Classification Trends Taxonomy of Model-based Testing for Software Product Line: A Systematic Literature Review

Rabatul Aduni Sulaiman[1], Dayang Norhayati Abang Jawawi[2*], and Shahliza Abdul Halim[2]
[1] Department of Software Engineering, Faculty of Computer Science and Information System, Universiti Tun Hussein Onn Malaysia, 86400 Parit Raja, Johor, Malaysia
[e-mail: rabatul@uthm.edu.my]
[2] Department of Software Engineering, School of Computing, Faculty Engineering Universiti Teknologi Malaysia, 81300 Skudai, Johor, Malaysia
* Corresponding author: Dayang Norhayati Abang Jawawi

Received January 26, 2022; revised March 31, 2022; accepted April 28, 2022; published May 31, 2022

Abstract

Context: Testing is one of the techniques that can assure the quality of software including the domain of Software Product Line (SPL). Various techniques have been deliberated to enhance the quality of SPL including Model-based Testing (MBT).
Objective: The objective of this study is to analyze and classify trends of MBT in SPL covering the solutions, issues and evaluation aspects by using taxonomy form.
Method: A Systematic Literature Review (SLR) was conducted involving 63 primary studies from different sources. The selected studies were categorized based on their common characteristics.
Results: Several findings can guide future research on MBT for SPL. The important finding is that the multiple measurements are still open to improving current metrics to evaluate test cases in MBT for SPL. The multiple types of measurement required a trade-off between maximization and minimization results to ensure the testing method which could satisfy multiple test criteria for example cost and effectiveness at the same time.

Keywords: model-based, model-based testing, software product line, variability

1. Introduction

Software Product Line (SPL) gathers related software features that share similar functionality yet vary from each other in terms of explicit features. New products can be derived by reusing and effectively combining software assets. This will produce a speedier and low-cost strategy that would still maintain high-quality products meeting the market demand, enabling it to compete with competitors. An explosion in the number of product variants could impede the management of products [1]. An increase in product size will result in difficulties in product management process. Testing in SPL typically examines the interactions and relations between core assets that are shared by many products derived from the product line [2].

One of the highly favored techniques for SPL testing is the Model-based Testing (MBT) technique which generates test cases by capturing requirements using models for SPL product artifacts. In this paper, we present a Systematic Literature Review (SLR) with the primary objective of providing, identifying, and interpreting MBT as a technique that handles SPL testing. This study aims to link the important points and present state-of-the-art advancements that have been accomplished in MBT for SPL. The specific focus of this paper is based on current issues of MBT in SPL, evaluation measures and solutions proposed by researchers. This study provides an overview of different perspectives of MBT for SPL in taxonomy format. Representation of MBT classification based on taxonomy can be considered the newest classification in the SPL field. In this study, the taxonomy format will combine different dimensions in a single figure. The rest of this SLR is structured as follows: Section 2 presents the introduction of existing literature. Section 3 discusses SLR methodology. Section 4 presents the result and discussion whereas Section 5 elaborates on the threats to validity. Section 6 presents the conclusion and future works.

2. Analysis of Existing Literatures

Five SLR studies were found to be related to the domain of MBT for SPL as shown in **Table 1**. In terms of SLR, Razak *et al.* [3] discussed the general overviews of MBT for SPL. They classified MBT in terms of domains, approaches, and existing models that have been applied in SPL. However, this work lacks in discussing the existing issues, solutions and evaluation methods of MBT.

Table 1. Summary of related SLR studies

Study Type	**References**	**Scope**	**Year of Publication**	**Total Papers Reviewed**	**Years Covered**
SLR	Razak *et al.* [3]	MBT in SPL	2017	25	2005-2015
	Saeed *et al.* [4]	Search-based for MBT	2016	72	2001-2013
	Jihyun *et al.* [5]	Test Criteria for SPL	2020	78	2014-2017
	Maíra *et al.* [6]	SPL	2020	60	1999-2017
	Matias *et al.* [7]	Variability Model	2021	43	2007-2018

Saeed *et al.* [4] specifically reviewed search-based techniques via MBT in the software engineering domain. The results indicated that many solutions and techniques exist, but the study did not consider how MBT research is being evaluated by existing researchers in the software engineering field. Jinyun *et al.* [5] conduct SLR on test coverage criteria for SPL. This study clarifies the test coverage based on 63 selected studies from the year 2014 until 2017. This study is differing from our aim since it is more on a test basis and test coverage.

Maíra *et al.* [6] cover SLR in terms of evolution. However, this study covers different evolution artifacts implemented in SPL for a variety of domains. Matias *et al.* [7] worked on SLR for the variability model in SPL. This study differs from our aim to categorize MBT for SPL in solution, issue and evaluation. All these SLRs have provided a good overview of the current issues in each domain. However, none of them provides a comprehensive inspection of the whole aspects of MBT in SPL in a single report. We believe that an SLR trends report can ease the researcher's work in exploring relevant literature. Based on the existing review, this study can provide new contributions in different facets such as in terms of weaknesses of previous approaches and issues with SPL core assets since this work focused on multiple facets of MBT in SPL.

3. Systematic Literature Review (SLR) Methodology

SLR is defined as the process of identifying, evaluating, and integrating findings from previous research to address research questions and interests of new research. In order to have an acceptable analysis from this SLR, protocol guidelines provided by [11] were strictly observed. The guidelines outline several steps in developing an SLR, including review protocol development, conducting reviews, results from the analysis, and conclusion discussion which helps focus to increase the transparency of researched products and reduce research biasness.

3.1 Research questions

Kitchenham [12] provided guidelines to frame research questions based on criteria defined by researchers in the Software Engineering domain. These guidelines have five main elements that lead to the best searchable question. The guidelines are based on the Population, Intervention, Comparison, Outcome and Context (PICOC) criteria collected from other sources. Based on guidelines, the research questions of this SLR have been constructed as displayed in **Table 2**.

Table 2. Summary of SLR's research questions

Research Question (RQ)	RQ Statement	Motivation
RQ1	What are the issues solved based on MBT for SPL?	To know the trends of issues highlighted for implementation of MBT for SPL.
RQ2	What are the current MBT solutions that have been applied to solve the issues in SPL?	To help other researchers to evaluate previous research solutions for their related experiments.
RQ3	What are the types of evaluation used to measure the quality of MBT results for SPL?	To helps researchers select types of suitable evaluation to verify the results of testing based on MBT.

3.2 Identification of relevant studies

The process of collecting relevant studies from selected search engines and online databases is depicted in **Fig. 1**. We divided the process into two phases. The first phase consists of collecting published studies. Next, the snowball process is used to combine the sources from the manual and automated search processes. Then, all irrelevant studies and unrelated papers were removed, resulting in the final selected studies. Detail for each phase is provided below:

3.2.1 Phase 1: Collecting Primary Studies

This phase includes the process of searching for an updated list of primary studies. We performed an automated search process on four main database engines: ScienceDirect, ACM, IEEEXplore and SpringerLink. Table 3 lists the search strings applied for automated and manual searches. The majority of publications in the Software Engineering field are found in Springer, ScienceDirect and IEEEXplore, while the majority of SPL studies are found in ScienceDirect and SpringerLink. From this task, we obtained 315 studies of automated and 71 studies manual.

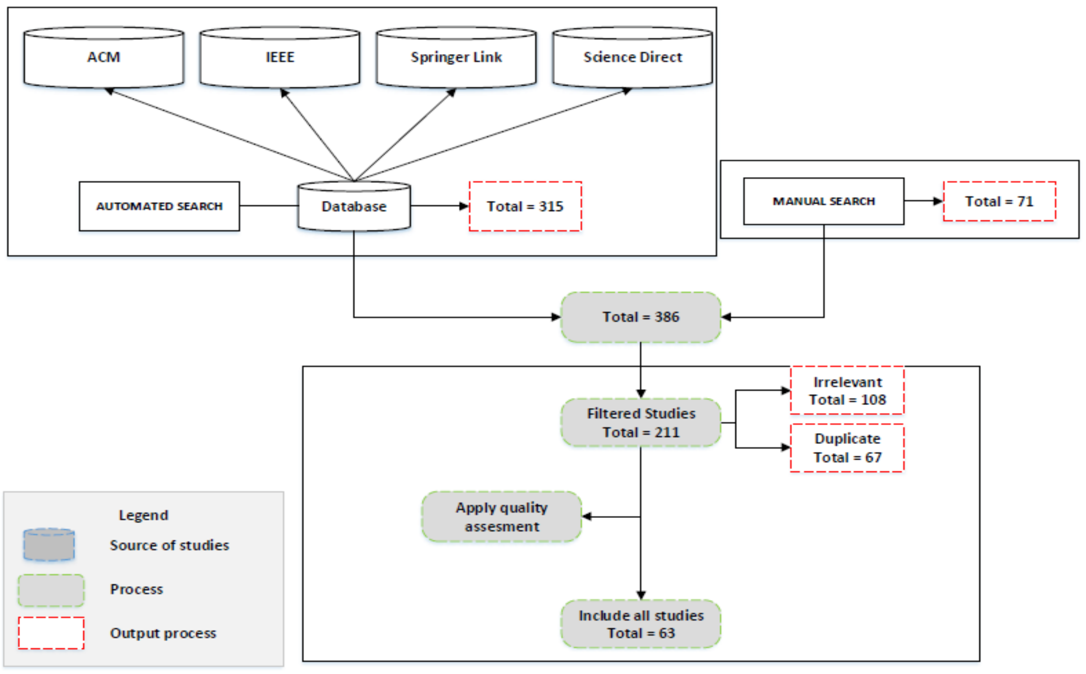

Fig. 1. Publication search procedures

Table 3. Search strings

Terms	Search strings
SPL terms:	"software product line", "software product family", "product line", "product families", "software family", "system families", "variability",
Testing:	"test", "testing", "software testing"
Model based terms:	"model-based", "model based", "MBT", "model driven"

3.2.2 Phase 2: Selection Strategies

In this phase, we identified duplications and unrelated topics of studies. Unrelated papers were removed upon scrutinizing the publications' titles, abstracts, and introductions. Next, inclusion and exclusion criteria were established to assess the best list of publications. We applied inclusion and exclusion criteria based on the titles and abstracts of published studies listed in Table 4. Next, for exclusion criteria for example out of scope, redundant papers, papers that did not focus on SPL testing, and papers that were written in languages other than English were also removed. This task aimed to ensure only relevant studies are included in this research. At the end of this phase, we ended up with 211 publications.

