The Journal of Systems and Software 154 (2019) 1–21

Contents lists available at ScienceDirect

The Journal of Systems and Software

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

FM-CF: A framework for classifying feature model building approaches

Ricardo Gacitúa[a,b,*], Samuel Sepúlveda[a,b], Raúl Mazo[c,d]

[a] Department of Computer Science and Informatics, Universidad de La Frontera (UFRO), Temuco, Chile
[b] CEIS-Centro de Estudio en Ingeniería de Software, UFRO, Temuco, Chile
[c] Centre de Recherche en Informatique (CRI), Université Panthéon-Sorbonne, Paris, France
[d] GiDITIC, Universidad Eafit, Medellín, Colombia

ARTICLE INFO

Article history:
Received 13 September 2017
Revised 5 April 2019
Accepted 7 April 2019
Available online 8 April 2019

Keywords:
Feature model
Software product lines
Framework
Classification
Models

ABSTRACT

Software product line engineering has emerged as a prominent software engineering paradigm, as it comprises a set of core assets sharing functionality and quality attributes. Feature modelling is one of the most frequently used techniques for modelling the variability within a software product line. There are several proposals for building Feature Models which rely on semi-automated or fully automated means. Unfortunately, automatic feature model construction has been addressed from different viewpoints, so it is not easy to know which is the best approach for automating the building of variability models. In fact, there is no clarity regarding common elements, and the main differences that characterise such approaches. Additionally, the wide variety of terms used to refer to the process of building a Feature Model (e.g. synthesis, location, re-engineering, and weaving) means that approaches are varied and very heterogeneous, making them complex to understand and classify. This paper introduces FM-CF, which is a Conceptual experience-based Framework for classifying approaches for the automatic building of Feature Models. The framework considers a set of categories mainly focused on characterising some aspects, such as input sources, methods and techniques, results, and types of evaluation. A literature review of (semi-) automated Feature Model construction was performed to identify approaches for building Feature Models by (semi-)automatic means, and the main terms used by those approaches. Then the completeness of the framework was evaluated by mapping the set of dimensions and their items, and the terms extracted from the literature.

The conceptual framework provides guidance to researchers for choosing the appropriate aspects with which to build Feature Models, and helps in the understanding and clarification of the proposed approaches.

1. Introduction

Software Product Line Engineering (SPLE) has emerged as a promising software development paradigm for increasing productivity and improving variability management, as it represents an exciting approach for achieving software reuse (Mannion et al., 2017). This paradigm is inspired by manual production processes, such as the production of planes and other vehicles, in the sense that a high percentage of the components used in their construction are shared. Software Product Lines (SPL) have emerged with the intention of avoiding the repetition of production processes, or aspects of them.

Pohl et al. (2005) define SPL as "*a group of software systems that share and administer a common set of characteristics that satisfy the specific needs of a particular market segment or purpose, and are developed from a common set of active codes in a predefined way*". One of the key concepts in SPL is *variability*, defined as the ability of the SPL to be changed, adjusted, configured or extended for use in a specific context (Chen et al., 2009). Models that allow for the representation of variability in a set of systems are fundamental to the development and management of SPL. Such models include related concepts with features (Kang et al., 1990), decisions (Schmid et al., 2011) or variation points (Griss et al., 1998) depending on the level of abstraction.

Feature Models (FMs) constitute one of the most used notations for representing variability and commonalities within a product line (Czarnecki et al., 2006). An FM is organised hierarchically, but also allows for transversal relationships among the characteristics or features of the corresponding product line. Its root node represents the name of the family of software products, and its leaves

* Corresponding author at: Department of Computer Science and Informatics, Universidad de La Frontera (UFRO), Temuco, Chile.
E-mail addresses: ricardo.gacitua@ufrontera.cl (R. Gacitúa), samuel.sepulveda@ufrontera.cl (S. Sepúlveda), raul.mazo@univ-paris1.fr (R. Mazo).

https://doi.org/10.1016/j.jss.2019.04.026
0164-1212/

represent individual or merged components that could be assembled in order to create a particular application. The manual construction of FMs is very work intensive in terms of man-hours, and is not exempt from error due to human imprecision (Bécan et al., 2014). As a consequence, approaches that allow for the partial or total automation of FM construction are required so that the strain involved can be reduced (Andersen et al., 2012). However, there remains little clarity regarding which factors must be taken into account in order to produce models that truly represent variability aspects and the required level of detail. In fact the broad variety of concepts used to refer to the FM construction process (such as: synthesis, creation, construction, weave, compose, aggregation, generation, extraction, re-engineering, and development) make it complicated to understand and classify the approaches that are used. This is especially true considering that each concept proposes a different approach and, in many cases, a particular context and domain (Bécan et al., 2015b).

This study presents an experience-based framework for classifying approaches related to the construction of FMs by automated or semi-automated means. In other words, this paper aims to conceptualise an analysis framework for work in the area of automated FM construction, and capture the current state of this area. More specifically, we are investigating the following research question: How can the automated approaches for constructing FMs be compared and classified? We detail this overall question by proposing an experience-based framework that comprises a set of categories to classify work that addresses automatic FM building. A systematic review was conducted to (i) explore the current situation of automated FM construction, and (ii) extract technical keywords used in these FM construction approaches. Technical keywords are an essential part of all technical and scientific writing. Each field and specialty typically uses a vocabulary that relays a variety of specialised concepts by means of technical language. The collection of keywords extracted from the literature is used to verify that our framework integrates the knowledge that exists in the current set of approaches for constructing FMs in an automatic mode. The experience-based framework presented in this paper has the following objectives:

1. Understanding and clarifying the diversity of existing approaches.
2. Locating the state of the practice of model construction, and
3. Aiding researchers in the development of better approaches to be used in FM construction.

The remainder of this paper is structured as follows: Section 2 introduces basic concepts and presents the relative dilemma of the broad variety of terms used when discussing the construction of FMs. Section 3 describes the conceptual framework and the methodology that characterises the process that led to the proposed framework. Section 4 presents the validation of the completeness of the framework, and shows a set of different proposals of considered research studies, analysing them under the conceptual framework. Section 5 analyses the strengths and weaknesses of the conceptual classification framework. Section 7 presents some related studies and shows that our proposed technique is innovative and up to the minute. Finally, Section 8 presents the conclusions of the study and an agenda for future work.

2. Background

We begin our the technical development with basic terminology used to refer to the construction of FMs by automatic means.

2.1. Feature model

Features are an abstract concept for describing commonality and the variability of a collection of products that belong to the same product line. A feature is a characteristic of a system relevant to some stakeholders. What this means precisely needs to be decided for each product line. Features can be for example requirements, technical functions or function groups, or non-functional (quality) characteristics, depending on the interests of the stakeholders. Feature Models (FMs) were first introduced in the Feature-Oriented Domain Analysis (FODA) method by Kang et al. (1990). An FM represents the information of all the possible products of a product line in terms of its features and relationships, and it is one of the most frequently used tools for modelling product lines. In fact, a recent survey about variability modelling showed that feature modelling is the most frequently reported notation in industry (Berger et al., 2013).

An FM is represented as a tree structure, with features forming the nodes of the tree composed of relationships between a parent (or compound) feature and its child features (or sub-features), and cross-tree (or cross-hierarchy) constraints that are typically inclusion or exclusion statements in the form: if feature `F` is included, then features `A` and `B` must also be included (or excluded). Fig. 1 depicts a simplified FM inspired by the coffee making industry which shows that every `coffee machine` must have a `Brewer`, a `Coffee Maker` and a `Creamer` component. However, it may or may not have a `Tea Maker` component. A coffee maker can provide `American` or `Espresso` types of coffee, or both, and the `Creamer` can be either `Regular` or `Irish`. Having an FM, a customer may ask for a specific configuration, i.e., a set of features. For example, enabling the set of features {`Coffee Machine, Brewer, Coffee Maker, American, Creamer, Regular`} form a *valid* configuration. A set of enabled feature is, therefore, a set of features whose grouping is valid Note that mandatory features must be present in a *valid* configuration if their parents are present. In contrast, the configuration {`Coffee Machine, Brewer, Coffee Maker, Creamer, Regular, Tea Maker`} is *not valid*, since no sub-feature of `Coffee Maker` is selected. Note that the whole `Coffee Machine` (the root) must be included in all valid configurations.

A valid configuration must include the root feature (i.e., `Coffee Machine` in Fig. 1). In addition to this concept, the FM presented in this section allows the following relationships among its features:

- Mandatory. A child feature has a mandatory relationship with its parent when the child is included in all products in which its parent feature appears. For instance, every coffee machine in our example must have a `Brewer`.
- Optional. A child feature has an optional relationship with its parent when the child can be included optionally in all products in which its parent feature appears. In the example, a coffee machine may optionally include a `Tea Maker`.
- Alternative. A set of child features has an alternative relationship with the parent when only one feature of the children can be selected if its parent feature is part of the product. In the example, a `Coffee Maker` may provide `American` or `Espresso` types of coffee.
- Disjunction. A set of child features has an or-relationship with their parent when one or more of them can be included in the products in which its parent feature appears. In Fig. 1, whenever `Creamer` is selected, `Regular` or `Irish` can be selected.

In addition to the parental relationships within features, an FM can also contain cross-tree constraints between features. These are typically in the form:

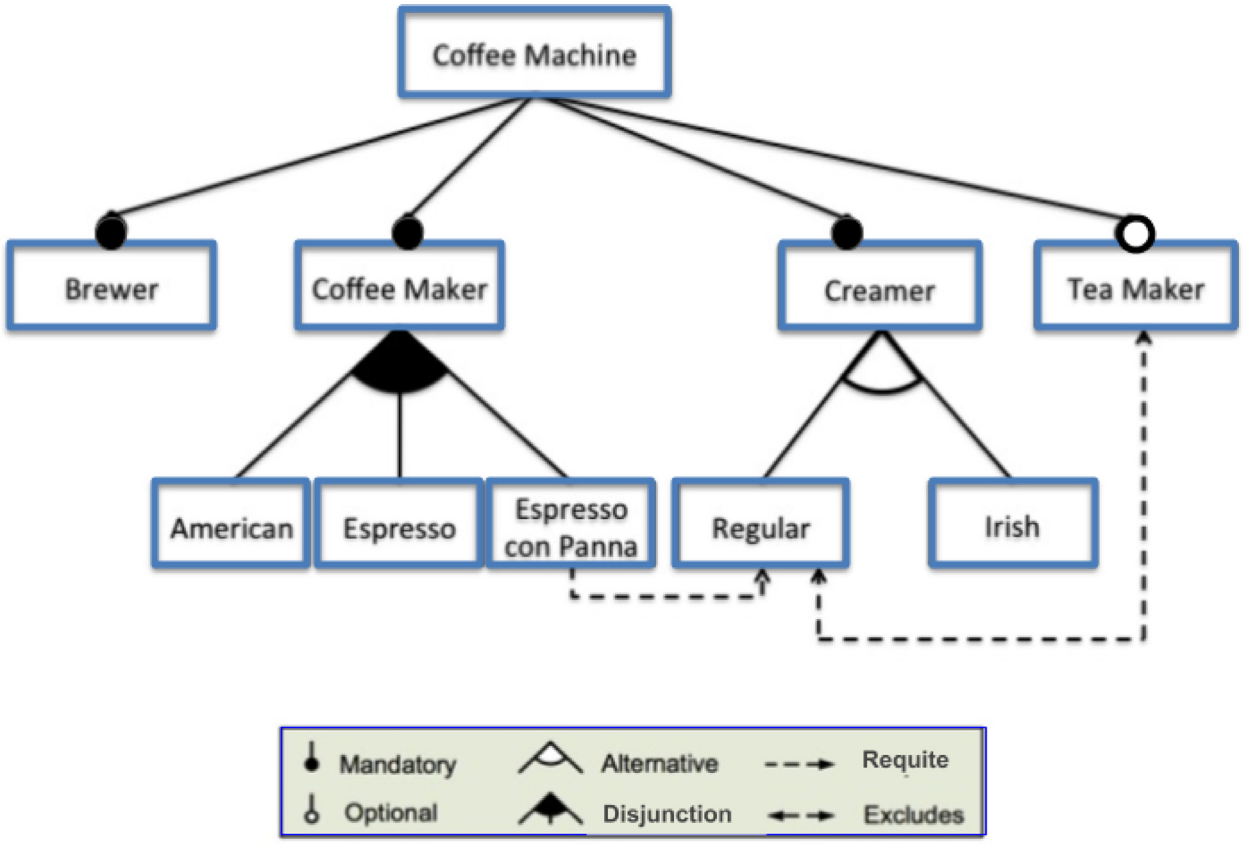

Fig. 1. The Coffee machine sample.

- Requisite. If a feature A requires a feature B, the inclusion of A in a product implies the inclusion of B too. `Espresso con Panna` requires the inclusion of `Regular Cream`.
- Exclude. If a feature A excludes a feature B, neither feature can be part of the same product. In the example presented in this section a `Tea Maker` and `Regular Cream` are incompatible.

Sometimes it is necessary to extend FMs in order to include more information about features. More complex cross-tree relationships have been proposed later in the literature allowing constraints in the form of generic propositional formulas (Batory, 2005) or constraints (Mazo et al., 2011), e.g. "(A and B) implies not C". Thus, the entire FM could be written as one constraint program or one propositional logic formula; however, the whole reason for doing the FM in the FODA-style of Fig. 1 is the interest in the structure and structural properties of the FM. Some authors propose to introduce new relationships with UML-like multiplicities, sometimes called *cardinalities* in order to enhance the conceptual completeness of FMs (Czarnecki et al., 2005; Riebisch et al., 2002). Extended FMs can also include complex constraints among their attributes and features such as: "if attribute A of feature F is lower than a value X, then feature T cannot be part of the product". Kang et al. (1998) make an explicit reference to what they call non-functional features related to feature attributes. Some authors propose the inclusion of attributes in FMs (Batory, 2005; Batory et al., 2006). Various extensions of the FODA-style were introduced to compensate for a degree of ambiguity and lack of precision, and also expressiveness. However, they did not receive formal semantics, which is the indicator of precision and lack of ambiguity, and a prerequisite for efficient and safe tool automation. Schobbens et al. (2007) contrary to popular belief, it has been formally proved that the extensions of FODA add absolutely no expressiveness to original FDs, but these are maximally expressive.

2.2. Terminology

The wide variety of terms used in reference to FM construction creates confusion when one needs to establish classifications, or make comparisons among approaches. As a result, it is necessary to provide clarity regarding the exact sense in which a term is used in order to reduce such confusion. In the literature, various terms referring to FM construction can be found, as illustrated in Table 1, many of which are often used as synonyms and are thus used indistinctly.

