Extending the SPES Modeling Framework for Supporting Role-specific Variant Management in the Engineering Process of Embedded Software

Tobias Kaufmann[1], Christian Manz[2], Thorsten Weyer[1]

[1]University of Duisburg-Essen
paluno – The Ruhr Institute for Software Technology
Gerlingstraße 16
45127 Essen, Germany
{tobias.kaufmann, thorsten.weyer}@paluno.uni-due.de

[2]Daimler AG
Research and Development
Wilhelm-Runge-Straße 11
89081 Ulm, Germany
christian.c.manz@daimler.com

Abstract: In many application domains embedded systems and the corresponding embedded software face an increase in sometimes mutual excluding stakeholder needs like requests from different customers or national legal obligations. In order to meet these needs variability is explicitly designed into the embedded software. Nowadays, in the engineering process of embedded software the variability information is explicitly documented in a dedicated variability model. Hence, the variability model comprises multiple variability-related concerns that are specific to different roles in the lifecycle of embedded software. Each role (e.g. product manager, requirements engineer, architect, maintenance engineer) requires only a specific subset of the variability information that is documented in the variability model to fulfill their certain responsibility. As a consequence, mechanisms for structuring the variability model with respect to the specific role-based variability-concerns are needed. In this paper we present an extension of a well-known model-based engineering framework for embedded software (the SPES Modeling Framework) in order to structure the overall variability model of the embedded software with respect to role-based variability-concerns.

1. Introduction

In many application domains (e.g. in the automotive domain) the management of variants of embedded software becomes increasingly important to address the different needs that are demanded from various stakeholders like customers, users or national authorities for legislation. In order to cope with the manifold and sometimes mutual exclusive stakeholder needs, the variability of the embedded software is explicitly considered with respect to the engineering artifacts that are created during the engineering process (e.g. requirements, functional design, and technical architecture).

This requires an explicit and continuous management of the different variant of the embedded software throughout the engineering process, or even better, throughout the whole lifecycle of the corresponding embedded software.

In recent years a consortium of 21 partners from academia and industry has developed the SPES Modeling Framework (or short: SPES MF, cf. [Br12]) that aims at supporting the seamless model-based engineering of embedded software. The SPES MF is built upon two powerful software engineering principles: "separation of concerns" and "divide and conquer" (cf. [GJM03]). The principle "separation of concerns" is manifested by distinguishing between the four different viewpoints: *requirements*, *functional design*, *logical architecture* and *technical architecture*. Each of these SPES viewpoints focuses on a set of role-specific concerns in the engineering process of embedded software. For instance, the requirements viewpoint addresses the concerns of the role "requirements engineer" in the engineering process, since he/she is responsible for the requirements specification of the embedded software. "Divide and conquer" is realized by multiple layers of systems' granularity. Typically, the granularity layers are defined by systematically decomposing the embedded software into ever more fine-grained building blocks like subsystems, components and system elements. Therefore, the coarse-grained engineering "problem" is step-wise decomposed into fine-grained engineering problems that are regarded in distinct engineering processes. Each of those engineering processes is in turn structured by the four SPES viewpoints (cf. [Br12]).

Originally, variability and variants were not considered in the SPES MF. Therefore, in [HKW13] we proposed a general solution idea how to extend the SPES MF in order to enable an explicit variant management throughout the engineering process of embedded software. Since variability has a crosscutting nature with respect to the embedded software we introduced the concept *variability perspective* that is orthogonal to the viewpoints and granularity layers of the original SPES MF which is described in [Br12]. The different roles in the lifecycle of embedded software require specific excerpts of the variability information to fulfill their responsibilities. To take this into account appropriately, we extend the concept of the variability perspective by suggesting an approach, which allows for a role-specific tailoring of the variability information.

The remainder of the paper is structured as follows: In Section 2, we present the conceptual basis for our approach. Section 3 explains requirements and concerns for structuring the variability perspective of the SPES MF. Then we introduce the concept for structuring the variability perspective and explain its applicability by using a variability model extract of an Advanced Driver Assistance System. We will then discuss the effects of the proposed concept in Section 4. The related work will be discussed in Section 5 and our conclusion will be presented in Section 6.

