negative aspects. One is that automakers must enhance their testing techniques because vehicle complexity is increasing. Testing typically consumes more than half of all development costs [4]. While testing a single software system is difficult, testing without prioritization is even more challenging due to the exponential number of products and the number of features. Today, software determines more than 90% of the functionality in automotive systems and software components are no longer handwritten [6].

Test case prioritization is a method to prioritize and schedule test cases. In this technique, test cases are run in the order of priority to minimize time, cost, and effort during the software testing phase. Every organization has its own methods to prioritize test cases. The automotive safety standard ISO26262 requires extensive testing with numerous test cases. To achieve a high productivity, the availability of quality assurance systems must be high [7].

Herein we use a systematic literature review to evaluate relevant publications on prioritization in the automotive industry. A systematic review aims to assess scientific papers in order to group concepts around a topic. Through analysis criteria, it allows the quality of research to be evaluated. Herein the system review aims to identify common techniques in automotive testing and to define new challenges.

The paper is structured as follows. Section II describes related works. The systematic literature review approach is detailed in Section III. Section IV presents the results obtained from the systematic review. Section V addresses potential threats to validity. Finally, Section VI lists the conclusions and the definitions for future work.

II. Related Works

Automakers have experienced the impact of the evolution of technology on automotive testing. Today, testing all systems manually is not only cost-intensive and time-consuming but nearly impossible. Automating the testing phase would significantly reduce the cost of software development [14]. Literature about the prioritization efforts in the automobile industry is scarce. Herein we focus on known techniques and their applicability to the investigated domain.

In the past few decades, numerous studies [5], [6], [7], [8], [9], [10], [13], [15], [16], [17], [19], [20], [24], [27] have demonstrated that vehicles are becoming increasingly more complex and more connected. For example, an empirical study, which aimed to investigate the potential regarding quality improvements and cost savings, employed data from 13 industry case studies as part of a three-year large-scale research project. This study identified major goals and strategies associated with (integrated) model-based analysis and testing as well as evaluated the improvements achieved [25].

The only study we found that reviews the literature about the benefits and the limitations of Automated Software Testing is presented by Mantyla et al, [3] (2012). Their review, which included 25 works, tries to close the gap by investigating academic and practitioner views on software testing regarding the benefits and the limits of test automation. They found that while benefits often come from stronger sources of evidence (experiments and case studies), limitations are more frequently reported in experience reports. Second, they conducted a survey of the practitioners' view. The results showed that the main benefits of test automation are reusability, repeatability, and reduced burden in test executions. Of the respondents, 45% agreed that the available tools are a poor fit for their needs and 80% disagreed with the vision that automated testing would fully replace manual testing.

III. Methodology

We started the systematic literature review by specifying our scope and searching only documents in the domain of automotive software testing that discuss issues related to prioritization in the field of testing. Topics that focus only on software testing without prioritizing the test cases are excluded. In this research, we followed the guidelines suggested in papers [1], [2] . This method is divided into three steps.

A. Research Questions

This study strives to answer the following research questions:

RQ1. What are the **publication trends** of *prioritization in Automotive Software Testing*?
This research question should characterize the interest and ongoing research on this topic. Additionally, it will identify relevant venues where results are being published and the contributions over time.

RQ2. What are the **methods used** for *prioritization in Automotive Software Testing*?
This research question should elucidate the different methods used for prioritization in Automotive Software Testing. The goal here is to determine the main methods and tools used by researchers.

RQ3. How are the studies distributed based on a **quality evaluation** of *prioritization in Automotive Software Testing*?
This research question should reveal the quality distribution of the selected primary studies and evaluate them accordingly.

RQ4. How does **existing research** on prioritization help with the *optimization of Automotive Software Testing*?
This research question should classify existing and future research on prioritization in Automotive Software Testing and assess current research gaps. This is the most important and challenging question as it aims to compile problems that have yet to solved.

B. Search and Selection Process

The search and selection process is a multi-stage process (Fig. 1). This multi-stage process allows us to fully control the number and characteristics of the studies that are considered during various stages.