The diversity of terms used to refer to FM construction processes makes it more difficult to understand proposals due to the fact that they highlight, on occasion, a disconnect between what is defined in terms of actions described by the authors, and the defined purpose of the study. For example, the concept of re-engineering in the field of software refers to the process of restructuring a system. This can be seen as a return to the cycle of development. Win et al. (Tun and Myo, 2015) proposed an approach for the "re-engineering" of an FM. However, considering that they presented a method for the extraction of variability from software products, in accordance with the previously supplied definition, the proposal cannot be considered re-engineering in the context of software. Due to the above, we utilised a taxonomy proposed by Chikofsky and Cross (1990) that defines and clarifies terms such as: forward engineering; reverse engineering; re-engineering; among others. This taxonomy provides a new characterisation approach for identifying the underlying objective of a given proposal when using a conceptual framework for the classification of FM construction proposals.

Fig. 2 presents the relationship among related terms used in the construction of models. The taxonomy defined by Chikofsky and Cross (1990), for the sake of simplicity, uses only three stages to

Table 1
Some terms referring to FM construction.

Name	Description	References
Reverse Engineering	The process of analysing a system in order to identify its components and their interrelation, and to create representations of the system in another form, or at a higher level of abstraction	(Haslinger et al., 2011; Lopez-Herrejon et al., 2012; She et al., 2011; Acher et al., 2013; 2011)
Re-Engineering	It is the examination or alteration of the system in order to rebuild it in a new way, and the subsequent implementation of its new form	(Tun and Myo, 2015)
Synthesis	The process of producing something through the combination of separate parts	(Bécan et al., 2014), (Andersen et al., 2012), (Bécan et al., 2015b)
Generation	The process of producing something	(Bae and Kang, 2007) (Casalánguida and Durán, 2012), (Wanderley et al., 2012), (Itzik and Reinhartz-Berger, 2014)
Mining	The process of discovering patterns	(Yi et al., 2012)(Ferrari et al., 2013)
Composition	The process of forming something through by putting parts together	(Fleurey et al., 2007) (Acher et al., 2010)
Recovery	The process for the extraction of architectural information from lower level representations of a software system	(Yang et al., 2009) (Al-Msie'Deen et al., 2012)
Extraction	The process of obtain something using a set of methods	(Ryssel et al., 2011) (Mefteh et al., 2015; Davril et al., 2013; Damasevicius et al., 2012; Acher et al., 2012)
Construction	The work of building something	(Chen et al., 2005)
Weaving	The process of build something moving backwards and forwards or from side to side	(Brown et al., 2006)(Horcas et al., 2016)
Identification	the act of recognising and naming someone or something	(Ziadi et al., 2012)

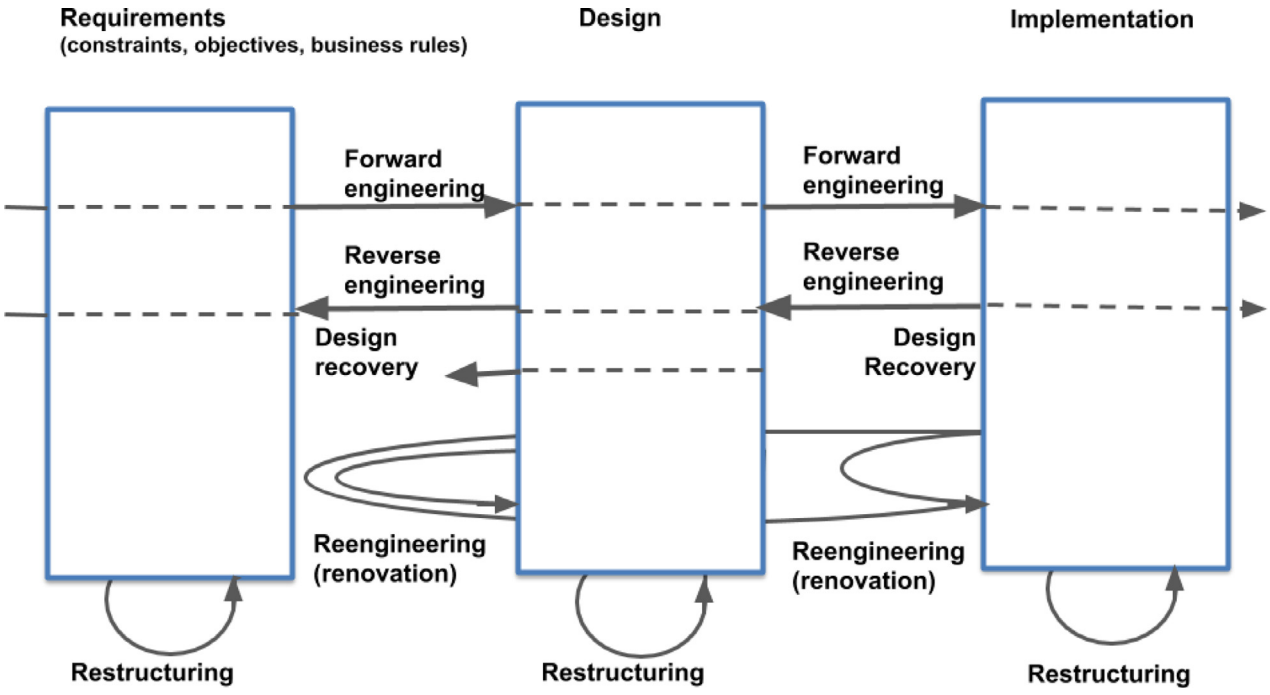

Fig. 2. Relationships between terms. Taken from Chikofsky and Cross (1990) .

describe a software product's life cycle with different levels of abstraction which are: i) Requirements of the problem to be solved, ii) Design (the specifying of the solution), and iii) Implementation (coding, trials, and delivery of the operating system).

Using this taxonomy, the following definitions can be established:

- **Forward engineering (aka direct engineering:** The traditional process of moving from a high level of abstraction and logic towards a level of design, and subsequently a level of physical implementation.
- **Reverse engineering (aka inverse engineering):** The process of analysing a system in order to identify its components and their interrelation, and to create representations of the system in another form, or at a higher level of abstraction. Reverse Engineering generally implies the extraction of design devices, and the construction or synthesis of abstractions that are less dependent on implementation. Reverse Engineering does not imply changes to the system, nor the creation of a new system based on a reversed version of the system. It is a process of examination and not a process of change or duplication. There are several sub-areas of Reverse Engineering. Two of them are: *Redocumentation and Design Recovery*. Redocumentation is the creation or revision of a semantically equivalent representation within the same level of abstraction. The resulting forms of the representation are considered alternative views. The prefix "re"

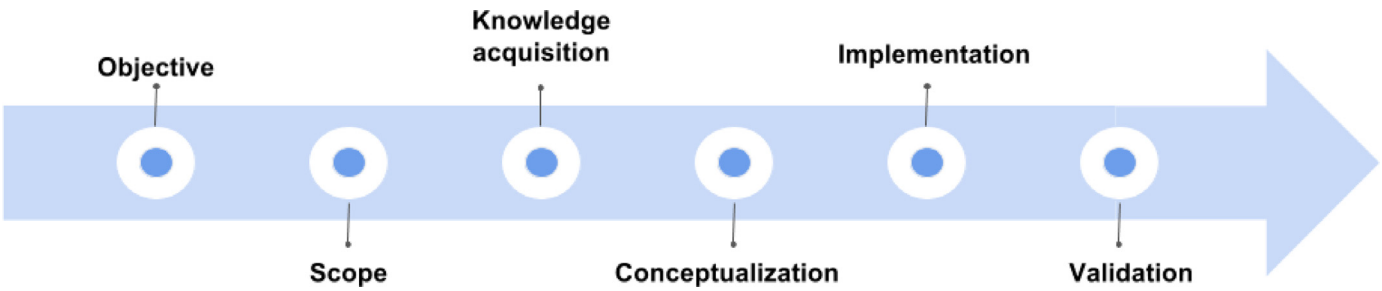

Fig. 3. The framework construction process.

implies the intention to recover documentation regarding a system that exists, or should have existed. Design Recovery is a subset of Inverse Engineering in which information is added to the domain. Information refers to external information or reasoning of the observations of the system, in order to identify significant abstractions at a higher level than those obtained by the system examining itself. Thus, Design Recovery recreates design abstractions from a combination of code, of existing design documentation, personal experience, and general knowledge relating to the problem and the application domain.

- **Restructuring:** The transformation from one representative form to another at the same level of abstraction, preserving the same external behavior of the system (functionality and semantics). The term has a broad meaning that recognises the application of similar transformations and recasting techniques (e.g. readjustments, re-shaping). The normalisation of data is an example of Restructuring.
- **Re-engineering:** Also known as Renovation. It is the examination or alteration of the system in order to rebuild it in a new way, and the subsequent implementation of its new form. Re-Engineering includes some form of Inverse Engineering in order to achieve a profile of higher abstraction, followed by some form of Direct Engineering or Restructuring. This could include modifications made in respect to new requirements that must be satisfied by the system.

Providing greater clarity regarding the terminology used in the field of FMs allows for a better connection between the objective sought by the approach and its associated description, whilst also allowing for classification to be made between approaches.

3. FM-CF: A framework for classifying approaches

This section presents the main contribution of this paper, a framework for classifying Feature Model building approaches. Firstly, a methodology for constructing the framework is presented, then a description of the framework and its dimensions.

3.1. Methodology

The methodology we followed for building the framework relies on two principles: (i) it is an inductive activity, which means that we, as researchers, have no preconceived ideas on how to prove or disprove the constructs of the framework; and (ii) it is made by 'constant comparisons', which means that we constantly compare concepts or instances of data that we have named as a specific category with other concepts or instances of data to see if the categories under construction fit. Based on these principles, we carried out the following six steps to build the framework: (1) objective definition, (2) scope definition, (3) knowledge acquisition, (4) conceptualisation, (5) implementation and (6) validation, as illustrated in Fig. 3.

The first step is to define the underlying objective of the development of the FM-CF framework. This must be characterised at the start, including its intended uses and end-users. The second step is to define the scope, which designates the area of interest covered by the framework. The knowledge acquisition step aims at gathering from different sources the knowledge for the framework construction. Then, in the conceptualisation step the knowledge is structured in a conceptual model, which comprises a set of categories to classify work that addresses automatic FM building. The implementation step aims to provide a specification of the conceptualisation (framework) in a formal language, such as RDF[1]. Finally, the validation step guarantees that the resulting conceptualisation corresponds to what it is supposed to represent.

The details of how the first steps were applied to building the framework are presented in the following subsections, and the last step (validation) is detailed in Section 4.

3.1.1. Objective

The main objective of the framework is to provide a generic and conceptual artifact containing knowledge about the automatic FM building process. This framework will provide support for classifying and analysing work that addresses automatic FM building. The framework will be a meta-view for different automatic FM building approaches. This should locate the state of practice in model construction, and help researchers in the development of better approaches to be used in FM construction.

3.1.2. Scope

The framework describes the automatic FM construction process in its high level aspects (i.e. input, processing (tasks), and output). The scope of this study includes all of the categories defined by the experts involved in this study, who have been working on SPL currently. As this study is focused on automatic FM building, manual approaches are not considered. A validation of the framework is addressed later in the paper in Section 4.

3.1.3. Knowledge acquisition

It is well-known that the main problem of FM construction is the bottleneck of information. The problem of the bottleneck of information refers to the difficulty of correctly extracting knowledge from an expert and expressing it as a knowledge base (Feigenbaum, 1984). In order to address these problems, two approaches have been identified in the literature:

1. **Developing methods, methodologies and tools for FM integration.** There are several approaches for utilising different FMs as a baseline, and for reusing them. The process of integration finds commonalities between the source FMs and derives a new FM from them. This process of integration may be done in the following ways:

[1] https://www.w3.org/TR/rdf-concepts/.

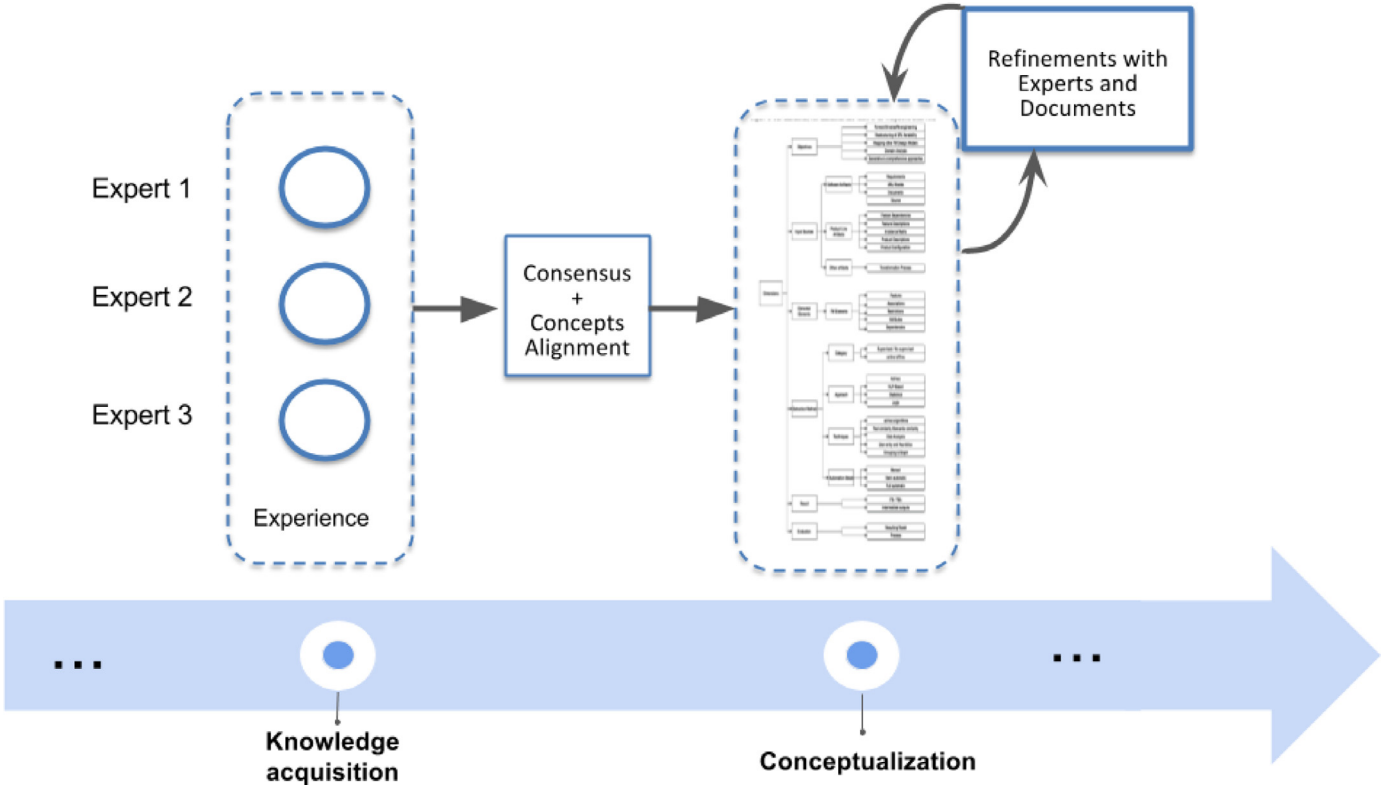

Fig. 4. Knowledge acquisition and conceptualisation steps.

