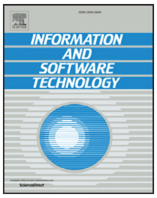
Information and Software Technology 105 (2019) 209–225

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof



Integration of feature models: A systematic mapping study

Vinicius Bischoff, Kleinner Farias*, Lucian José Gonçales, Jorge Luis Victória Barbosa

Graduate Program in Applied Computing (PPGCA), University of Vale do Rio dos Sinos (Unisinos), São Leopoldo, Rio Grande do Sul, Brazil

ARTICLE INFO

Keywords:
Feature models
Model integration
Systematic mapping study

ABSTRACT

Context: The integration of feature models has been widely investigated in the last decades, given its pivotal role for supporting the evolution of software product lines. Unfortunately, academia and industry have overlooked the production of a thematic analysis of the current literature. Hence, a thorough understanding of the state-of-the-art works remains still limited.
Objective: This study seeks to create a panoramic view of the current literature to pinpoint gaps and supply insights of this research field.
Method: A systematic mapping study was performed based on well-established empirical guidelines for answering six research questions. In total, 47 primary studies were selected by applying a filtering process from a sample of 2874 studies.
Results: The main results obtained are: (1) most studies use a generic notation (68.09%, 32/47) for representing feature models; (2) only one study (2%, 1/47) compares feature models based on their syntactic and semantics; (3) there is no preponderant use of a particular integration technique in the selected studies; (4) most studies (70%, 33/47) provide a product-based strategy to evaluate the integrated feature models; (5) majority (70%, 33/47) automates the integration process; and (6) most studies (90%, 42/47) propose techniques, rather than focusing on producing practical knowledge derived from empirical studies.
Conclusion: The results were encouraging and suggest that integration of feature models is still an evolving research area. This study provides insightful information for the definition of a more ambitious research agenda. Lastly, empirical studies exploring the required effort to apply the current integration techniques in real-world settings are highly recommended in future work.

1. Introduction

The adoption of *feature models* has advanced in software-development projects in industry [11,21,29], since such models play a chief role in several development activities, e.g., representing variability in Software Product Lines (SPL) [13], grouping domain concepts in terms of their commonalities and differences within a family of software systems [2], helping the derivation of products from SPL [23], or even guiding developers to extract valid combinations of features [74].

A feature can be briefly defined as a functionality (or behavioral) that a software product should provide [13,29]. A feature model is, in turn, a diagram representing which characteristics are parts of a software product and how such characteristics relate with each other. By using feature models, software architects can illustrate different product configurations that can be derived from an SPL. According to Kang [52], a software product line consists of a software family formed by a set of features. In this sense, a feature model can be applied to represent several configurations of software systems. In practice, the feature models can be created in parallel by different software-development teams, so that team members concentrate on parts more relevant to them. However, at some point, the parts created in parallel need to be integrated to form an overview of the variabilities found in an SPL. For this reason, many integration techniques have been proposed over the last years (e.g., [1,10,19,73]).

The term *integration of feature models* can be briefly defined as a set of activities that should be performed over two (or more) input feature models, FM_A and FM_B, to produce an output-desired feature model, FM_{AB}. Unfortunately, the integration of FM_A and FM_B does not produce an FM_{AB}. Instead, an output-integrated feature model, FM_{IM}, with inconsistencies is generated (i.e., $FM_{IM} \neq FM_{AB}$). Thus, developers need to invest effort to detect and resolve such inconsistencies in FM_{IM}, so that FM_{AB} can be produced.

Regardless of the integration technique used, the models to be integrated (FM_A and FM_B) inevitably end up having conflicting parts. If such conflicts are improperly tamed, the conflicting parts are converted into inconsistencies, i.e., contradictions between properties found in FM_{AB}

[*] Corresponding author.
E-mail addresses: viniciusbischof@edu.unisinos.br (V. Bischoff), kleinnerfarias@unisinos.br (K. Farias), lucianj@edu.unisinos.br (L.J. Gonçales), jbarbosa@unisinos.br (J.L. Victória Barbosa).

https://doi.org/10.1016/j.infsof.2018.08.016
Received 5 October 2016; Received in revised form 31 January 2018; Accepted 31 August 2018
Available online 1 September 2018

and FM_{IM}. The automated resolution of such conflicts is challenging as the meaning of each feature is rarely formally represented. As a result, the integration techniques are often unable to solve such conflicts automatically. As such, the integration of models is still be considered a highly-intensive manual task [67].

In addition, the integration of feature models has widely been studied in practice, given its pivotal role for supporting the evolution of software product lines. For this reason, both academia and industry have proposed several works in recent years, such as [16,19,21]. Unfortunately, such techniques have not demonstrated to be effective enough to support software architects in real-world settings [18,35,55,59], requiring further improvements.

Although many works have been done in the last years (e.g., [13,19,22,68]), the literature has overlooked the production of a panoramic view of the current literature. Hence, a thorough understanding of the state-of-the-art techniques remains still limited and inconclusive. For example, not much is known about the notations used to represent feature models, the comparison techniques used to identify equivalences between feature model elements, and the integration techniques often adopted. In addition, lacking an understanding of where and how often studies, involving feature model integration, have been published over the past few years. Furthermore, it is neither known what types of research (e.g., controlled experiment, case study, survey) have been done nor which topics (e.g., tool support) they have explored most frequently.

This study seeks, therefore, to create a panoramic view about the current literature regarding the integration of feature models. Furthermore, it aims at pinpointing open issues and supplying insights for further improvements of the state-of-the-art works. This thematic analysis is crucial to outline which research issues have been more explored, summarize the notations used, present how the studies have occurred over the years, grasp the most-used research methods, understand the types of comparison and integration techniques, as well as pinpoint to what extent the current works provide tool support. In addition, this study lists important gaps, which might be used by researchers for building a future research agenda with emerging themes.

To put this research in practice, a systematic mapping study (SMS) was designed and performed following well-known empirical guidelines in Kitchenham and Co-workers [56–58,62,63]. We defined six research questions and identified a set of key terms related to the research field of feature integration to retrieve a bunch of potentially-relevant articles. After three review cycles, we selected 47 primary studies by applying a careful filtering process to a sample of 2874 candidate studies retrieved from 6 electronic databases.

In particular, these studies were carefully scrutinized for bringing out (1) the notations used to represent feature models, (2) the comparison strategies used to identify equivalences between feature model elements, (3) the types of integration techniques, (4) the evaluation techniques used to check inconsistencies, (5) the kind of support tools, and finally (6) the research methods most commonly used to evaluate the works published. Investigating these six issues can be seen as a first step for a more ambitious agenda on how to characterize and improve integration techniques of feature models.

The main results obtained are: (1) most studies use a generic notation (68.09%, 32/47) for representing feature models; (2) only one study (2%, 1/47) compares feature models based on their syntactic and semantics; (3) there is no preponderant use of a particular integration technique in the selected studies; (4) most studies (70%, 33/47) provide a product-based strategy to evaluate the integrated feature models; (5) majority (70%, 33/47) automates the integration process; and (6) most studies (90%, 42/47) propose techniques, rather than focusing on producing practical knowledge derived from empirical studies.

This SMS brings a series of benefits for researchers and practitioners. First, it is valuable and useful by providing an overview about the research field of feature model integration. Second, this study provides a starting point for PhD students who need to organize their works. Third, it empowers developers with information regarding improvements that need to be introduced into the current integration tools. Fourth, this study can reduce learning curve and bias to perform a literature review study, since its protocol can be reused. Moreover, this SMS uncovers *when* and *where* the most representative studies have been published, as well as shows how such studies are distributed over the last years (Section 7.1). In addition, it outlines some further challenges, which may server as a basis to build a richer research agenda.

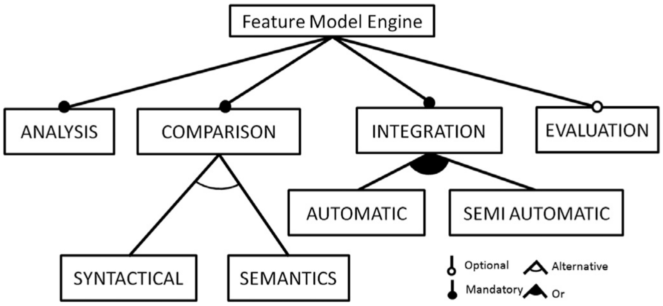


Fig. 1. A simplified view of a feature model.

The remainder of the paper is organized as follows. Section 2 introduces the main concepts and knowledge that are going to be used and discussed throughout the article. Section 3 contrasts this study with related works. Section 4 presents the study methodology. Section 5 carefully describes the study filtering process. Section 6 reports the main results. Section 7 presents some additional discussions. Section 8 discusses how the threats to validity were mitigated. Finally, Section 9 presents some concluding remarks and future work.

2. Background

This Section provides an overview about the key concepts concerning the integration of feature models. Section 2.1 describes the concepts of feature modeling and their main purposes. Section 2.2 defines feature integration.

2.1. Feature modeling

The goal of *feature modeling* is to represent the commonalities and differences among all products of a software product line [2]. The output of this activity is a compact representation of all potential products, so-called feature model. A feature can be seen as a distinctive characteristic of a product [7,15]. Fig. 1 illustrates a feature model using FODA (Feature-Oriented Domain Analysis) [21] notation. The feature model is represented as a tree, where each node is a feature, and the edges represent the relationship between features. The relationship expresses the variability among features, which can be mandatory, optional or alternative.

Alternative relationships are represented by an arc, where a black-filled arc means that any feature combination can be derived (i.e., *Or*). Otherwise, an empty arc means only one among the features can be derived (i.e., *Alternative*). A mandatory relationship is represented by a relationship with a black circle at the end, indicating that it must be implemented. Finally, an optional feature, represented by a relationship with an end empty circle, indicates that it can be absent.

2.2. Generic model integration process

To facilitate the understanding of feature model integration, Fig. 2 illustrates a four-step generic model integration process. We have reported this process in [43]. The steps to integrate the feature models are described as follows:

- **Step 1: analysis**: During the analysis process, the techniques often verify if the feature input models are represented using a same kind

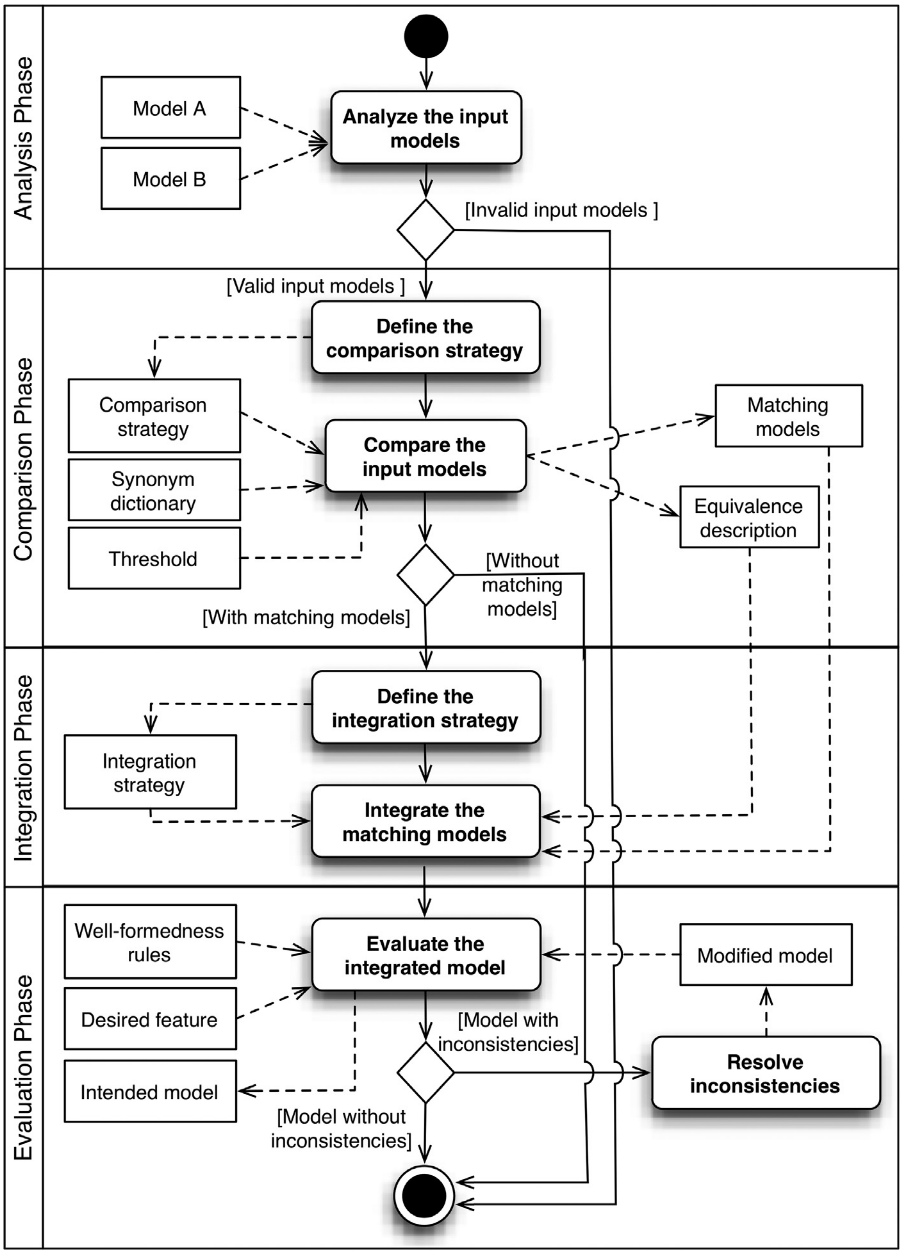


Fig. 2. A generic model integration process (from Farias et al. [43]).

of notations, e.g., if both feature models are represented using the FODA notation.