Table 4. List of exclusion criteria

Exclusion Criteria	Details
Out of Scope	By checking the abstract and introduction sections, unrelated topics is figure out.
Abstract Paper	We exclude papers based on their posters, tutorials and editor keynotes.
Language	We exclude papers that use other languages aside from English.
Comparison Paper	We exclude comparison papers that only perform comparison of previous studies.

Next, a quality assessment was done on the preliminary collection of papers. Data extracted for this assessment include year, title, authors, source and types of publication, objectives, problems solved and notes. This screening process was done by two researchers to prevent any bias in the selection process. Discussions were done to settle any disagreement and uncertainty related to this assessment process. A set of questionnaires based on practice of conducting empirical research especially in Software Engineering and Critical Appraisal Skill Programed (CASP). As listed in **Table 5**, the questionnaire used consists of 10 questions. Upon completion of this phase, only 63 studies have been selected. The primary studies were categorized according to publication types, which include journal, conference paper, workshop, book and thesis. In **Table 6**, out of the 63 studies, 36 consists of conference papers, 19 journal articles, seven theses, one article and one book.

Table 5. Quality assessment questions

No	Details	Issue
1.	Assessment questions	Report
2.	Does the paper provide clear research aims?	Report
3.	Does the paper have clear problem statements?	Relevance
4.	Are there adequate descriptions on the context carried out?	Relevance
5.	Does the paper state the current issues to be solved?	Rigor
6.	Are the proposed strategies clearly described?	Rigor
7.	Is the case study implemented in the paper?	Relevance
8.	Does the research contributes to the research community?	Credibility
9.	Does the paper provide sufficient data analysis process?	Credibility
10	Are there clear findings statements in the paper?	Relevance

Table 6. Summary of primary studies collected based on publication type and year

	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	All
Journal	1	1	3	-	2	2	3	3	1	1	1	1	19
Conference	3	1	3	5	8	2	4	1	2	3	2	2	36
Workshop	-	-	-	1	-	-	-	-	-	-	-	-	1
Thesis	-	-	1	-	2	-	1	1	-	1	1	-	7
Book	-	1	-	-	-	-	-	-	-	-	-	-	1
Total	4	3	7	6	12	4	8	5	3	4	4	3	**63**

4. Results and Discussion

In this section, firstly, the overview of MBT in SPL is elaborated according to taxonomy as illustrated in **Fig. 2**. The purpose of this step is to assess state-of-the-art MBT in SPL research. Each RQ1 until RQ3 describes one facet of the taxonomy.

4.1 Classification Schema

This section discusses how the taxonomy classification facets are defined. Issues facet reviews the issues raised by the researchers. Solutions facet refers to the proposed solutions to solve issues and achieve good testing results in SPL. This facet covers solutions provided in different processes. Evaluations facet refers to assessments carried out by researchers to evaluate their proposed work in the primary studies. This facet is used to assess types of case studies, faults, coverage and cost of the proposed solutions. The next section will describe details on each MBT trend based on the facets defined earlier.

4.2 RQ1. What are the issues solved based on MBT for SPL?

Issues facet comprises two categories, which are core assets and test cases quality. Each category will be discussed as follows:

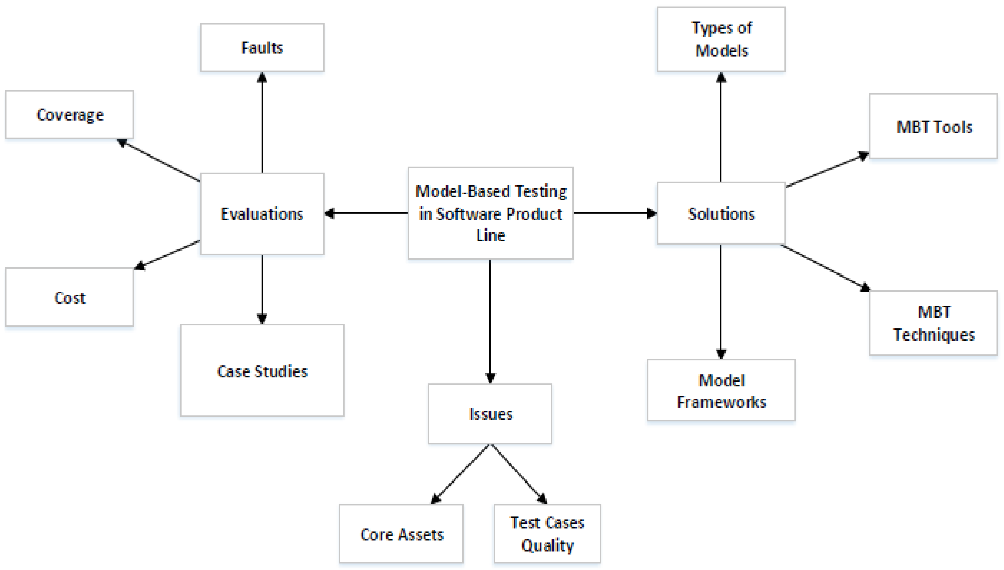

Fig. 2. Classification trends of MBT for SPL in taxonomy form

Core Assets

As depicted in **Fig. 3**, core assets were highlighted in 23 out of 63 primary studies (36.5%) covering variability requirement and reusability issues. Variability issue was reported in 11 studies [13]–[20] whereas requirement issue was reported in three studies [21]–[23]. From the analysis, it is evident that variability is highly emphasized by researchers, indicating the importance of managing variability in SPL through any means including MBT. Techniques for handling variability need to be kept simple and efficient to manage variability abstraction effectively. In addition, there are issues related to variability modeling that was purposely used to handle variability complexity. Reusability in SPL also highlights concerns by previous researchers. This issue has been discussed in nine studies [15],[17],[22]–[28]. Reusability is still an open issue in SPL to satisfy high-level quality attributes. Since the key purpose of SPL is to develop the core assets, it requires a reusability attribute in the core assets to ensure that common functionality and variability can be captured among products.

Test Case Quality

Test case quality is also one of the issues that are frequently highlighted by researchers in primary studies. We found that there are four types of test cases related to quality issues comprising performance, usability, functional suitability, and reliability. Quality attributes were emphasized in 31 out of 63 primary studies (49.20%), and some studies focused on more than two quality attributes. Performance is the most popular quality attribute issue. This issue is related to the efficiency of an approach to handle a large number of product variants. This issue often occurs in SPL as end users are incapable to customize SPL product variants. The results showed that performance attribute was highly investigated from 2004 until 2015. Performance was highlighted in 13 studies [16],[19],[27]–[37].

Functional suitability is also one of the quality attributes emphasized in five studies [27],[30],[34],[38],[39]. Reliability attribute related to faults is one of the quality issues in MBT for SPL. The fundamental issue in software testing is to relate results with software faults. In this SLR, reliability issue related with faults is covered in eight primary studies [34],[35],[40]–[45]. Next, usability attribute is also one of the quality attributes highlighted where in this SLR, the attribute was highlighted in five studies [15],[20],[21],[41],[43].

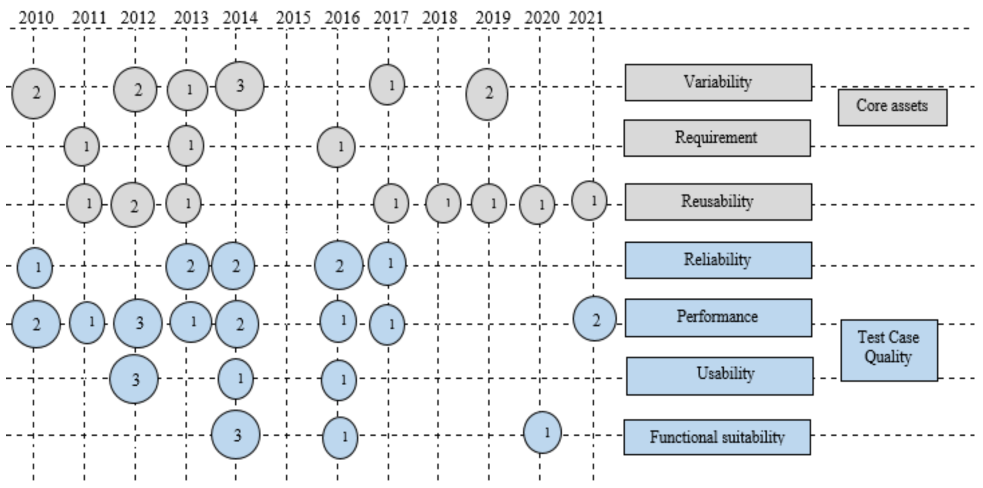

Fig. 3. Bubble plot of MBT in SPL issues

4.3 RQ2. What Are the Current MBT Solutions That Have Been Applied to Solve Issues in SPL?

As depicted in **Fig. 4**, solutions sub-category is further divided into four main parts consisting types of models, MBT techniques, MBT tools, and model frameworks. Detail of each sub-category will be described in the following sections:

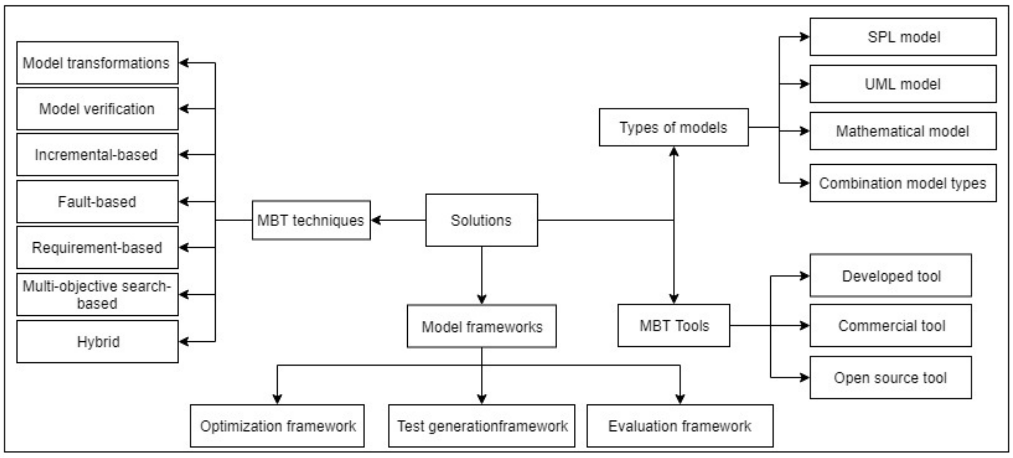

Fig. 4. Classification trends of solutions

Types of Models

Types of models refer to the model that was used to improve the limitations of previous studies. Different model styles lead to different levels of efficiency and functionalities. The various models implemented in the primary studies are provided in **Fig. 5**. The results show that the UML model is mostly utilized compared to other models. In total, 26 out of 63 (41.26%) primary studies used multiple UML types in their studies. Commonly utilized UML models in MBT for SPL include statecharts, use cases, activity diagrams, class diagrams and sequence diagrams. Statechart was found in 17 primary studies whereas use cases were found in six primary studies.

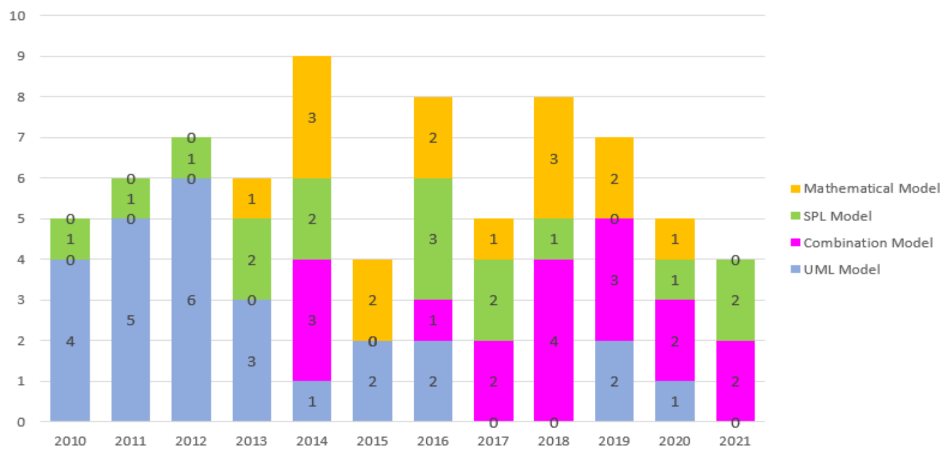

Fig. 5. Types of models implemented in MBT for SPL

Meanwhile, sequence diagrams were found in three primary studies [31],[42],[46]. Class diagrams were found in two primary studies. However, similar to sequence diagrams, class diagrams have also been integrated with other models to form a single approach for generating test cases [23],[47]. SPL model comprises FM, OVM and delta model. SPL model was found in 16 out of 63 primary studies. There is only one study reported the usage of OVM and three studies reported the use of the delta model. Instead of capturing variability, additional functionalities were used to identify valid product configurations [48], configure MBT model parameters [17], integrate with other models such as UML [49], and capture overall test cases

[50]. We found that in these recent years, there are studies that implemented delta modeling as an independent transformations variability approach for SPL development. The method is used to capture commonality and variability by using deltas as operations [51],[52].

Combination types refer to primary studies that combine the usage of more than one type of model in a single approach. There are 17 out of 63 studies using combination model types. In this area, many studies combined the functionality of the SPL model with UML [58] or mathematical model [53]. A mathematical model was used in 15 out of 63 primary studies. This category comprise four mathematical-based models, encompassing Markov chain, Parnas table, directed graph, and Label Transition System (LTS). LTS model was used in eight out of ten primary studies. Feature Transition System (FTS) and Modal Transition System (MTS), which are extensions of LTS were used in two out of five LTS model studies. Markov chain was used in three out of ten mathematical model categories. Meanwhile, the directed graph was used in one out of ten mathematical models. In addition, one primary study based on the set of tables was presented.

MBT Techniques

For the MBT technique, there are seven sub-categories comprising model transformation, model verification, incremental-based, fault-based, requirement-based, multi-objective search-based, and hybrid techniques. **Fig. 6** shows the types of MBT techniques implemented in SPL. There are 28 studies that applied and proposed techniques in solving product line issues. The model transformation was applied in seven out of 63 primary studies from 2010 until 2019.

Four types of model transformations process were found, consisting of flattening, oracle, code generation, and data generation. Several researchers proposed a model transformation by flattening the model into a propositional formula [23],[54]. Other primary studies covered model-to-code transformations based on metamodel [47],[55]. For test generation, several studies proposed model transformation in delta modelling [20] and dynamic model [41],[56]. Model verification technique or also known as MBT model checker has been implemented in eight out of 63 primary studies comprising code [45],[57] and design model checking [33],[37],[46],[58],[59]. Several works studies proposed model checking that improves variability-aware analyses by encoding SPL variability to simulate test cases [44]. The incremental-based approach was also applied in SPL through MBT where two out of 63 primary studies [16],[35]. The fault-based technique was found in six out of 63 primary studies. The technique is used to identify faults in modelling [60],[61] and improve the quality of testing [45],[46],[62],[63]. Fault-based MBT is proposed to manage SPL variability. Mutation testing was introduced to enhance the quality of this technique. Meanwhile, the requirement-based technique was found in two out of 63 primary studies. The technique is used to generate test cases for each SPL requirement [30],[46].

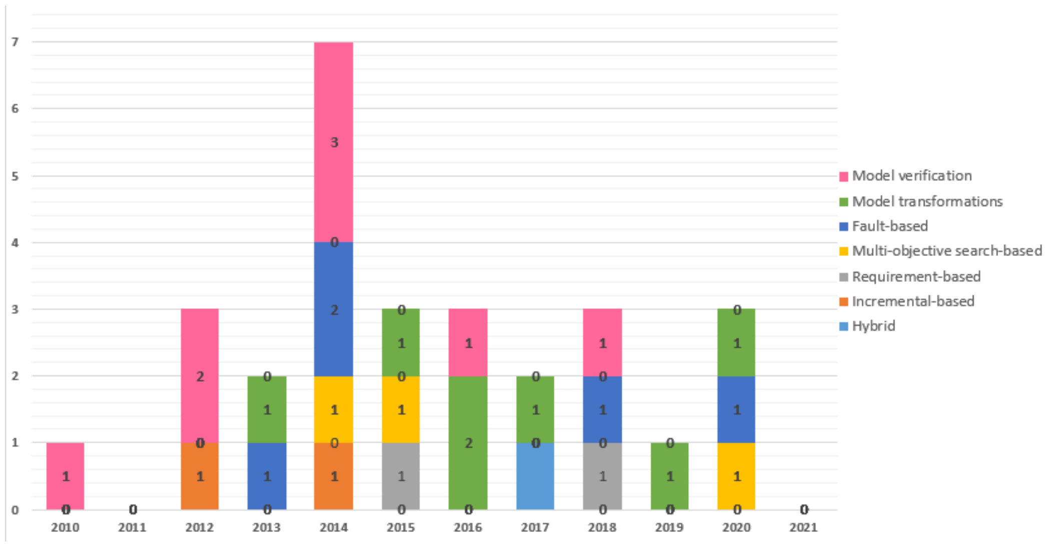

Fig. 6. MBT techniques for SPL

In addition, four studies implemented multi-objective search-based technique to achieve high coverage [64], optimization [37],[62] and generation of test cases [65]. Optimization strategies in MBT are often implemented by using search-based algorithms to handle test selection, minimization, and prioritization. In terms of algorithms, CIT and heuristic have been used to solve optimization problems. In this SLR, MBT techniques that are used in combination are considered as hybrid approaches. Two primary studies implemented a hybrid approach in MBT for SPL. Tripartite-oriented technique comprising aspect-oriented, feature-oriented, and object-oriented techniques in a single approach was used to support MBT [66],[67]. The hybrid technique implemented as a single approach remains deficient in handling a large number of SPL product features.

MBT Tools

As depicted in **Fig. 7**, MBT tools were used in 19 out of 63 primary studies, comprising commercial tools, open-source tools, and developed tools. A commercial tool refers to any tool that requires an official license before installation. Open-source tool refers to any tool that is available freely and can be downloaded from the internet. Meanwhile, a developed tool refers to any tool that is developed by researchers in primary studies for academic purposes.

