A Metamodel-based Classification of Variability Modeling Approaches [*]

Paul Istoan[1], Jacques Klein[2], Gilles Perouin[3], and Jean-Marc Jézéquel[4]

[1] CRP Gabriel Lippmann, Luxembourg - LASSY, University of Luxembourg, Luxembourg-Université de Rennes 1, France,
[2] SnT - University of Luxembourg, Luxembourg, Luxembourg
[3] PRECISE, University of Namur, Namur, Belgium
[4] IRISA, Université de Rennes 1, France
istoan@lippmann.lu, gilles.perrouin@fundp.ac.be,
jacques.klein@uni.lu, jezequel@irisa.fr

Abstract. Software Product Line Engineering (SPLE) is an emerging paradigm taking momentum that proposes to address flexibility and shorter time-to-market by maximizing software reuse. The key characteristic of SPLE is the effective modelling and management of variability, for which a number of Variability Modeling (VM) techniques have been developed during the last two decades. Therefore, understanding their commonalities and differences is important for selecting the most suitable technique. In this paper, we propose a metamodel-based classification of VM techniques gathered through a survey of relevant literature.

Keywords: Variability Modeling Approaches, Model-Driven Engineering, Survey

1 Introduction

Constant market evolution triggered an exponential growth in the complexity and variability of modern software solutions. It is frequently the case that software development is actually a redevelopment process, with many products being partially built before. *Software Product Lines* (SPL), or *software families*, are rapidly emerging as an important and viable software development paradigm designed to handle such issues [34]. Use of SPL approaches has allowed renowned companies like Hewlett-Packard, Nokia or Motorola to achieve considerable quantitative and qualitative gains in terms of productivity, time to market and customer satisfaction [1]. Their increasing success relies on the capacity to offer software suppliers/vendors *ways to exploit the existing commonalities in their software products*. SPL engineering focuses on capturing the *commonality* and *variability* between several software products [12]. This new concept started to draw the attention of the software community when software began to be massively integrated into hardware product families, with cellular phones [28] probably being the most well known example. More generally, automotive systems, aerospace or telecommunications are some of the areas targeted by SPL research.

[*] This work has been funded by the SPLIT project (FNR + CNRS, FNR/INTER/CNRS/08/02), the IAP Programme of the Belgian State, Belgian Science Policy (MoVES project) and the Walloon Region (NAPLES project) and the FNR CORE project MITER (C10/IS/783852)

Variability is seen as the key feature that distinguishes SPL engineering from other software development approaches [9]. In common language use, the term *variability* refers to ”the ability or the tendency to change”. It is a central concern in SPL development [19] and covers the entire development life cycle, from requirements elicitation to product testing. When talking about *SPL variability*, two concepts immediately stand out [23]: *commonalities* (assumptions true for each family member) and *variabilities* (assumptions about how individual family members differ). Variability management is thus growingly seen as being complex process that requires increased attention.

A traditional way used by scientists to master the increasing complexity and variability of real-world phenomena is to resort to *modelling*. In software engineering, *models* allow to express both problems and solutions at a higher abstraction level than code [24]. Model Driven Engineering (MDE) treats *models* as first-class elements for application development. The goal of MDE is to reduce design complexity and make software engineering more efficient by shifting the focus from implementation to modelling. *Models* are created based on concepts defined in a *meta-model*, which defines the concepts, relationships and (static) semantics of a domain. The relation between a model and its meta-model is defined as a conformity relation.

In recent years, several variability modelling techniques have been developed, aiming to explicitly and effectively represent SPL variability. The existing differences between them render each method unique, suitable for a particular domain and in a specific context. Hence the question of *which approach is the most suitable with respect to a particular context?* is of great interest to SPL engineers. There is a stringent need to extract, synthesize and analyse in a critical manner the research literature on SPL variability modelling. A review of all contributions related to this topic, outlining the individual characteristics of each method and possibilities of improvement, can facilitate and guide SPL engineers in the selection of a particular technique suitable for their specific development context. Furthermore, such a comparative analysis can provide practitioners with a qualified portfolio of available techniques and therefore play an important role in the transfer of knowledge from research to industry. In this context, this paper addresses the following research questions:

– RQ1. How can variability be modelled in SPLs?
– RQ2. How can existing techniques be classified?