As mentioned in [1] and [2], we used three of the largest scientific databases and indexing systems in software engineering: ACM Digital Library, IEEE Xplore, and Scopus. These were selected because they are common, effective in systematic literature reviews in software engineering, and capable of exporting the search results. Further, these databases provide mechanisms to perform keyword searches. We did not specify a fixed time frame when conducting the search. To cover as many significant studies as possible, the systematic literature search query was very generic and considered the object of our research (i.e., Prioritization in Automotive Software Testing).

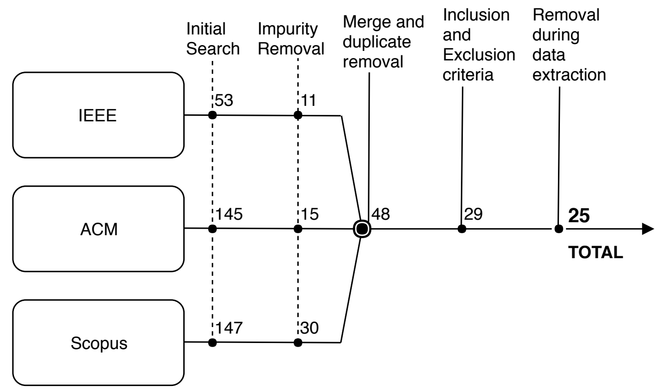

Fig. 1. Paper selection process

1) **Initial Search***:* We performed a search in three of the largest and most complete scientific databases and indexing systems in software engineering: ACM Digital Library, IEEE Xplore, and Scopus. We searched these databases using a search string that included the important keywords in our four research questions. Further, we augmented the keywords with their synonyms, producing the following search string:

((”automobile” OR ”automotive” OR ”car”)
AND
(”software” OR ”program” OR ”code”)
AND
(”prioritization” OR ”priority” OR ”case selection”)
AND
(test*))

For consistency, we executed the query on titles, abstracts, and keywords of papers in all the data sources at any time and any subject area.

2) **Impurity Removal***:* Due to the nature of the involved data sources, the search results included some elements that were clearly not research papers such as abstracts, international standards, textbooks, etc. In this stage, we manually removed these results.

3) **Merge and Duplicate Removal***:* Here we combined all studies into a single dataset. Duplicated entries were matched by title, authors, year, and venue of publication.

4) **Inclusion and Exclusion Criteria***:* We considered all the selected studies and filtered them according to a set of well-defined selection criteria. The inclusion and exclusion criteria of our study are:

Inclusion criteria:

- Studies focusing on software testing specific to the automotive industry.
- Studies providing a solution for prioritizing Automotive Software Testing.
- Studies in the field of software engineering.
- Studies written in English.

Exclusion criteria:

- Studies that focus on the automotive industry, but do not explicitly deal with software testing.
- Studies where software testing is only used as an example.
- Studies not available as full-text.
- Studies not presented in English.
- Studies that are duplicates of other studies.

5) **Removal during Data Extraction***:* When reviewing the primary studies in detail to extract information, all the authors agreed that four studies were semantically beyond the scope of this research. Consequently, they were excluded.

C. Data extraction

Relevant information was extracted to answer the research questions from the primary studies. We used data extraction forms to make sure that this task was carried out in an accurate and consistent manner. The data was collected and stored in a spreadsheet using MS Excel to list the relevant information of each paper. This technique helps extract and view data in a tabular form.

The following information was collected from each paper:

- Publication title
- Publication year
- Publication venue
- Problems faced by the authors
- Testing method used
- Limitations in field
- Detail of the proposed solution
- Results obtained
- Rating of quality issues
- Verification and validation
- Future work suggested by the authors
- Conclusions
- Answers to research questions

IV. Analysis Results

This section presents the analysis and each sub-section answers the previously presented research questions. We used the R software environment and Microsoft Excel to perform basic statistical operations and draw charts.