Five commercial tools were found in the primary studies comprising IBM software products [29],[37],[59],[68], Pure::Variants [17],[34],[50],[52],[53], AspectT [60], Test Development for UML (TDE/UML) [49], and Automatic Efficient Test Generator (AETG) [39],[70]. Five primary studies used open-source tools to develop models, generate test cases, and execute tests. The tools comprise of FeatureIDE [48],[52],[71],[72], PLEDGE [65], TopCased [73], Microsoft Spec Explorer [74] and NModel [49][47]. For developed tools, nine tools were proposed by researchers for academic research comprising Unified Modelling Language All Purposes Transformer (UMLAUT) [68], Test Generation Language (TGL) [43], Partition Test Generator (ParTeG) [75], Common Variability Language (CVL) [47], Import Plugin and Transformation (IPT) [66], DeltaTables [67], Transformation-based Tool for UML-based Testing (TRUST) [34],[64],[76], AGEDIS [51],[77] and, Alloy [69],[78],[79].

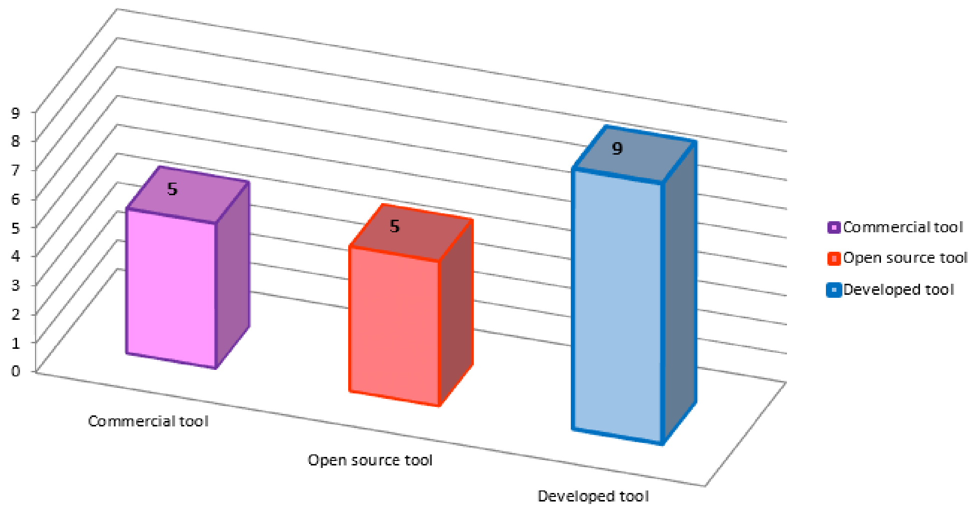

Fig. 7. MBT tools for SPL

Model Framework

There is also model framework proposed to conduct MBT for SPL. There are 12 out of 63 studies that discussed processes involving testing. As depicted in **Fig. 8**, we categorize model framework into three main categories which are optimization framework [28], test generation framework [19],[23],[28],[37],[76]–[78], and evaluation framework [15],[30],[40],[43]. In terms of test generation framework, seven MBT frameworks were proposed, covering mainly delta-oriented processes and test case derivation processes. Four studies proposed MBT evaluation frameworks to evaluate MBT techniques implemented in SPL. In terms of optimization framework, formal frameworks were proposed to optimize test suites from a set of product variants by using linear programming.

4.4 RQ3. What are the types of evaluations used to measure the quality of MBT results for SPL?

As depicted in **Fig. 9**, four evaluation types were used by researchers to evaluate their proposed solutions, namely types of case study, cost, faults, and effectiveness. These evaluations are often used to validate the effectiveness of testing from test cases generated. The following sections describe each evaluation type.

Types of Case Studies

Types of case studies are used to measure the strengths and weaknesses of the proposed work. As depicted in **Fig. 10**, case studies can be divided into three types comprising experimental, industrial, and random case studies. Out of the three categories, an experimental-based case study was chosen by a majority of researchers in 25 out of 63 (39%) studies, followed by an industrial case study in 20 out of 63 (31%) studies, and lastly, a random case study in 19 out of 63 (30.0%) studies.

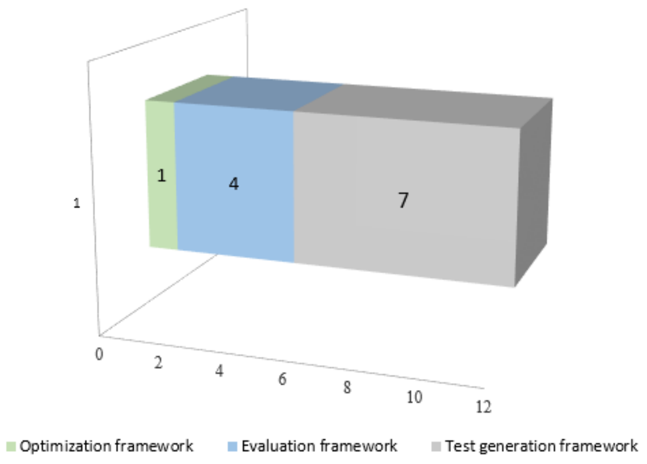

Fig. 8. MBT frameworks for SPL

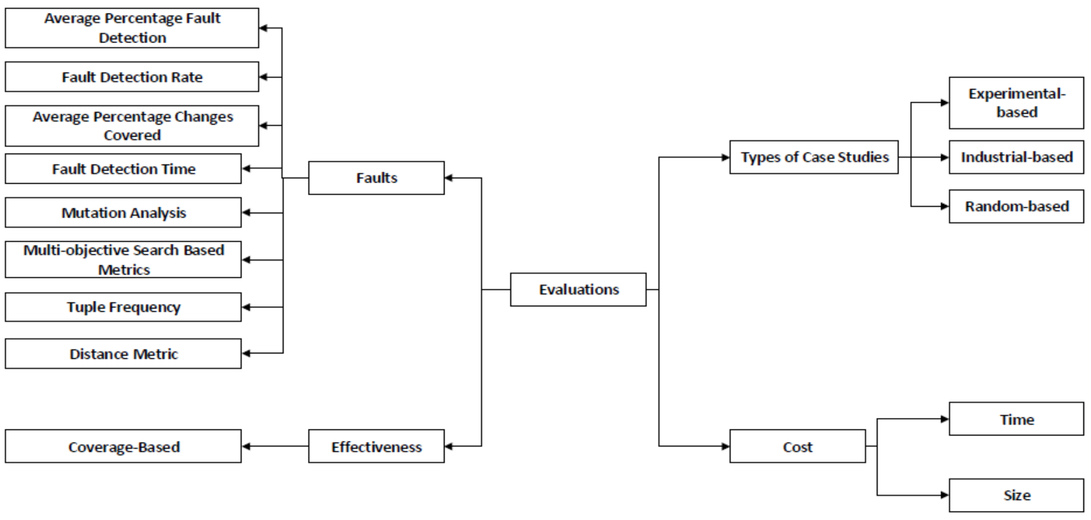

Fig. 9. MBT evaluation trends

Further, it is observed that case studies which are one of the quantitative research methods depended heavily on the types of SPL products being researched. An experimental-based case study is a popular choice among researchers to conduct SPL research followed by an industrial case study. This shows that MBT is not only applicable in the academic sphere but also useful in the industrial sphere. In contrast, some researchers choose to pick random case studies to be used in SPL research, for example, open-access case studies hosted in the SPLOT repository. This process is a legitimate approach to conducting a research. Occasionally, some of these randomly handpicked case studies have been used in previous SPL research as benchmarks. Results show that industrial and academic research are highly concerned in solving SPL issues based on MBT.

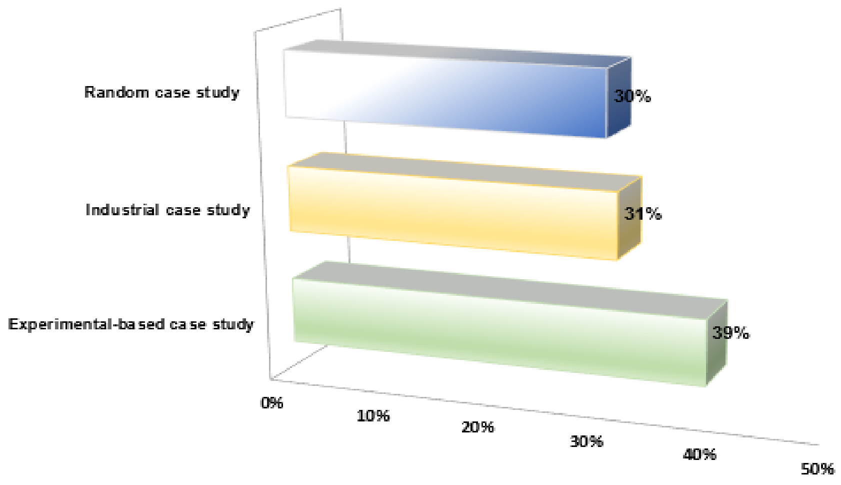

Fig. 10. Types of case studies

Cost

Cost analysis is related to the cost of selection, generation, and execution of test cases [80]. In this SLR, the cost was determined by inferring previous techniques used to measure cost. Based on the analysis, 57 studies (90.4%) were found to utilize either a single technique or a combination of more than one technique to measure cost. Referring to **Table 7**, test suite size is the most preferable technique to measure cost and was used in 20 out of 63 empirical studies. This method is used to measure cost for test generation by comparing the size of the test suite in pre and post-application of proposed solutions. Specifically, statistical analysis was used to compare the size of test suites generated, including Vargha and Delaney statistics, Kruskal-Wallis, and Mann-Whitney U test.