In this paper, we argue that *VM techniques can be classified according to how variability is handled at the meta-model and model levels*. These two levels refer to both the product line artifacts and the product line variability. After having surveyed the relevant literature, we provide a classification framework that applies this two-level analysis to sort the VM techniques discussed, and highlight the fundamental differences between them in the way they capture variability. This classification provides a better understanding of these approaches and helps the engineers find the appropriate VM technique.

The remainder of this paper is structured as follows: Section 2 details how the survey of existing VM approaches was carried out. Section 3 presents our classification of VM approaches and briefly discusses them. Section 4 outlines some relevant related work while Section 5 concludes the paper.

2 Survey Protocol

With variability modelling being a major concern in SPL engineering, a plethora of methods have been developed by research and industry. So, in order to answer our first research question, a valid selection of relevant work on SPL variability modelling must first be performed. In this section we briefly explain the selection process followed to identify relevant contributions in the field.

The search process was performed in three steps. First, a thorough on-line research of relevant papers was performed using the Google search engine, using *search strings* based on the main concepts of the topic investigated. The search area was enlarged by using synonyms or other terms directly related to the topic of SPL variability as search strings. Similar searches were repeated on the main digital sources of research literature: ACM Digital Library, Lecture Notes in Computer Science, SpringerLink, SCOPUS (Elsevier), Web of Knowledge (ISI), IEEE Xplore, IEEE Computer Society Digital Library and ScienceDirect. In a second step, we performed a manual search in specific conference proceedings known to be classical venues of publication for SPL research: Software Product Line Conference (SPLC) and Product Family Engineering (PFE) conferences, Variability Modelling of Software-Intensive Systems (VaMoS) workshop. Finally, we also analysed other research projects addressing SPL engineering and variability to see which papers they considered relevant. The result of the search process produced a list of 236 papers.

Separately, we analysed the research literature for other surveys addressing the topic of SPL variability. Twelve papers were found: [10, 40, 32, 7, 16, 15, 20, 3, 26, 43, 33, 44]. For each of them, we extracted the list of referenced papers and regrouped them in a unique list, containing all papers cited in at least 2 surveys. Each paper on this list was assigned a value representing the number of surveys it appeared in. Based on this criteria, the list was ordered, resulting in a total of 55 papers. This selection criterion is relevant as it regroups the knowledge and expertise of other authors from SPLE.

The final list of papers to be analysed was obtained by comparing the previous two results. We identified 38 papers common to both lists. To obtain the final result, containing 20 papers, we also took into account the specific classification criteria we propose and discuss later on in this paper, and mapped them on the list of 38 papers.

3 Classification of Variability Modeling Methods

As variability is extensively used in SPL engineering, variability-related concepts can be gathered in a separate, dedicated language. In MDE, the structure of a domain is explicitly captured in a *meta-model*. Working at the level of *models* and *meta-models* makes it possible to analyse and classify SPL variability modelling methods at a high level of abstraction and objectiveness, and to extract general observations valid for an entire class of variability modelling approaches. We identify and analyse the central concepts used by a wide variety of VM techniques and show how they relate to each other. The analysis is performed at two levels: *meta-model* and *model*.

SPLs are usually characterized by two distinct concepts: a *set of core assets* or reusable components used for the development of new products (**assets model**); a *means* *to represent the commonality and variability* between SPL members (**variability model**). Our classification is based on these two concepts. A thorough analysis of the research literature revealed two major directions in SPL variability modelling:

- Methods that use a **single (unique) model to represent the SPL assets and the SPL variability**:
  A. Annotate a base model by means of extensions: [11, 18, 35, 45]
  B. Combine a general, reusable variability meta-model with different domain meta-models: [31]
- Methods that **distinguish and keep separate the assets model from the variability model**:
  A. Connect Feature Diagrams to model fragments: [36, 13, 27, 2]
  B. Orthogonal Variability Modelling: [38, 30]
  C. ConIPF Variability Modeling Framework (COVAMOF): [42, 41]
  D. Decision model-based approaches: [14, 29, 17, 39, 4]
  E. Relate a common variability language with different base languages: [22]