- **Step 2: comparison**: The integration technique identifies the equivalence between the elements of the input feature models, FM_A (Model A) and FM_B (Model B).
- **Step 3: integration**: The technique will integrate similar elements. However, this step may generate some inconsistencies in the output feature model, resulting in a Composed Feature Model, F_{CM}, which differs from a desired feature model, FM_{AB}.
- **Step 4: evaluation**: The evaluation process checks if well-formedness rules are challenged. If the integration operators used offer guarantees by construction, this step is not needed. Finally, the process ends when the FM_{AB} is produced.

Developers use integration techniques of feature models so that a desired feature model can be produced with as little effort as possible [41]. If an integration technique produces output feature models with inconsistencies, then developers need to detect and resolve each inconsistency produced so that the desired feature model can be obtained [39]. If the number of inconsistencies is high, then the performance of technique in real-world settings is questionable, where time is tight. The current literature (e.g., [24,65,76] defines inconsistency as the violation of built-in constraints in models. An inconsistency in a feature model might be seen, for example, as a violation of constraints defined to format the elaboration of products.

3. Related work

Several works have defined the concept of features in the last decades [11,21,29,52,54]. Berger et al. [21] aimed at dealing with feature models in the field of software product lines. This work highlights the attention that enterprises have invested on features as a manner for supporting the development of their products. The authors also explored the environments of these enterprises to investigate the different aspects of the use of features in real-world settings. They concluded enterprises do not have common practices and guidelines to maintain and manage features throughout their product life cycle. Typical operations of inclusion and exclusion have been overlooked, for example. Therefore, understanding the field of integration of feature models is necessary to pinpoint research gaps and develop a better support to overcome these limitations.

Some works have highlighted the importance to provide solid knowledge about issues concerning the pivotal characteristics on steps of feature model integration (e.g., [16,47,61]). In addition, composition and differentiation techniques of feature models (e.g., [1,7,70]) were proposed in the last years. In [1], Acher et al. proposed syntactic and semantic operators for integrating two feature input models. Furthermore, other authors proposed syntactic and semantic operators but with the purpose of differentiating feature models [7]. In contrast, Segura et al. [70] proposed using graph data structures to automate the integration of feature models.

Moreover, many literature reviews have been produced to provide a solid knowledge base regarding issues of the integration of feature models [19,22,51,68]. These studies endorse the importance of the emerging key roles of feature models in mainstream software-development projects. Integration of feature models is a crucial task, since the parallel manipulation of these artifacts has become more frequently. Thus, industry will demand precise and effective integration techniques. To this end, researchers and practitioners need guidelines and panoramic view of the tasks of feature model integration. Moreover, a precise technique for model integration is needed due to developers demanding more effort when using an inappropriate integration technique [41].

Hubaux et al. indirectly address a recurrent problem related to the scalability of feature models [51]. As feature models grow, they present a limited scalability. When they are used to represent the variability of realistic software product lines, usually having hundreds and even thousands, the models become hard for understanding, imprecise for automated reasoning about large feature models, confuse for creating, updating, and interpreting. Thus, many approaches have been proposed to improve the separation of concerns in feature models as a way to mitigate the scalability problem. Hubaux et al. report that there is no consensus about what the main concerns of feature models are and how these concerns may be managed. Therefore, the authors investigate two issues [51]: (1) what are the key concerns that should be recognized and separated in feature models? and (2) what guidelines are provided by the existing feature diagramming techniques for achieving separation of concerns in feature models? Although it is a review of the literature, it does not explore any research question investigated in our article. As a result, there are no overlaps in relation to the research questions investigated.

To summarize, not much is known about important issues regarding integration of feature models. Although many works have been proposed, little has been done to map systematically the integration techniques of feature models regarding their respective studies, and pinpoint further research directions. Lastly, we identified six research gaps related to lack of knowledge about (1) the notations often used to represent feature models, (2) comparison techniques applied to detect similarities between feature models, (3) how integration techniques can be classified, (4) the current evaluation techniques applied to check the consistency of integrated feature models, (5) the kind of tools used to support the integration of feature models, and finally (6) which methods have been most widely used. To overcome these gaps, we have created six research questions, which are presented in the following Section.

4. Planning

This Section describes the experimental procedures adopted to plan and run the systematic mapping study. For this, we have adopted well-established guidelines described in [56–58,62,63]. Section 4.1 presents the objective and research questions (RQ) investigated. Section 4.2 introduces search strategy and digital libraries used to retrieve representative studies. Section 4.3 outlines inclusion and exclusion criteria used to filter the retrieved studies. Finally, Section 4.4 introduces all procedures adopted to extract data from the selected studies.

4.1. Goal and research questions

The goal of this SMS is twofold: (1) to provide a classification of the current literature on integration of feature models (Section 6); and (2) to identify some promising research directions for further investigations (Section 7). For this, six research questions were defined, so that each facet of this goal could be carefully investigated. Table 1 shows the six research questions investigated, their motivations, as well as the variable investigated in each research question.

4.2. Search strategy

The next step was to define a search strategy to retrieve a representative sample of studies that could be used to answer the specified RQs. This search strategy was performed in two steps: (1) the construction of *search string* (SS), and (2) the definition of *search scope*. To do this, we have followed well-known guidelines [56,57,63].

4.2.1. Construction of the search string

Search strings consist of the concatenation of terms used in search engines to retrieve potentially relevant studies in the current literature. Table 2 presents the terms chosen and their synonyms. The following steps were taken to define such terms: (1) define the main keywords; (2) identify alternative words, synonyms or terms relating to the main keywords; (3) check keywords against ones found in already published research articles; and (4) associate synonyms, alternative words or terms with the logical operators “AND” and “OR”. The main keywords investigated in this mapping study are *integration, feature, model*, and *tool*. Although several other synonyms could be considered, we listed those that were most effective in recovering potential studies. The search string produced is introduced as follows:

(*Merging OR Integration OR Disambiguation OR Decomposition OR Production OR Composition*)
AND (*Feature OR Functionality OR Characteristic*)
AND (*Model OR View OR Viewpoint OR Design OR Diagram*)
AND (*Tool OR Applicable OR Procedure OR Technique*)

4.2.2. Definition of search scope

The search scope refers to the mechanisms used to retrieve studies from the formulated search string. In this study, six electronic databases were used, which are listed in Table 3. We used these electronic databases because they cover the most relevant journals, conferences, and workshops. Applying the search string into these electronic databases, an extensive list of works was produced. So, the next step was to apply inclusion and exclusion criteria to filter the works retrieved. Such criteria are presented in the following Section.

4.3. Inclusion and exclusion criteria

This section aims at establishing inclusion criteria (IC) and exclusion criteria (EC) used to filter the potentially relevant articles retrieved from the search engine used (Table 3). The IC presents what should be considered to include a particular work in our sample of representative articles. On the other hand, the EC comes up with the requirements to support the removal of works deemed inadequate to answer the RQs. The IC sought to select studies that were:

- **IC1**: related to the search string and research question goal;
- **IC2**: written in English;
- **IC3**: published until June 2017;
- **IC4**: available in electronic digital libraries.

Next, the EC, in turn, sought to throw away works that:

Table 1
Description of the investigated research questions.

Research question	Motivation	Variable
RQ1: What are the *notations* used to represent feature models?	*Reveal* the notations often used to represent feature models.	Feature notations.
RQ2: What are *comparison techniques* used to identify equivalences between feature models?	*Uncover* comparison techniques used to identify equivalences between feature model elements.	Comparison techniques.
RQ3: What are the types of *integration techniques* adopted?	*Classify* integration techniques often used to integrate feature models.	Integration techniques.
RQ4: What are the *evaluation techniques* adopted to check consistency of feature models?	*List* evaluation techniques used to check consistency of the integrated feature models.	Evaluation techniques.
RQ5: What *integration tools* have been used?	*Reveal* which tools have been used to support the integration of feature models.	Tool support.
RQ6: What *research methods* have been used?	*Identify* research methods often applied in the current studies.	Research methods.

Table 2
Description of main terms and their synonyms.

Main terms	Synonym
Integration	Merging, Disambiguation, Decomposition, Production, Composition
Feature	Functionality, Characteristic
Model	View, Viewpoint, Design, Diagram
Tool	Technique, Applicable, Procedure

Table 3
List of electronic databases.

Data sources	Electronic address
1 - ACM Digital Library	http://portal.acm.org
2 - IEEE Xplore	http://ieeexplore.ieee.org
3 - Google Scholar	https://scholar.google.com
4 - CiteSeerX	http://citeseerx.ist.psu.edu
5 - Springer Link	http://www.springerlink.com
6 - Science Direct	http://www.sciencedirect.com

- **EC1**: appeared in duplicate;
- **EC2**: were not written in English;
- **EC3**: match the keywords defined in the Search String but the context is different from the research purposes;
- **EC4**: reported only a summary, conference calls, or patents;
- **EC5**: were not within the context of Software Engineering domain;
- **EC6**: do not meet the motivation of the research questions described in Section 4.1.

4.4. Data extraction procedures

After establishing the inclusion and exclusion criteria, the next step was to define how to extract data from the selected works. These procedures should guide the authors in such a way that the articles' data are properly extracted, thereby avoiding misinterpretations. Note that the extracted data are the basis to answer the formulated RQs. For this, we have defined a classification scheme and a data extraction form (Fig. 3). Both were used to gather data so that the RQs might be properly answered.