- Merging FMs to create a unique and coherent FM.
- Aligning FMs to establish a correlation between them, and to allow for the reuse of information from one to another.
- Mapping FMs and finding correlations in each one.

Examples of this approach can be found in Mazo (2011) and (Segura et al., 2008).

2. **Developing methods, methodologies and tools for semi-automated FM construction.** At this point, we introduce the concept of Feature Model Learning as a synonym of FM extraction, FM generation or FM acquisition, in order to specify the automated creation or semi-automated creation of FMs, including the automated extraction of features and the relationships between them from any source, as well as their representation or coding in some formal language to facilitate their recovery. An example of this approach can be found in Braganca and Machado (2007) and Weston et al. (2009).

The conceptual experience-based framework presented in this study focuses on the second approach. In terms of knowledge acquisition sources, the framework is based on the experience of experts obtained in building FMs. To the best of our knowledge, there are no frameworks existing in the literature to classify FM construction approaches. Fig. 4 illustrates the knowledge acquisition step and part of the conceptualisation step, starting with the consensus of experts' knowledge of the main concepts related to the FM construction process, the concept alignments, and the conceptualisation with the help of experts and documents. The knowledge and the conceptualisation steps were performed manually, relying essentially on brainstorming sessions.

3.1.4. Conceptualization

In this step, the knowledge is structured in a conceptual model. Over the last few years, some approaches and systems for the automated or semi-automated creation of FMs have been published. For example, some studies propose the use of automated procedures for the extraction of logical dependencies, and a set of product features from existing software artifacts, such as code sources, configuration files, or requirements (Haslinger et al., 2011; Weston et al., 2009). The approaches and previous systems differ from each other due to several factors. These factors were classified by the experts as six main categories for classifying work that addresses automatic FM building, which are called: *dimensions*. The set of dimensions was derived from a generic process definition (i.e. input, tasks, and output) as illustrated in Fig. 5. Each part of the generic process comprises one dimension at least. For each dimension some characteristics were selected that together describe the proposed framework. The framework considers some approaches might select features based on functional requirements and non-functional properties. A functional requirement (FR) defines what a system is supposed to accomplish. Functional requirements are supported by non-functional requirements (NFRs), which impose constraints on design or implementation (such as performance requirements, security, or reliability). They are also known as quality requirements.

3.1.5. Implementation

This step pursues to provide a specification of the conceptualisation (framework) in a formal language. Even though we already encode the framework in RDF, the resulting encode model will be used in further work. An example of future work from the implementation of the framework could be a recommender system. This system could provide a set of recommendations for building or reasoning a feature model from a given set of inputs determined by the problem domain and available software artifacts. For example, if a repository of source code and a list of requirements are available for a specific software product line, the recommender system would establish more suitable data extraction methods or techniques, and then recommend the elements of the feature model that could be derived from the information collected, for example

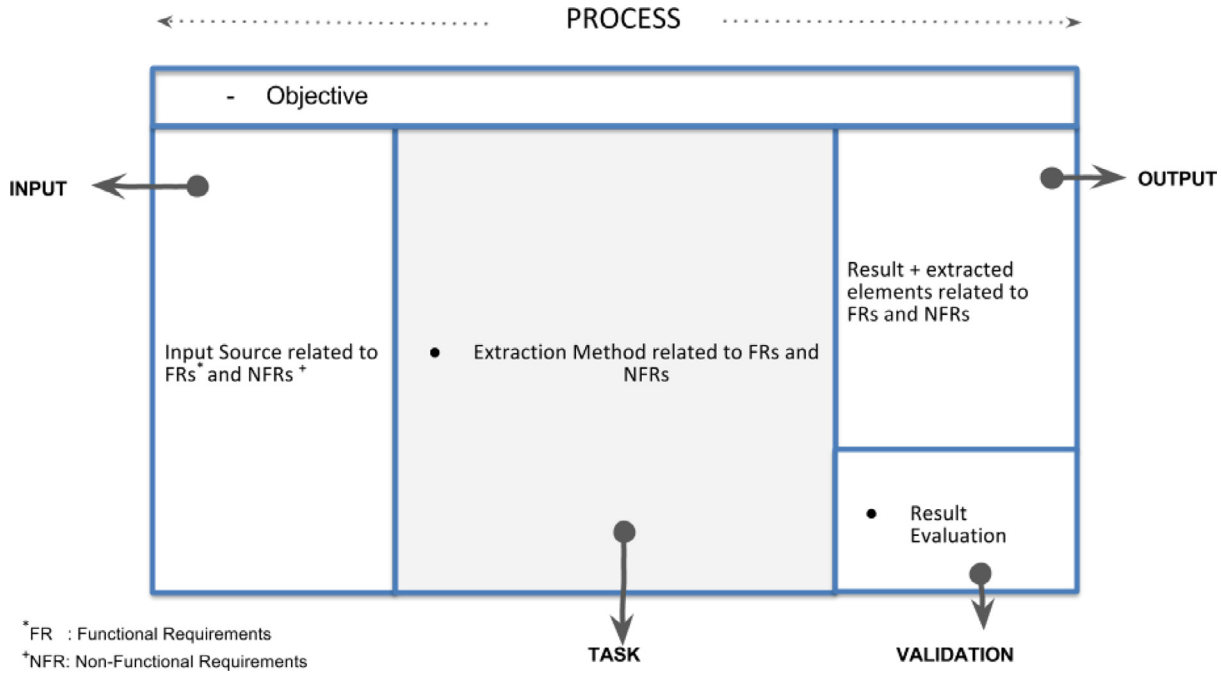

Fig. 5. Knowledge acquisition and conceptualization steps.

features and their attributes, dependency relationships and restrictions between features.

3.1.6. Validation

The validation of the framework is presented in Section 4. This step guarantees that the resulting model correspond to what it is supposed to represent. A collection of keywords extracted from the literature is used to verify that our framework integrates the knowledge that exists in the current set of approaches for constructing FMs in an automatic mode.

The reader will find details on all dimensions covered by the framework in the next section.

3.2. Description of dimensions of FM-CF

Fig. 6 shows the dimensions and sub-dimensions of the framework, as well as its values. The dimensions and sub-dimensions are shown as rectangles, and the values are shown as rectangles with rounded borders.

3.2.1. Proposal

This dimension should provide a response to the question *"What is the objective of the approach?"* In accordance with what existing studies have been reviewed, five objectives can be identified: (i) some engineering approaches (e.g. forward engineering, reverse engineering, re-engineering or FM restructuring), (ii) FM restructuring or merging, (iii) mapping of other FM design models, (iv) domain analysis and (v) generative and comprehensive approaches.

3.2.2. Input

This dimension provides the response to the question "Which data sources initiate element extraction?" The process of feature extraction uses the baselines of previous knowledge, and acquires new knowledge elements from several sources. Input elements vary by type and characteristics. These input sources are categorized into 3 sub-dimensions: software artifacts, product line artifacts, and other artifacts, each with their own respective values, for example:

- *Software artifacts.* Software artifacts are one of many kinds of tangible by-products produced during the development of software related to FRs and NFRs, such as requirements, UML models; NFR models (e.g, i* models and Kaos models), documents and source code. For instance, Alves et al. (2008) have proposed a systematic framework for abstracting existing requirements specifications in an FM. Siegmund et al. (2012) have presented an approach named SPL Conqueror to configure a software product automatically with respect to a set of predefined NFRs. Horcas et al. (2016) defined an Aspect-Oriented SPL to inject customized FQAs into the applications.
- *Product line artifacts.* A product line artifact is a product derived from FMs, such as feature dependencies, feature descriptions, incidence matrix, product descriptions, and product configurations. For example, Lora-Michiels et al. (2010) proposed a method based on statistical algorithms and data mining for building FMs from a set of products presented via one, or several, product tables known as a (multi) Bill of Material (BOM).
- *Other artifacts.* Other artifacts are products not derived from FMs or used as a source for building FMs. Examples of this kind of product are business process models. For instance, Bae and Kang (2007) proposed a method for the generation of an FM from a Business Process Model, focusing exclusively on the domain of business applications.

3.2.3. Tasks

This dimension provides the answer to the question "What extraction method is used to construct an FM?" In order to describe this dimension completely, it is necessary to define four subdimensions as follows:

- *Supervision*. This sub-dimension refers to the type of learning used to train algorithms in those applications, and to whether the approach is supported by some online or offline software application. We have classified applications into two types: online and offline. In terms of learning, we have classified learning as either supervised or unsupervised learning. Supervised learning requires data sets to be provided in order to train the algorithms, and to obtain the desired results. In contrast,

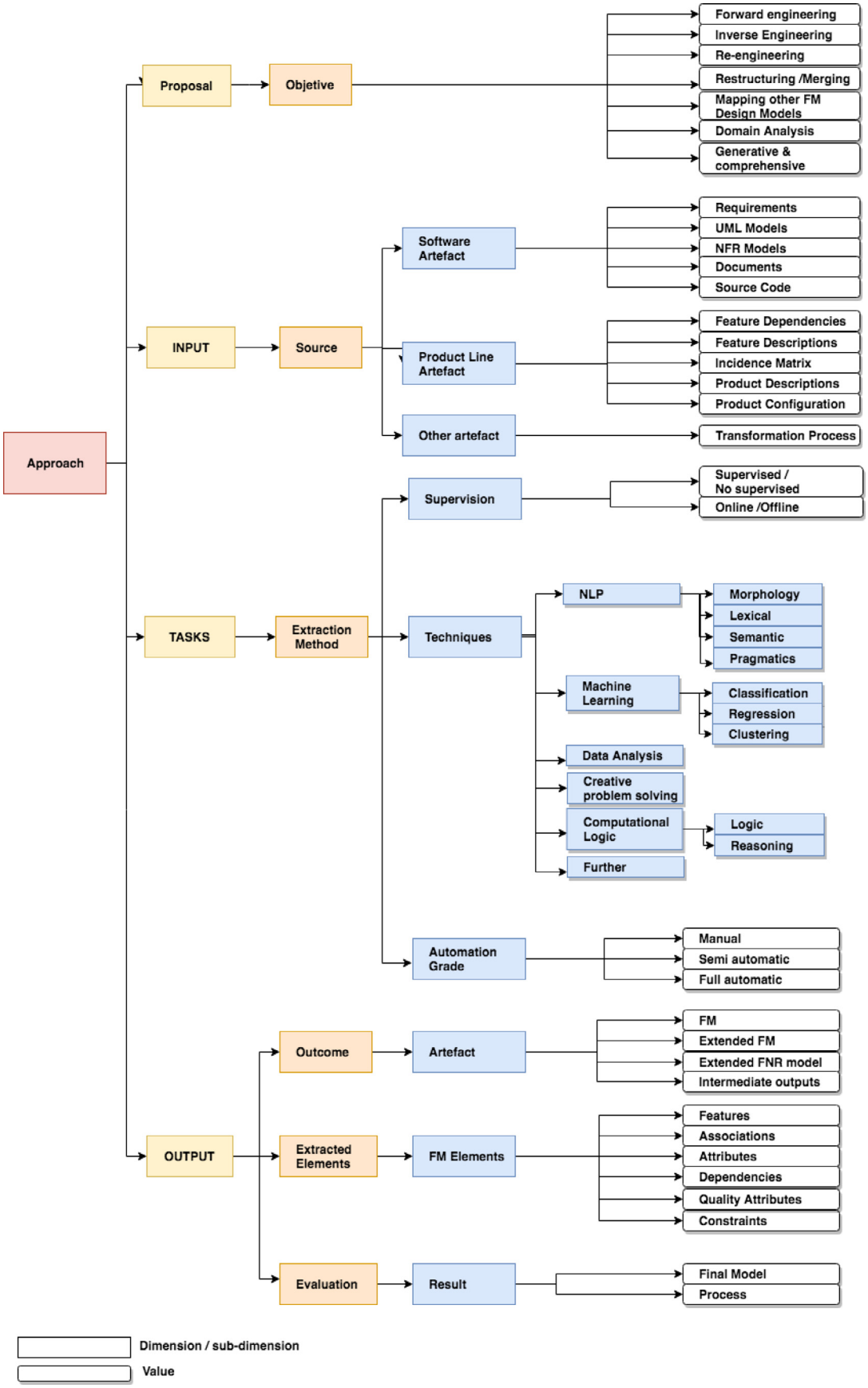

Fig. 6. The dimensions, sub-dimensions and values of the framework.

unsupervised learning does not require data sets upon input. Instead, data is grouped in different classes. Therefore, depending on the type of technique that is being used, one can generally also deduce the type of supervision that is being applied. For example, clustering techniques are an example of unsupervised learning. Bécan et al. (2015a) have proposed a procedure based on ontological aspects that uses heuristics for clustering, and for the consideration of logical, syntactic and semantic relationships between the names of features.

- *Techniques.* This sub-dimension refers to the category of techniques used by the approaches and utilised in diverse contexts. We have defined six sub-categories:
  1. *NLP techniques*. The NLP techniques are categorised into four sub-dimensions: (i) Morphology Techniques, which are focused on Words and forms; (ii) Lexical Techniques, which are focused on Grammar. It means sentences, clauses and phrases; (iii) Semantic Techniques, which are focused on meanings; and (iv) Pragmatics techniques, which are focused on language use. It means, situation and context. For instance, Acher et al. (2012) use a morphology technique (i.e. Tokenization) to extract feature models from product description. Alves et al. (2008) explore the suitability of information retrieval (IR) techniques (e.g. Vector Space Model) for scalable identification of commonalities and variabilities in requirement specifications for software product lines. Wang (2015) extracts semantic information to model the variability and commonality of functional requirements in the domain engineering. Li et al. (2017) present an overview of NLP techniques used to identify features and their relations.
  2. *Machine learning techniques*. The Machine Learning (ML) techniques are categorised into three sub-dimensions:
     (i) Classification techniques. These techniques are used to identify to which of a set of categories a new observation belongs. In terms of FM, classification techniques can be used, for instance, to identify if a domain concept is a feature. For instance, Wang (2016) uses the K-nearest neighbor approach to classify labels and identify features based on word sense. (ii) Regression techniques. These techniques refer to a set of statistical processes for estimating the relationships among variables. For instance, Li et al. (2018) use neural network to extract features from requirements.
     (iii) Clustering techniques. These kind of techniques involve the grouping of data points. Given a set of data points, a clustering algorithm is used to classify each data point into a specific group. For instance, Dumitru et al. (2011) utilise text mining and a novel incremental diffusive clustering algorithm to discover domain-specific features.
  3. *Data analysis techniques*. These techniques examine data sets in order to draw conclusions about the information they contain, increasingly with the aid of specialized systems and software. For instance, Yu et al. (2015) propose an rule-based approach that map goal models to feature models to avoid creating the feature model from scratch.
  4. *Creative problem solving techniques*. These techniques refer to the mental process of searching for an original and previously unknown solution to a problem. For instance, Hamza and Walker (2015) present an approach focuses on the design and implementation of an recommendation system to automatically recommend features for software product lines, the types of these features, and how they could be related to each other, based on NLP techniques and heuristics.
  5. *Computational logic techniques (Techniques of formal reasoning)*. These techniques refer to the use of logic to perform or reason about computation. This sub-dimension are categorised into two sub-dimensions: (i) Logic and (ii) Reasoning. For instance, Bagheri et al. (2010) present an approach to show how the structure and constraints of a feature model can be modelled uniformly through Propositional Logic extended with concrete domains. Czarnecki and Wasowski (2007) have proposed an approach for FM extraction from propositional formulas. This proposal provides an automated procedure for constructing an FM from a logical formula.
  6. *Further techniques*. Other techniques no classified as previous categories.