Table 7. Cost Measurements

Types	Cost Measure	Related studies
Size	Test suite size	[20],[27],[30],[33],[36],[40],[44],[45],[50],[61],[62],[64],[65],[81]–[87]
	Cost of single test case	[18],[29],[37],[42],[50],[53],[80],[88]
Time	Test execution time	[27],[28],[30],[41],[48],[50],[59],[61],[70],[88]
	Optimization time	[35],[36],[48],[53],[89]
	Bounded cost of execution time	[36],[62],[64],[86]
	Time for finding defects	[35],[37],[45],[53],[60],[67],[80],[84],[89],[90]

Aside from that, mathematical formulas have also been used to measure the extent of similarity of test configurations. Jaccard distance [65] computes the size of intersection similarity that exists in a sample test suite. The second most reported cost measurement technique is execution times to find defects and test execution times with 10 studies. This cost measurement technique is used to measure the total time taken by test execution to generate test suites. Quantifying the cost of a single test case was observed in eight studies. These two techniques are not commonly used to measure cost since MBT is commonly used to generate test suites. In addition, optimization is one of the most recent strategies in MBT. However, only five studies have implemented the cost of optimization time for SPL. Meanwhile, bounded cost of the execution time was observed in only four primary studies.

Faults

In this SLR, evaluation metrics used to measure faults discovered were also reviewed. As depicted in **Fig. 11**, eight evaluation metrics were used in the primary studies to measure faults. These metrics were arranged based on year type. Overall, 27 studies (42.8%) have either reused existing evaluation metrics or proposed new metrics. From the analysis, Average Percentage of Fault Detection (APFD) is the most popular metric used in six out of 63 (9.52%) primary studies, where it is used to measure fault detection capability [39],[53],[67],[86],[90],[91]. On the other hand, four out of 63 studies (6.34%%) utilized multi-objective search-based metrics [17],[53],[64],[90] and four studies used distance metrics (6.34%) [84],[89],[92],[93]. Other metrics include three studies (4.761%) on Fault Detection Rate (FDR), mutation analysis and tuple frequency [45],[86],[94]. There is one study (1.58%) on Average Percentage of Changes Covered (APCC) and Fault Detection Time (FDT) [36],[95]. There are two studies (3.17%) that proposed reusability metrics to evaluate reusability [96].

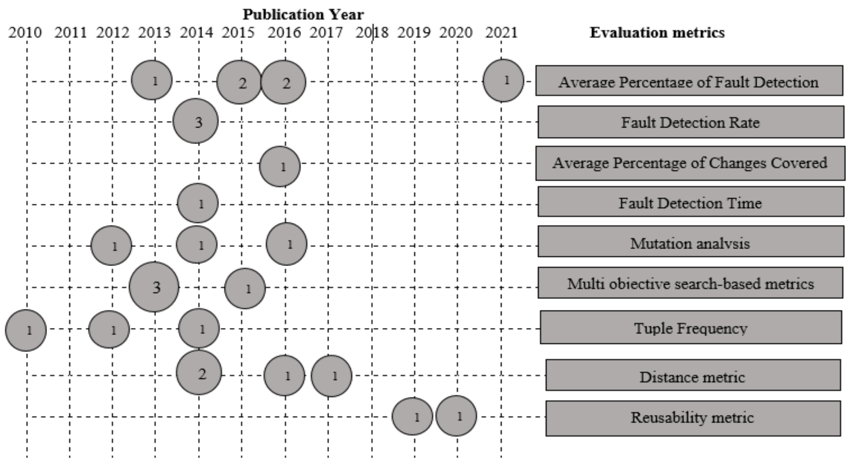

Fig. 11. Bubble plot of fault evaluation metrics

Based on the analysis, several studies utilized more than one evaluation metric in order to compare the results of test case generation, optimization, and selection. APFD is frequently used to evaluate the detection faults in each test suite. APFD is utilized in various MBT areas, comprising test case generation and test case optimization. Meanwhile, multi-objective search-based metrics are also popular since there is a demand to evaluate fault, cost, and effectiveness for test case generation. There is only one study that developed new evaluation metrics, namely FDT and APCC metrics. APCC is also a new metric that was proposed to validate faults, which was adapted from APFD [45]. The metric was proposed since no failure information was available to measure the changes in product variants. The reusability metric has been implemented in SPL to measure the core assets. There are two studies that have proposed a reusability metric in MBT for SPL.

Effectiveness

Effectiveness in SPL is commonly measured by using coverage-based criteria. Coverage-based criteria establish the parameters of effective testing based on the goal defined by researchers. As depicted in **Fig. 12**, four sub-categories of coverage criteria are used in primary studies, comprising fault-based coverage, model-based coverage, data coverage, and feature-based coverage. Fault-based coverage was adopted to solve issues and errors that cannot be solved by n-wise coverage [94]. Based on analysis results, nine out of 63 studies applied fault-based coverage in MBT for SPL by implementing mutation testing. This method is rarely used since it involves expensive process [92]. Model-based coverage that covers six criteria, comprising path, all-states, all-action, all-transition, branch, and statement coverage was used in 14 out of 63 primary studies. Path coverage exists in many MBT test generators. It is an efficient method to use when there are a large number of products to test. All-state coverage was found in a limited number of studies since this coverage is recognized as the weakest coverage criteria [83]. The all-states coverage process requires each state to be visited at least once. Next, data coverage refers to n-wise coverage that is commonly applied in MBT to automate test generation and fulfil n-wise test adequacy criteria in two primary studies [27],[39]. This method is used to minimize combinatorial space by considering all product configurations with the primary aim of detecting faults in testing. In terms of test generation, researchers have explored strategies to satisfy the n-wise coverage of a test suite generated.

Next, 12 of 63 primary studies applied for feature coverage in testing. Three out of 63 studies utilized multiple types of coverage to evaluate test cases and model-based coverage [28],[40],[89]. Meanwhile, nine out of 63 primary studies measured the quality of MBT by using coverage and fault. On the other hand, six primary studies applied three types of evaluations, whereas three primary studies applied coverage and cost. Based on these trends, cost, fault, and coverage are highly emphasized in MBT for SPL research. Coverage acts as a proxy to validate the quality of the test suite. Coverage is frequently used in MBT for SPL as it is often used as an indicator to measure the effectiveness of the test suite. However, sole implementation of coverage as a completeness indicator is inadequate to yield a good testing quality. It is so because other test qualities also need to be measured such as effectiveness in terms of fault and cost of implementation. For the cost, many empirical studies quantified cost in terms of test suite size and execution time. Whereas, in fault measurement, APFD is widely used to quantify fault detection. In this recent years, two other quality indicators namely cost and fault is relatively new in MBT for SPL. Thus, much work is needed to implement all three types of quality measures (coverage, cost, and fault) in a single empirical study.

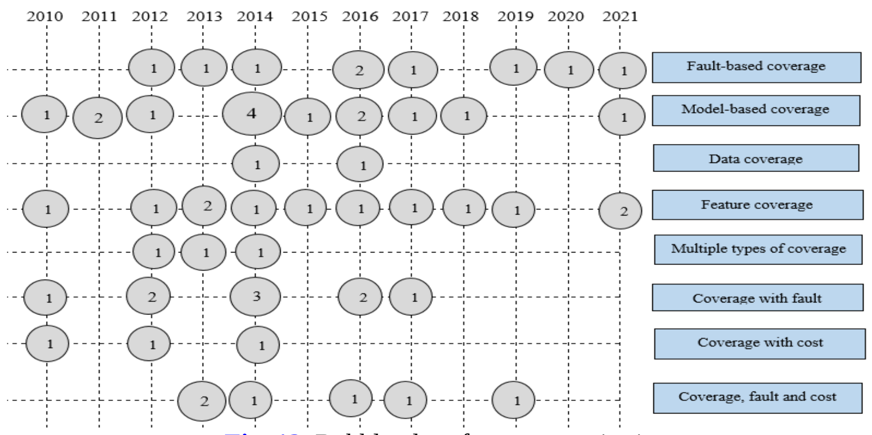

Fig. 12. Bubble plot of coverage criteria

4.5 Research Findings

A detailed summary is given in this section based on the review analyses. Industry players expect to generate test cases efficiently, utilizing shorter duration and minimal implementation cost. Due to this, researchers often seek to obtain the best test case generation solutions that fulfill those requirements. Many studies focused on MBT test case generation techniques as solutions to manage variabilities. The current MBT techniques still require further enhancements since there is an open challenge on scalability with related to large SPL product space. Furthermore, there are also issues in the performance of existing MBT techniques, such as the implementation of multiple model types in a single approach and the validity of a model to be used as a medium for test generation, which is sometimes overlooked in testing. Aside from that, current testing strategies also need to be improved to better handle product variants. Test cases with maximal coverage that can be generated in minimal execution time. The multi-objective problem is used similarly with this evaluation measure in MBT for SPL. Next, a summary of this SLR is provided below:

MBT research is still broadly open: From the analysis, a new improvement to existing approaches are constantly implemented. From the results, researchers proposed different kinds of solutions based on the problems in terms of scalability, finding defects, and model enhancements. There are many open issues related with these problems that have been investigated by researchers. Therefore, it can be concluded that further investigation is required in the field of MBT for SPL research towards discovering further elements that can contribute towards the enhancement of SPL testing strategy.