4.4.1. Classification scheme

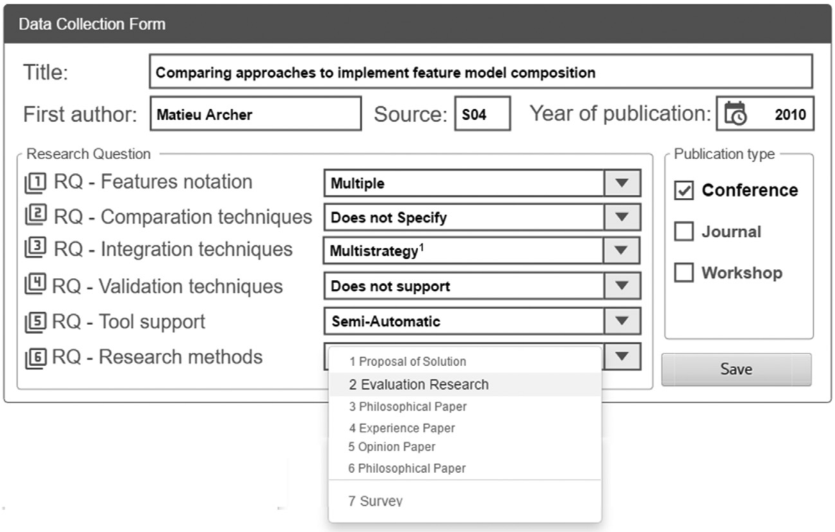
The proposed classification scheme, shown in Table 4, relates the RQs, the investigated variables and the possible answers to the RQs. The answers are the possible values assumed by the variables. Before beginning the data extraction, the authors collaboratively elaborated this scheme. The authors independently read a sample of previous studies (e.g., [2,12,72]), related works (Section 3) and representative studies (discussed in Section 5) to identify possible answers for each research question. After this initial iteration, the collected answers were analyzed and categorized, generating the classification scheme. This scheme was obtained (Table 4) based on the consensual understanding of the authors. To reach this consensus, two meeting cycles were held. In addition, we emphasize that the structure of this classification scheme is also based on the authors' experience with studies related to the integration of design models (e.g., [38–42]). In order to mitigate threats to validity related to the elaborated scheme, we looked for previously validated schemes to take them as a basis, such as one proposed in [44]. We briefly describe each answer presented in Table 4 as follows.

Feature notations (RQ1). The feature models are based on a set of diagrams, forming a connected tree through relationships among their nodes, i.e., the own features. These models can be represented using different notations, which are rarely categorized. The following the categories of notations used to classify the collected studies are shown:

- Algebra: Feature Oriented Software Development (FOSD) uses modern mathematics as a modeling language to express the design and synthesis of programs in software product lines (e.g., [14,16]). It treats features as algebraic operations.
- Cardinality-based: Cardinality-based Feature Modeling (CBFM) is used to represent commonality and variability between the products of a domain in terms of features (e.g., [33]).
- Feature-Algebra: The use algebraic integration constraints linking features (e.g., [50]).
- Generic: A generic feature model provides a basis for developing parameterizing and configuring reusable assets (e.g., [4,8]).
- Multiple: Approaches that use multiple notations, including Feature-Oriented Reuse Method (FORM), Reuse-driven Software Engineering Business Features (RSEB), Product Line Use Case Modeling for Systems and Software Engineering (PLUSS), Common Variability Language (CVL), Text-based Variability Language (TVL), the Model-Driven Product Lines Engineering (AMPLE), Generative Programming (GP), and Feature Oriented Product Line Software Engineering (FOPLE).

Comparison techniques (RQ2). There are many comparison operations to evaluate feature models in the current literature. We have identified the following techniques:

- Heuristic: It performs the mappings between feature models and problem frames using mapping heuristics rather than a formal approach (.e.g., [34]).
- Manual: Demonstration of formalization rules between feature models, as well as the steps to perform operations between models manually (e.g.,[1]).
- Multiple: It implements configuration, formalization and specification aspects when comparing models (e.g., [51]).
- Name-based: Supporting different matching/merging strategies and semantic properties (being related to configuration or ontological aspects) (e.g., [6]).
- Semantic: It takes into account the meaning of the model elements (e.g., [28]).
- Semantic and Syntactic: It considers the meaning of the model elements and takes into account the syntax of the feature models (e.g., [7]).



Data Collection Form

Title: Comparing approaches to implement feature model composition

First author: Matieu Archer Source: S04 Year of publication: 2010

Research Question

1 RQ - Features notation	Multiple
2 RQ - Comparation techniques	Does not Specify
3 RQ - Integration techniques	Multistrategy[1]
4 RQ - Validation techniques	Does not support
5 RQ - Tool support	Semi-Automatic
6 RQ - Research methods	

Publication type

☑ Conference
□ Journal
□ Workshop

Fig. 3. Data extraction form.

Table 4
The classification scheme used to extract data from the studies.

Question	Variable	Answers
RQ1	Feature model notations	1. Algebra, 2. Cardinality-based, 3. Feature-Algebra, 4. Generic, 5. Multiple, 6. Does not Specify
RQ2	Comparison techniques	1. Heuristic, 2. Manual, 3. Multiple, 4. Name-based, 5. Semantic, 6. Semantic and Syntactic, 7. Does not Specify
RQ3	Integration techniques	1. Algebra, 2. Merge, 3. Multiple, 4. Multistrategy, 5. Semantic, 6. Slice Operator, 7. Synthesis, 8. Does not specify
RQ4	Evaluation techniques	1. Family-based, 2. Product-Based, 3. Multiple, 4. Does not specify
RQ5	Tool Support	1. Automatic, 2. Semi-automatic, 3. Multiple, 4. Does not specify
RQ6	Research Methods	1. Proposal of Solution, 2. Evaluation Research, 3. Survey Paper

Integration techniques (RQ3). The integration of two features models combines the model elements so that an integrated model can be produced. We have identified the following operations used for this purpose:

- Algebra: It envolves works related to algebraic integration constraints linking features, approaching a wide class of integration constraint formulations (e.g., [16,50]).
- Merge: Detect interactions automatically using specifications of combined components and automated theorems (e.g., [48,69]).
- Multiple: The papers present multiple approaches on the integration of architectures for feature models (e.g., [6,25]).
- Multistrategy: Modeling techniques that support the separation and composition of resource models using a set of operators (merge, union, intersection and insertion) (e.g., [1,3]).
- Semantic: The works span modularity and display compatibility and display reconciliation techniques to connect different views of a product line (e.g. [28,35]).
- Slice operator: The works describe a set of complementary operators (aggregate, merged, slice) in the data carrier for separating feature models. So with, a set of techniques for specifying, viewing, and verifying the coverage of a set of views (e.g., [5,51]).
- Synthesis: They present an algorithmic and parametrizable approach for computing a and appropriate hierarchy of features, including feature groups, typed feature attributes, domain values and relations among these attributes (e.g., [17]).

Evaluation techniques (RQ4). Despite the difficulty of analyzing product lines, some kinds of evaluation (or analysis) techniques have been proposed, such as product-based and family-based ones:

- Family-based: Variability-aware analysis techniques that can be applied to analyze the variable part of a product line [60,71]. Considering the variable part of a product line, these techniques can produce sound results.
- Product-based: Product-aware analysis techniques that can be applied to analyze each product of a product line individually [60,71].
- Multiple: Works that consider more than one strategy of analysis.

Tool support (RQ5). To know the kinds of tool support provided to users, we have investigated the current works from three perspectives:

- Automatic: It does not require any human interaction.
- Semi-automatic: It does not require users to specify configuration parameters before differentiating the input model elements.
- Multiple: Works that can make use of more than one strategy to run the integration of feature models.

Research methods (RQ6). This issue provides an overview of the direction of the current studies, i.e., the type of studies produced. We have used the categories proposed in [57] to classify the selected works. These guidelines enabled us to classify the works properly. Thus, we classified the primary studies as following:

- Proposal of solution: Works that proposes a new solution.
- Evaluation research: Works that performs empirical studies.
- Survey paper: assumes a general knowledge of the area (e.g., [51]).

4.4.2. Data extraction form

A data extraction form was elaborated to streamline the data collection process. This form is based on the classification scheme shown in Table 4. To create this form, the authors met again to collaboratively define the structure and content of the form. The result of this third meeting was the form presented in Fig. 3. This form was essential to guide and standardize the data extraction. Moreover, we have used spreadsheets to store the collected data to produce statistics for further analysis. These statistics help us to understand, characterize and summarize the state-of-the-art works about integration of feature models.

The data extraction process itself was performed in three review cycles, with all authors to avoid false-positives or false-negatives, and to

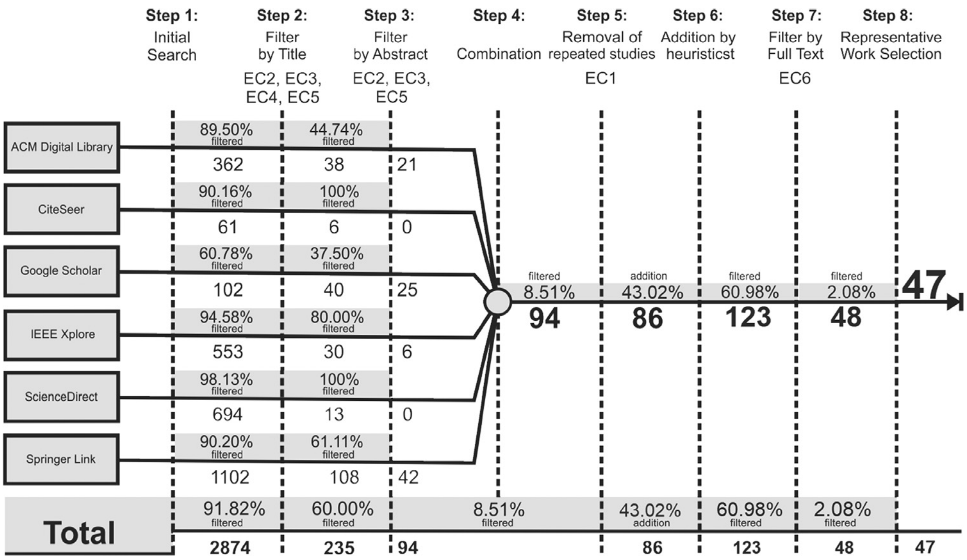

Step 1: Initial Search
Step 2: Filter by Title EC2, EC3, EC4, EC5
Step 3: Filter by Abstract EC2, EC3, EC5
Step 4: Combination
Step 5: Removal of repeated studies EC1
Step 6: Addition by heuristicst
Step 7: Filter by Full Text EC6
Step 8: Representative Work Selection
ACM Digital Library 89.50% filtered 362 44.74% filtered 38 21
CiteSeer 90.16% filtered 61 100% filtered 6 0
Google Scholar 60.78% filtered 102 37.50% filtered 40 25
IEEE Xplore 94.58% filtered 553 80.00% filtered 30 6
ScienceDirect 98.13% filtered 694 100% filtered 13 0
Springer Link 90.20% filtered 1102 61.11% filtered 108 42
filtered 8.51% 94
addition 43.02% 86
filtered 60.98% 123
filtered 2.08% 48
47
Total 91.82% filtered 2874 60.00% filtered 235 94 8.51% filtered 43.02% addition 86 60.98% filtered 123 2.08% filtered 48 47

Fig. 4. The process of primary study selection.

cover important open issues. In the first cycle, at least two authors reviewed each study. For this, they used the classification scheme and the data extraction form. In the second review cycle, the forms with data extracted from the articles were consolidated into a spreadsheet. For this, the authors met in person to discuss and resolve conflicts between the collected data. Finally, the authors performed a third parallel review cycle to find any possible inconsistencies in the results. In the following Section we present how the retrieved studies from the search engine were filtered.

5. Study filtering

This section describes the process followed to filter the potentially relevant studies retrieved after applying the search string to six electronic databases (Table 3). This study filtering process is formed for eight steps. In each step, we have applied the inclusion and exclusion criteria previously presented in Section 4.3. The search and filtering considered articles published between January/2000 and June/2017. Fig. 4 illustrates the results obtained in each filtering process step. The results obtained in the eight steps defined are outlined as follows:

- *Step 1: initial search.* A broad list of studies was retrieved after applying the search string to the six electronic databases (see Table 3). In total, 2874 articles were found in the initial search.
- *Step 2: filter by title.* This step aimed at analyzing the articles considering their title. We checked if they were related to variables listed in Table 1. For this, the exclusion criteria EC2, EC3, C4 and EC5 were applied. The 2874 candidate studies were filtered, removing 2639 articles (91.8%). At the end of this step, 235 articles remained.
- *Step 3: filter by abstract.* This filter was pivotal to determine the study relevance to our research. Exclusion criteria EC2, EC3, and EC5 were applied. 141 articles (60%) were filtered. As a result, only 94 articles went through the filtering process.
- *Step 4: combination.* All the filtered studies from the previous phase were brought together. These 94 articles represented approximately 3.27% of the 2874 articles initially retrieved.
- *Step 5: removal of repeated studies.* It consists of applying the first exclusion criteria (EC1), i.e., repeated studies were discarded. After checking duplicated articles, 8 articles (8.51%) were removed from 94 studies.
- *Step 6: addition by heuristics.* We inserted 37 relevant studies from other sources, totaling 123 articles (43.02% more). These articles were added heuristically, i.e., retrieved from sources based on our knowledge in a manual way.