A. Publication Trends (RQ1)

Figure 2 presents the distribution of publications over time. The most common publication types are conference papers (17/25) followed by workshop papers (5/25), journals (2/25), and symposiums (1/25). The high number of conference papers may indicate that prioritization of automotive software testing is maturing. A small but constant number of publications were published until 2014. However, prioritization has become an important and eye-catching aspect in terms of research since 2014. The interest in prioritization of automotive software testing has rapidly increased in the last few years.

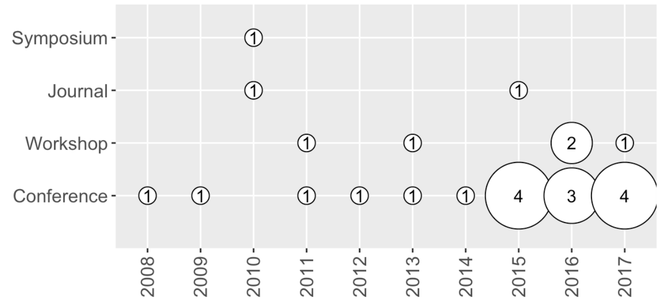

Fig. 2. Primary studies distributed by type of publication over the years

TABLE I
APPLIED RESEARCH STRATERGIES

Res. strategies	#Studies	Studies
Solution Proposal	20	P4, P5, P6, P7, P8, P9, P10, P11, P12, P16, P17, P18, P19, P20, P21, P22, P23, P24, P26, P27
Evaluation Research	15	P5, P8, P10, P11, P13, P15, P16, P17, P18, P19, P20, P21, P22, P26, P28
Validation Research	14	P5, P8, P9, P10, P11, P12, P16, P18, P20, P21, P22, P24, P26, P28
Opinion Paper	4	P13, P14, P25, P27
Survey Paper	1	P25

Studies published before 2015 refer to slightly different perspectives on prioritization than more recent papers. The number of papers has drastically increased since 2014. . [25] [11] [4] [6] [10] used model-based testing to improve the prioritization by increasing the effectiveness. On the other hand, [6] [10] showed potential improvements and proposed new model-based methods.

Many researchers provided solution proposals (20/25) and evaluation research (15/25) (Table I), indicating that today's researchers focus on industry and practitioner-oriented studies (e.g., industrial case study, action research). Another common research strategy is validation research (14/25), highlighting the fact that there is some level of evidence (e.g., simulations, experiments, prototypes, etc.) supporting the proposed solutions. However, Table I also shows that few studies employ surveys (1/25), suggesting that future studies should fill this gap.

B. Methods Used (RQ2)

Due to the requirement of connected services for vehicles, an interesting method is model-based testing. Figure 3 depicts a histogram of the distribution of the most common techniques in the literature. The most common testing methods are model-based testing (7/25), regression testing (6/25), and black box testing (5/25), followed by hardware in the loop testing

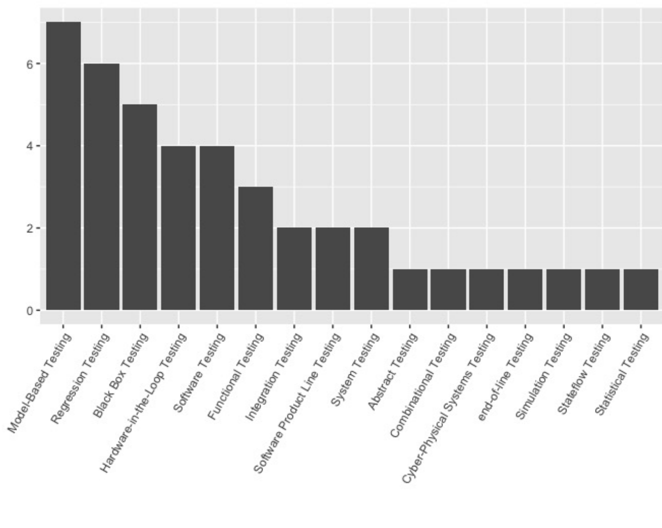

Fig. 3. Testing Methods

(4/25), software testing (4/25), functional testing (3/25), and other.