Advantages and limitations of individual technique to solve variability issue: Several techniques can be used to represent SPL requirements in terms of test scenarios, test behavior, transition probability, and mutation operators. However, certain techniques have limitations in terms of requirement process and pre-processing developments. These limitations could lead to issues that may affect the quality of generated test cases. Limitations include usage of too many model types in single approach, complex model transformation, and clumsy model notations indicate that much enhancement is required to improve the issues.

Single quality metrics are inadequate to validate the effectiveness of the testing results. There is demand by the industry to consider cost and fault as metrics to assess testing techniques and for the development of efficient test case generation solutions. Researchers have started to measure the quality of MBT test case generation by using cost and fault-based metrics, however, the research is still in the early phase since most of the primary studies chose to use traditional common metrics.

5. Threats to Validity

There are limitations in terms of publication bias, robustness of taxonomy, size of publications reviewed and data extraction inaccuracy. The search process was restricted to finding publications published in English language only. However, some publications in other languages still appeared in the search. Thus, each paper was manually checked to ensure that the study was published in English. Issues including unclear studies descriptions, ambiguous objectives, and vague conclusions had led difficulties to classify and summarize findings. Furthermore, due to the nature of academic research, this SLR does not guarantee that all primary studies concerning MBT for SPL have been included. There is an obvious challenge to capture all suitable search strings that reflect the entire review topic. Despite this, the SLR has tried to capture most of the publications related with the topic. Another potential threat that affects this SLR is the robustness of the taxonomy concerning its ability to review all primary studies adequately.

6. Conclusion

The purpose of this study is to define SLR based on classification trends and issues of MBT in SPL. The outcome of this SLR will assist researchers to generate new ideas concerning the research of MBT for SPL domain. In addition, the taxonomy also can give new researchers to understand the roadmap of MBT in SPL. The following is a list of important points drawn out from this study where possible improvements and enhancements can be made :-

1. Issues on scalability, reusability and test model to represent variability as test model artefacts need to be investigated further. The quality of test cases from proposed tools, model notation, and approaches required multiple measurements.
2. There are limited number of studies focusing on model-based test execution in application engineering process compared to requirement gathering and test generation processes.
3. In recent years, there is an emerging trend of implementing search-based algorithm as a solving technique in MBT for SPL involving the process of test case generation and prioritization.
4. The balance trade-off demands for cost and effectiveness measures in MBT for SPL can be achieved by implementing multi-objective optimization.

Acknowledgement

This research was supported by Universiti Tun Hussein Onn Malaysia (UTHM) through Tier 1 Grant (Vot H937).

References

[1] I. D. C. Machado, J. D. McGregor, Y. C. Cavalcanti, and E. S. De Almeida, "On strategies for testing software product lines: A systematic literature review," *Inf. Softw. Technol.*, vol. 56, no. 10, pp. 1183–1199, 2014. Article (CrossRef Link)

[2] M. Fathi and S. Khoshnevis, "Reusability Metrics in Search-Based Testing of Software Product Lines: An Experimentation," in *Proc. of 26th Int. Comput. Conf. Comput. Soc. Iran, CSICC 2021*, 2021. Article (CrossRef Link)

[3] M. A. Isa, S. A. Razak, D. Norhayati, A. Jawawi, and O. L. Fuh, "Model-Based Testing for Software Product Line : A Systematic Literature Review," *Int. J. Softw. Eng. Technol.*, vol. 02, no. 2, pp. 27–34, 2017. Article (CrossRef Link)

[4] A. Saeed, S. H. Ab Hamid, and M. B. Mustafa, "The experimental applications of search-based techniques for model-based testing: Taxonomy and systematic literature review," *Appl. Soft Comput. J.*, vol. 49, pp. 1094–1117, 2016. Article (CrossRef Link)

[5] J. Lee, S. Kang, and P. Jung, "Test coverage criteria for software product line testing: Systematic literature review," *Inf. Softw. Technol.*, vol. 122, no. January, p. 106272, 2020. Article (CrossRef Link)

[6] M. Marques, J. Simmonds, P. O. Rossel, and M. C. Bastarrica, "Software product line evolution: A systematic literature review," *Inf. Softw. Technol.*, vol. 105, no. January, pp. 190–208, 2019. Article (CrossRef Link)

[7] M. Pol'la, A. Buccella, and A. Cechich, "Analysis of variability models: a systematic literature review," *Softw. Syst. Model.*, vol. 20, no. 4, pp. 1043–1077, 2021. Article (CrossRef Link)

[8] M. Azanza and L. Montalvillo, "20 years of Industrial Experience at SPLC : a Systematic Mapping Study," in *Proc. of Int. Syst. Softw. Prod. Line Conf.*, vol. A, pp. 172–183, 2021. Article (CrossRef Link)

[9] P. Ardimento, N. Boffoli, and G. Superbo, “Multi software product lines: A systematic mapping study,” in *Proc. of ENASE 2020 - Proc. 15th Int. Conf. Eval. Nov. Approaches to Softw. Eng.*, no. Enase, pp. 470–476, 2020. Article (CrossRef Link)
[10] K. L. Petry, E. OliveiraJr, and A. F. Zorzo, “Model-based testing of software product lines: Mapping study and research roadmap,” *J. Syst. Softw.*, vol. 167, 2020. Article (CrossRef Link)
[11] S. El-Sharkawy, N. Yamagishi-Eichler, and K. Schmid, “Metrics for analyzing variability and its implementation in software product lines: A systematic literature review,” *Information and Software Technology*, vol. 106, pp. 1-30, 2019. Article (CrossRef Link)
[12] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, “Systematic literature reviews in software engineering - A systematic literature review,” *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, 2009. Article (CrossRef Link)
[13] N. Anquetil et al., “Traceability for Model Driven, Software Product Line Engineering,” in *Proc. of ECMDA Traceability Workshop Proceedings*, vol. 12, pp. 77–86, 2008. Article (CrossRef Link)
[14] S. Pradhan, M. Ray, and S. K. Swain, “Transition coverage based test case generation from state chart diagram,” *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 3, pp. 993-1002, 2022. Article (CrossRef Link)
[15] H. Samih, H. Le Guen, R. Bogusch, M. Acher, and B. Baudry, “Deriving usage model variants for model-based testing: An industrial case study,” in *Proc. of IEEE Int. Conf. Eng. Complex Comput. Syst. ICECCS*, pp. 77–80, 2014. Article (CrossRef Link)
[16] G. Perrouin, S. Oster, S. Sen, J. Klein, B. Baudry, and Y. le Traon, “Pairwise testing for software product lines: Comparison of two approaches,” *Softw. Qual. J.*, vol. 20, no. 3–4, pp. 605–643, 2012. Article (CrossRef Link)
[17] S. Wang, S. Ali, T. Yue, and M. Liaaen, “Using feature model to support model-based testing of product lines: An industrial case study,” in *Proc. of 2013 13th International Conference on Quality Software*, pp. 75–84, 2013. Article (CrossRef Link)
[18] H. Samih et al., “An Approach to Derive Usage Models Variants for Model-based Testing,” in *Proc. of The 26th IFIP International Conference on Testing Software and Systems*, pp. 80–96, 2016. Article (CrossRef Link)
[19] X. Devroey, “Behavioural Model Based Testing of Software Product Lines,” in *Proc. of Softw. Prod. Line Conf.*, no. August, pp. 1–8, 2014. Article (CrossRef Link)
[20] M. Lochau, S. Lity, R. Lachmann, I. Schaefer, and U. Goltz, “Delta-oriented model-based integration testing of large-scale systems,” *J. Syst. Softw.*, vol. 91, no. 1, pp. 63–84, 2014. Article (CrossRef Link)
[21] H. Cichos, S. Oster, M. Lochau, and A. Schuerr, “Model-Based Coverage-Driven Test Suite Generation for Software Product Lines,” *Model Driven Eng. Lang. Syst.*, vol. 6981, pp. 425–439, 2011. Article (CrossRef Link)
[22] S. Weißleder and H. Lackner, “Top-Down and Bottom-Up Approach for Model-Based Testing of Product Lines,” *Electron. Proc. Theor. Comput. Sci.*, vol. 111, no. Mbt, pp. 82–94, 2013. Article (CrossRef Link)
[23] C. S. Gebizli and H. Sozer, “Model-Based Software Product Line Testing by Coupling Feature Models with Hierarchical Markov Chain Usage Models,” in *Proc. of 2016 IEEE Int. Conf. Softw. Qual. Reliab. Secur. QRS-C 2016*, pp. 278–283, 2016. Article (CrossRef Link)
[24] F. Sadia, M. Hasan, N. Nahar, and M. Rokonuzzaman, “A New Process Model of Incremental Asset Building for Software Project Management,” in *Proc. of 19th Int. Conf. Softw. Eng. Res. Manag. Appl.*, pp. 86–90, 2021. Article (CrossRef Link)
[25] M. Varshosaz and G. Schneider, "Test Models and Algorithms for Model-Based Testing of Software Product Lines," no. 30. 2017. Article (CrossRef Link)
[26] R. Sulaiman, D. N. A. Jawawi, and S. A. Halim, “Features and Behaviours Mapping in Model-based Testing in Software Product Line,” in *Proc. of IOP Conf. Ser. Mater. Sci. Eng.*, vol. 884, no. 1, 2020. Article (CrossRef Link)
[27] A. Anjorin, S. Oster, I. Zorcic, and A. Sch, “Optimizing Model-Based Software Product Line Testing with Graph Transformations,” in *Proc. of Electron. Commun. EASST 11th Int. Work. Graph Transform. Vis. Model. Tech. (GTVMT 2012)*, vol. 47, 2012. Article (CrossRef Link)