Table 5
The classification of the feature model notations (RQ1).

Notation	Number of articles	Percentage	Article ID
Algebra	2	4.26%	[S14], [S15]
Cardinality-based	1	2.13%	[S20]
Feature algebra	2	4.26%	[S29], [S11]
Generic	32	68.09%	[S01], [S05], [S06], [S07], [S08], [S09], [S10], [S13], [S16], [S18], [S19], [S21], [S22], [S27], [S28], [S30], [S31], [S32], [S34], [S35], [S36], [S37], [S38], [S39], [S40], [S41], [S42], [S43], [S44], [S45], [S46], [S47]
Multiple	9	19.15%	[S02], [S03], [S04], [S12], [S17], [S23], [S24], [S25], [S26]
Does not specify	1	2.13%	[S33]
Total	47	100%	

- *Step 7: filter by full text.* After reading the full text to produce and check the content based on the data extraction procedures, the six research questions might be answered. Here the EC6 was applied. This step was thrown away 75 articles after a careful analysis because they did not address issues related to our research questions. This removal represents a reduction of 60.98%.
- *Step 8: representative work selection.* Finally, a complete list of representative studies was produced. A total of 47 studies were selected, hereafter called *primary studies* (Appendix A).

6. Results

This Section presents the results obtained. Tables are used to provide an overview of the data gathered applying the methodology presented in Section 4. These Tables allow us to analyze the impact of aspects on the current literature. All of them related to the RQs formulated. Each Table focuses on presenting data regarding a particular RQ.

6.1. RQ1: feature notations

This question investigates the notations for representing the feature models. Table 5 displays the data obtained. The data indicate that the Generic notation is the most used one, being present in 32 primary studies (68.09%, 32/47). The Multiple notation is the second one most used, corresponding to 19.15% (9/47) of the primary studies. The other notations were adopted by different studies, such as Algebra (4.26%, 2/47),

Table 6
The evolution of the feature notations.

Notation	Category	Years	Description
FODA [52]	Generic	1990	The FODA is a notation with representativeness, and represents the primary model variability.
FORM [53]	Generic	1998	Extension of FODA notation. It provides additional, components to represent specific architectural components in a particular domain.
RSEB [46]	Generic	1998	It inherits many characteristics of FODA notation, having generic representations. The great difference is that features are based on Use Case diagrams.
GP [31]	Multiple	2000	GP focuses on the generation of software systems from feature models.
Hein [49]	Multiple	2000	An extension of FODA notation to enable the application of features in industry demands. For this, they add the support for cross links, to enable the relation of more than one-parent relation.
Capilla [27]	Feature algebra	2001	An extension of FODA notation to support the variability of distributed architectures. This specially enables the specification of Quality of Service attributes on the variability tree.
Van Gurp [75]	Feature algebra	2001	A feature notation to support the representation of functional and quality requirements.
GP-Extended [30]	Multiple	2002	An improvement of GP notation for providing support on embedded systems.
Riebisch [66]	Multiple	2002	An alternative feature notation that implements multiplicity, which defines amount of features in a set.
CBFM [32]	Cardinality based	2004	It brings some improvements together: cardinality, feature-diagram references and attributes.
Benavides [20]	Generic	2005	It implements a formalized feature notation. It is possible to execute reasoning, i.e., asking inherent questions to the model, such as the number of products, and the optimum product.
OVM [26]	Cardinality based	2005	It defines a notation to enable the orthogonal relations between features, and supports cardinality.
PLUSS [37]	Generic	2005	A feature notation based on use case diagrams.
VFD [75]	Generic	2007	It proposes a more precise feature notation based on previous notations. To enable this, a Free Feature Diagram is a generalization based on previous notations.
CVL [45]	Generic	2008	A generic language to define and describe feature diagrams. It can be adapted to any domain specific language.
Co-Nets [9]	Feature Algebra	2009	It is based on high-level Petri's nets. This enables the runtime adaptability. Thus, features can be weaved on running . components dynamically.

Legend: Benavindes: Notation proposed by Benavindes, Capilla: Notation proposed by Capilla, CBFM: Cardinality-Based Feature Model, Co-Nets: Convolutional Networks, CVL: Common Variability Language, FODA: Feature-Oriented Domain Analysis, FORM: Feature-Oriented Reuse Method, GP: Generative Program, Hein: Notation proposed by Hein, GP Extended: Generative Program Extended, OVM: Orthogonal Variability Model, VFD: Varied Feature Diagram, PLUSS: Product Line Use Case Modeling for Systems and Software Engineering, RSEB: Reuse-Driven Software Engineering Business, TVL: Text-based Variability Language, Van Gurp: Notation proposed by Van Gurp.

Feature Algebra (4.26%, 2/47), and Cardinality-based (2.13%, 1/47). Note that a article (i.e., [S33]) was classified as “Does not Specify” because it was a literature review article.

Table 6 shows the evolution of the feature notations over the years. This table was reformulated to better represent the evolution of the feature notations. This evolution was formulated based on the information extracted from the primary studies (Appendix A). Each study cites which work inspired it or on which it is based. Having such information at hand, it was possible to define how the techniques evolved. Between 1990 to 2010, the academia has been proposing new feature notations. The last notation focused on implementing cardinality between relationships, and improving the expressiveness to avoid ambiguity. For example, FODA notation (published in 1990) provides the representation of the feature relationships through the mandatory, optional, alternative, and OR. The CBFM published in 2004 implements the cardinality concept in feature models. Another interesting observation is that the CBFM is known as an extension of FODA that is more expressive to describe commonality and variability by introducing the cardinal concept in feature models. Finally, we can observe that there is no standard notation for feature diagrams until now, and the integration rules are maintained separately from the feature diagram.

6.2. RQ2: comparison techniques

This question investigates the strategies used in the literature to compare and identify the equivalences between two input feature models. The output of this step serves as a basis for the integration step. Then, the aspects used in this step have a strong impact on the integration results. The integration technique will integrate elements of the feature models incorrectly if incorrect equivalence relationships are defined.

Table 7 displays the data obtained in our research on the comparison techniques. The results show that great part of the primary studies adopts the semantic strategy (13%, 6/47) to compare features. Next, some approaches follow a name-based comparison strategy (15%, 7/47), and manual strategy (6.52%, 3/46). To refine the similarity, only one approach (2%, 1/47) combined the syntactic and semantic strategies. Furthermore, two heuristic approaches (4%, 2/47) were proposed. In addition, several works discuss multiples strategies (12%, 5/47) for

Table 7
Distribution of primary studies by comparison techniques (RQ2).

Comparison Techniques	Number of articles	Percentage	Article ID
Heuristic	2	4%	[S21], [S47]
Manual	3	6%	[S03], [S15], [S28]
Multiple	5	11%	[S12], [S23], [S24], [S25], [S33]
Name-based	7	15%	[S02], [S04], [S05], [S08], [S09], [S14], [S43]
Semantic	6	13%	[S13], [S18], [S19], [S20], [S22], [S29]
Syntactic and Semantic	1	2%	[S10]
Does not Specify	23	49%	[S01], [S06], [S07], [S11], [S16], [S17], [S26], [S27], [S30], [S31], [S32], [S34], [S35], [S36], [S37], [S38], [S39], [S40], [S41], [S42],[S44], [S45], [S46]
Total	47	100%	

Table 8
Distribution of primary studies by integration techniques (RQ3).

Integration strategies	Number of articles	Percentage	Article ID
Algebra	4	9%	[S11], [S14], [S15], [S29]
Merge	3	6%	[S27], [S28], [S43]
Multiple	7	15%	[S09], [S12], [S17], [S23], [S24], [S33], [S47]
Multistrategy^1	5	11%	[S02], [S04], [S05], [S10], [S42]
Multistrategy^2	1	2%	[S03]
Semantic	4	9%	[S18], [S19], [S20], [S31]
Slice operator	5	11%	[S06], [S07], [S08], [S30], [S32]
Synthesis	1	2%	[S16]
Does not specify	16	34%	[S01], [S13], [S21], [S22], [S25], [S26], [S34], [S35], [S36], [S37], [S38], [S39], [S40], [S41], [S44], [S45]
Total	47	100%	

Legend:
Multistrategy^1: Merge, union and intersection.
Multistrategy^2: Merge, union, intersection and insertion.

comparing features. Finally, most studies (49%, 23/47) do not specify any comparison technique.

These results suggest that more comparison techniques might be proposed in order to refine the granularity of the comparison, and improve the precision of the output integrated feature model.

6.3. RQ3: integration techniques

This question investigates the current techniques applied to the integration of feature models. As mentioned previously, the integration step combines the equivalent features, and then generates the output-integrated feature model. In general, there are well-known heuristics for model integration. However, to the best of our knowledge, these heuristics have not widely discussed until now.

Table 8 shows the results obtained in our research about the integration techniques. Although the majority of the studies (34%, 16/47) does not specify which integration strategy, some discuss multiple strategies (15%, 7/47). Moreover, the results show that 11% (5/47) of the primary studies make use of the slice operator, while another 11% (11%, 5/46) adopts the multistrategy to integrate features. We might also highlight that 11% (5/47) aim at using Slice Operator, which is applied to decompose feature models. In addition, one study focuses on one operator for Synthesis of views (2%, 1/47), and three studies (6%, 3/47) focus only on merge strategy. The results suggest that there is plenty of scope for new work, focused on proposing hybrid approaches, while being applicable to different types of feature model notations.

6.4. RQ4: evaluation techniques

After integrating two feature models, the integration techniques need to check whether all desired products may be produced. With this in mind, the evaluation step seeks to identify inconsistencies between the integrated feature model and desired feature model, by checking whether all well-formedness rules (from FM_A and FM_B) are satisfied. If such rules are not satisfied, then inconsistencies should be reported. SAT and CSP solvers are examples of techniques that could be used to check such rules. Note that if conflicts arise during model integration, the users of integration techniques must deal with each conflict in an appropriate way so that inconsistencies are generated.

Table 9 presents the evaluation strategies for the variability of the feature models. The overall results suggest that the integration approaches prefer adopting family analysis rather than product base analysis. The family-based analysis consists of verifying consistency on modules related to correspondent features. Whereas, the product-based strategy aims at verifying the consistency on final generated products from the feature models [72].

The majority of the studies (71%, 33/71) implements the product-based analysis, while a smaller amount aims at family-based evaluation (19%, 9/47). A few works (6%, 3/47) use multiple strategies to analyze the variability of feature models. Family-based analysis techcniques take advantage of the inherent variability of a product line for producing complete analysis results [60,71,72]. However, they are often computationally expensive. This issue may explain the greater adoption of product-based analysis techniques. Finally, very few studies (4%, 2/47) do not specify which kind of analysis is applied.

In addition to the analysis strategies, another factor that is involved in the evaluation is the constraint language. Table 10 shows the constraint languages for evaluation of feature models. It is observed that several studies invest efforts on using a multistrategy approach, including a combination of SAT and BDD (9%, 4/47), CSP and SAT (4%, 2/47), and SAT and CNF (2%, 1/47). Nevertheless, the overarching view is that there is no consensus on the use of constraint language.

For practically all other languages are adopted by only one study respectively. These languages are AFD (2%, 1/47), CFD (2%, 1/47), SAT (6%, 3/47), CNF (2%, 1/47), correctness argument (2%, 1/47), CSP (4%, 2/47), Maude (2%, 1/47), Ontology (2%, 1/47), Gaph-based (2%, 1/47), Rule-based (2%, 1/47), properties ensured by construction (6%, 3/47), restriction function (2%, 1/47), T-wise (2.17%, 1/47), and

Table 9
Distribution of studies by evaluation strategies (RQ4).