Approaches that use model-based testing are found in [4] [6] [9] [10] [11] [25] [26]. Techniques listed as "OTHER" refer to the use of integration testing [10] [24], software product line testing [4] [11], system testing [5] [8], abstract testing [17], combinational testing [18], cyber-physical system testing [13], end-of-line testing [7], simulation testing [12], stateflow testing [15], and statistical testing [20].

Different methods or combination of methods are used in multiple studies (Table II). These papers frequently target model-based testing (7/25), regression testing (6/25), and black box testing (5/25). Model-based development is an efficient, reliable, and cost-effective paradigm to design and implement complex embedded systems. The software determines more than 90% of the functionality of automotive systems and up to 80% of the automotive software can be automatically generated from models [6]. Additionally, model-based testing is a common solution to test embedded systems in automotive engineering. Regression testing is undertaken every time a model is updated to verify quality assurance, which is time-consuming as it reruns an entire test suite after every minor change. Test case selection for regression testing after new releases is an important task to maintain the availability [7]. Typically studies focus on black-box testing scenarios because the source code is often unavailable in the automotive domain such as an OEM-supplier scenario [19]. Hardware in the loop testing (4/25) and software testing (4/25) are the next most used methods. The most common method of testing the software and the Electronic Control Units (ECU) is the use of Hardware-In-the-Loop (HIL) simulation [27]. Software testing presents an approach to automatically generate test cases for a software product. Functional testing (3/25) strives to demonstrate the correct implementation of functional requirements and is one of the most important approaches to gain confidence in the correct functional behavior of a system [28]. Integration testing (2/25), software product line testing (2/25), and system testing (2/25) are used as the time donation when the testing process is limited. A negative highlight of this systematic review is the fact that only one paper directly employs a simulation testing method [12]. If a simulation environment can imitate the key criteria of the real-world

TABLE II
APPLIED TESTING METHODS

Testing Methods	#Studies	Studies
Model-Based Testing	7	P4, P6, P9, P10, P11, P25, P26
Regression Testing	6	P5, P6, P7, P9, P11, P19
Black-Box Testing	5	P5, P8, P9, P16, P19
Hardware in the Loop Testing Testing	4	P8, P16, P27, P28
Software Testing	4	P14, P21, P23, P27
Functional Testing	3	P16, P22, P28
Integration Testing	2	P10, P24
Software Product Line Testing Testing	2	P4, P11
System Testing	2	P5, P8
Abstract Testing	1	P17
Combinational Testing	1	P18
Cyber-Physical Systems Testing	1	P13
End-of-Line Testing	1	P7
Simulation Testing	1	P12
Stateflow Testing	1	P15
Statistical Testing	1	P20

TABLE III
RATING OF REVIEWED ARTICLES

Reference	QI1	QI2	QI3	QI4	Total
4	Y	Y	Y	Y	4
5	Y	P	P	Y	3
6	Y	P	Y	P	3
7	Y	Y	P	Y	3.5
8	Y	P	P	Y	3
9	Y	Y	P	P	3
10	Y	Y	N	P	2.5
11	P	Y	P	Y	3
12	Y	P	Y	P	3
13	Y	Y	Y	Y	4
14	Y	P	Y	Y	3.5
15	Y	P	Y	P	3
16	Y	Y	P	Y	3.5
17	Y	Y	Y	Y	4
18	P	P	Y	Y	3
19	Y	Y	Y	Y	4
20	Y	N	P	P	2
21	Y	P	P	Y	3
22	Y	Y	P	Y	3.5
23	P	N	Y	N	1.5
24	Y	Y	Y	Y	4
25	Y	Y	Y	P	3.5
26	Y	Y	P	Y	3.5
27	Y	Y	Y	N	3
28	Y	P	Y	P	3
				Average	3.2