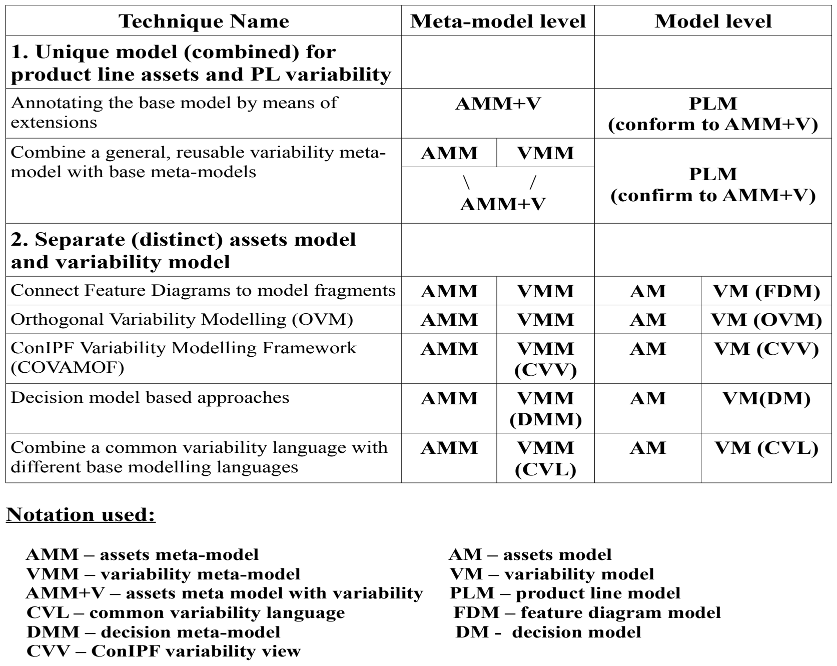
In this classification, the terms *assets meta-model (AMM)* and *assets model (AM)* cover a broad spectrum, depending on the point of view of the different authors. They are further refined for each particular class of methods. Table 1 summarizes the proposed classification and the newly introduced concepts. It briefly describes what happens at meta-model and model level for the identified classes of variability modelling techniques. The papers cited here are analysed in more detail in the following.

3.1 Single model to describe the product line assets and the product line variability

This category contains techniques that extend a language or a general purpose meta-model with specific concepts that allow designers to describe variability. Their core characteristic is the mix of variability and PL assets concepts into a unique model. Concepts regarding variability and those describing the assets model are combined into a new language, that may either have a new, mixed syntax, or one based on that of the base model extended by the syntax of the variability language. This applies at both meta-model and model level. We further distinguish:

A. Annotate a base model by means of extensions [11, 45, 18, 35]: standard languages are not created to explicitly represent all types of variability. Therefore, SPL models are frequently expressed by extending or annotating such standard languages (models). The annotated models are unions of all specific models in a model family and contain all necessary variability concepts. Regarding our classification, we distinguish at meta-model level an assets meta-model enhanced with variability concepts (AMM+V).In this case, the term ”assets meta-model” (AMM) refers to a *base* or a *domain meta-model* (meta-model of standard language used, eg. UML). Then, at model level, product line models (PLM) can be derived. They conform to the AMM+V defined at meta-model level. Typical examples from this category are methods that extend UML with profiles and stereotypes: [11, 18, 35, 45].

B. Combine a general, reusable variability meta-model with different domain meta-models [31, 37]: this approach addresses in particular the meta-model level, where a



Technique Name	Meta-model level		Model level	
1. Unique model (combined) for product line assets and PL variability				
Annotating the base model by means of extensions	**AMM+V**		**PLM (conform to AMM+V)**	
Combine a general, reusable variability meta-model with base meta-models	**AMM**	**VMM**	**PLM (confirm to AMM+V)**	
	\ **AMM+V**	/		
2. Separate (distinct) assets model and variability model				
Connect Feature Diagrams to model fragments	**AMM**	**VMM**	**AM**	**VM (FDM)**
Orthogonal Variability Modelling (OVM)	**AMM**	**VMM**	**AM**	**VM (OVM)**
ConIPF Variability Modelling Framework (COVAMOF)	**AMM**	**VMM (CVV)**	**AM**	**VM (CVV)**
Decision model based approaches	**AMM**	**VMM (DMM)**	**AM**	**VM(DM)**
Combine a common variability language with different base modelling languages	**AMM**	**VMM (CVL)**	**AM**	**VM (CVL)**