[28] H. Baller, S. Lity, M. Lochau, and I. Schaefer, "Multi-objective test suite optimization for incremental product family testing," in *Proc. of IEEE 7th Int. Conf. Softw. Testing, Verif. Validation, ICST 2014*, pp. 303–312, 2014. Article (CrossRef Link)
[29] M. Lochau, S. Oster, U. Goltz, and A. Schürr, "Model-based pairwise testing for feature interaction coverage in software product line engineering," *Softw. Qual. J.*, vol. 20, no. 3–4, pp. 567–604, 2012. Article (CrossRef Link)
[30] A. Reuys, E. Kamsties, K. Pohl, and S. Reis, "Model-Based System Testing of Software Product Families," in *Proc. of Int. Conf. Adv. Inf. Syst. Eng.*, pp. 519–534, 2005. Article (CrossRef Link)
[31] E. Kamsties, K. Pohl, S. Reis, and A. Reuys, "Testing Variabilities in Use Case Models," in *Proc. of Int. Work. Softw. Prod. Eng.*, pp. 6–18, 2010. Article (CrossRef Link)
[32] M. Lochau, I. Schaefer, J. Kamischke, and S. Lity, "Incremental model-based testing of delta-oriented software product lines," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7305 LNCS, pp. 67–82, 2012. Article (CrossRef Link)
[33] S. Wang, S. Ali, and A. Gotlieb, "Cost-effective test suite minimization in product lines using search techniques," *J. Syst. Softw.*, vol. 103, pp. 370–391, 2015. Article (CrossRef Link)
[34] H. Lackner, "Model-based product line testing: Sampling configurations for optimal fault detection," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 9369, pp. 238–251, 2015. Article (CrossRef Link)
[35] G. Kaur and A. V. Rajoriya, "Approaches to Face Challenges in Product Line Model Based Testing," *Int. J. Comput. Syst.*, vol. 03, no. 03, pp. 207–210, 2016.
[36] M. Farrag, "Colored Model Based Testing for Software Product Lines (CMBT-SWPL)," *Technical University of Ilmenau*, 2010. Article (CrossRef Link)
[37] Devroey, X., Perrouin, G., Legay, A., Schobbens, P. Y., and Heymans, P., "Search-based Similarity-driven Behavioural SPL Testing," in *Proc. of the Tenth International Workshop on Variability Modelling of Software-intensive Systems*, pp. 89–96, 2016. Article (CrossRef Link)
[38] J. Simmonds and M. C. Bastarrica, "Modeling variability in software process lines," 2011. Article (CrossRef Link)
[39] S. Oster, "Feature Model-based Software Product Line Testing," Ph.D. thesis, Technische Universität, 2012. Article (CrossRef Link)
[40] F. Belli, C. J. Budnik, A. Hollmann, T. Tuglular, and W. E. Wong, "Model-based mutation testing—Approach and case studies," *Sci. Comput. Program.*, vol. 120, pp. 25–48, 2016. Article (CrossRef Link)
[41] J. Font, L. Arcega, Ø. Haugen, and C. Cetina, "Leveraging variability modeling to address metamodel revisions in Model-based Software Product Lines," *Comput. Lang. Syst. Struct.*, vol. 48, pp. 20–38, 2017. Article (CrossRef Link)
[42] I. Schaefer, M. Al-Hajjaji, R. Lachmann, F. Fürchtegott, and S. Lity, "Fine-grained test case prioritization for integration testing of delta-oriented software product lines," in *Proc. of the 7th International Workshop on Feature-Oriented Software Development*, vol. 1, no. 212, pp. 1–10, 2016. Article (CrossRef Link)
[43] S. J. Prowell, "Using Markov Chain Usage Models to Test Complex Systems," in *Proc. of 38th Annu. Hawaii Int. Conf. Syst. Sci.*, vol. 00, no. C, pp. 318c-318c, 2005. Article (CrossRef Link)
[44] C. Kästner, A. Von Rhein, S. Erdweg, S. Apel, and K. Ostermann, "Toward Variability-Aware Testing," in *Proc. 4th Int. Work. Featur. Softw. Dev.*, ACM, pp. 1-8, 2012. Article (CrossRef Link)
[45] I.Machado, "Fault Model-Based Variability Testing," Ph.D. Thesis, Universidade Salvador, 2014. Article (CrossRef Link)
[46] H. Lackner, M. Thomas, F. Wartenberg, and S. Weißleder, "Model-based test design of product lines: Raising test design to the product line level," in *Proc. of IEEE 7th Int. Conf. Softw. Testing, Verif. Validation, ICST 2014*, pp. 51–60, 2014. Article (CrossRef Link)
[47] B. García, R. García-carmona, Á. Navas, H. A. Parada-gélvez, F. Cuadrado, and J. C. Dueñas, "An automated Model-based Testing Approach in Software Product Lines Using a Variability Language," 2010. Article (CrossRef Link)

[48] C. D. N. Damasceno, M. R. Mousavi, and A. da S. Simao, “Learning by sampling: learning behavioral family models from software product lines,” *Empir. Softw. Eng.*, vol. 26, no. 1, 2021. Article (CrossRef Link)
[49] E. D. . Rodrigues, “PLETS - A PRODUCT LINE OF MODEL-BASED TESTING TOOLS,” 2013. Article (CrossRef Link)
[50] S. Wang, A. Gotlieb, S. Ali, and M. Liaaen, “Automated Test Case Selection Using Feature Model: An Industrial Case Study,” *Model-Driven Engineering Languages and Systems*, pp. 237–253, 2013. Article (CrossRef Link)
[51] X. Devroey, G. Perrouin, M. Cordy, H. Samih, A. Legay, and P. S. Patrick, “Statistical prioritization for software product line testing : an experience report,” *Softw. Syst. Model.*, vol. 16, no. 1, pp. 153–171, 2017. Article (CrossRef Link)
[52] F. Ensan, E. Bagheri, and D. Gasevic, “Evolutionary Search-based Test Generation for Software Product Line Feature Models,” in *Proc. of International Conference on Advanced Information Systems Engineering*, pp. 613–628, 2012. Article (CrossRef Link)
[53] S. Lity, S. Nahrendorf, T. Thüm, C. Seidl, and I. Schaefer, “175 % Modeling for Product-Line Evolution of Domain Artifacts,” *ACM Comput. Mach.*, no. 1, pp. 27–34, 2018. Article (CrossRef Link)
[54] U. Kelter, C. Pietsch, T. Kehrer, D. Reuling, and M. Ohrndorf, “SiPL -- A Delta-Based Modeling Framework for Software Product Line Engineering,” in *Proc. of 30th IEEE/ACM Int. Conf. Autom. Softw. Eng.*, pp. 852–857, 2015. Article (CrossRef Link)
[55] G. Perrouin et al., “Automatic and Scalable T-wise Test Case Generation Strategies for Software Product Lines,” in *Proc. of Third international conference on software testing, verification and validation IEEE*, pp. 459–468, 2010. Article (CrossRef Link)
[56] K. L. Petry, E. OliveiraJr, and A. F. Zorzo, “Model-based testing of software product lines: Mapping study and research roadmap,” *J. Syst. Softw.*, vol. 167, p. 110608, 2020. Article (CrossRef Link)
[57] P. de A. dos Santos Neto, R. Britto, R. de A. L. Rabêlo, J. J. de A. Cruz, and W. A. L. Lira, “A hybrid approach to suggest software product line portfolios,” *Appl. Soft Comput. J.*, vol. 49, pp. 1243–1255, 2016. Article (CrossRef Link)
[58] H. Yadav, A. Charan Kumari, and R. Chhikara, “Feature selection optimisation of software product line using metaheuristic techniques,” *Int. J. Embed. Syst.*, vol. 13, no. 1, pp. 50–64, 2020. Article (CrossRef Link)
[59] A. O. Elfaki, S. Phon-Amnuaisuk, and C. K. Ho, “Modeling variability in software product line using first order logic,” in *Proc. of 7th ACIS International Conference on Software Engineering Research, Management and Applications, SERA09*, pp. 227–233, 2009. Article (CrossRef Link)
[60] S. Wang and S. Ali, “Modeling bCMS product line using feature model, component family model and UML,” in *Proc. of CEUR Workshop Proc.*, vol. 1076, 2013. Article (CrossRef Link)
[61] H. Lackner and M. Schmidt, “Towards the assessment of software product line tests: a mutation system for variable systems,” in *Proc. of 18th Int. Softw. Prod. Line Conf. Companion Vol. Work. Demonstr. Tools - SPLC ’14*, vol. 2, pp. 62–69, 2014. Article (CrossRef Link)
[62] R. A. Sulaiman, D. N. A. Jawawi, and S. A. Halim, “Derivation of test cases for model-based testing of software product line with hybrid heuristic approach,” *Adv. Intell. Syst. Comput.*, vol. 1073, pp. 199–208, 2020. Article (CrossRef Link)
[63] D. Sundmark, P. Wallin, and C. Amlinger, “Model-Based Product Line Engineering in an Industrial Automotive Context : An Exploratory Case Study,” in *Proc. of the 22nd International Conference on Systems and Software Product Line*, vol. v2, pp. 56–63, 2018. Article (CrossRef Link)
[64] A. Arrieta, G. Sagardui, and L. Etxeberria, “A model-based testing methodology for the systematic validation of highly configurable cyber-physical systems,” in *Proc. of 6th International Conference on Advances in System Testing and Validation Lifecycle, VALID 2014*, pp. 66–72, 2014. Article (CrossRef Link)