TABLE IV
VALIDATION,TECHNIQUES/TOOLS, GAPS, AND MAIN OUTCOMES OF STUDIES

Validation	No. of Papers	Study	Techniques/Tools	Gaps	Main Outcome(s)
Industrial Case Study	14 (56%)	P5	Test case selection based a on Stochastic model	Find better heuristic clustering approaches	Regression effort can be minimized
		P8	Test case selection based on a component and communication model	Change deployed libraries	82.3% reduction in tested functions
		P10	Taster tool, Proposed new framework for MBT	Focus on the optimization of the classification structure used within the priority assignment procedure	New framework for the MBT
		P11	Dissimilarity-based TCP	Investigate the fault detection capabilities of the approach	Dissimilarity-based TCP issues are solved
		P15	Test selection algorithms	Develop optimal guidelines to divide test oracle budget across the output-based selection algorithms	Output-based algorithms consistently outperform coverage-based algorithms in revealing faults
		P16	Evolutionary Testing Framework, modularHiL, MESSINA	Investigate how to configure the evolutionary testing system to reduce the number of pre-tests	Solution provide complements systematic testing in that it generates test cases for situations that would otherwise be unforeseen by testers
		P17	Migration from traditional to abstract testing	Extend the formalism to handle non-functional requirements	Abstract testing is comparable in effectiveness
		P18	Equivalence Class Partitioning (ECP), Boundary Value Analysis (BVA), Choice Relationship Framework (CRF)	Investigatee efficient test case generation and discover more feasible tools and empirical studies to work	Efficient reduction in the final effective number of test cases by 42 (88% reduction)
		P19	Test case combination	Six approaches are presented to improve test efficiency	Machine learning approach in black-box testing
		P20	Combination of test models in MATLAB/Simulink	*NO GAPS MENTIONED*	Higher coverage is achieved compared with manually created test cases
		P21	Automatically generate test cases	Refine the functional coverage model	Improvement actions are identified for test case generation
		P22	Proposed unified model	Implement a large survey on software specifications in Johnson Controls company	More than 90% of the requirements are represented by the model
		P26	End-to-end test framework	Investigate the nature of the test model and the relevance to the generated test cases	Automation of executable test script generation
		P28	Search-based testing principles	Further investigate safety requirements	Promising approach to ensure safety requirements
Technique Comparison	6 (24%)	P4	Similarity-Based Product Prioritization w.r.t Deltas	Include more Solution space information (Such as Source code)	Improvement in the effectiveness of SPL testing
		P6	New model-based method for test case prioritization	Implement performance evaluation with a large-scale case study	Future regression testing can be sped up
		P7	Combined fault diagnosis and test case selection	Evaluate using an end-of-line test system at a real assembly line	System can find test cases to increase the test coverage
		P13	Weight-based search algorithms	Study weight tuning, different fitness functions, and cost-effectiveness measures	Results suggest that all the search algorithms outperform Random Search
		P23	Model slicing technique for optimal test case generation	*NO GAPS MENTIONED*	Complexity of Simulink models can be reduced
		P24	OUTFIT tool	Evaluate it with other domains such as medical and avionic systems	Potential defects can be effectively identified
Statistical Evaluation	1 (4%)	P9	Dependences between the components of embedded systems	Study exhaustiveness of the path search and correctness of path search	Reduction in test-cases for regression testing
Simulation	1 (4%)	P12	Parallelly execute loosely coupled segments	Fully automate the process of segmentation and instrumentation	Reduce the simulation testing time for both successful and failed runs
Others	3 (12%)	P14	Three approaches to automatically generate MC/DC test cases	Focus on MC/DC test case generation from formal specifications	Approaches can be combined to support different kinds of decisions
		P25	Survey paper	Publish more detailed description of the applied evaluation approach	Improvements that are possible with MBAT technologies
		P27	Study on requirements	*NO GAPS MENTIONED*	Synect provides easy test requirement specifications and management of the test results

environment, it should be used to provide early feedback on the vehicle's design.

C. Quality Evaluation (RQ3)

According to [29], a ranking was created to rate papers based on the relevance to the topic and the quality of the paper. Quality Issues (QI) are:

- QI1 Is the paper's goal clear?
- QI2 Does the assessment approach match the goals?
- QI3 Can the method be replicated?
- QI4 Are results shown in detail?