Notation used:

AMM – assets meta-model	**AM – assets model**
VMM – variability meta-model	**VM – variability model**
AMM+V – assets meta model with variability	**PLM – product line model**
CVL – common variability language	**FDM – feature diagram model**
DMM – decision meta-model	**DM - decision model**
CVV – ConIPF variability view	

Fig. 1. Classification of variability modelling techniques - meta-model and model level

two-step process is applied. Initially, two separate meta-models are created: an assets meta-model and a general, reusable variability meta-model. In a second step, they are combined, resulting in a unique assets meta-model extended with variability concepts. In this case, the term AMM denotes a domain meta-model (meta-model of domains specific language used for modelling). As for the previous category, at model level, PL models can be derived. A representative approach from this category comes from Morin et al. [31]. They propose a reusable variability meta-model describing variability concepts and their relations independently from any domain meta-model. Using Aspect-Oriented Modelling (AOM) techniques, variability can be woven into a given base meta-model, allowing its integration in a semi-automatic way into a wide range of meta-models.

3.2 Separate the assets model from the variability model

Techniques in this category have separate representations for the variability and the assets model. Elements from the variability model relate to assets model elements either by referencing or by other techniques. The key characteristic of such methods is the clear separation of concerns, which applies at both meta-model and model level. Some advantages of such approaches are: each asset model may have more than one variability model; designers can focus on modelling the SPL core assets and address the SPL variability separately; possibility for a standardized variability model. We further identify five sub-categories of methods pertaining to this category. The essential difference between all these sub-categories is the different type of variability model (meta-model) each one uses.

A. Connect Feature Diagrams to model fragments [36, 13, 27, 2]: Feature Diagrams (FD) [25] are the most popular VM technique in the SPL community. They organise features hierarchically in a tree-like structure where variability is defined via operators (or, xor, and) applied on child features. They also allow to model additional relations (mutual exclusion or dependence) via cross-tree constraints and have been subject to formalisation [5] and automated analyses [8]. Yet, how we associate model fragments to features is an emerging research direction. Different model fragment types can be associated to features. In this context, the feature diagram defines the PL variability, with each feature having an associated implementation. Concerning our classification, we notice a clear distinction between assets and variability related concepts at meta-model level. This situation extends to model level: separate assets and variability models exist. For this category, the assets model consists of a set of software artefact/asset fragments. The particular variability model used is a Feature Diagram.

B. Orthogonal Variability Modelling [38, 30]: as for all approaches in this category, the assets model and the variability model are distinct. The differentiating factor is the type of variability model used: an orthogonal variability model (OVM). There is also a difference regarding the assets model, which in this case is a compact software development artefact and no longer a set of model fragments. The variability model relates to different parts of the assets model using *artefact dependencies*. Pohl et al. [38] proposed the OVM concept, defined as: a model that defines the variability of a SPL separately and then relates it to other development artefacts like use case, component and test models. OVM provides a view on variability across all development artefacts. A slightly different OVM proposal comes from Metzger et al. [30].

C. ConIPF Variability Modeling Framework (COVAMOF) [42, 41]: this category contains the COVAMOF method proposed by Sinnema et al. Concerning our classification, we identify, at the meta-model level, separate variability and assets meta-models. This reflects also at model level, where a separate variability model, called COVAMOF Variability View (CVV), and an assets model can be distinguished. Sinnema et al. identify four requirements they considered essential for a variability modelling technique: uniform and first class representation of variation points at all abstraction levels; hierarchical organization of variability representation; first-class representation of dependencies; explicit modelling of interactions between dependencies. An analysis of existing variability approaches performed by Sinnema et al. revealed that none supported all four criteria. As a result they propose COVAMOF, an approach designed to uniformly model variability in all abstraction layers of a SPL.

D. Decision model based approaches: this class of approaches differs by using *decision models* as variability model. Decision-oriented approaches were designed to guide the product derivation process based on *decision models*. For Bayer et al. it is a model that ”captures variability in a product line in terms of open decisions and possible resolutions” [6]. A decision model is basically a table where each row represents a decision and each column a property of a decision. The most well-known approach in this category is DOPLER [14]. It was designed to support the modelling of both problem space variability (stakeholder needs) using decision models, and solution space variability (architecture and components of technical solution) using asset models and also to assure traceability between them.