[65] C. Henard, “Enabling Testing of Large Scale Highly Configurable Systems with Search-based Software Engineering : The Case of Model-based Software Product Lines," *Dissertation Defense Committee, Universite Du Luxemborg*, 2015. Article (CrossRef Link)
[66] S. Wang, S. Ali, and A. Gotlieb, “Automated product line methodologies to support model-based testing,” in *Proc. of CEUR Workshop Proc.*, vol. 1115, pp. 56–60, 2013. Article (CrossRef Link)
[67] F. Damiani, D. Faitelson, C. Gladisch, and S. Tyszberowicz, “A novel model-based testing approach for software product lines,” *Softw. Syst. Model.*, vol. 16, no. 4, pp. 1223–1251, 2017. Article (CrossRef Link)
[68] E. Cartaxo and P. D. L. Machado, “Test case generation by means of UML sequence diagrams and Label Transition System,” in *Proc. of 2007 IEEE International Conference on Systems, Man and Cybernetics*, pp. 1292–1297, 2007. Article (CrossRef Link)
[69] W. K. G. Assunção, S. R. Vergilio, and R. E. Lopez-Herrejon, “Automatic extraction of product line architecture and feature models from UML class diagram variants,” *Inf. Softw. Technol.*, vol. 117, no. March 2019, 2020. Article (CrossRef Link)
[70] F. Damiani, M. Lienhardt, and L. Paolini, “A formal model for Multi Software Product Lines,” *Sci. Comput. Program.*, vol. 172, no. 644298, pp. 203–231, 2019. Article (CrossRef Link)
[71] D. ÖZTÜRK, “A MODEL-BASED TEST GENERATION APPROACH FOR AGILE SOFTWARE PRODUCT,” *Doctoral dissertation, Izmir Institute of Technology (Turkey)*, 2020. Article (CrossRef Link)
[72] R. M. Hierons, M. Li, X. Liu, J. A. Parejo, S. Segura, and X. Yao, “Many-objective test suite generation for software product lines,” *ACM Trans. Softw. Eng. Methodol.*, vol. 29, no. 1, pp. 1–46, 2020. Article (CrossRef Link)
[73] M. Bernardino, E. M. Rodrigues, A. F. Zorzo, and L. Marchezan, “Systematic mapping study on MBT: tools and models,” *IET Softw.*, vol. 11, no. 4, pp. 141–155, 2017. Article (CrossRef Link)
[74] M. Utting, A. Pretschner, and B. Legeard, “A taxonomy of model-based testing approaches,” *Softw. Test. Verif. Reliab.*, vol. 22, no. 5, pp. 297–312, 2012. Article (CrossRef Link)
[75] S. Weißleder, D. Sokenou, and B.-H. Schlingloff, “Reusing State Machines for Automatic Test Generation in Product Lines,” in *Proc. of 1st Work. Model. Test. Pract. (MoTiP ’08)*, vol. 6, p. 10, 2008. Article (CrossRef Link)
[76] P. Arcaini, O. Inverso, and C. Trubiani, “Automated model-based performance analysis of software product lines under uncertainty,” in *Proc. of the 25th ACM International Systems and Software Product Line Conference - Volume A*, p. 112, 2021. Article (CrossRef Link)
[77] T. Buchmann, A. Dotor, and B. Westfechtel, “MOD2-SCM: A model-driven product line for software configuration management systems,” *Inf. Softw. Technol.*, vol. 55, no. 3, pp. 630–650, 2013. Article (CrossRef Link)
[78] S. Wang, A. Gotlieb, S. Ali, and M. Liaaen, “Automated Selection of Test Cases using Feature Model for Product Lines : An Industrial Case Study,” *Model-Driven Engineering Languages and Systems*, pp. 237–253, 2013. Article (CrossRef Link)
[79] M. A. Jamil, M. K. Nour, A. Alhindi, N. S. Awang Abhubakar, M. Arif, and T. F. Aljabri, “Towards Software Product Lines Optimization Using Evolutionary Algorithms,” *Procedia Comput. Sci.*, vol. 163, pp. 527–537, 2019. Article (CrossRef Link)
[80] S. Wang, “Systematic Product Line Testing : Methodologies, Automation, and Industrial Application,” PhD Thesis, University of Oslo (UiO), 2015. Article (CrossRef Link)
[81] K. Petry, E. OliveiraJr, L. Costa, A. Zanin, and A. Zorzo, “SMartyTesting: A Model-Based Testing Approach for Deriving Software Product Line Test Sequences,” in *Proc. of Int. Conf. Enterp. Inf. Syst.*, vol. 2, no. Iceis, pp. 165–172, 2021. Article (CrossRef Link)
[82] S. Oster, Feature Model-based Software Product Line Testing, 2012. Article (CrossRef Link)
[83] X. Devroey, G. Perrouin, A. Legay, M. Cordy, P. Schobbens, and P. Heymans, “Coverage Criteria for Behavioural Testing of Software Product Lines,” in *Proc. of 6th Int. Symp. Leveraging Appl. Form. Methods, Verif. Valid. (to Appear.)*, pp. 336–350, 2014. Article (CrossRef Link)
[84] S. Wang and S. Ali, “Minimizing Test Suites in Software Product Lines Using Weight-based Genetic Algorithms,” in *Proc. of 15th Annu. Conf. Genet. Evol. Comput*, ACM, pp. 1493–1500, 2013. Article (CrossRef Link)

[85] R. A. Sulaiman, D. N. A. Jawawi, and S. A. Halim, "A dissimilarity with dice-jaro-winkler test case prioritization approach for model-based testing in software product line," *KSII Trans. Internet Inf. Syst.*, vol. 15, no. 3, pp. 932–951, 2021. Article (CrossRef Link)

[86] C. Henard, M. Papadakis, G. Perrouin, J. Klein, and Y. Le Traon, "Assessing software product line testing via model-based mutation: An application to similarity testing," in *Proc. of IEEE 6th Int. Conf. Softw. Testing, Verif. Valid. Work. ICSTW 2013*, pp. 188–197, 2013. Article (CrossRef Link)

[87] C. Henard, M. Papadakis, G. Perrouin, and J. Klein, "Multi-objective Test Generation for Software Product Lines," in *Proc. of 17th Int. Softw. Prod. Line Conf. ACM*, pp. 62-71, 2013. Article (CrossRef Link)

[88] M. Al-Hajjaji, "Scalable and Efficient Sampling for Product-Line Testing," Tech. Rep. FIN-003-2014, Univ. Magdeburg, Ger. 2014, 2014. Article (CrossRef Link)

[89] R. A. Sulaiman, D. N. A. Jawawi, and S. A. Halim, "Coverage-based approach for model-based testing in Software Product Line," *Int. J. Eng. Technol.*, vol. 7, no. 4, pp. 63–68, 2018. Article (CrossRef Link)

[90] X. Lian, L. Zhang, J. Jiang, and W. Goss, "An Approach for Optimized Feature Selection in Large-scale Software Product Lines," *J. Syst. Softw.*, vol. 137, pp. 636-651, 2018. Article (CrossRef Link)

[91] A. Knapp and M. Roggenbach, "On the Use of Test Cases in Model-Based Software Product Line Development," in *Proc. of 18th Int. Softw. Prod. Line Conf. 1. ACM*, no. 3, pp. 247–251, 2014. Article (CrossRef Link)

[92] R. A. Matnei Filho and S. R. Vergilio, "A multi-objective test data generation approach for mutation testing of feature models," *J. Softw. Eng. Res. Dev.*, vol. 4, no. 1, p. 4, 2016. Article (CrossRef Link)

[93] S. Wang, S. Ali, A. Gotlieb, and M. Liaaen, "A systematic test case selection methodology for product lines: results and insights from an industrial case study," *Empir. Softw. Eng.*, vol. 21, no. 4, pp. 1586–1622, 2016. Article (CrossRef Link)

[94] D. Reuling, J. Bürdek, S. Rotärmel, M. Lochau, and U. Kelter, "Fault-based product-line testing," in *Proc. of SPLC - Int. Conf. Softw. Prod. lines*, pp. 131–140, 2015. Article (CrossRef Link)

[95] A. Arrieta, S. Wang, G. Sagardui, and L. Etxeberria, "Test Case Prioritization of Configurable Cyber-Physical Systems with Weight-Based Search Algorithms," in *Proc. of 2016 Genet. Evol. Comput. Conf. - GECCO '16*, pp. 1053–1060, 2016. Article (CrossRef Link)

[96] J. S. Her, J. H. Kim, S. H. Oh, S. Y. Rhew, and S. D. Kim, "A framework for evaluating reusability of core asset in product line engineering," *Inf. Softw. Technol.*, vol. 49, no. 7, pp. 740–760, 2007. Article (CrossRef Link)

R.ADUNI SULAIMAN received PhD degree in computer science from Universiti Teknologi Malaysia, Johor, Malaysia. She was a system engineer at Infineon Technologies Malaysia before she joined the Department of Software Engineering at Universiti Tun Hussein Onn. Currently, her research interest on software engineering, software testing and soft computing.

DAYANG N. A. JAWAWI is a Professor in School of Computing, Faculty of Engineering, Universiti Teknologi Malaysia (UTM). A major part of her research projects focuses on rehabilitation and mobile robotics, healthcare application, real-time embedded systems and precision farming applications.

SHAHLIZA ABDUL HALIM received the B.Sc., M.Sc., and Ph.D. degrees from Universiti Teknologi Malaysia, Johor Bahru, Malaysia, in 1997, 1998, and 2016, respectively. She is currently a Senior Lecturer at the same university, where she is currently involved in the Software Product Line and Engineering Education research area.