For each quality issue, articles were rated as: Yes (Y) when the issue is addressed in the text, Partial (P) when the issue is partially addressed in the text, and No (N) when the issue is not addressed in the text. These ratings were scored as Yes = 1 point, Partial = 0.5, and No = 0. Table III shows the papers that were analyzed in this SLR and their respective scores based on the Quality Issues discussed above.

D. Existing Research (RQ4)

Here we discuss the recurring problems that are targeted by primary studies, which methods described in RQ2 are validated, and the gaps mentioned in the research.

Recurring problems are time consumption (15/25), cost (13/25), and complexity (14/25) followed by test case selection (3/25) and quality improvement (3/25) (Table V). Because the testing time is expensive, it should be reduced without an uncontrolled reduction of reliability. The entire test suite must be rerun each time the system is updated or modified. Consequently, each modification makes the testing process more time-consuming. Automotive systems are becoming more complex due to a higher rate of integration and shared usage. The high complexity results in numerous interfaces, and many signals must be processed inside the system [9]. Testing activities can account for a considerable part of the software production costs. However, only two studies discuss improving efficiency [18] and safety [26] which is a negative highlight.

Table IV presents the studies within each category, Techniques/Tools, gaps, and the main study outcomes. Most studies are focus on industrial case studies (n = 14) and technique comparisons (n = 6).

TABLE V
TARGET PROBLEMS

Problems	#Studies	Studies
Time Consumption	15	P5, P6, P8, P9, P10, P12, P13, P14, P15, P16, P17, P19, P20, P24, P27
Complexity	14	P5, P6, P7, P8, P9, P10, P13, P15, P16, P17, P19, P20, P24, P27
Cost	13	P5, P6, P8, P9, P10, P13, P15, P16, P17, P19, P20, P24, P27
Test Case Selection	3	P7, P11, P28
Quality Improvement	3	P22,P25,P26
Test Generation	2	P21,P23
Problem Space Information	1	P4
Improving Efficiency	1	P18
Safety	1	P26

TABLE VI
VALIDATION

Techniques	#Studies	Studies
Industrial Case Study	14	P5, P8, P10, P11, P15, P16, P17, P18, P19, P20, P21, P22, P26, P28
Technique Comparison	6	P4, P6, P7, P13, P23, P24
Statistical Evaluation	1	P9
Simulation	1	P12
Others	3	P14, P25, P27

Table VI lists the techniques used to validate the selected studies. The most common are industrial case studies (14/25) followed by technique comparisons (6/25), statistical evaluations (1/25), simulations (1/25), and others (3/25). Technique comparisons include studies that propose and then compare a new method to an old one. Others include three studies, which [14] [27] talk about the advantages, limitations, and requirements of different approaches. [25] is a survey paper from 13 industry case studies.

V. Threats to Validity

The analysis was conducted by a single person. Thus, one threat is that some information may be omitted. Moreover, the analysis is limited by the analytical skills of that single person.

VI. Conclusion

This paper overviews the Prioritization in Automotive Software Testing. The results should help companies that are planning to incorporate prioritization into their strategies. Researchers can also benefit because this study depicts the limitations and gaps in current research. Additionally, the trends in other embedded and non-embedded domains must be investigated as this should provide a more detailed picture and lessons learned regarding prioritization in Automotive Software Testing. Future work includes (i) a qualitative study to better understand test execution, test case generation, test case selection, and test analysis and (ii) addressing the identified research gaps.

References

[1] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," 2007.

[2] B. Kitchenham and P. Brereton, "A systematic review of systematic review process research in software engineering," *Inf. Softw. Technol.*, vol. 55, no. 12, pp. 2049–2075, Dec. 2013. [Online]. Available: http://dx.doi.org/10.1016/j.infsof.2013.07.010

[3] D. M. Rafi, K. R. K. Moses, K. Petersen, and M. V. Mäntylä, "Benefits and limitations of automated software testing: Systematic literature review and practitioner survey," in *2012 7th International Workshop on Automation of Software Test (AST)*, June 2012, pp. 36–42.