E. Relate a common variability language with different base languages [22]: methods belonging to this category propose a generic variability language which can relate to different base models, extending them with variability. Regarding our classification, at meta-model level there is a separate generic variability meta-model and an assets meta-model (AMM). The AMM is actually the meta-model of the base language on which the *common variability language* is applied. At model level, elements of the variability model relate to assets model elements by referencing and using substitutions. A representative approach in this category is the Common Variability Language (CVL) proposed by Haugen et al. [22].

4 Related Work

We identified several other surveys and studies that address to some extent the subject of product line variability modelling. In this section, the most relevant proposals are briefly analysed and compared to our work.

In [10] Chen et al. present the findings of their systematic literature review of papers on variability management in SPL engineering. The focus of the paper seems to be more to reveal the chronological background of various approaches and the history of variability management research rather than to classify the actual methods. Out paper differs significantly from the one of Chen et al. in this aspect, as our goal is not to detail the individual steps of a systematic review, but to focus on the actual classification of methods. In the conclusion of their paper, Chen et al. state that one of the aspects that needs immediate attention from SPL researchers and practitioners is to provide a classification of the different variability modelling approaches. This point summarizes precisely the contribution and focus of our work.

In [32] Mujtaba et al. use a systematic method to develop a SPL variability map and classify relevant literature accordingly. The main contributions of their work are: identification of emphasized and neglected SPL research areas, classification of contributions made by different approaches, providing an example of how to adapt systematic mapping studies to software engineering. They focus mostly on presenting the research methodology used. In contrast, our contribution is of a more practical nature: introduce general concepts regarding SPL variability and classify how exactly each of them captures variability.

In the technical report [44] Trigaux et al. present and compare different notations for modelling SPL variability: feature modelling, use cases, class diagrams. The criteria used for comparison are: representation of common and variable parts, distinction between types of variability, representation of dependencies between variable parts, support for model evolution, understandability and graphical representation. In our paper we cover a much broader spectrum of approaches and also classify them according to a model driven framework.

Another technical report that discusses SPL variability is [3]. Asikainen identifies the concepts suitable for modelling configurable SPLs, what is their semantics and what kind of language or modelling method can support these concepts. The core part of their discussion on previous existing literature consists of an analysis and comparison of methods for modelling variability. The evaluated methods fall in three categories:

feature-based, architecture-based and other methods. Compared to their work, we provide a clear classification of the methods studied from a model-driven perspective and point out the particular ways in which they express variability.

In [21] Haugen et al. introduce a reference model used for comparing system family modelling approaches. The proposed reference model is based on the distinction between the *generic sphere* (feature models, product line models) and the *specific sphere* (feature selection, product model). The authors identify three major approaches for modelling system families: using standard languages, annotating a general language, using dedicated domain-specific languages. Although some of the methods presented overlap in some way with methods we present in our paper, we use a different set of criteria for classifying variability modelling approaches.

In [43] Svahnberg et al. discuss the factors that need to be considered when selecting an appropriate technique for implementing variability. This paper focuses on how to implement variability in architecture and implementation artefacts, like the software architecture design and the components and classes of a software system. Their main contribution is to provide a taxonomy of techniques that can be used to implement variability. Svahnbeg et al. focus on discussing the actual implementation of variability, mostly at code level, while we discuss variability modelling at the higher level of abstraction of languages and models.

5 Conclusion

Initiated more than two decades ago and developed by an active research community, variability modelling became the key concern in SPL engineering and important research topic in software engineering in general. Therefore, a lot of efforts of the SPL community were in this direction. As a result, the number of variability modelling approaches proposed by research or industry quickly increased. Such techniques are needed in ever growing number of applications, from complex manufacturing activities to online configurators needed for e-commerce websites. Thus, it is of the utmost importance to review VM techniques and to understand their fundamental characteristics in order to choose the most appropriate one for a particular application context. The classification provided in this paper is a first step in this direction, outlining major trends in variability modelling and declining them at the metamodel and model levels. Future work includes the evaluation of the surveyed approaches against a set of criteria enabling a fine-grained comparison and giving practical insights to engineers who need to ground their decisions. We also plan to apply the surveyed approaches on different examples, which would a allow for a more pertinent comparison and also point out the relative advantages and disadvantages of each individual approach.