  An overview of techniques is presented in Table 9.
- *Automation grade*. This sub-dimension refers to the grade to which the FM construction process is automated. For instance, Lora-Michiels et al. (2010) have proposed a fully automated technique to extract FMs from BOMs. However, this study uncovered only the typical restrictions of FMs, such as optional, mandatory, requires, excludes and group cardinalities. This means that whoever uses this study must discover for themselves the other implicit relationships (such as the individual cardinalities in the BOMs), and must make a graphical schematic from the relationships they have uncovered.

3.2.4. Output

This dimension provides an answer to the question "Which outcomes deliver the proposed approach?" This dimension defines three sub-dimensions as follows:

- *Outcome.* This dimension describes the type of resulting artifact that the approach generates. In some cases they will be FMs, in others they will be intermediate outputs that precede the creation of an FM. Some approaches have proposed an extended FM in order to support NFRs. For instance: The proposal of Lora-Michiels et al. (2010) delivered a result of hierarchical relationships (optional and obligatory), transverse relationships (required and exclusive) and relationships of cluster cardinalities. Jarzabek et al. (2006) have proposed an integrated modelling framework called a feature-softgoal interdependency graph (F-SIG). In order to support the concept of quality in SPLs, the authors extended the Feature-Oriented Domain Analysis (FODA) method with a goal-oriented approach.
- *Extracted elements.* This part responds to the question *"What type of conceptual structure is extracted?"*. The extracted elements are mainly ones that compose an FM such as features, associations, restrictions and attributes. FM elements refer to the those that an FM is comprised of, such as features, associations, restrictions, attributes, quality attributes, and dependencies. For instance, Dumitru et al. (2011) identified only features for a specific domain. Martinez et al. (2014) proposed a new representation of FMs, in the form of graphs depicting the proximity of relationships within features which facilitate the configuration process and allow for the expression of proximity relationships within features.
- *Evaluation.* This dimension responds to the question "What elements are evaluated by the proposed approach?" The focus of the evaluation of results is defined as: the evaluation of the final product, or the evaluation of the process, techniques, or algorithm used to build the FM. For example: Weston et al. (2009) proposed an approach and a tool called ArborCraft to create FMs automatically from requirements documents. The resulting FMs are evaluated by experts of the domain corresponding to the resulting FM.

4. Validation of the framework

This section presents the validation step, which assesses the extent to which the resulting framework corresponds to what it is

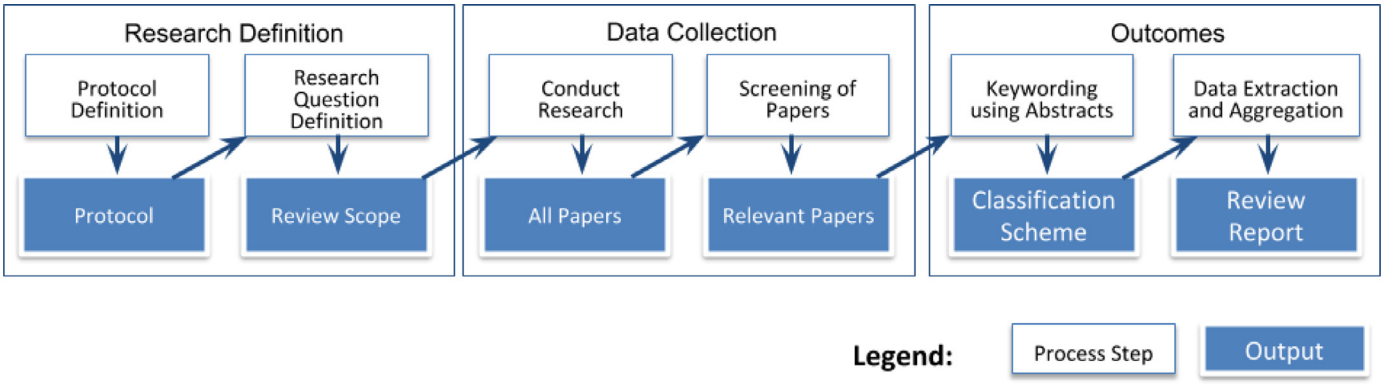

Fig. 7. The literature review process.

supposed to represent. A literature review was conducted to gather an up to the minute view of FM refactoring (and similar work), and to obtain some input to validate the framework. A set of keywords was extracted from the literature review in order to evidence the completeness of the framework. The keywords were extracted from the abstracts of the primary studies, and used to verify that our framework integrate the knowledge that exist in the current set of approaches to constructing FMs in an automatic mode. Additionally, a set of selected approaches to building FMs by automatic means was classified and compared to illustrate the usefulness of the framework.

4.1. Literature review

This paper addresses the problem of the classification of approaches to building FMs by automatic means. In particular, we are interested in exploring how the automatic building of FMs is supported by existing approaches. A literature review is driven to synthesise the current state of research reported in the literature on the automated support for building FMs. To this aim, a systematic literature review was conducted following the guidelines and the process proposed in Kitchenham et al. (2009). The process comprises three major phases: Research Definition, Data Collection, and Outcomes, as illustrated in Fig. 7.

The *Research Definition phase* was focused on specifying the review study protocol, which describes some activities for gathering available evidence. It provides a definition of research questions, search strategy for gathering relevant studies, criteria for primary studies, inclusion and exclusion, the screening of papers' process, and data extraction analysis and synthesis.

The *Data Collection phase* was focused on the application of the research protocol for searching primary studies, extracting data, and synthesizing relevant knowledge related to automated FM building approaches.

The *Outcomes phase* reports the review study findings. In this phase, all information is consolidated and presented as a report.

As a summary, the basic steps of the process are: definition of the protocol and the research questions, execution of a search of relevant articles, filtration of articles, search of key concepts used in summaries and, finally, the extraction of data and the processes of aggregate of the extracted data. Each process produces an output. The final output of the process is the review report.

4.1.1. First phase: Research definition

This section presents the first phase of the literature study process, in which the protocol and research questions are defined.

(a) *Protocol*. The purpose of the protocol was to manage the research objectives and to define clearly how they can be achieved by defining the research questions, and planning how the sources and selected studies would be used to respond to these questions. We have adopted this task from literature review guidelines (Kitchenham et al., 2009). The first activity in this study was to develop a protocol, i.e. a strategy defining the basic study procedure.

(b) *Research questions*. The objectives of literature review are reflected by the Research Questions (RQ). The research questions were framed by four criteria:
(i) Population: Scientific literature that shows approaches to building FMs in an automatic way. *(ii) Study design*: Describes the issue with a focus on the study. In this case, it is defined as automatic approaches to building FMs. *(iii) Intervention*: Any study that shows some degree of automation in building FMs. *(iv) Outcomes*: Quantity and type of related evidence regarding automation in building FMs. Table 2 shows the research questions, which correspond to this study's objective and are in accordance with the standards indicated by Petersen et al. (2008). In order to answer the research questions, data was collected from the literature. This task, in general terms, involved defining a search strategy, identifying data sources, selecting studies, and analysing and synthesising data.

4.1.2. Second phase: Data collection

In order to answer the research questions, data were collected from the research literature. These activities involved conducting research and the screening of papers. To conduct this research we developed a search strategy, and identified data sources.

(a) **Search strategy.** The search strategy was developed through the review of the data needed to respond to each research question. Primary studies were identified using search strings on scientific databases, or searching manually through conference proceedings or publications in specialised journals. An initial set of keywords were refined after a preliminary search that retrieved 5 many results of little relevance. Several combinations of search items were used until an appropriate set of keywords was reached. The search chain consisted of Boolean expressions composed of key words that described the FM construction process. Table 3 presents the composition of the search string applied. Terms within a row are 5 connected by the operation "OR", while the different parts of the search string are connected by the operator "AND" in order to improve the results' completeness.

(b) **Data sources.** The search included important journals and related conferences, with the research topics related to automatic FM building. The search was restricted to studies published between the years 2005 and 2018 in order to achieve wide coverage of the study area. The initial step was conducting a search,

Table 2
Definition of the research questions (Scope of the literature review).

ID	Questions	Objectives
RQ1	Where were the studies published?	To determine the place of a study's publication (e.g. journals, conferences, others). This makes it possible to know where the publications are concentrated and, using that information, determine the maturity level of their results.
RQ2	What are the terms used to represent the automatic FM construction process?	Identifying the current terminology used to denote an FM construction process. This makes it possible to know the terms related to an FM construction process.
RQ3	What type of research study was conducted?	To determine what type of study was performed in order to show an approach to building FMs. For example, Experiment, Case Study, Project Description, among others. This allows the published studies to be categorised.

Table 3
Table of search terms.

No.	Terms
1	Software Product Line, SPL, Software Product Family, Domain Engineering, Domain Analysis, Product Line Engineering, SPL, SPLE
2	Variability Model, Feature Model, Variability Modelling, Feature Modelling, FM
3	Building, Extraction, Construction, Derivation, Merging, Synthesis, Reverse Engineering, Mining, Weaving

using the terms previously described, via digital library search engines. Publications obtained from SCOPUS, and ACM Digital Library were considered. The second step was the search of international review journals with articles considered relevant, which were published by ISI-WOS as they are considered high-level publications (McGregor et al., 2010). Conference proceedings were also searched. In the event that a conference maintained its proceedings on a published website, this was also accessed. The search for conference proceedings and journals produced many results that had already been obtained through the digital library search. In this case, the last results were discarded and only the first were considered, given that these results had already been included in the final list. After the search was executed for conferences and journals using digital libraries and proceedings, it became possible to identify which known publications are commonly referenced by other studies in this area, such as technical reports and theses, and had not been included in the resulting final list.

(c) **Screening of papers.** The screening of papers involved selecting studies to analyse, and data analysis and synthesis.

(a) *Study selection*. The set of search strings was applied to the search engines, specifically to those mentioned in the previous section. Criteria of inclusion and exclusion were used to filter studies that were not relevant for answering the research questions. Inclusion criteria were used to select all the studies during the search stage. Afterwards, the criteria of exclusion were mainly applied to the titles of studies, and then to the summaries and conclusions. Regarding the inclusion criteria, the studies were only considered if they included an approach to building FMs. Studies were excluded if they met the following criteria:

- *Research focus unrelated to feature modelling*. The article is outside the field of feature modelling.
- *Research focus related to feature modelling but with insufficient information regarding the construction process*. The construction process is not part of the contribution of the article, and the terms are only mentioned in the introductory sentence of the summary.
- *Research focus related to FM but not related to the construction process*. The article is within the field of FMs, but the approach to building FMs is not part of the contribution of the article.
- *Duplicate studies*. When the same study was published in different articles, only the most recent was included.
- *The study had already been included from another source.*
- *The article was in a language other than English*. All studies not written in English were filtered.
- *Technical reports, both short and position papers and theses*. Given that they neither ensure an in-depth peer review nor are widely validated by the scientific community, technical reports, short papers, position papers and theses were excluded. In the case of theses and technical reports, it was assumed that, if they offered some contributions, these originated from other publications, explaining why they were excluded from this study.

The selection of studies involved a process of analysis composed of three filters which were intended to select the most appropriate results, as the probability of retrieving inappropriate studies could have been high. Table 4 describes what was considered in each filter. Additionally, the figure presents the number of items that were obtained after the application of each filter.

(b) *Data extraction*. The method for data extraction was designed to extract all the information needed to answer the research questions. The following information was extracted from each article: *authors; source; conference/journal; publication year; summary; a brief opinion regarding its strengths and weaknesses, and the study's objectives*. It was decided that when several studies were reported in the same paper, each relevant study would be treated separately. However, this situation did not occur. The search for key concepts is one way of reducing the time necessary to develop a classification of approaches, and ensure that the search considers all articles. This search was conducted in two steps. First, the reviewers read the article summaries. Then, they looked for keywords and concepts that reflected the contribution of the article, and subsequently identified the research context. Once the above had been completed, a set of keywords was obtained from different summaries, which were combined in order to achieve a high level of understanding with respect to the nature, and contribution, of the article.

4.1.3. Third phase: Outcomes

This section describes the classification framework (or schema) used, and the results of data extraction. Once the framework was defined, the relevant studies were ordered according to the framework. The output of this stage is the study map, which is presented at the end of this section together with the discussion. We

Table 4
Stages of the article selection process.