Evaluation Strategies	Number of Articles	Percentage	Article ID
Family-based	9	19%	[S01], [S02], [S11], [S16], [S34], [S36], [S39], [S42], [S44]
Product-Based	33	70%	[S03], [S04], [S05], [S06], [S07], [S08], [S09], [S10], [S13], [S14], [S15], [S17], [S18], [S19], [S20], [S21], [S22], [S24], [S25], [S26], [S28], [S29], [S30], [S31], [S32], [S37], [S38], [S40], [S41], [S43], [S45], [S46], [S47]
Multiple	3	6%	[S12], [S33], [S35]
Does not specify	2	4%	[S23], [S27]
Total	47	100%	

Table 10
Distribution of primary studies by constraint formula (RQ4).

Constraint Formulas	Number of articles	Percentage	Article ID
AFD	1	2%	[S16]
CFD	1	2%	[S44]
SAT	3	6%	[S14], [S31], [S32]
CNF	1	2%	[S47]
Correctness argument	1	2%	[S21]
CSP	2	4%	[S26], [S40]
Maude	1	2%	[S11]
Multiple	4	9%	[S01], [S12], [S25], [S35]
Ontology	1	2%	[S13]
Graph-based	1	2%	[S34]
Rule-based	1	2%	[S36]
Multistrategy (CSP and SAT)	2	4%	[S19], [S39]
Multistrategy (SAT and BDD)	4	9%	[S02], [S10], [S30], [S42]
Multistrategy (SAT and CNF)	1	2%	[S37]
Properties ensured by construction	3	6%	[S15], [S24], [S38]
Restriction function	1	2%	[S46]
T-Wise	1	2%	[S41]
Alloy	1	2%	[S45]
Does not Specify	17	38%	[S03], [S04], [S05], [S06], [S07], [S08], [S09], [S17], [S18], [S20], [S22], [S23], [S27], [S28], [S29], [S33], [S43]
Total	47	100%	

Alloy (2.17%, 1/46). Additionally, 9% of the studies (4/47) discussed multiple constraint languages.

Some studies claimed that the number of works in the evaluation of feature models increased [8,18,19,36]. Although it was found a relevant number of approaches to validate the models in this study, the majority of them (38%, 17/47) did not specify a constraint language. Finally, we might point out that an efficient execution of the evaluation step depends on the model size analyzed. So improving efficiency in validating large models is a major research challenge.

6.5. RQ5: tool support

This question aims to investigate the kind of tool support provided to the integration of models. We have identified three categories: (1) *automatic*, representing the tools that do not suffer human interference; (2) *semi-automatic*, representing the techniques that allow interactions between man and machine; and (3) *multiple*, the integration process is conducted in an automatic or semi-automatic way.

Table 11 displays the automation categories proposed by the literature. The results show that majority of the primary studies (70%, 33/47) present tools that provide automated support. Next, the semi-automatic support is the second more used (13%, 6/47). Moreover, the support for both automation strategies is the minority (6%, 3/46). Finally, some investigated studies do not specify any tool support (11%, 05/46).

Although the automatic tools are the most applied, no experiment was found that evaluated the precision and accuracy of the proposed techniques. The semi-automatic tools do not present a diagnosis of conflicts, nor do they allow the editing of the feature models. Thus, a lot of effort is still required by developers to resolve unwanted models and errors propagated during a bad integration that can affect the cost of design and maintenance. These observations point out that there are promising research directions regarding automated computational support for feature model integration. Furthermore, an intelligible guidance might be proposed to support developers in practice of combining heterogeneous feature models.

6.6. RQ6: research methods

We classified the primary studies based upon the research methods presented in Section 4.4. Table 12 shows the results of this classification. The results indicate that majority of the studies (90%, 42/47) concerned on proposing innovative solutions, while a small amount addressed evaluation research (6%, 3/46) and only two articles are related to literature review, representing 4% (2/47) of the sample. This means that little has been done to evaluate the current integration techniques empirically.

Table 11
Distribution of primary studies by automation degree (RQ5).

Tool support	Number of articles	Percentage	Article ID
Automatic	33	70%	[S01], [S02], [S05], [S06], [S07], [S08], [S09], [S10], [S11], [S13], [S14], [S16], [S17], [S18], [S19], [S20], [S21], [S22], [S29], [S30], [S31], [S32], [S34], [S38], [S39], [S40], [S41], [S42], [S43], [S44], [S46], [S47]
Semi-automatic	6	13%	[S03], [S04], [S15], [S28], [S36], [S37]
Multiple	3	6%	[S12], [S23], [S24]
Does not support	5	11%	[S25], [S26], [S27], [S33], [S35]
Total	47	100%	

Table 12
Distribution of primary studies by research methods (RQ6).

Research method	Number of articles	Percentage	Article ID
Proposal of solution	42	90%	[S01], [S02], [S03], [S05], [S06], [S07], [S08], [S09], [S10], [S11], [S13], [S14], [S15], [S16], [S17], [S18], [S19], [S20], [S21], [S22], [S23], [S24], [S26], [S27], [S28], [S29], [S30], [S31], [S32], [S34], [S36], [S37], [S38], [S39], [S40], [S41], [S42], [S43], [S44], [S45], [S46], [S47]
Evaluation research	3	6%	[S04], [S12], [S35]
Survey paper	2	4%	[S25], [S33]
Total	47	100%	

That is, little has been done to discuss the state-of-the-art techniques of feature models. Finally, the lack of a massive amount of empirical studies may indicate that the evaluation of the integration techniques has been largely based on expert reflection, rather than on empirical evidence.

7. Additional discussion

This Section seeks to reveal *when* and *where* the primary studies have been published (Section 7.1), and outline a *panoramic view* and highlight some *further challenges*, which may be taken as a basis to build a richer research agenda (Section 7.2).

7.1. Distribution of the primary studies

This Section investigates when and where the primary studies were published. The main objective is to identify publication trends in specific research venues, as well as uncover how these publications are distributed over the years. For this, the publication year and research venue of each primary study were collected using the data extraction form (Fig. 3). In terms of research venue, each study was classified into conference paper, journal paper, and workshop paper. Fig. 5 shows how the primary studies are distributed over the years. Note that the dashed black line summarizes the number of primary studies published per year from 2002 until June 2017.

7.1.1. Distribution over venues

Considering the venue where the primary studies have been published, there is a predominance of publications in conference (68.09%, 32/47). In particular, researchers have chosen SPLC (Systems and Software Product Line Conference) as the main venue to publish their research results. In total, 11 articles were published in SPLC from 2002 to 2016, including S01, S12, S13, S16, S20, S21, S26, S27, S28, S40, and S43. On the other hand, 20 articles are spread through 13 conferences, including ECBS (S11, S22), RE (S37), GPCE (S14), COMPSAC (S36), SLE (S03), ICST (S41), SAC (S39), SC (05), ECMFA (S04), SIGPLAN (S15, S35), ASE (S06, S07), CAISE (S10), AOSD (S08), IFM (S38), MODELS (S09), RCIS (S23), ICICS (S34) and ICSE (S47). Next, few studies have been published in Journals (10/47, 21.28%), including Journal of Systems and Software (S24 and S45), SOSYM (S29 and S31), ACM Computing Surveys (S33), Science of Computer Programming (S19), Software Quality Journal (S02), Domain Engineering (S30), Journal of Computer and Communications (S17) and STTT (S25). Lastly, a small number of studies (5/47, 10.64%) was published in workshops, including FMSPLE (S18, S44), EA (S32), VAMOS (S42), and ME (S46).

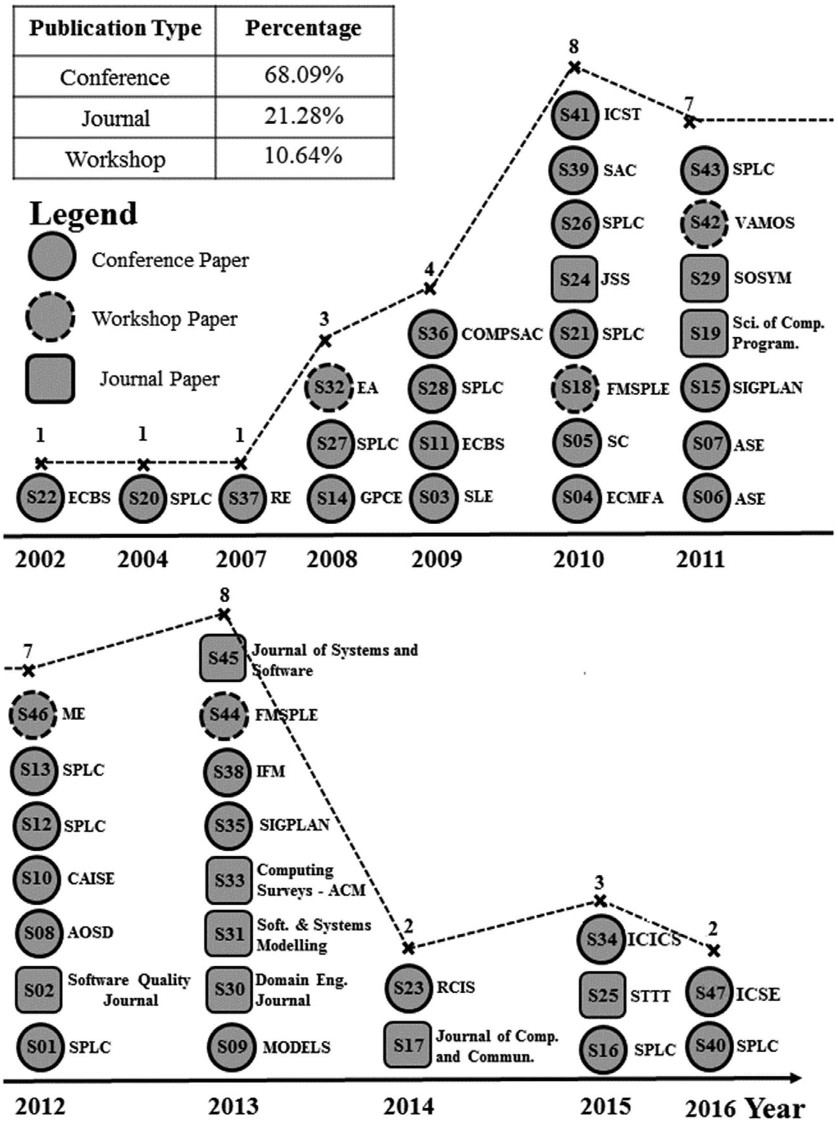
Based on the collected data, we might observe that just one article was published at the International Conference on Software Engineering. Perhaps, this is motivated by the absence of more robust studies, approaching different types of empirical methods, such as controlled experiment, case studies and survey. For example, it would be interesting to carry out complementary studies so that cross-analysis might be run.

7.1.2. Distribution over years

Although 47 studies have been published until now, the number of studies is still small. Only three articles were published from 2002 to 2007, whereas nothing was published in 2003, 2005 and 2006.

We observed a growing trend in the number of publications from 2008 to 2013, when the number of publications fell again. Specifically, the majority of studies (80.43%, 37/47) were published during this period. In 2008, three studies were published, i.e., an increase of 200%. Next, from 2009 to 2013 at least four articles were published, reaching a peak of 8 articles in 2010 and 2013, respectively. In 2014, the amount, in turn, fell from 8 to 2 articles published, representing an expressive drop.

Given that integration of software artifacts (e.g., source code, conceptual models, among others) is widely known as a wicked problem [67], and the number of open gaps is still huge, more studies should be



Publication Type	Percentage
Conference	68.09%
Journal	21.28%
Workshop	10.64%

Fig. 5. Distribution of the primary studies over the past years.

still in progress. Therefore, new publications should occur in upcoming editions of conferences, workshops, and journals. A kick off of potential effective, breakthrough techniques may occur in the next years, caused by evolution of the integration techniques. Today, no easy-to-use and production-ready tool exists, while commercial viability is unproven.

7.2. Panoramic view and further challenges

Fig. 6 presents a Bubble chart, which is a variation of a scatter chart. The axis-x represents the research issues considered, i.e., the range of research method used (left) and the type of techniques proposed (right). The axis-y represents the publication year. An additional dimension of the data is represented by the size of the bubbles, which represents the number of primary studies published.