[4] M. Al-Hajjaji, S. Lity, R. Lachmann, T. Thüm, I. Schaefer, and G. Saake, "Delta-oriented product prioritization for similarity-based product-line testing," in *2017 IEEE/ACM 2nd International Workshop on Variability and Complexity in Software Design (VACE)*, May 2017, pp. 34–40.

[5] I. Alagöz, T. Herpel, and R. German, "A selection method for black box regression testing with a statistically defined quality level," in *2017 IEEE International Conference on Software Testing, Verification and Validation (ICST)*, March 2017, pp. 114–125.

[6] A. Morozov, K. Ding, T. Chen, and K. Janschek, "Test suite prioritization for efficient regression testing of model-based automotive software," in *2017 International Conference on Software Analysis, Testing and Evolution (SATE)*, Nov 2017, pp. 20–29.

[7] S. Abele and M. Weyrich, "A combined fault diagnosis and test case selection assistant for automotive end-of-line test systems," in *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*, July 2016, pp. 1072–1077.

[8] S. Vöst and S. Wagner, "Trace-based test selection to support continuous integration in the automotive industry," in *2016 IEEE/ACM International Workshop on Continuous Software Evolution and Delivery (CSED)*, May 2016, pp. 34–40.

[9] P. Caliebe, T. Herpel, and R. German, "Dependency-based test case selection and prioritization in embedded systems," in *2012 IEEE Fifth International Conference on Software Testing, Verification and Validation*, April 2012, pp. 731–735.

[10] L. Krejčí and J. Novák, "Model-based testing of automotive distributed systems with automated prioritization," in *2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, vol. 2, Sept 2017, pp. 668–673.

[11] R. Lachmann, S. Lity, M. Al-Hajjaji, F. Fürchtegott, and I. Schaefer, "Fine-grained test case prioritization for integration testing of delta-oriented software product lines," in *Proceedings of the 7th International Workshop on Feature-Oriented Software Development*, ser. FOSD 2016. New York, NY, USA: ACM, 2016, pp. 1–10. [Online]. Available: http://doi.acm.org/10.1145/3001867.3001868

[12] M. A. Al Mamun and J. Hansson, "Reducing simulation testing time by parallel execution of loosely coupled segments of a test scenario," in *Proceedings of International Workshop on Engineering Simulations for Cyber-Physical Systems*, ser. ES4CPS '14. New York, NY, USA: ACM, 2007, pp. 33:33–33:37. [Online]. Available: http://doi.acm.org/10.1145/2559627.2559635

[13] A. Arrieta, S. Wang, G. Sagardui, and L. Etxeberria, "Test case prioritization of configurable cyber-physical systems with weight-based search algorithms," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, ser. GECCO '16. New York, NY, USA: ACM, 2016, pp. 1053–1060. [Online]. Available: http://doi.acm.org/10.1145/2908812.2908871

[14] S. Kangoye, A. Todoskoff, M. BARREAU, and P. GERMANICUS, "Mc/dc test case generation approaches for decisions," in *Proceedings of the ASWEC 2015 24th Australasian Software Engineering Conference*, ser. ASWEC ' 15 Vol. II. New York, NY, USA: ACM, 2015, pp. 74–80. [Online]. Available: http://doi.acm.org/10.1145/2811681.2811696

[15] R. Matinnejad, S. Nejati, L. C. Briand, and T. Bruckmann, "Effective test suites for mixed discrete-continuous stateflow controllers," in *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, ser. ESEC/FSE 2015. New York, NY, USA: ACM, 2015, pp. 84–95. [Online]. Available: http://doi.acm.org/10.1145/2786805.2786818

[16] P. M. Kruse, J. Wegener, and S. Wappler, "A highly configurable test system for evolutionary black-box testing of embedded systems," in *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '09. New York, NY, USA: ACM, 2009, pp. 1545–1552. [Online]. Available: http://doi.acm.org/10.1145/1569901.1570108