Filter	Activities	#papers
1	To identify relevant studies using defined search items and refine the search by applying the inclusion criteria to the study title.	410
2	To exclude studies on the basis of exclusion criteria applied to abstracts and conclusions	189
3	To obtain primary studies for performing a critical appraisal	31

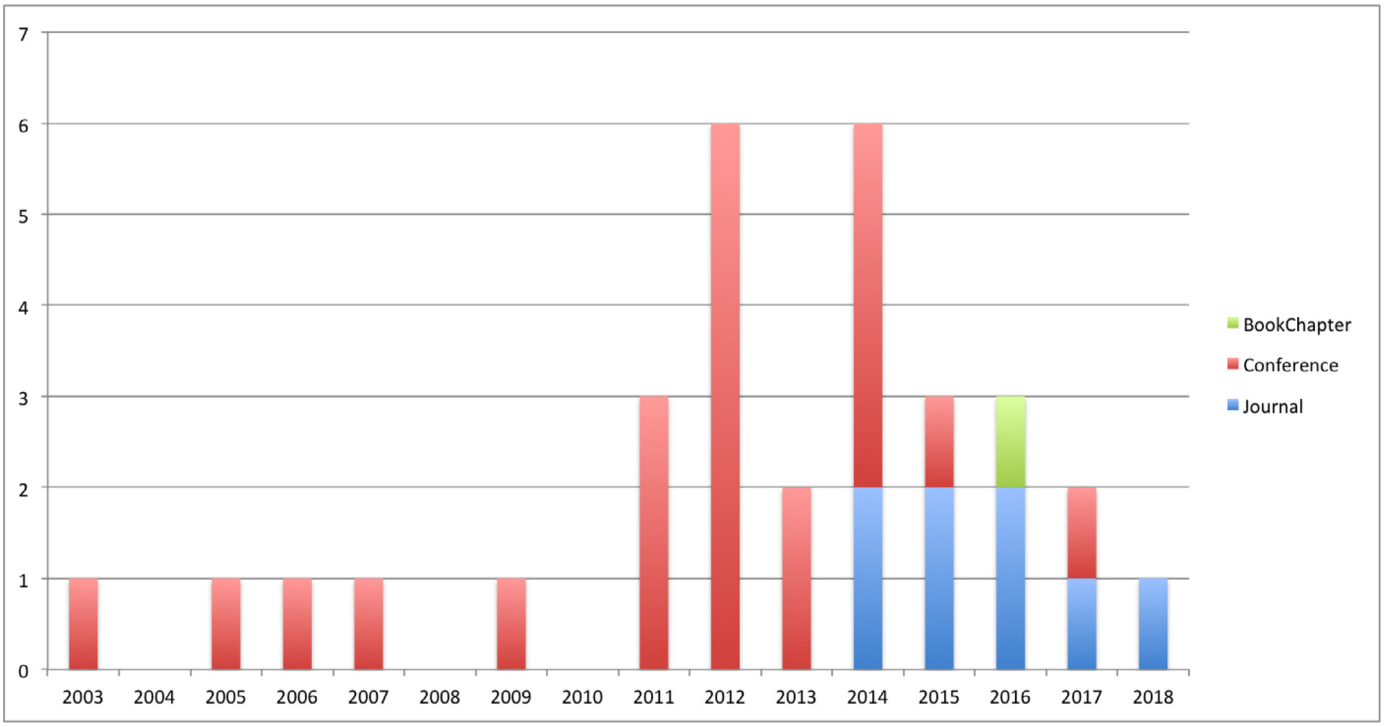

Fig. 8. Distribution by years.

categorise studies in facets. In our case, we defined two facets. One facet examined the type of research, and the other arranged the topic in terms of the research questions. The search for key concepts is a strategy to ensure that the searching considers all articles. This search was conducted in two steps. First, the reviewers read the article summaries. Then they looked for keywords and concepts that reflected the contribution of the article, and subsequently identified the research context. Once the above had been completed a set of key words was obtained from different summaries, which were combined in order to achieve a high level of understanding with respect to the nature and contribution of the article, in addition to its reported approach to building FMs.

4.2. Analysis of the literature review results

In this section, we present the elements that were found through the analysis of the selected articles with the intention of answering the research questions. The evidence obtained from the data extraction process is highlighted, as are the particularities found. Prior to the above, a classification of the collected studies analysed is included. The final list of primary studies used in this study is shown in Table 8.

(a) **Classification of studies**
A classification of the studies obtained is presented below. The classification process was intended to determine the distribution of articles by year. Fig. 8 presents the distribution of articles by year. The years 2012 and 2014 comprise the greatest amount of research work on building FMs by automatic means.

Table 5
Terms used to represent an automatic feature model construction process.

Terms	
Addressing (1)	Analyzing (1)
Augmenting (1)	Automating (1)
Constructing/construction (3)	Extraction (6)
Generating/Generation (4)	Identification (1)
Intersection (1)	Managing (1)
Mining (2)	Modelling (1)
Re-factoring(1)	Reverse Engineering (1)
Selection (2)	Synthesis (4)

(b) **Research questions**
The answers to the research questions are presented as follows.

- **RQ1** - Where were the research studies published?
  The 31 publications analysed between the years 2002 and 2018, were distributed as follows: 8 papers (26%) were published in journals and 22 papers were published in conferences (71%).
- **RQ2** - What are the most frequent terms used to represent an automatic feature model construction process?
  The terms most frequently used to represent an automatic feature model construction process were derived from the title of the 31 analysed articles, as they described the main objective of the proposed approach. The list of terms ordered in alphabetical order are shown in the Table 5.
  The terms most frequently used in the titles of analysed studies are: *Extraction, Generation/Generating* and *Synthesis*. In addition, a list of terms was extracted from the abstracts

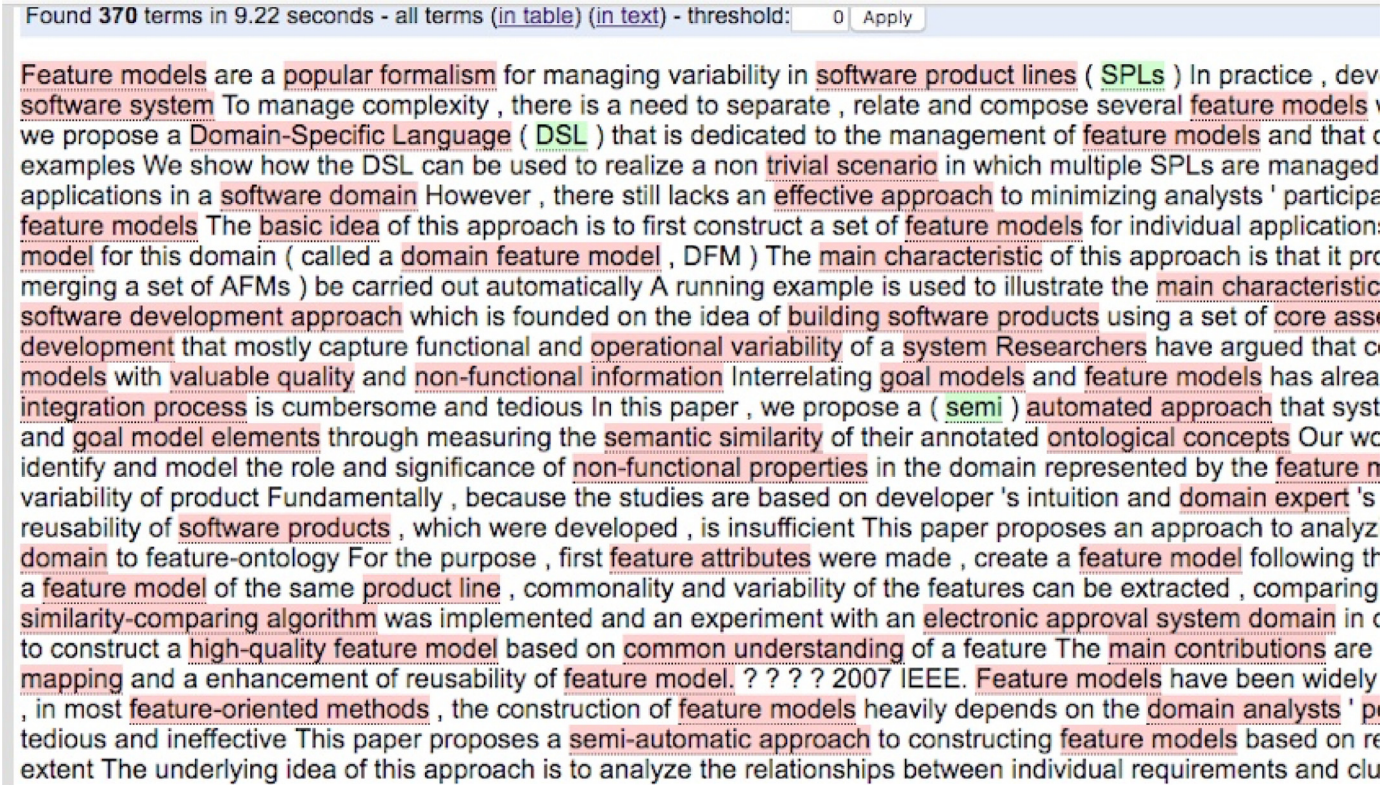

Fig. 9. A screenshot of the terms obtained by using the web service Termine.

Table 6
The Top-30 terms extracted from the abstracts of the analysed articles.

Terms (score)		
feature model (114.59)	variability model (6.0)	domain feature model (3.16)
product line (51.91)	product configuration (6.0)	feature tree (3.87)
software product line (49.77)	variant feature (5.0)	customer preference information (3.16)
feature modelling (15.8)	reverse engineering (5.0)	multi-level feature tree (3.16)
domain analysis (12.75)	product variant (5.0)	product configuration process (3.16)
software product line engineering (8.0)	formal concept analysis (4.75)	domain expert (3.0)
goal model (7.0)	domain analysis process (4.75)	feature group (3.0)
feature model synthesis (6.33)	feature identification (4.0)	feature dependency (3.0)
software system (6.0)	software reuse (4.0)	natural language processing (3.0)
core assets (6.0)	software development (4.0)	feature diagram (3.0)

of the analysed articles. To do this, we used a web service called Termine, as it is freely available from the academic domain provided by the University of Manchester[2]. A total of 370 relevant terms was obtained, as shown in Fig. 9. A top 30 list of terms is shown as an example in Table 6 ordered by relevance (score) in descending order. Termine is based on the C-/NC-value method (Frantzi et al., 2000), which is an efficient domain-independent multi-word term recognition method which combines linguistic and statistical knowledge. This method stipulates that using more statistical information than the pure frequency of occurrence of candidate terms, improves the precision of the extracted multi-words terms and that using information from the context of the candidate terms, improve their distribution in the extracted list, i.e. real terms tend to appear closer to the top, while non-terms concentrate closer to the bottom of the list.

- **RQ3** - What type of research study was conducted?
  The classification of research approaches proposed by Wieringa et al. (2005) was used to relate the type of research approach of a particular study. This classification was used with the intention of determining the type of research proposed. This classification identified six types of articles, defined as follows:
  (a) **Research evaluation**. Techniques are implemented in practice and an evaluation is conducted. This means that the way in which the technique is implemented (implementation of the solution) is presented, as are the consequences of the implementation in terms of its benefits and inconveniences (evaluation of the implementation). This also includes identifying problems within industry.
  (b) **Proposed solution**. A solution is proposed for the problem. The solution could be a new technique, or the extension of an already existing one. The potential benefits and the applicability of the solution are shown through a small example, or a short but effective argument.
  (c) **Research validation**. The techniques researched are recent and have still not been implemented in practice. The techniques used are, for example, experiments or, in other words, work performed in a laboratory.

[2] http://www.nactem.ac.uk/software/termine/.

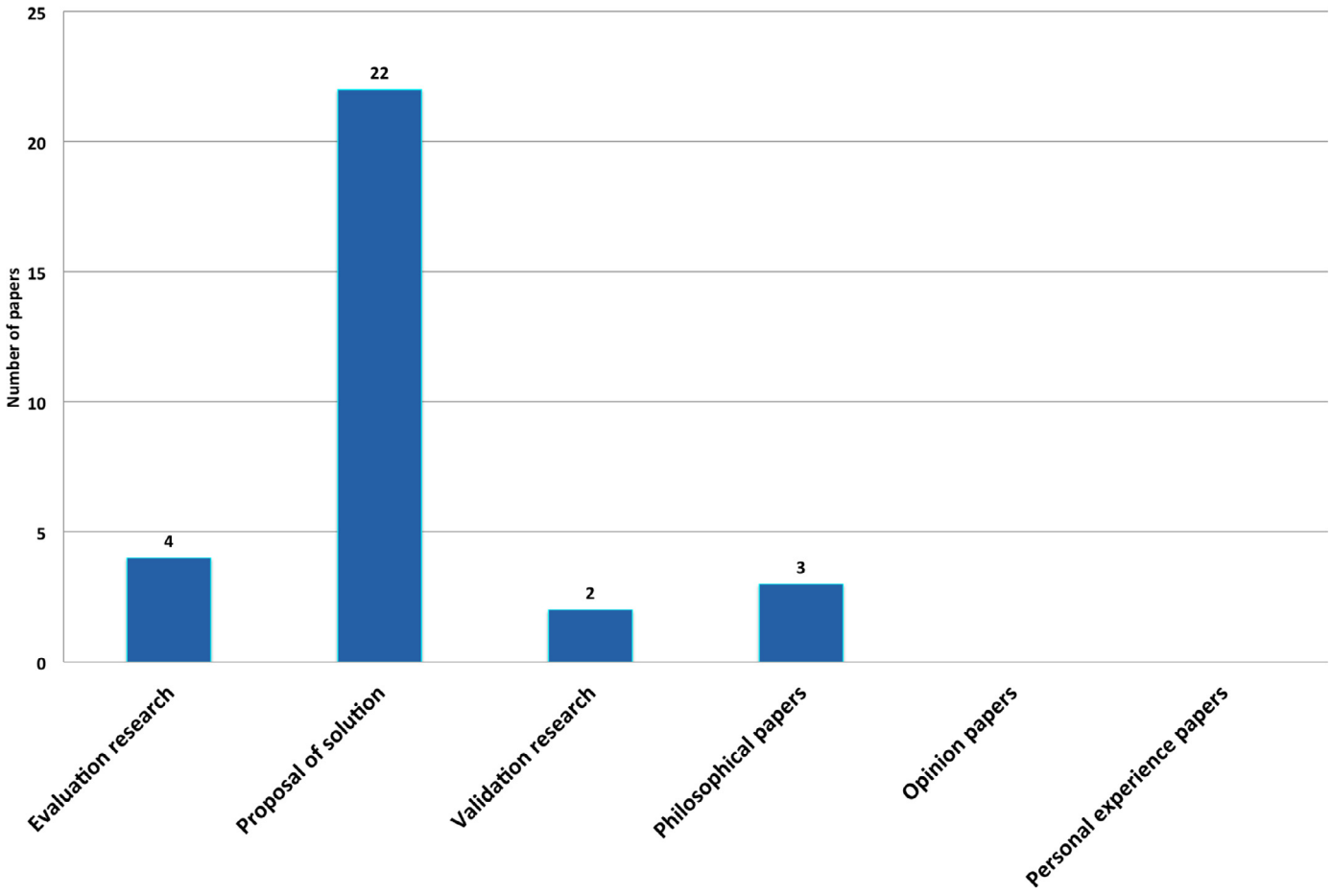

Fig. 10. Classification of research approaches.

(d) **Philosophical study**. These articles establish a new way of analysing current topics of interest by structuring the area in a taxonomic way, or through a conceptual framework.
(e) **Opinion articles**. These articles express the personal opinion of someone with respect as to whether a certain technique is good or bad, or in regards to how it should be performed. These opinions are not based on related studies or research methodologies.
(f) **Personal experience study**. Articles based on personal experience that explain what, and how, something was done in practice.

Fig. 10 presents a classification of research approaches according to the framework proposed by Wieringa et al. (2005). The category of "Proposal of Solution" comprises the greatest number of articles, followed by "Evaluation Research".

4.3. Validation strategies

This sections presents two strategies that validate our framework. Firstly, we verified that our framework integrated the knowledge that exists in the current set of approaches to constructing FMs in an automatic mode. To do this, we used the set of keywords extracted from the literature review in order to evidence the completeness of the framework. Secondly, the set of analyzed articles were evaluated using the proposed framework in order to compare and classify them to illustrate the usefulness of the framework.

Table 7
Table of the precision metric.