The main feature observed in this chart is the presence of a distribution pattern considering the use of research methods over the years. The “proposal of solution” works have been the most adopted research method, being found in 85.11% (40/47) of the primary studies, whereas the remaining two research methods used registered together the 14.89% of the cases (7/47), i.e., survey paper (4.26%, 2/47) and evaluation research (10.64%, 05/47). This superiority can mean that this research field is still in an initial stage, in which the number of works is predominantly higher than one related to empirical evaluation. Additionally, we have also observed the primary studies have mostly focused on proposing integration techniques (61.70%, 29/47), comparison techniques (14.89%, 7/47), verification and validation techniques (14.89%, 7/47) and tool support (8.51%, 4/47).

In Fig. 6 (left side), for example, in 2012 one article (out of 7) was related to evaluation research (ER) and six articles were proposal of solution (PS). Still in Fig. 6 (right side), five studies are dedicated to integration techniques (IT), two studies are related to comparison techniques (CT) and validation technique (VT). In the period of 2008–2013, 37 publications were realized, representing the greatest volume of publication recorded. In particular, we might highlight that most of them focused on the development of proposal of solution (86.49%, 32/37) in the context of integration techniques (67.57%, 25/37).

Moreover, seven studies proposed integration techniques of feature models, addressing the union, difference, and intersection integration strategies. We would also highlight that such studies are recent, since they were published in the last five years. This reinforces that the studies selected are pioneering studies. In fact, only proof-of-concept techniques have been proposed, rather than user-friendly, handy techniques supported by empirical data derived from mainstream software-development projects. These data are essential to support developers

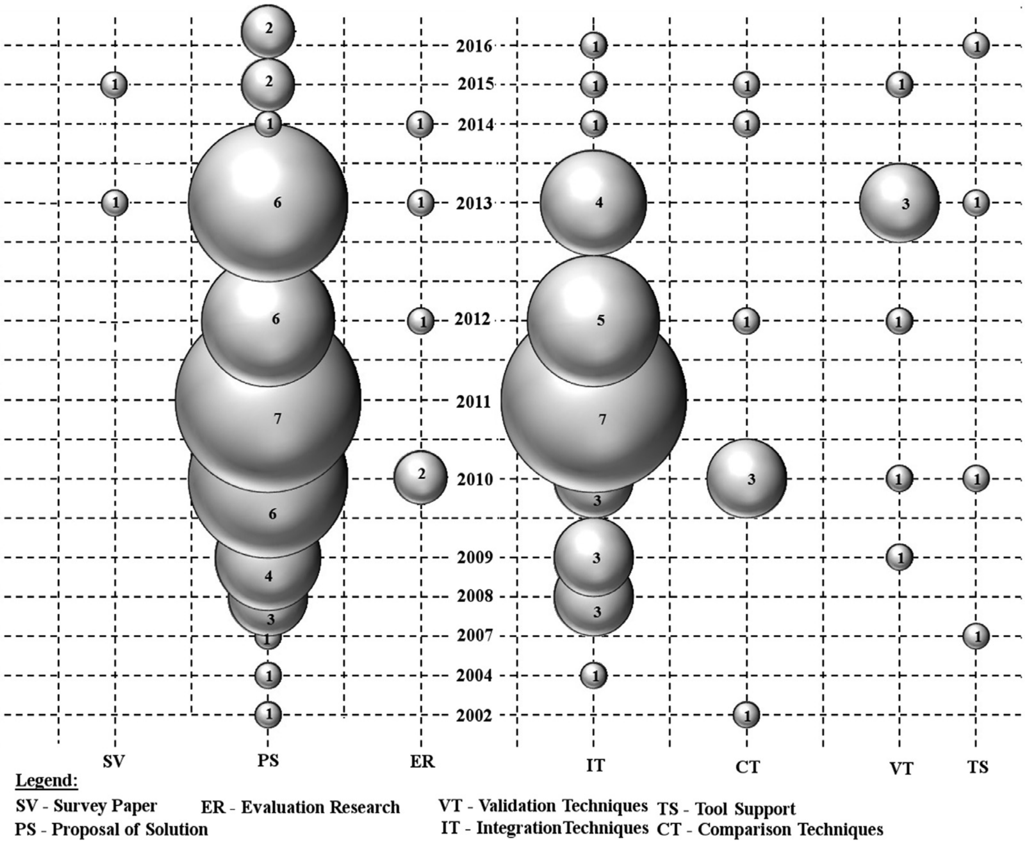


Fig. 6. Overview of the publications in terms of research issues over the years.

to make decisions on adopting (or not) such techniques in mainstream software projects, where time and cost are tight.

Taking a closer look at the integration techniques, most of them provide primitive approaches. Even though they can deal with a set of elementary integration cases, supporting structural, syntactic and semantic issues, they are still error-prone as more critical integration cases are not properly supported. In fact, usually they are unable to pinpoint similarity, or differences, caused by restructuring changes. This type of modifications is typically found in more severe cases of evolution of feature models, which can appear from the refactoring tasks or the refinement of core features in SPL (e.g., including, changing, or deleting features due to project requirement changes). A very interesting research direction is about how to improve the technical comparison between the feature models, which are aware of the structural changes, syntactic and semantics in the evolution of new models.

We have observed that the integration of feature models depends on a critical step, namely the comparison between the model elements following a flexible, multi-strategy approach, where structural, syntactic, semantic, lexical, and layout issues can be contemplated. If the level of detail to compare feature models might be adjusted, developers would compare feature models at different levels of abstraction. This would allow to have a better control about the types of conflicting changes that could occur.

Moreover, another interesting research direction would be the production of empirical knowledge on the use of the integration techniques in real-world settings. Today, such techniques are adopted without empirical evidence regarding important issues, such as granularity, flexibility, accuracy, scalability, efficiency, customizability, or even how cost-effective they are. Integration techniques have been adopted based on reflection and expert opinion rather than evidence derived from empirical studies, such as case studies, controlled experiment, and quasi-experiment. Unfortunately, the experts' opinions often diverge, and such opinions cannot be endorsed by evidence.

We highlight that little is known about the effort that developers invest to integrate feature models as well as how they lead with conflicting feature models. If conflicts between the input elements of feature models are solved improperly, then developers will be able to deal with defects (or inconsistencies). Given that the current modeling techniques are not to detect and resolve broad spectrum of inconsistencies, developers end up having to deal with feature models with problems. With this in mind, an important gap in the knowledge about using feature models is concerned with the effects of poor-quality feature models on quality issues (e.g., effort, comprehensibility and reuse) in practice.

8. Threats to validity

This Section discusses the strategies used for mitigating some threats to validity.

8.1. Construct validity

Some works highlighted that literature review may do not identify and include all the relevant works within the research field [44,64]. This can be caused by a mismatch between the search string and the keywords reported in the related works. As such, relevant works cannot be retrieved, or even irrelevant can be also identified. To mitigate this problematic, we carefully adopted rigorous procedures (reported in Section 4) for retrieving and filter primary studies. Search strings and their synonyms were defined according to well-established methods found in [56,63]. In addition, we also included a considerable range of electronic databases, including ACM Digital Library, IEEE Xplore, Google Scholar, Scopus, Springer, and Science Direct.

8.2. Conclusion validity

This threat concerns on problems that can affect the reliability of our conclusions. To mitigate it, we have followed rigorously the steps provided by well disseminated systematic mapping study protocol [56–58,62,63]. As such, the conclusions outlined are derived from data produced from experimental procedures widely recognized for conducting the systematic mapping study. Finally, all the conclusions were made after collecting the results, then avoiding the fish-and-error measures.

9. Conclusions and future works

This work sought to understand, characterize and summarize the current literature about integration of feature models. For this, we performed a systematic mapping study for investigating six research questions. We selected 47 primary studies by applying a careful filtering process to a sample of 2874 studies searched from 6 electronic databases.

We summarized the main findings as follows. Regarding notations used, the majority of the studies (68.09%, 32/47) chose the generic notation for modeling features diagrams. This result showed a strong trend toward modeling features in an informal way. As such, studies considering both comparison and integration steps are still lacking. The comparison of feature models techniques proposed in the literature covered mainly semantic (12.77%, 6/47 studies), name-based (6.38%, 3/47), and manual (6.38%, 3/47). Regarding the integration techniques, most of them are concentrated on proposing slice operator (14.89%, 7/47), a combination with merge, union, and intersection strategy (10.64%, 5/47), and semantic (8.51%, 4/47). Furthermore, some works also focused on merge strategy (6.38%, 3/47), algebra (8.51%, 4/47), a multistrategy combination with merge, union, intersection, and insertion (2.13%, 1/47), and synthesis (2.13%, 1/47).

The question concerned the validation techniques investigated the analysis strategies and the constraint languages. Regarding the analysis strategies, the majority of integration techniques adopted family-based analysis (19.57%, 9/47) instead of product-based analysis (6.38%, 3/47). This is because family-based analysis is a more efficient approach to analyze the validity of feature models than product-based one. On the other side, the results of the validation techniques showed that there are many kinds of validation strategies (i.e., techniques such as AFD, CFD, SAT, CSP, and even the combination between some of them). Specifically, these categories were discussed in a relevant number (23,40%, 11/47) of the studies. Moreover, there are also articles arguing about alloy (2.13%, 1/47), and ontology (2.13%, 1/47) exclusively.

Results related to tool support point out that the majority (70.21%, 33/47) of the primary studies have proposed automated tools to support integration of feature models. Furthermore, semi-automatic support is also covered but by a minority of studies (12.77%, 6/47). Nevertheless, a considerable portion (10.64%, 5/47) of works does not provide any tool. In addition, tool support to all integration steps proposed was not found in the current literature. Regarding the research methods, results showed that the majority (85.11%, 40/47) of the primary studies was a proposal of solutions. Given the lack of empirical studies in real-world settings, more hands-on studies, including experiments and case studies, should be validated in the industrial context.

As future work, we need more empirical and experimental research on the precision and scalability of integration approaches, not only regarding conflict and inconsistency detection, but also regarding the amount of time and effort required to resolve them. We suggest as further research to manage the number of reported integration conflicts as the feature models grow in size and complexity. Finally, we hope that the findings discussed throughout the article can encourage researchers and practitioners to explore the findings reported. Moreover, this study can be seen as a first step for a more ambitious agenda on how to characterize and improve the integration techniques of feature models.

Acknowledgments

This work was funded by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) - Universal Project 14/2013 (grant number 480468/2013-3) Brasil. This study was also financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nivel Superior - Brasil (CAPES) - Finance Code 001.

Appendix A. List of primary studies

The 46 articles selected as primary studies in the systematic mapping study are listed below.

S01. Andersen, N., Czarnecki, K., She, S., and Wasowski, A. (2012) Efficient synthesis of feature models. Proceedings of the 16th International Software Product Line Conference-Volume 1, pp. 106–115.

S02. Acher, M., Collet, P., Gaignard, A., Lahire, P., Montagnat, J., France, R. B., 2012. Composing multiple variability artifacts to assemble coherent workflows. Software Quality Journal 20 (3–4), 689–734.

S03. Acher, M., Collet, P., Lahire, P., France, R., 2009. Composing feature models. In: International Conference on Software Language Engineering. Springer, pp. 62–81.

S04. Acher, M., Collet, P., Lahire, P., France, R., 2010. Comparing approaches to implement feature model composition. In: European Conference on Modelling Foundations and Applications. Springer, pp. 3–19.

S05. Acher, M., Collet, P., Lahire, P., France, R., 2010. Managing variability in workflow with feature model composition operators. In: International Conference on Software Composition. Springer, pp. 17–33.

S06. Acher, M., Collet, P., Lahire, P., France, R. B., 2011. Decomposing feature models: language, environment, and applications. In: Proceedings of the 2011 26th IEEE/ACM International Conference on Automated Software Engineering. IEEE Computer Society, pp. 600–603.

S07. Acher, M., Collet, P., Lahire, P., France, R. B., 2011. Slicing feature models. In: Proceedings of the 2011 26th IEEE/ACM International Conference on Automated Software Engineering. IEEE Computer Society, pp. 424–427.