References

1. Software product line conference - hall of fame. `http://splc.net/fame.html`
2. Apel, S., Janda, F., Trujillo, S., Kästner, C.: Model superimposition in software product lines. In: ICMT '09: Proceedings of the 2nd International Conference on Theory and Practice of Model Transformations. pp. 4–19. Springer-Verlag, Berlin, Heidelberg (2009)

3. Asikainen, T., Soininen, T.: Modelling methods for managing variability of configurable software product families (2004)
4. Atkinson, C., Bayer, J., Muthig, D.: Component-based product line development: the kobra approach. In: Proceedings of the first conference on Software product lines : experience and research directions: experience and research directions. pp. 289–309. Kluwer Academic Publishers, Norwell, MA, USA (2000)
5. Batory, D.S.: Feature models, grammars, and propositional formulas. In: SPLC. pp. 7–20 (2005)
6. Bayer, J., Flege, O., Gacek, C.: Creating product line architectures. In: IW-SAPF. pp. 210–216 (2000)
7. Bayer, J., Gerard, S., Haugen, Ø., Mansell, J.X., Møller-Pedersen, B., Oldevik, J., Tessier, P., Thibault, J.P., Widen, T.: Consolidated product line variability modeling. In: Software Product Lines, pp. 195–241 (2006)
8. Benavides, D., Segura, S., Ruiz-Cortés, A.: Automated analysis of feature models 20 years later: A literature review. Information Systems 35(6), 615 – 636 (2010), `http://www.sciencedirect.com/science/article/pii/S0306437910000025`
9. Bosch, J., Florijn, G., Greefhorst, D., Kuusela, J., Obbink, J.H., Pohl, K.: Variability issues in software product lines. In: PFE '01: Revised Papers from the 4th International Workshop on Software Product-Family Engineering. pp. 13–21. Springer-Verlag, London, UK (2002)
10. Chen, L., Ali Babar, M., Ali, N.: Variability management in software product lines: a systematic review. In: Software Product Line Conference. pp. 81–90. Carnegie Mellon University, Pittsburgh, PA, USA (2009)
11. Clauss, M.: Generic modeling using uml extensions for variability. In: OOPSLA (2001)
12. Coplien, J., Hoffman, D., Weiss, D.: Commonality and variability in software engineering. IEEE Software 15(6), 37–45 (1998)
13. Czarnecki, K., Antkiewicz, M.: Mapping features to models: A template approach based on superimposed variants. In: GPCE. pp. 422–437 (2005)
14. Dhungana, D., Grünbacher, P., Rabiser, R.: The dopler meta-tool for decision-oriented variability modeling: a multiple case study. Automated Software Engineering 18(1), 77–114 (2010)
15. Djebbi, O., Salinesi, C.: Criteria for comparing requirements variability modeling notations for product lines. In: Proceedings of the Fourth Internationa Workshop on Comparative Evaluation in Requirements Engineering. pp. 20–35. IEEE Computer Society, Washington, DC, USA (2006)
16. Elena Alana, A.R.: Domain engineering methodologies survey. Tech. rep., CORDET (2007)
17. European Software Engineering Institute Spain, IKV++ Technologies AG Germany: Master: Model-driven architecture instrumentation , enhancement and refinement. Tech. Rep. IST-2001-34600, IST (2002)
18. Gomaa, H., Shin, M.E.: Multiple-view modelling and meta-modelling of software product lines. IET Software 2(2), 94–122 (2008)
19. Halmans, G., Pohl, K.: Communicating the variability of a software-product family to customers. Inform., Forsch. Entwickl. 18, 113–131 (2004)
20. Harsu, M.: A survey on domain engineering. Tech. rep., Institute of Software Systems, Tampere University of Technology (2002)
21. Haugen, .o., Pedersen, B.M., Oldevik, J.: Comparison of System Family Modeling Approaches. In: Software Product Lines, 9th International Conference. Lecture Notes in Computer Science, vol. 3714, pp. 102–112. Springer (2005)
22. Haugen, O., Moller-Pedersen, B., Oldevik, J., Olsen, G.K., Svendsen, A.: Adding standardized variability to domain specific languages. Software Product Line Conference, International 0, 139–148 (2008)