Keyword	Precision
50	81%
100	83%
200	83%
300	79%

4.3.1. Using terms to evidence that the framework integrates relevant knowledge

To determine how well our framework integrates relevant knowledge related to the construction of FMs by automatic means, we used the set of keywords extracted from the analysed articles in order to compare it against the set of dimensions of the framework. Each keyword was analysed to determine under what dimension of the framework it might be classified. Then we used the Precision metric, which is defined as a useful measure of relevancy. Precision (P) is defined as the number of true positives (T_p), over the number of true positives plus the number of false positives (F_p).

$$P = \frac{T_p}{(T_p + F_p)}$$

We used several sets of keywords (i.e. 50, 100, 200 and 300), from the total of 370 keywords extracted from the abstracts of the articles, to calculate the Precision metric. Table 7 presents the results obtained. No filters were used on the set of keywords. Some keywords represent general concepts such as "Feature Modelling"

Table 8
Overview of all reviewed papers, ordered by year of appearance.

ID	Title	Year	Source
P01	Employing fuzzy logic in feature diagrams to model variability in software product-lines	2003	(Robak and Pieczynski, 2003)
P02	An approach to constructing feature models based on requirements clustering	2005	(Chen et al., 2005)
P03	Refactoring product lines	2006	(Alves et al., 2006)
P04	An approach to analysing commonality and variability of features using ontology in a software product line engineering	2007	(Lee et al., 2007)
P05	A use case based approach to feature models' construction	2009	(Wang et al., 2009)
P06	A domain-specific language for managing feature models	2011	(Acher et al., 2011)
P07	Extraction of feature models from formal contexts	2011	(Ryssel et al., 2011)
P08	Reverse engineering feature models	2011	(She et al., 2011)
P09	Domain specific feature modelling for software product lines	2012	(Hofman et al., 2012)
P10	Efficient synthesis of feature models	2012	(Andersen et al., 2012)
P11	Generating feature model from creative requirements using model driven design	2012	(Wanderley et al., 2012)
P12	Intersection of feature models	2012	(van den Broek, 2012)
P13	On extracting feature models from product descriptions	2012	(Acher et al., 2012)
P14	Usage scenarios for feature model synthesis	2012	(She et al., 2012)
P15	Feature model extraction from large collections of informal product descriptions	2013	(Davril et al., 2013)
P16	Mining complex feature correlations from software product line configurations	2013	(Zhang and Becker, 2013)
P17	Addressing non-functional properties in feature models: A goal-oriented approach	2014	(Noorian et al., 2014)
P18	Automated feature identification in web applications	2014	(Marciuska et al., 2014)
P19	Automated generation of computationally hard feature models using evolutionary algorithms	2014	(Segura et al., 2014)
P20	Automating variability model inference for component-based language implementation	2014	(Vacchi et al., 2014)
P21	Generating Feature Models from Requirements: Structural vs. Functional Perspectives	2014	(Itzik and Reinhartz-Berger, 2014)
P22	WebFML: Synthesising feature models everywhere	2014	(Bécan et al., 2014)
P23	Extracting variability-safe feature models from source code dependencies in system variants	2015	(Assunção et al., 2015)
P24	Heterogeneous feature models and feature selection applied to bearing fault diagnosis	2015	(Rauber et al., 2015)
P25	Synthesis of attributed feature models from product descriptions	2015	(Bécan et al., 2015)
P26	Breathing ontological knowledge into feature model synthesis: an empirical study	2016	(Bécan et al., 2016)
P27	Mining feature models from functional requirements	2016	(Mefteh et al., 2016)
P28	Software Features Extraction From Object-Oriented Source Code Using an Overlapping Clustering Approach	2016	(Araar and Seridi, 2016)
P29	Augmenting feature model through customer preference mining by hybrid sentiment analysis	2017	(Zhou et al., 2017)
P30	Automated Extraction of Feature Models from Android Based Portable Devices	2017	(Yildirim and Sözer, 2017)
P31	Feature selection optimisation based on atomic set and genetic algorithm in software product line	2018	(Zhan et al., 2018)

or "Software Product Line Engineering". These types of concepts could not be matched with the set of dimensions as they represent a general category related to all the approaches. For this reason, we cannot expect to get a 100% precision rate. However, since the precision metric results are over 79% we believe most of the keywords obtained from the abstracts are matched in our framework.

4.3.2. Analysed articles were evaluated using the proposed framework

The search protocol used in the literature review (Kitchenham et al., 2009) generated a list of approaches to be evaluated. The conceptual framework was used to classify the approaches found in the literature review as shown in Fig. 11. The dimensions, sub-dimensions, and some possible values for

	PROPOSAL	INPUT				TASKS			OUTPUT		
	Objective	Source				Extraction Method			Outcome	Extracted Elements	Evaluation
		Software Artifact	Product Line Artifact	Other Artifact	Supervision	Category	Techniques	Automation Grade	Artefact	FM elements	Result
P1	Restructuring			Customer profiles		Logic-based	Computational Logic / Fuzzy Logic	manual	Extended FM	Features (variant)	FM
P2	Forward	Requirements			Supervised	Machine Learning	Machine Learning /Clustering	semi	FM	Features	FM
P3	Restructuring		FM			Ad-hoc	Creative problem solving / Heuristics	manual		Features	FM
P4	Mapping			ontology	Supervised	NPL-based	NLP/Semantic Model	semi	Meta FM		FM
P5	Forward	UML model	individual FMs			Ad-hoc	Further	semi	FM		FM
P6	Domain Analysis	Language				Ad-hoc	Further	manual	Intermediate FM		Process (Rationale)
P7	Forward		incidence matrix			NPL-based	NLP/ FCA	semi	FM		FM
P8	Reverse		FM			Ad-hoc	Creative problem solving /Heuristics	automatic		Features, Associations	Process
P9	Forward	Documents				Ad-hoc	Creative problem solving / Intuitive Judgment	semi		Features	Process
P10	Forward		Constraints			Ad-hoc	Computational Logic / Propositional logic	automatic		Associations	FM
P11	Domain Analysis	Requirements				Ad-hoc	Creative problem solving /Heuristics	semi		FM	FM
P12	Merge		FM			Ad-hoc	Creative problem solving /Heuristics	automatic	FM		Process
P13	Domain Analysis		Product descriptions			Ad-hoc	Creative problem solving / Rule of thumbs	automatic	FM		Process
P14	Forward		Features			Ad-hoc	Creative problem solving /Heuristics	semi	FM		Process
P15	Forward		Product descriptions			Ad-hoc	Creative problem solving /Heuristics	semi	FM		Proces
P16	Forward		Product configuration			Ad-hoc	Data Analysis / Association rule learning	manual		Features (correlations)	Process
P17	Merge		Goal model	Ontology	Supervised	NLP-based	NLP / semantic similarity	semi	Extended FM		Process
P18	Forward	Web application				Ad-hoc	Creative problem solving /Heuristics	semi		Features	Process
P19	Comprehensive		FM			Ad-hoc	Further / Evolutionary algorithm	semi	FM		Process
P20	Forward	Languaje components				Ad-hoc	Creative problem solving /Heuristics	semi		Features, Associations	
P21	Forward	Requirements			Supervised	NLP-Based	NLP / Semantic (ontology)	automatic	FM		Process
P22	Forward		Dependencies, matrix			Logic	Machine Learning / Clustering heuristics	automatic	FM		Process
P23	Reverse	Source Code				Ad-hoc	Creative problem solving /Heuristics	automatic	FM		Process
P24	Forward		Signal documents			Statistical	Machine Learning /classifier, vector machine	semi		Features	Process
P25	Forward		Product descriptions			Ad-hoc	Creative problem solving /Heuristics	semi	FM		FM
P26	Merge		Dependencies			Statistical	Machine Learning /classifier, heuristics	semi		Associations	Process
P27	Forward	Requirements			Supervised	NLP-based	NLP /FCA	automatic	FM		Process
P28	Comprehensive	Source Code			Supervised	NPL-based	Machine Learning / clustering	automatic		Features	Process
P29	Merge		Customer information		Supervised	NP-Based	NLP /Sentiment Analysis	automatic	FM		Process
P30	Domain Analysis		Product specification			Ad-hoc	Creative problem solving /Heuristics	automatic	FM		Process
P31	Comprehensive		Features			Ad-hoc	Genetic algorithm	automatic		Features	Process

Fig. 11. Classification of analysed studies using the framework.

Table 9
Overview of techniques.

Techniques		
NLP	Morphology	Tokenization
	Lexical	Part of Speech (POS)
		Lemmatization
		Stemming
		C-Value/NC-Value
		TF-IDF
		RAI
	Semantic	Latent Semantic Analysis (LSA)
		Vector Space Model (VSM)
		Contrastive Analysis
		Syntactical Heuristics
		Semantic Model
		Latent Dirichlet Allocation
		Semantic Role Labelling
		Formal Concept Analysis (FCA)
	Pragmatics	Abduction
		discourse analysis
Machine Learning	Classification	Support Vector Machine
		Discriminant Analysis
		Naive Bayes
		Nearest Neighbor
		Trees
		Logistic Regression
	Regression	Linear Regression
		GLM
		SVR
		GPR
		Ensemble Methods
		Decision Trees
	Clustering	K-means
		K-Medoids
		SPK-Means
		Hidden Markov Model
		Gaussian Mixture
		Fuzzy C-Means
		Self Organising Map Clustering
		Neural Network
		Hierarchical Agglomerative Clustering
		Incremental Diffusive Clustering
Data Analysis		Tracking patterns
		Anomaly or outlier detection
		Associate rule learning
		Sequential patterns
		Range Ranking
		Rule-based classifier
Creative problem solving		Heuristics
		intuitive judgment
		Stereotyping
		Educated guess
		Rule of thumb
		Profiling
		Common sense
		Greedy Algorithm
Computational Logic	Logic	Propositional logic
		classical predicate logic
	Reasoning	non-monotonic and fuzzy logic
		causal reasoning
		case-based reasoning
		reasoning by analogy
Further		Edmond SFI Algorithm
		Maximum Entropy Method
		Sequential Patterns
		Evolutionary Algorithm
		Other

those dimensions and sub-dimensions are presented in Section 3. In Fig. 6, the columns show the dimensions and sub-dimensions of the FM-CF, and the rows present the proposals that have been selected for examination in this study. As is shown in the table, some columns that represent dimensions (e.g. Input sources, Extracted elements, and Method) of the framework are divided into smaller columns, indicating that they are sub-dimensions, such as software devices, SPL devices, others, FM elements, Category, Approach, Techniques, and Automation grade. Each cell in the intersection of a column (dimension), and a row (a proposal), indicates the value(s) of the dimension for the proposal. In Fig. 11 it was discerned that the majority of the proposed approaches fell

into the categories of Forward engineering. In terms of input sources, it was found that the source code, the requirements, FMs, features, and text documents corresponded to the most used input sources. The extracted elements were generally FMs, features and certain conditions found in the FMs, such as inconsistencies and restrictions. In general, the proposals defined their own approaches (ad-hoc), and some proposals used approaches based on natural language (NLP-based). In terms of the results, the focus was generally on producing FMs, or on identifying features. The evaluation was more aligned with evaluating the ad-hoc approach (e.g. algorithm, process) than evaluating the result (e.g. resulting FM or feature validity).

5. Discussion

The objective of the conceptual framework presented in this article is the classification of proposals related to the construction of FMs by (semi-)automated means.

From a practical perspective, the framework can be used by engineers and researchers to choose, for instance, the appropriate elements with which to construct FMs for their own domains. Thus, the use of the framework provides the following advantages:

1. *Helping to understand and clarify the diverse existing approaches in the literature.* In effect, the variety of the published proposals in the literature can be classified under a common framework.
2. *Situating the state of practice in model construction.* It is possible to determine with high precision what have been the contributions of the proposals, and to determine their principal focus as well as their results.
3. *Assisting researchers in the development of better approaches for the construction of FMs.* Given that one can determine the empirical evidence of the results and the advances in the automation process of the diverse proposals, it is possible to define new approaches taking into consideration the advances achieved at the present moment, and to assume, as a challenge, the creation of new techniques that improve the current state of development.
4. *Allowing for the identification of research areas that have not been addressed.* The act of providing a common framework for the classification of the diverse proposals that have been published allows for the clear identification of those areas and aspects that the scientific community has not presently considered.
5. *Offering a guide for the identification of future research challenges.* The use of the conceptual framework permits the derivation of a research agenda in which research challenges can be identified, along with their priority level and their likely impact, in the field of SPLs and feature modelling.

Even though the conceptual framework accounts for the classification of FM construction proposals by (semi-)automated means, there exist two aspects which threaten the validity of the proposed conceptual framework.

First, the lack of standardised terminology in the field of feature modelling for referring to the construction processes (e.g., inverse engineering, synthesis, localisation, and mining are, among others, essentially synonyms) along with the constant evolution of terms and the use of new concepts stemming from related fields (such as Requirements engineering; Information retrieval; Web semantic and Artificial intelligence) making it possible that these new concepts could fail to be considered by the proposed framework in its current state. Second, the constant research and search for new methods and techniques on the part of researchers in the field could result in the extension of the current defined values of each dimension, as well as the possible inclusion of current values in new sub-dimensions. However, the summary of approaches presented in Table 8 offers evidence that it is possible to develop detailed descriptions of the currently published approaches for FM construction.

6. Threats to validity

We considered the possible threats to validity and took measures to counteract them, as presented in the following paragraphs.

External validity. It is related to the question "Results will generalize beyond the specific situations studied?". The first threat related to external validity consists in having a set of primary studies that is not representative of the whole research on FM building approaches. We mitigated this potential threat by following a systematic search process with respect to Petersen et al. (2008) and Kitchenham et al. (2009)'s guidelines. The selected studies have been explored in depth.

Regarding the 'relevance' of the papers included in the framework validation, each author did an individual classification of 100 terms extracted from the abstracts of the primary studies, and used them to verify that our framework integrated the knowledge that exists in the current set of approaches to constructing FMs in an automatic mode. Then we reviewed the classification of the other two co-authors (200 terms, from a total of 300) that were classified according to the significance metric (score) provided by the C-Value method [68]). Then, we discussed the differences in each classification proposal by tracking rates of inter-researchers' agreement. The classification made from the 300 most relevant terms in the papers selected to evaluate our framework is available on the Internet[3]. However, we acknowledge that this approach is subjective because it relies on the abstracts of the selected papers, and depends on the suitability of the selected terms in representing each paper.

A further threat related to external validity can be associated by the fact that, gray literature (e.g., white papers, non-reviewed publications or books, etc.) is not included in our research since we want to focus exclusively on the state of the art presented in high-quality scientific studies.

Internal validity. It refers to the influence of extraneous variables on the design of the study and is related to the question "Can we be sure our results really follow from the data?". This potential threat to validity has been mitigated by defining and validating our classification framework by carefully following a well-structured elaboration process (see Section 3). Regarding the influences of procedural and situational variables in the experiment, no important events happen during the study and the same observation and measurement instruments were applied during the data collection and analysis phases.