S08. Acher, M., Collet, P., Lahire, P., France, R. B., 2012. Separation of concerns in feature modeling: support and applications. In: Proceedings of the 11th annual international conference on Aspect-oriented Software Development. ACM, pp. 1–12.

S09. Acher, M., Combemale, B., Collet, P., Barais, O., Lahire, P., France, R. B., 2013. Composing your compositions of variability models. In: International Conference on Model Driven Engineering Languages and Systems. Springer, pp. 352–369.

S10. Acher, M., Heymans, P., Collet, P., Quinton, C., Lahire, P., Merle, P., 2012. Feature model diferences. In: International Conference on Advanced Information Systems Engineering. Springer, pp. 629–645.

S11. Aoumeur, N., Barkaoui, K., Saake, G., 2009. Validating and dynamically adapting and composing features in concurrent product-lines applications. In: Engineering of Computer Based Systems, 2009. ECBS 2009. 16th Annual IEEE International Conference and Workshop on the. IEEE, pp. 138–146.

S12. Asadi, M., Bagheri, E., Mohabbati, B., Gašević, D., 2012. Requirements engineering in feature oriented software product lines: an initial analytical study. In: Proceedings of the 16th International Software Product Line Conference-Volume 2. ACM, pp. 36–44.

S13. Barais, O., Baudry, B., Viana, W., Andrade, R., et al., 2012. An approach for semantic enrichment of software product lines. In: Proceedings of the 16th International Software Product Line Conference-Volume 2. ACM, pp. 188–195.

S14. Batory, D., 2008. Using modern mathematics as an fosd modeling language. In: Proceedings of the 7th international conference on Generative programming and component engineering. ACM, pp. 35–44.

S15. Batory, D., Höfner, P., Kim, J., 2011. Feature interactions, products, and composition. In: ACM SIGPLAN Notices. Vol. 47. ACM, pp. 13–22.

S16. Bécan, G., Behjati, R., Gotlieb, A., Acher, M., 2015. Synthesis of attributed feature models from product descriptions. In: Proceedings of the 19th International Conference on Software Product Line. ACM, pp.1–10.

S17. Budiardjo, E. K., Zamzami, E. M., et al., 2014. Feature modeling and variability modeling syntactic notation comparison and mapping. Journal of Computer and Communications 2 (02), 101.

S18. Clarke, D., Proença, J., 2010. Towards a theory of views for feature models. In: Proceedings of the First Intl. Workshop on Formal Methods in Software Product Line Engineering (FMSPLE 2010).

S19. Classen, A., Boucher, Q., Heymans, P., 2011. A text-based approach to feature modelling: Syntax and semantics of tvl. Science of Computer Programming 76 (12), 1130–1143.

S20. Czarnecki, K., Helsen, S., Eisenecker, U., 2004. Staged configuration using feature models. In: International Conference on Software Product Lines. Springer, pp. 266–283.

S21. Dao, T. M., Kang, K. C., 2010. Mapping features to reusable components: A problem frames-based approach. In: International Conference on Software Product Lines. Springer, pp. 377–392.

S22. de Bruin, H., Van Vliet, H., 2002. Top-down composition of software architectures. In: Engineering of Computer-Based Systems, 2002. Proceedings. Ninth Annual IEEE International Conference and Workshop on the. IEEE, pp. 147–156.

S23. Dehmouch, I., 2014. Towards an agile feature composition for a large scale software product lines. In: Research Challenges in Information Science (RCIS), 2014 IEEE Eighth International Conference on. IEEE, pp. 1–6.

S24. Dhungana, D., Grünbacher, P., Rabiser, R., Neumayer, T., 2010. Structuring the modeling space and supporting evolution in software product line engineering. Journal of Systems and Software 83 (7), 1108–1122.

S25. Eichelberger, H., Schmid, K., 2015. Mapping the design-space of textual variability modeling languages: a refined analysis. International Journal on Software Tools for Technology Transfer, 17(5):559-584.

S26. Guo, J., Wang, Y., 2010. Towards consistent evolution of feature models. In: International Conference on Software Product Lines. Springer, pp. 451–455.

S27. Hartmann, H., Trew, T., 2008. Using feature diagrams with context variability to model multiple product lines for software supply chains. In: Software Product Line Conference, 2008. SPLC'08. 12th International. IEEE, pp. 12–21.

S28. Hartmann, H., Trew, T., Matsinger, A., 2009. Supplier independent feature modelling. In: Proceedings of the 13th International Software Product Line Conference. Carnegie Mellon University, pp. 191–200.

S29. Höfner, P., Khedri, R., Möller, B., 2011. An algebra of product families. Software & Systems Modeling 10 (2), 161–182.

S30. Hubaux, A., Acher, M., Tun, T. T., Heymans, P., Collet, P., Lahire, P., 2013. Separating concerns in feature models: Retrospective and support for multi-views. In: Domain Engineering. Springer, pp. 3–28.

S31. Hubaux, A., Heymans, P., Schobbens, P.-Y., Deridder, D., Abbasi, E. K., 2013. Supporting multiple perspectives in feature-based configuration. Software & Systems Modeling 12 (3), 641–663.

S32. Hubaux, A., Heymans, P., Unphon, H., 2008. Separating variability concerns in a product line re-engineering project. In: Proceedings of the 2008 AOSD workshop on Early aspects. ACM, p. 4.

S33. Hubaux, A., Tun, T. T., Heymans, P., 2013. Separation of concerns in feature diagram languages: A systematic survey. ACM Computing Surveys (CSUR) 45 (4), 51.

S34. Khalfaoui, K., Kerkouche, E., Chaoui, A., Foudil, C., 2015. Automatic generation of spl structurally valid products: An approach based on progressive composition of partial configurations. In: Information and Communication Systems (ICICS), 2015 6th International Conference on. IEEE, pp. 25–31.

S35. Kolesnikov, S., von Rhein, A., Hunsen, C., Apel, S., 2013. A comparison of product-based, feature-based, and family-based type checking. In: ACM SIGPLAN Notices. Vol. 49. ACM, pp. 115–124.

S36. Mannion, M., Savolainen, J., Asikainen, T., 2009. Viewpoint-oriented variability modeling. In: Computer Software and Applications Conference, 2009. COMPSAC'09. 33rd Annual IEEE International. Vol. 1. IEEE, pp. 67–72.

S37. Metzger, A., Pohl, K., Heymans, P., Schobbens, P.-Y., Saval, G., 2007. Disambiguating the documentation of variability in software product lines: A separation of concerns, formalization and automated analysis. In: Requirements Engineering Conference, 2007. RE'07. 15th IEEE International. IEEE, pp. 243–253.

S38. Millo, J.-V., Ramesh, S., Krishna, S. N., Narwane, G. K., 2013. Compositional verification of software product lines. In: International Conference on Integrated Formal Methods. Springer, pp. 109–123.

S39. Nakajima, S., 2010. Semi-automated diagnosis of foda feature diagram. In: Proceedings of the 2010 ACM Symposium on Applied Computing. ACM, pp. 2191–2197.

S40. Nešić, D., Nyberg, M., 2016. Multi-view modeling and automated analysis of product line variability in systems engineering. In: Proceedings of the 20th International Systems and Software Product Line Conference. ACM, pp. 287–296.

S41. Perrouin, G., Sen, S., Klein, J., Baudry, B., Le Traon, Y., 2010. Automated and scalable t-wise test case generation strategies for software product lines. In: Software Testing, Verification and Validation (ICST), 2010 Third International Conference on. IEEE, pp. 459–468.

S42. Rosenmüller, M., Siegmund, N., Thüm, T., Saake, G., 2011. Multidimensional variability modeling. In: Proceedings of the 5th Workshop on Variability Modeling of Software-Intensive Systems. ACM, pp. 11–20.

S43. Scholz, W., Thüm, T., Apel, S., Lengauer, C., 2011. Automatic detection of feature interactions using the java modeling language: an experience report. In: Proceedings of the 15th International Software Product Line Conference, Volume 2. ACM, p. 7.

S44. Seidl, C., Schaefer, I., Aßmann, U., 2013. Variability-aware safety analysis using delta component fault diagrams. In: Proceedings of the 17th International Software Product Line Conference co-located workshops. ACM, pp. 2–9.

S45. Teixeira, L., Borba, P., Gheyi, R., 2013. Safe composition of configuration knowledge-based software product lines. Journal of Systems and Software 86 (4), 1038–1053.

S46. Urli, S., Blay-Fornarino, M., Collet, P., Mosser, S., 2012. Using composite feature models to support agile software product line evolution. In: Proceedings of the 6th International Workshop on Models and Evolution. ACM, pp. 21–26.

S47. Schröter, R., Krieter, S., Thüm, T., Benduhn, F., Saake, G., 2016. Feature-model interfaces: the highway to compositional analyses of highly-configurable systems. In: 2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE), pp. 667–678.

References

[1] M. Acher, P. Collet, P. Lahire, R. France, Composing feature models, in: International Conference on Software Language Engineering, Springer, 2009, pp. 62–81.

[2] M. Acher, P. Collet, P. Lahire, R. France, Comparing approaches to implement feature model composition, in: European Conference on Modelling Foundations and Applications, Springer, 2010, pp. 3–19.

[3] M. Acher, P. Collet, P. Lahire, R.B. France, Comparing approaches to implement feature model composition., ECMFA 10 (2010) 3–19.

[4] M. Acher, P. Collet, P. Lahire, R.B. France, Managing variability in workflow with feature model composition operators., in: Software Composition, Vol. 6144, Springer, 2010, pp. 17–33.

[5] M. Acher, P. Collet, P. Lahire, R.B. France, Separation of concerns in feature modeling: support and applications, in: Proceedings of the 11th Annual International Conference on Aspect-Oriented Software Development, ACM, 2012, pp. 1–12.

[6] M. Acher, B. Combemale, P. Collet, O. Barais, P. Lahire, R.B. France, Composing your compositions of variability models, in: International Conference on Model Driven Engineering Languages and Systems, Springer, 2013, pp. 352–369.

[7] M. Acher, P. Heymans, P. Collet, C. Quinton, P. Lahire, P. Merle, Feature model differences, in: International Conference on Advanced Information Systems Engineering, Springer, 2012, pp. 629–645.

[8] N. Andersen, K. Czarnecki, S. She, A. Wasowski, Efficient synthesis of feature models, in: 16th International Software Product Line Conference-Volume 1, ACM, 2012, pp. 106–115.

[9] N. Aoumeur, K. Barkaoui, G. Saake, Validating and dynamically adapting and composing features in concurrent product-lines applications, in: 16th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems, IEEE, 2009, pp. 138–146.

[10] S. Apel, J.M. Atlee, L. Baresi, P. Zave, Feature interactions: the next generation (Dagstuhl seminar 14281), Dagstuhl Rep. 4 (7) (2014).

[11] S. Apel, C. Kästner, An overview of feature-oriented software development, J. Object Technol. 8 (5) (2009) 49–84.

[12] M. Asadi, E. Bagheri, B. Mohabbati, D. Gašević, Requirements engineering in feature oriented software product lines: An initial analytical study, in: Proceedings of the 16th International Software Product Line Conference - Volume 2, in: SPLC '12, ACM, New York, NY, USA, 2012, pp. 36–44.

[13] D. Batory, Feature models, grammars, and propositional formulas, in: International Conference on Software Product Lines, Springer, 2005, pp. 7–20.

[14] D. Batory, Using modern mathematics as an fosd modeling language, in: Proceedings of the 7th International Conference on Generative Programming and Component Engineering, ACM, 2008, pp. 35–44.

[15] D. Batory, D. Benavides, A. Ruiz-Cortes, Automated analysis of feature models: challenges ahead, Commun. ACM 49 (12) (2006) 45–47.

[16] D. Batory, P. Höfner, J. Kim, Feature interactions, products, and composition, ACM SIGPLAN Not. 47 (3) (2011) 13–22.

[17] G. Bécan, R. Behjati, A. Gotlieb, M. Acher, Synthesis of attributed feature models from product descriptions, in: Proceedings of the 19th International Conference on Software Product Line, ACM, 2015, pp. 1–10.