[17] F. Merz, C. Sinz, H. Post, T. Gorges, and T. Kropf, "Bridging the gap between test cases and requirements by abstract testing," *Innov. Syst. Softw. Eng.*, vol. 11, no. 4, pp. 233–242, Dec. 2015. [Online]. Available: http://dx.doi.org/10.1007/s11334-015-0245-7

[18] J. S. Eo, H. R. Choi, R. Gao, S. y. Lee, and W. E. Wong, "Case study of requirements-based test case generation on an automotive domain," in *2015 IEEE International Conference on Software Quality, Reliability and Security - Companion*, Aug 2015, pp. 210–215.

[19] R. Lachmann and I. Schaefer, "Towards efficient and effective testing in automotive software development," in *Lecture Notes in Informatics (LNI), Proceedings - Series of the Gesellschaft fur Informatik (GI)*, vol. P-232, 2014, pp. 2181–2192, cited By :2. [Online]. Available: www.scopus.com

[20] S. Siegl, K. S. Hielscher, and R. German, "Modeling and statistical testing of real time embedded automotive systems by combination of test models and reference models in matlab/simulink," in *2011 21st International Conference on Systems Engineering*, Aug 2011, pp. 180–185.

[21] R. Awedikian and B. Yannou, "Design of a validation test process of an automotive software," *International Journal on Interactive Design and Manufacturing*, vol. 4, no. 4, pp. 259–268, 2010, cited By :2. [Online]. Available: www.scopus.com

[22] R. Awedikian, B. Yannou, P. Lebreton, L. Bouclier, and M. Mekhilef, "A simulated model of software specifications for automating functional tests design," in *Proceedings DESIGN 2008, the 10th International Design Conference*, 2008, pp. 561–570, cited By :1. [Online]. Available: www.scopus.com

[23] Z. Jiang, X. Wu, Z. Dong, and M. Mu, "Optimal test case generation for simulink models using slicing," in *Proceedings - 2017 IEEE International Conference on Software Quality, Reliability and Security Companion, QRS-C 2017*, 2017, pp. 363–369. [Online]. Available: www.scopus.com

[24] D. Holling, A. Hofbauer, A. Pretschner, and M. Gemmar, "Profiting from unit tests for integration testing," in *Proceedings - 2016 IEEE International Conference on Software Testing, Verification and Validation, ICST 2016*, 2016, pp. 353–363, cited By :1. [Online]. Available: www.scopus.com

[25] M. Klas, T. Bauer, A. Dereani, T. Soderqvist, and P. Helle, "A large-scale technology evaluation study: Effects of model-based analysis and testing," in *Proceedings - International Conference on Software Engineering*, vol. 2, 2015, pp. 119–128, cited By :4. [Online]. Available: www.scopus.com

[26] J. Lasalle, F. Peureux, and J. Guillet, "Automatic test concretization to supply end-to-end mbt for automotive mechatronic systems," in *2011 International Workshop on End-to-End Test Script Engineering, ETSE 2011 - Proceedings*, 2011, pp. 16–23, cited By :4. [Online]. Available: www.scopus.com

[27] A. Bansal, M. Muli, and K. Patil, "Taming complexity while gaining efficiency: Requirements for the next generation of test automation tools," in *AUTOTESTCON (Proceedings)*, 2013, pp. 123–128, cited By :1. [Online]. Available: www.scopus.com

[28] F. Lindlar and A. Windisch, "A search-based approach to functional hardware-in-the-loop testing," in *Proceedings - 2nd International Symposium on Search Based Software Engineering, SSBSE 2010*, 2010, pp. 111–119, cited By :7. [Online]. Available: www.scopus.com

[29] U. Kanewala and J. M. Bieman, "Testing scientific software: A systematic literature review," *Inf. Softw. Technol.*, vol. 56, no. 10, pp. 1219–1232, Oct. 2014. [Online]. Available: http://dx.doi.org/10.1016/j.infsof.2014.05.006