Construct validity. It concerns the validity of extracted data with respect to the research questions. A key validity concern related to the question "are we building the right framework?" is the degree to which the set of classification factors is complete. We call it 'complete' if the factors sufficiently account for the major differences between the methods found. We judged the completeness of the framework when using it for classifying the results from a literature review on the same topic: (semi-) automated construction of Feature Models. Thus, the completeness of the framework is evaluated by mapping the set of dimensions and their items, and the concepts extracted from the literature. We do, however, acknowledge that this judgment is subjective. The factors could have been more fine-grained (for instance including fine-grained details about the functional and NFRs represented with FMs, but it was beyond the purpose of our framework to classify FM construction approaches), as can be concluded from the observation that methods falling in the same group still differ. It is

[3] http://londres.ceisufro.cl/papers/jss-d-17-00719/.

possible, therefore, that researchers posing different research questions, or with a different scope, might want to include further factors within our classification framework.

The framework was improved with the data found from the literature review and to mitigate this source of threats, we performed an automated search on multiple electronic databases to avoid potential biases due to publishers' policies and business concerns. To improve it, we applied the framework on 31 approaches found in the literature review and completed it with the (sub) dimensions needed to take into account the relevant concepts of each approach and be able to classify them with the framework.

Conclusion validity. It concerns the relationship between the extracted data and the obtained results. Threats to conclusion validity have been mitigated by applying well-assessed research protocol throughout our study. The corresponding process has been formalized and documented in Section 3, so this study can be replicated by other researchers interested in collaborative configuration of product lines. Moreover, other researchers may identify dimensions and attributes different from the ones captured in our framework. We mitigated this potential threat by iteratively validating the framework (with two of the co-authors and one external researcher who can be considered as experts on the domain by their experience and publications related to the FM building). Finally, we claim that our framework is extensible and open as it can encompass new concepts (dimensions and attributes) characterizing new aspects that may be proposed by future FM building approaches.

7. Related work

To the best of our knowledge, there have been no published studies which propose frameworks for the classification of FM construction proposals, and which respond to the questions defined in Section 5. However, some researchers have tried to clarify some aspects of FMs. For instance, Saba et al. (2015) described and mentioned some extraction methods. Li et al. (2017) investigate the current state-of-the-art of feature and variability extraction from natural language (NL) documents. They performed a Systematic Literature Review (SLR) because they wanted to characterize the state-of-the-art and maturity of the analyzed approaches. Lee et al. (2002) clarified the concepts of the features and goals of FMs, and provided practical guidelines for successful product line software engineering implementation. Czarnecki et al. (2012) clarified the relation between FMs and Decision Models (DMs). They also compared multiple aspects of FMs and DMs, ranging from historical origins and rationale through to syntactic and semantic richness, identifying commonalities and differences. Finally, Chen and Ali Babar (2011) provided a systematic review of variability management approaches. They assessed 34 papers, dating from 1990 to 2008. The study identified several gaps such as scalability, which is an issue that has not been addressed.

8. Conclusion

This study aims to conceptualise an analysis framework for work in the area of automated FM construction, and capture the current state of this area. A conceptual framework for the classification of research proposals related to the (semi-)automated construction of FMs is presented. Six dimensions are considered that respond to the following questions: What is the purpose of the approach?, Which data sources initiate element extraction?, What type of conceptual structure is extracted?, What procedure or technique is used to construct an FM?, Which result delivers the proposed approach? What elements are evaluated by the proposed approach?. These questions are categorised as dimensions and subdimensions.

As a preliminary validation of the conceptual framework we evaluated 31 proposals that employ the use of re-engineering and synthesis, two of the most often used concepts to denote FM construction by (semi-)automated means as a baseline. The results show that the description of each proposal is simplified, and that it is possible to establish the main elements of the approach, and the evidence of their results, with greater precision. The proposed conceptual framework helps in the understanding and clarification of the diverse existing approaches found in the literature, and allows for the identification of research areas that have not been addressed. Thus, the framework builds a useful guide for the identification of research challenges in the area of FM construction by (semi-)automated means.

Future studies will amplify the base of the description of the proposal in a way that clearly establishes it as an up to the minute method for FM construction. In addition, a secondary connected study will be developed relating to the proposals for FM construction/collection, which will allow for the extension of the proposed framework, and will verify its validity through the participation of a greater base of users who are experts in FM construction, and are internationally recognised.

Acknowledgements

This study was supported by the *Vicerrectoría de Investigación y Postgrado, Universidad de La Frontera*, Chile, PROY. DI15-0020, PROY. DI17-0099, by the Department of *Ingenieria de sistemas, Universidad Eafit*, Colombia, and by the *Centre de Recherche en Informatique (CRI)* of the Panthéon-Sorbonne University, France.

References

Acher, M., Baudry, B., Heymans, P., Cleve, A., Hainaut, J.-L., 2013. Support for reverse engineering and maintaining feature models. In: Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems. ACM, New York, NY, USA. 20:1–20:8 10.1145/2430502.2430530.

Acher, M., Cleve, A., Collet, P., Merle, P., Duchien, L., Lahire, P., 2011. Reverse Engineering Architectural Feature Models. In: Springer (Ed.), 5th European Conference of Software Architecture (ECSA). Springer, Essen, Germany, pp. 220–235. doi:10.1007/978-3-642-23798-0_25.

Acher, M., Cleve, A., Perrouin, G., Heymans, P., Vanbeneden, C., Collet, P., Lahire, P., 2012. On extracting feature models from product descriptions. In: Proceedings of the Sixth International Workshop on Variability Modeling of Software-Intensive Systems. ACM, New York, NY, USA, pp. 45–54. doi:10.1145/2110147.2110153.

Acher, M., Cleve, A., Perrouin, G., Heymans, P., Vanbeneden, C., Collet, P., Lahire, P., 2012. On extracting feature models from product descriptions. In: Proceedings of the Sixth International Workshop on Variability Modeling of Software-Intensive Systems. ACM, pp. 45–54.

Acher, M., Collet, P., Lahire, P., France, R., 2010. Software Language Engineering: Second International Conference, SLE 2009, Denver, CO, USA, October 5 –6, 2009, Revised Selected Papers. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 62–81. Ch. Composing Feature Models

Acher, M., Collet, P., Lahire, P., France, R.B., 2011. A domain-specific language for managing feature models. In: Proceedings of the 2011 ACM Symposium on Applied Computing. ACM, pp. 1333–1340.

Al-Msie'Deen, R., Seriai, A.D., Huchard, M., Urtado, C., Vauttier, S., Salman, H.E., 2012. An approach to recover feature models from object-oriented source code. Actes de la Journée Lignes de Produits 15–26.

Alves, V., Gheyi, R., Massoni, T., Kulesza, U., Borba, P., Lucena, C., 2006. Refactoring product lines. In: Proceedings of the 5th international conference on Generative programming and component engineering. ACM, pp. 201–210.

Alves, V., Schwanninger, C., Barbosa, L., Rashid, A., Sawyer, P., Rayson, P., Pohl, C., Rummler, A., 2008. An exploratory study of information retrieval techniques in domain analysis. In: Software Product Line Conference, 2008. SPLC '08. 12th International, pp. 67–76. doi:10.1109/SPLC.2008.18.

Andersen, N., Czarnecki, K., She, S., Wäsowski, A., 2012. Efficient synthesis of feature models. In: Proceedings of the 16th International Software Product Line Conference-Volume 1. ACM, pp. 106–115.

Andersen, N., Czarnecki, K., She, S., Wasowski, A., 2012. Efficient synthesis of feature models. In: Proceedings of the 16th International Software Product Line Conference - Volume 1. ACM, New York, NY, USA, pp. 106–115. doi:10.1145/2362536.2362553.

Araar, I.E., Seridi, H., 2016. Software features extraction from object-oriented source code using an overlapping clustering approach. Informatica (Slovenia) 40 (2).

Assunção, W.K., Lopez-Herrejon, R.E., Linsbauer, L., Vergilio, S.R., Egyed, A., 2015. Extracting variability-safe feature models from source code dependencies in sys-

tem variants. In: Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation. ACM, pp. 1303–1310.
Bécan, G., Acher, M., Baudry, B., Ben Nasr, S. , 2015a. Breathing ontological knowledge into feature model synthesis: an empirical study . Empir . Softw . Eng . 51. doi:10.1007/s10664-014-9357-1.
Bécan, G., Acher, M., Baudry, B., Nasr, S.B., 2016. Breathing ontological knowledge into feature model synthesis: an empirical study. Empir . Softw . Eng . 21 (4), 1794–1841.
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.
Bécan, G., Behjati, R., Gotlieb, A., Acher, M. , 2015b. Synthesis of attributed feature models from product descriptions. In: Proceedings of the 19th International Conference on Software Product Line. ACM, New York, NY, USA, pp. 1–10. doi:10.1145/2791060.2791068.
Bécan, G., Ben Nasr, S., Acher, M., Baudry, B., 2014. Webfml: Synthesizing feature models everywhere. In: Proceedings of the 18th International Software Product Line Conference: Companion Volume for Workshops, Demonstrations and Tools - Volume 2. ACM, New York, NY, USA, pp. 112–116. doi:10.1145/2647908.2655974.
Bécan, G., Ben Nasr, S., Acher, M., Baudry, B., 2014. Webfml: synthesizing feature models everywhere. In: Proceedings of the 18th International Software Product Line Conference: Companion Volume for Workshops, Demonstrations and Tools-Volume 2. ACM, pp. 112–116.
Bae, J., Kang, S., 2007. A method to generate a feature model from a business process model for business applications. In: Computer and Information Technology, 2007. CIT 2007. 7th IEEE International Conference on, pp. 879–884. doi:10.1109/CIT.2007.78.
Bagheri, E., Di Noia, T., Ragone, A., Gasevic, D., 2010. Configuring software product line feature models based on stakeholders' soft and hard requirements. In: Proceedings of the 14th International Conference on Software Product Lines: Going Beyond. Springer-Verlag, Berlin, Heidelberg, pp. 16–31. http://www.dl.acm.org/citation.cfm?id=1885639.1885642
Batory, D., 2005. Feature models, grammars, and propositional formulas. In: Proceedings of the 9th International Conference on Software Product Lines. Springer-Verlag, Berlin, Heidelberg, pp. 7–20. doi:10.1007/11554844_3.
Batory, D., Benavides, D., Ruiz-Cortes, A., 2006. Automated analysis of feature models: challenges ahead . Commun. ACM 49 (12), 45–47. doi:10.1145/1183236.1183264.
Berger, T., Rublack, R., Nair, D., Atlee, J.M., Becker, M., Czarnecki, K., Wąsowski, A., 2013. A survey of variability modeling in industrial practice. In: Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems. ACM, New York, NY, USA doi:10.1145/2430502.2430513. 7:1–7:8
Braganca, A., Machado, R.J., 2007. Automating mappings between use case diagrams and feature models for software product lines. In: Proceedings of the 11th International Software Product Line Conference. IEEE Computer Society, Washington, DC, USA, pp. 3–12. doi:10.1109/SPLC.2007.11.
Brown, T.J., Gawley, R., Bashroush, R., Spence, I.T.A., Kilpatrick, P., Gillan, C., 2006. Weaving behavior into feature models for embedded system families. In: Software Product Lines, 10th International Conference, SPLC 2006, Baltimore, Maryland, USA, August 21 –24, 2006, Proceedings, pp. 52–64. doi:10.1109/SPLINE.2006.1691577.
Casalánguida, H., Durán, J.E., 2012. Automatic generation of feature models from uml requirement models. In: Proceedings of the 16th International Software Product Line Conference - Volume 2. ACM, New York, NY, USA, pp. 10–17. doi:10.1145/2364412.2364415.
Chen, L., Ali Babar, M., 2011. A systematic review of evaluation of variability management approaches in software product lines. Inf. Softw. Technol. 53 (4), 344–362. doi:10.1016/j.infsof.2010.12.006.
Chen, L., Ali Babar, M., Ali, N., 2009. Variability management in software product lines: a systematic review. In: Proceedings of the 13th International Software Product Line Conference. Carnegie Mellon University, pp. 81–90.
Chen, K., Zhang, W., Zhao, H., Mei, H., 2005. An approach to constructing feature models based on requirements clustering. In: 13th IEEE International Conference on Requirements Engineering (RE'05), pp. 31–40. doi:10.1109/RE.2005.9.
Chen, K., Zhang, W., Zhao, H., Mei, H., 2005. An approach to constructing feature models based on requirements clustering. In: Requirements Engineering, 2005. Proceedings. 13th IEEE International Conference on. IEEE, pp. 31–40.
Chikofsky, E.J., Cross, J.H., 1990. Reverse engineering and design recovery: a taxonomy. IEEE Softw . 7 (1), 13–17. doi:10.1109/52.43044.
Czarnecki, K., Grünbacher, P., Rabiser, R., Schmid, K., Wąsowski, A., 2012. Cool features and tough decisions: A comparison of variability modeling approaches. In: Proceedings of the Sixth International Workshop on Variability Modeling of Software-Intensive Systems. ACM, New York, NY, USA, pp. 173–182. doi:10.1145/2110147.2110167.
Czarnecki, K., Helsen, S., Eisenecker, U., 2005. Formalizing cardinality-based feature models and their specialization. Softw . Proc . Improve . Pract . 10 (1), 7–29. doi:10.1002/spip.213.
Czarnecki, K., Hwan, C., Kim, P., Kalleberg, K.T., 2006. Feature models are views on ontologies. In: Oibrien, L. (Ed.), 10th International Software Product Lines Conference, SPLC'06. IEEE Computer Society, Baltimore, Maryland, pp. 41–51. doi:10.1109/SPLINE.2006.1691576.
Czarnecki, K., Wasowski, A., 2007. Feature diagrams and logics: There and back again. In: Software Product Line Conference, 2007. SPLC 2007. 11th International, pp. 23–34. doi:10.1109/SPLINE.2007.24.
Damasevicius, R., Paskevicius, P., Karciauskas, E., Marcinkevicius, R., 2012. Automatic extraction of features and generation of feature models from java programs.. ITC 41 (4), 376–384.
Davril, J.-M., Delfosse, E., Hariri, N., Acher, M., Cleland-Huang, J., Heymans, P., 2013. Feature model extraction from large collections of informal product descriptions. In: Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering. ACM, New York, NY, USA, pp. 290–300. doi:10.1145/2491411.2491455.
Davril, J.-M., Delfosse, E., Hariri, N., Acher, M., Cleland-Huang, J., Heymans, P., 2013. Feature model extraction from large collections of informal product descriptions. In: Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering. ACM, pp. 290–300.
Dumitru, H., Gibiec, M., Hariri, N., Cleland-Huang, J., Mobasher, B., Castro-Herrera, C., Mirakhorli, M., 2011. On-demand feature recommendations derived from mining public product descriptions. In: Proceedings of the 33rd International Conference on Software Engineering. ACM, New York, NY, USA, pp. 181–190. doi:10.1145/1985793.1985819.
Feigenbaum, E., 1984. Knowledge engineering. Ann. N. Y. Acad. Sci. 426 (1), 91–107. doi:10.1111/j.1749-6632.1984.tb16513.x.
Ferrari, A., Spagnolo, G.O., Dell'Orletta, F., 2013. Mining commonalities and variabilities from natural language documents. In: Proceedings of the 17th International Software Product Line Conference. ACM, New York, NY, USA, pp. 116–120. doi:10.1145/2491627.2491634.
Fleurey, F., Baudry, B., France, R., Ghosh, S., 2007. A generic approach for automatic model composition. In: International Conference on Model Driven Engineering Languages and Systems. Springer, pp. 7–15.
Frantzi, K., Ananiadou, S., Mima, H., 2000. Automatic recognition of multi-word terms:. the c-value/nc-value method. Int . J . Digit . Lib . 3 (2), 115–130. doi:10.1007/s007999900023.
Griss, M.L., Favaro, J., Alessandro, M.d., 1998. Integrating feature modeling with the rseb. In: Proceedings of the 5th International Conference on Software Reuse. IEEE Computer Society, Washington, DC, USA, p. 76.
Hamza, M., Walker, R.J., 2015. Recommending features and feature relationships from requirements documents for software product lines. In: Proceedings of the Fourth International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering. IEEE Press, Piscataway, NJ, USA, pp. 25–31. http://www.dl.acm.org/citation.cfm?id=2820668.2820675
Haslinger, E.N., Lopez-Herrejon, R.E., Egyed, A., 2011. Reverse engineering feature models from programs' feature sets. In: 2011 18th Working Conference on Reverse Engineering, pp. 308–312. doi:10.1109/WCRE.2011.45.
Hofman, P., Stenzel, T., Pohley, T., Kircher, M., Bermann, A., 2012. Domain specific feature modeling for software product lines. In: Proceedings of the 16th International Software Product Line Conference-Volume 1. ACM, pp. 229–238.
Horcas, J.-M., Pinto, M., Fuentes, L., 2016. An automatic process for weaving functional quality attributes using a software product line approach. J. Syst. Softw. 112 (C), 78–95. doi:10.1016/j.jss.2015.11.005.
Itzik, N., Reinhartz-Berger, I., 2014. Generating feature models from requirements: Structural vs. functional perspectives. In: Proceedings of the 18th International Software Product Line Conference: Companion Volume for Workshops, Demonstrations and Tools - Volume 2. ACM, New York, NY, USA, pp. 44–51. doi:10.1145/2647908.2655966.
Itzik, N., Reinhartz-Berger, I., 2014. Generating feature models from requirements: Structural vs. functional perspectives. In: Proceedings of the 18th International Software Product Line Conference: Companion Volume for Workshops, Demonstrations and Tools-Volume 2. ACM, pp. 44–51.
Jarzabek, S., Yang, B., Yoeun, S., 2006. Addressing quality attributes in domain analysis for product lines. IEE Proc .-Softw . 153 (2), 61–73.
Kang, K.C., Cohen, S.G., Hess, J.A., Novak, W.E., Peterson, A.S., 1990. Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical Report. Carnegie-Mellon University Software Engineering Institute.
Kang, K.C., Kim, S., Lee, J., Kim, K., Shin, E., Huh, M., 1998. Form: a feature-oriented reuse method with domain-specific reference architectures . Ann. Softw. Eng. 5, 143–168.
Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., Linkman, S., 2009. Systematic literature reviews in software engineering - a systematic literature review. Inf. Softw. Technol. 51 (1), 7–15. doi:10.1016/j.infsof.2008.09.009.
Lee, K., Kang, K.C., Lee, J., 2002. Concepts and guidelines of feature modeling for product line software engineering. In: Proceedings of the 7th International Conference on Software Reuse: Methods, Techniques, and Tools. Springer-Verlag, London, UK, UK, pp. 62–77. http://www.dl.acm.org/citation.cfm?id=645547.658853.
Lee, S.-B., Kim, J.-W., Song, C.-Y., Baik, D.-K., 2007. An approach to analyzing commonality and variability of features using ontology in a software product line engineering. In: Software Engineering Research, Management & Applications, 2007. SERA 2007. 5th ACIS International Conference on. IEEE, pp. 727–734.
Li, Y., Schulze, S., Saake, G., 2017. Reverse engineering variability from natural language documents: A systematic literature review. In: Proceedings of the 21st International Systems and Software Product Line Conference - Volume A. ACM, New York, NY, USA, pp. 133–142. doi:10.1145/3106195.3106207.
Li, Y., Schulze, S., Saake, G., 2018. Extracting features from requirements: Achieving accuracy and automation with neural networks. In: 2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER), pp. 477–481. doi:10.1109/SANER.2018.8330243.
Lopez-Herrejon, R.E., Galindo, J.A., Benavides, D., Segura, S., Egyed, A., 2012. Reverse engineering feature models with evolutionary algorithms: An exploratory study.