[18] D. Benavides, A. Felfernig, J.A. Galindo, F. Reinfrank, Automated analysis in feature modelling and product configuration, in: International Conference on Software Reuse, Springer, 2013, pp. 160–175.

[19] D. Benavides, S. Segura, A. Ruiz-Cortés, Automated analysis of feature models 20 years later: a literature review, Inf. Syst. 35 (6) (2010) 615–636.

[20] D. Benavides, P. Trinidad, A. Ruiz-Cortés, Automated reasoning on feature models, in: International Conference on Advanced Information Systems Engineering, Springer, 2005, pp. 491–503.

[21] T. Berger, D. Lettner, J. Rubin, P. Grünbacher, A. Silva, M. Becker, M. Chechik, K. Czarnecki, What is a feature?: A qualitative study of features in industrial software product lines, in: Proceedings of the 19th International Conference on Software Product Line, in: SPLC '15, 2015, pp. 16–25.

[22] T. Berger, R. Rublack, D. Nair, J.M. Atlee, M. Becker, K. Czarnecki, A. Wasowski, A survey of variability modeling in industrial practice, in: Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems, ACM, 2013, p. 7.

[23] D. Beuche, M. Dalgarno, Software product line engineering with feature models, Overload J. 78 (2007) 5–8.

[24] Y. Brun, R. Holmes, M.D. Ernst, D. Notkin, Proactive detection of collaboration conflicts, in: 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering, in: ESEC/FSE '11, ACM, New York, NY, USA, 2011, pp. 168–178.

[25] E.K. Budiardjo, E.M. Zamzami, et al., Feature modeling and variability modeling syntactic notation comparison and mapping, J. Comput. Commun. 2 (02) (2014) 101.

[26] S. Bühne, K. Pohl, Domain requirements engineering, Softw. Prod. Line Eng. (2005) 193–216.

[27] R. Capilla, J.C. Dueñas, Modelling variability with features in distributed architectures, in: International Workshop on Software Product-Family Engineering, Springer, 2001, pp. 319–329.

[28] D. Clarke, J. Proença, Towards a theory of views for feature models, in: Proceedings of the First Intl. Workshop on Formal Methods in Software Product Line Engineering (FMSPLE 2010), 2010.

[29] A. Classen, P. Heymans, P.-Y. Schobbens, What's in a feature: A requirements engineering perspective, in: International Conference on Fundamental Approaches to Software Engineering, Springer, 2008, pp. 16–30.

[30] K. Czarnecki, T. Bednasch, P. Unger, U.W. Eisenecker, Generative programming for embedded software: an industrial experience report, in: GPCE, 2, Springer, 2002, pp. 156–172.

[31] K. Czarnecki, U.W. Eisenecker, K. Czarnecki, Generative Programming: Methods, Tools, and Applications, vol. 16, Addison Wesley Reading, 2000.

[32] K. Czarnecki, S. Helsen, U. Eisenecker, Staged configuration using feature models, in: SPLC, vol. 3154, Springer, 2004, pp. 266–283.

[33] K. Czarnecki, S. Helsen, U. Eisenecker, Formalizing cardinality-based feature models and their specialization, Softw. Process 10 (1) (2005) 7–29.

[34] T.M. Dao, K.C. Kang, Mapping features to reusable components: a problem frames-based approach, in: International Conference on Software Product Lines, Springer, 2010, pp. 377–392.

[35] I. Dehmouch, Towards an agile feature composition for a large scale software product lines, in: 2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS), IEEE, 2014, pp. 1–6.

[36] H. Eichelberger, C. Kroher, K. Schmid, An analysis of variability modeling concepts: expressiveness vs. analyzability, in: International Conference on Software Reuse, Springer, 2013, pp. 32–48.

[37] M. Eriksson, J. Börstler, K. Borg, The pluss approach–domain modeling with features, use cases and use case realizations, Softw. Prod. Lines (2005) 33–44.

[38] K. Farias, Empirical Evaluation of Effort on Composing Design Models, Department of Informatics, PUC-Rio, 2012 Ph.D. thesis. Rio de Janeiro, Brazil

[39] K. Farias, A. Garcia, C. Lucena, Effects of stability on model composition effort: an exploratory study, Softw. Syst. Model. 13 (4) (2013) 1473–1494.

[40] K. Farias, A. Garcia, J. Whittle, C. Chavez, C. Lucena, Evaluating the effort of composing design models: a controlled experiment, in: 15th International Conference on Model-Driven Engineering Languages and Systems (MODELS'12), 2012, pp. 676–691.

[41] K. Farias, A. Garcia, J. Whittle, C. Chavez, C. Lucena, Evaluating the effort of composing design models: a controlled experiment, Softw. Syst. Model. 14 (4) (2015) 1349–1365.

[42] K. Farias, A. Garcia, J. Whittle, C. Lucena, Analyzing the effort of composing design models of large-scale software in industrial case studies, in: 6th International Conference on Model-Driven Engineering Languages and Systems (MODELS'13), 2013, pp. 639–655.

[43] K. Farias, L. Gonçales, M. Scholl, T.C. Oliveira, M. Veronez, Toward an architecture for model composition techniques, in: H. Xu (Ed.), 28th International Conference on Software Engineering and Knowledge Engineering, 2015, pp. 656–659.

[44] A.M. Fernández-Sáez, M. Genero, M.R. Chaudron, Empirical studies concerning the maintenance of uml diagrams and their use in the maintenance of code: a systematic mapping study, Inf. Softw. Technol. 55 (7) (2013) 1119–1142.

[45] F. Fleurey, Ø. Haugen, B. Møller-Pedersen, G.K. Olsen, A. Svendsen, X. Zhang, A Generic Language and Tool for Variability Modeling, Technical Report, University of Oslo, 2009.

[46] M.L. Griss, J. Favaro, M. d'Alessandro, Integrating feature modeling with the RSEB, in: Software Reuse, 1998. Proceedings. Fifth International Conference on, IEEE, 1998, pp. 76–85.

[47] J. Guo, Y. Wang, Towards consistent evolution of feature models, in: Proceedings of the 14th International Conference on Software Product Lines: Going Beyond, in: SPLC'10, 2010, pp. 451–455.

[48] H. Hartmann, T. Trew, A. Matsinger, Supplier independent feature modelling, in: Proceedings of the 13th International Software Product Line Conference, Carnegie Mellon University, 2009, pp. 191–200.

[49] A. Hein, M. Schlick, R. Vinga-Martins, Applying feature models in industrial settings, Softw. Prod. Lines (2000) 47–70.

[50] P. Höfner, R. Khedri, B. Möller, An algebra of product families, Softw. Syst. Model. 10 (2) (2011) 161–182.

[51] A. Hubaux, T.T. Tun, P. Heymans, Separation of concerns in feature diagram languages: a systematic survey, ACM Comput. Surv. 45 (4) (2013) 51.

[52] K.C. Kang, S.G. Cohen, J.A. Hess, W.E. Novak, A.S. Peterson, Feature-Oriented Domain Analysis (FODA) Feasibility Study, Technical Report, DTIC Document, 1990.

[53] K.C. Kang, S. Kim, J. Lee, K. Kim, E. Shin, M. Huh, Form: a feature-oriented reuse method with domain-specific reference architectures, Ann. Softw. Eng. 5 (1) (1998) 143.

[54] K.C. Kang, J. Lee, P. Donohoe, Feature-oriented product line engineering, IEEE Softw. 19 (4) (2002) 58.

[55] K. Khalfaoui, E. Kerkouche, A. Chaoui, C. Foudil, Automatic generation of SPL structurally valid products: an approach based on progressive composition of partial configurations, in: 6th International Conference on Information and Communication Systems (ICICS), IEEE, 2015, pp. 25–31.

[56] B. Kitchenham, S. Charters, Guidelines for Performing Systematic Literature Reviews in Software Engineering, 2007.

[57] B. Kitchenham, R. Pretorius, D. Budgen, O.P. Brereton, M. Turner, M. Niazi, S. Linkman, Systematic literature reviews in software engineering–a tertiary study, Inf. Softw. Technol. 52 (8) (2010) 792–805.

[58] B.A. Kitchenham, D. Budgen, O.P. Brereton, Using mapping studies as the basis for further research–a participant-observer case study, Inf. Softw. Technol. 53 (6) (2011) 638–651.

[59] S. Kolesnikov, A. von Rhein, C. Hunsen, S. Apel, A comparison of product-based, feature-based, and family-based type checking, ACM SIGPLAN Not. 49 (3) (2014) 115–124.

[60] S. Kolesnikov, A. von Rhein, C. Hunsen, S. Apel, A comparison of product-based, feature-based, and family-based type checking, ACM SIGPLAN Not. 49 (3) (2014) 115–124.

[61] L. Passos, K. Czarnecki, S. Apel, A. Wasowski, C. Kästner, J. Guo, Feature-oriented software evolution, in: Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems, in: VaMoS '13, 2013, pp. 1–17.

[62] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, Systematic mapping studies in software engineering, in: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering, in: EASE'08, 2008, pp. 68–77.

[63] K. Petersen, S. Vakkalanka, L. Kuzniarz, Guidelines for conducting systematic mapping studies in software engineering: an update, Inf. Softw. Technol. 64 (2015) 1–18.

[64] D. Qiu, B. Li, S. Ji, H. Leung, Regression testing of web service: a systematic mapping study, ACM Comput. Surv. 47 (2) (2014). 21:1–21:46

[65] A. Reder, A. Egyed, Determining the cause of a design model inconsistency, IEEE Trans. Softw. Eng. 39 (11) (2013) 1531–1548.

[66] M. Riebisch, K. Böllert, D. Streitferdt, I. Philippow, Extending feature diagrams with uml multiplicities, in: 6th World Conference on Integrated Design & Process Technology (IDPT2002), vol. 23, 2002, pp. 1–7.

[67] M. Rosa, M. Dumas, R. Uba, R. Dijkman, Business process model merging: an approach to business process consolidation, ACM Trans. Softw. Eng. Methodol. 22 (2) (2013) 11–42.

[68] I. Schaefer, R. Rabiser, D. Clarke, L. Bettini, D. Benavides, G. Botterweck, A. Pathak, S. Trujillo, K. Villela, Software diversity: state of the art and perspectives, Int. J. Softw. Tools Technol. Trans. 14 (5) (2012) 477–495.

[69] W. Scholz, T. Thüm, S. Apel, C. Lengauer, Automatic detection of feature interactions using the java modeling language: an experience report, in: Proceedings of the 15th International Software Product Line Conference, Volume 2, ACM, 2011, p. 7.

[70] S. Segura, D. Benavides, A. Ruiz-Cortés, P. Trinidad, Automated merging of feature models using graph transformations, in: Generative and Transformational Techniques in Software Engineering II, Springer, 2008, pp. 489–505.

[71] T. Thüm, S. Apel, C. Kästner, M. Kuhlemann, I. Schaefer, G. Saake, Analysis Strategies for Software Product Lines, Technical Report, School of Computer Science, University of Magdeburg, Tech. Rep. FIN-004-2012, 2012.

[72] T. Thüm, S. Apel, C. Kästner, I. Schaefer, G. Saake, A classification and survey of analysis strategies for software product lines, ACM Comput. Surv. 47 (1) (2014). 6:1–6:45

[73] T. Thum, D. Batory, C. Kastner, Reasoning about edits to feature models, in: 2009 IEEE 31st International Conference on Software Engineering, 2009, pp. 254–264.

[74] T. Thum, C. Kastner, S. Erdweg, N. Siegmund, Abstract features in feature modeling, in: Software Product Line Conference (SPLC), 2011 15th International, 2011, pp. 191–200.

[75] J. Van Gurp, J. Bosch, M. Svahnberg, On the notion of variability in software product lines, in: Software Architecture, 2001. Proceedings. Working IEEE/IFIP Conference on, 2001, pp. 45–54.

[76] V. Weber, K. Farias, L. Gonçales, V. Bischoff, Detecting inconsistencies in multi-view uml models, Int. J. Comput. Sci. Softw. Eng. 5 (12) (2016) 260–264.