23. Jean-Christophe TRIGAUX, P.H.: Modelling variability requirements in software product lines: a comparative survey. Tech. rep., FUNDP Namur (2003)
24. Jézéquel, J.M.: Model driven design and aspect weaving. Software and System Modeling 7(2), 209–218 (2008)
25. Kang, K.C., Cohen, S.G., Hess, J.A., Novak, W.E., Peterson, A.S.: Feature-oriented domain analysis (foda) feasibility study. Tech. rep., Carnegie-Mellon University Software Engineering Institute (November 1990)
26. Khurum, M., Gorschek, T.: A systematic review of domain analysis solutions for product lines. J. Syst. Softw. 82 (December 2009)
27. Laguna, M.A., González-Baixauli, B.: Product line requirements: Multi-paradigm variability models. In: 11th Workshop on Requirements Engineering WER (2008)
28. Maccari, A., Heie, A.: Managing infinite variability in mobile terminal software: Research articles. Softw. Pract. Exper. 35(6), 513–537 (2005)
29. Mansell, J.X., Sellier, D.: Decision model and flexible component definition based on xml technology. In: PFE. pp. 466–472 (2003)
30. Metzger, A., Pohl, K., Heymans, P., Schobbens, P.Y., Saval, G.: Disambiguating the documentation of variability in software product lines: A separation of concerns, formalization and automated analysis. Requirements Engineering, IEEE International Conference on 0, 243–253 (2007)
31. Morin, B., Perrouin, G., Lahire, P., Barais, O., Vanwormhoudt, G., Jézéquel, J.M.: Weaving variability into domain metamodels. In: MoDELS. pp. 690–705 (2009)
32. Mujtaba, S., Petersen, K., Feldt, R., Mattsson, M.: Software product line variability: A systematic mapping study (2008)
33. Myllymäki, T.: Variability management in software product lines. Tech. rep., Tampere University of Technology Software Systems Laboratory ARCHIMEDES (2001)
34. Northrop, L.: A framework for software product line practice. In: Proceedings of the Workshop on Object-Oriented Technology. pp. 365–376. Springer-Verlag London, UK (1999)
35. de Oliveira Junior, E.A., de Souza Gimenes, I.M., Huzita, E.H.M., Maldonado, J.C.: A variability management process for software product lines. In: CASCON. pp. 225–241 (2005)
36. Perrouin, G., Klein, J., Guelfi, N., Jézéquel, J.M.: Reconciling automation and flexibility in product derivation. In: SPLC '08: Proceedings of the 2008 12th International Software Product Line Conference. pp. 339–348. IEEE Computer Society, Washington, DC, USA (2008)
37. Perrouin, G., Vanwormhoudt, G., Morin, B., Lahire, P., Barais, O., Jézéquel, J.M.: Weaving variability into domain metamodels. Software and Systems Modeling pp. 1–23 (2010)
38. Pohl, K., Böckle, G., van der Linden, F.J.: Software Product Line Engineering: Foundations, Principles and Techniques. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2005)
39. Schmid, K., John, I.: A customizable approach to full lifecycle variability management. Sci. Comput. Program. 53, 259–284 (December 2004)
40. Sinnema, M., Deelstra, S.: Classifying variability modeling techniques. Inf. Softw. Technol. 49 (July 2007)
41. Sinnema, M., Deelstra, S., Hoekstra, P.: The covamof derivation process. In: ICSR. pp. 101–114 (2006)
42. Sinnema, M., Deelstra, S., Nijhuis, J., Bosch, J.: Covamof: A framework for modeling variability in software product families. In: SPLC. pp. 197–213 (2004)
43. Svahnberg, M., Gurp, J.V., Bosch, J.: A taxonomy of variability realization techniques. Software Practice and Experience 35, 705–754 (2005)
44. Trigaux, J.C., Heymans, P.: Modelling variability requirements in software product lines: a comparative survey. Tech. rep., University of Namur, Computer Science Institute (2003)
45. Ziadi, T., Jézéquel, J.M.: Software Product Lines, chap. Product Line Engineering with the UML: Deriving Products, pp. 557–586. Springer Verlag (2006)