In: Search Based Software Engineering - 4th International Symposium, SSBSE 2012, Riva del Garda, Italy, September 28 –30, 2012. Proceedings, pp. 168–182.
Lora-Michiels, A., Salinesi, C., Mazo, R., 2010. A Method based on Association Rules to Construct Product Line Model. In: 4th International Workshop on Variability Modelling of Software-intensive Systems (VaMos), p. 50. https://www.hal.archives-ouvertes.fr/hal-00707527. Linz, Austria
Mannion, M., Kaindl, H., Savolainen, J., 2017. Product line strategies and feature reuse. In: Proceedings of the 21st International Systems and Software Product Line Conference - Volume A. ACM, New York, NY, USA doi:10.1145/3106195.3106231. 252–252
Marciuska, S., Gencel, C., Abrahamsson, P., 2014. Automated feature identification in web applications. In: International Conference on Software Quality. Springer, pp. 100–114.
Martinez, J., Ziadi, T., Mazo, R., Bissyandé, T.F., Klein, J., Traon, Y.L., 2014. Feature relations graphs: A visualisation paradigm for feature constraints in software product lines. In: Software Visualization (VISSOFT), 2014 Second IEEE Working Conference on, pp. 50–59. doi:10.1109/VISSOFT.2014.18.
Mazo, R., 2011. A generic approach for automated verification of product line models. Ph.D. thesis, Université Paris 1 Panthéon - Sorbonne Ph.D. thesis.
Mazo, R., Salinesi, C., Diaz, D., Lora-Michiels, A., 2011. Transforming attribute and clone-enabled feature models into constraint programs over finite domains. In: 6th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE). Springer Press, pp. 188–199.
McGregor, J.D., Muthig, D., Yoshimura, K., Jensen, P., 2010. Guest editors' introduction: successful software product line practices . Softw . IEEE 27 (3), 16–21. doi:10.1109/MS.2010.74.
Mefteh, M., Bouassida, N., Ben-Abdallah, H., 2015. Implementation and evaluation of an approach for extracting feature models from documented uml use case diagrams. In: Proceedings of the 30th Annual ACM Symposium on Applied Computing. ACM, New York, NY, USA, pp. 1602–1609. doi:10.1145/2695664.2695907.
Mefteh, M., Bouassida, N., Ben-Abdallah, H., 2016. Mining feature models from functional requirements.
Noorian, M., Asadi, M., Bagheri, E., Du, W., 2014. Addressing non-functional properties in feature models: a goal-oriented approach. Int. J. Softw. Eng. Knowl. Eng. 24 (10), 1439–1487.
Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M., 2008. Systematic mapping studies in software engineering. In: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering. BCS Learning & Development Ltd., Swindon, UK, pp. 68–77.
Pohl, K., Böckle, G., Linden, F.J.v.d., 2005. Software product line engineering: Foundations, principles and techniques . Springer-Verlag New York, Inc., Secaucus, NJ, USA.
Rauber, T.W., de Assis Boldt, F., Varejão, F.M., 2015. Heterogeneous feature models and feature selection applied to bearing fault diagnosis. IEEE Trans. Ind. Electron. 62 (1), 637–646.
Riebisch, M., Böllert, K., Streitferdt, D., Philippow, I., 2002. Extending feature diagrams with uml multiplicities. In: 6th World Conference on Integrated Design & Process Technology (IDPT2002), 23, pp. 1–7.
Robak, S., Pieczynski, A., 2003. Employing fuzzy logic in feature diagrams to model variability in software product-lines. In: Engineering of Computer-Based Systems, 2003. Proceedings. 10th IEEE International Conference and Workshop on the. IEEE, pp. 305–311.
Ryssel, U., Ploennigs, J., Kabitzsch, K., 2011. Extraction of feature models from formal contexts. In: Proceedings of the 15th International Software Product Line Conference, Volume 2. ACM, New York, NY, USA, pp. 4:1–4:8. doi:10.1145/2019136.2019141.
Ryssel, U., Ploennigs, J., Kabitzsch, K., 2011. Extraction of feature models from formal contexts. In: Proceedings of the 15th International Software Product Line Conference, Volume 2. ACM, p. 4.
Saba, P., Mehran, M., Alasti, A.A.A., 2015. A review of feature model position in the software product line and its extraction methods . Int . J . Comp . Sci .Secur . (IJCSS) 9 (5), 274.
Schmid, K., Rabiser, R., Grünbacher, P., 2011. A comparison of decision modeling approaches in product lines. In: Proceedings of the 5th Workshop on Variability Modeling of Software-Intensive Systems. ACM, New York, NY, USA, pp. 119–126. doi:10.1145/1944892.1944907.
Schobbens, P.-Y., Heymans, P., Trigaux, J.-C., Bontemps, Y., 2007. Generic semantics of feature diagrams. Comput. Netw. 51 (2), 456–479. doi:10.1016/j.comnet.2006.08.008.
Segura, S., Benavides, D., Ruiz-Cortés, A., Trinidad, P., 2008. Automated merging of feature models using graph transformations. Post-proceed . Sec . Summer SchoolGenerat . Transform . Tech . Softw . Eng . 5235, 489–505.
Segura, S., Parejo, J.A., Hierons, R.M., Benavides, D., Ruiz-Cortés, A., 2014. Automated generation of computationally hard feature models using evolutionary algorithms. Expert Syst. Appl. 41 (8), 3975–3992.
She, S., Czarnecki, K., Wasowski, A., 2012. Usage scenarios for feature model synthesis. In: Proceedings of the VARiability for You Workshop: Variability Modeling Made Useful for Everyone. ACM, pp. 15–20.
She, S., Lotufo, R., Berger, T., Wasowski, A., Czarnecki, K., 2011. Reverse engineering feature models. In: Proceedings of the 33rd International Conference on Software Engineering. ACM, New York, NY, USA, pp. 461–470. doi:10.1145/1985793.1985856.
She, S., Lotufo, R., Berger, T., Wasowski, A., Czarnecki, K., 2011. Reverse engineering feature models. In: Software Engineering (ICSE), 2011 33rd International Conference on. IEEE, pp. 461–470.
Siegmund, N., Rosenmüller, M., Kuhlemann, M., Kästner, C., Apel, S., Saake, G., 2012. Spl conqueror: toward optimization of non-functional properties in software product lines . Softw . Qual . J . 20 (3 –4), 487–517. doi:10.1007/s11219-011-9152-9.
Tun, W.P.P., Myo, K.M., 2015. Re-engineering based feature model management for software product line. In: Proceedings of 2015 International Conference on Future Computational Technologies (ICFCT'2015), pp. 229–235.
Vacchi, E., Cazzola, W., Combemale, B., Acher, M., 2014. Automating variability model inference for component-based language implementations. In: Proceedings of the 18th International Software Product Line Conference-Volume 1. ACM, pp. 167–176.
van den Broek, P., 2012. Intersection of feature models. In: Proceedings of the 16th International Software Product Line Conference-Volume 2. ACM, pp. 61–65.
Wanderley, F., da Silveira, D.S., Araujo, J.a., Lencastre, M., 2012. Generating feature model from creative requirements using model driven design. In: Proceedings of the 16th International Software Product Line Conference - Volume 2. ACM, New York, NY, USA, pp. 18–25. doi:10.1145/2364412.2364416.
Wanderley, F., da Silveira, D.S., Araujo, J., Lencastre, M., 2012. Generating feature model from creative requirements using model driven design. In: Proceedings of the 16th International Software Product Line Conference-Volume 2. ACM, pp. 18–25.
Wang, Y., 2015. Semantic information extraction for software requirements using semantic role labeling. In: 2015 IEEE International Conference on Progress in Informatics and Computing (PIC), pp. 332–337. doi:10.1109/PIC.2015.7489864.
Wang, Y., 2016. Automatic semantic analysis of software requirements through machine learning and ontology approach 21 (6), 692–701. doi:10.1007/s12204-016-1783-3. Exported from on 2018/09/05.
Wang, B., Zhang, W., Zhao, H., Jin, Z., Mei, H., 2009. A use case based approach to feature models' construction. In: Requirements Engineering Conference, 2009. RE'09. 17th IEEE International. IEEE, pp. 121–130.
Weston, N., Chitchyan, R., Rashid, A., 2009. A framework for constructing semantically composable feature models from natural language requirements. In: Proceedings of the 13th International Software Product Line Conference. Carnegie Mellon University, Pittsburgh, PA, USA, pp. 211–220.
Wieringa, R., Maiden, N., Mead, N., Rolland, C., 2005. Requirements engineering paper classification and evaluation criteria: a proposal and a discussion . Requir. Eng. 11 (1), 102–107. doi:10.1007/s00766-005-0021-6.
Yang, Y., Peng, X., Zhao, W., 2009. Domain feature model recovery from multiple applications using data access semantics and formal concept analysis. In: Reverse Engineering, 2009. WCRE'09. 16th Working Conference on. IEEE, pp. 215–224.
Yi, L., Zhang, W., Zhao, H., Jin, Z., Mei, H., 2012. Mining binary constraints in the construction of feature models. In: 2012 20th IEEE International Requirements Engineering Conference (RE), pp. 141–150. doi:10.1109/RE.2012.6345798.
Yildirim, İ., Sözer, H., 2017. Automated extraction of feature models from android based portable devices. In: Software Quality, Reliability and Security Companion (QRS-C), 2017 IEEE International Conference on. IEEE, pp. 441–448.
Yu, D., Chen, Z., Zhang, Y., 2015. From goal models to feature models: A rule-based approach for software product lines. In: 2015 Asia-Pacific Software Engineering Conference (APSEC), pp. 277–284. doi:10.1109/APSEC.2015.22.
Zhan, Z., Luo, W., Guo, Z., Liu, Y., 2018. Feature selection optimization based on atomic set and genetic algorithm in software product line. Adv . Intell . Syst . Comp . 686, 93–100.
Zhang, B., Becker, M., 2013. Mining complex feature correlations from software product line configurations. In: Proceedings of the Seventh International Workshop on Variability Modelling of Software-intensive Systems. ACM, p. 19.
Zhou, F., Jiao, J.R., Yang, X.J., Lei, B., 2017. Augmenting feature model through customer preference mining by hybrid sentiment analysis. Expert Syst. Appl. 89, 306–317.
Ziadi, T., Frias, L., da Silva, M.A.A., Ziane, M., 2012. Feature identification from the source code of product variants. In: Software Maintenance and Reengineering (CSMR), 2012 16th European Conference on, pp. 417–422. doi:10.1109/CSMR.2012.52.