2. Fundamentals

The SPES MF is based on an architecture framework that is proposed in [III11]. In order to describe the extension concept for addressing role-specific variability-concerns we give some insights in the IEEE Std. 42010 and the SPES MF as well as in the general

idea of extending the SPES MF by a variability perspective that has already been published in [HKW13].

2.1. IEEE Std. 42010-based Viewpoint Specifications

IEEE Std. 42010 [III11] provides a conceptual framework for defining *viewpoints*, allowing separation of concerns in an architectural description. Viewpoints cover multiple concerns of different stakeholders (e.g. the logical system architecture). A viewpoint is a specification supporting the structured derivation one *view* on a system under development. Hence, multiple viewpoints are required to fully describe the architecture of such systems. The viewpoints also specify how they are interrelated w.r.t. concerns cutting across multiple viewpoints. IEEE Std. 42010 allows sharing architectural artifacts across multiple views and hence considers crosscutting concerns. This concept can be interpreted as one possible implementation of the concept *perspective*, which was introduced by ROZANSKI and WOODS [RW12]. A perspective is defined as *"[...] a collection of architectural activities, tactics, and guidelines that are used to ensure that a system exhibits its particular set of related properties that require consideration across a number of the system's architectural views"* and allows the orthogonal consideration of crosscutting concerns w.r.t. the viewpoints.

2.2. The SPES Modeling Framework

The SPES Modeling Framework (cf. [Br12]) consists of four viewpoints: The *Requirements Viewpoint* addresses the structured documentation and analysis of requirements. The *Functional Viewpoint* addresses the structured documentation and analysis of system functions. The L*ogical Viewpoint* addresses the structured documentation and analysis of the logical solution, whereas the *Technical Viewpoint* addresses the structured documentation and analysis of the technical solution. All four viewpoints cover multiple layers of granularity, which can be individually defined (see figure 1) according to the needs of the engineering process. A new granularity layer is created, whenever a coarse-grained engineering artifact is decomposed into multiple finer grained engineering subjects. For each of the engineering subjects the four viewpoints are applied to ensure a structured engineering path on the lower granularity layers. In addition, the SPES MF explicitly considers crosscutting system properties (e.g. safety, real-time).

2.3. Variability Modeling in the SPES Modeling Framework

Variability is a first class concept and needs to be documented explicitly in a separate variability model (cf. [CN02], [PBL05]). This paradigm originates from the software product line community and is based on two ontological concepts: *variability subject* and *variability object*. The *variability subject* is defined as a variable item of the real world or a variable property of such an item, e.g. the paint of a car. The *variability object* is defined as a particular instance of a variability subject, e.g. red paint (cf. [PBL05]). A software-variant is constituted of a selected set of variability objects. These two concepts are supported by multiple relationships between variability subjects and variability ob-

jects. Dependencies are defined to express optional or mandatory variability objects and alternative groups. Constraints between variability subjects and variability objects are used to express *requires*- or *excludes*- relationships. Variability modeling is essential to continuous variant management in the engineering of variable software for embedded systems, because the variability cuts across the engineering process. Thus, all SPES viewpoints and several roles participating in the engineering of such systems are affected. In [HKW13] the SPES MF was extended by the *variability perspective* to consider the crosscutting nature of variability. The variability perspective documents the variability information orthogonally to the SPES viewpoints in one variability model. This variability perspective can be understood as an instantiation of the concept *perspective* proposed by ROZANSKI and WOODS [RW12] and defines the ontological concepts for variability modeling in the SPES MF in accordance to [PBL05]. In consequence, the variability perspective does not prescribe specific variability modeling languages, which leaves the concept open to project or company specific instantiations.

3. Structuring the Variability Perspective

In this section we will elaborate on requirements for structuring the variability perspective and present concerns related to specific roles in an engineering process. Based on these requirements and concerns we propose an IEEE Std. 42010-compliant approach to structure the variability perspective of the SPES MF. We demonstrate the application of the approach by a simplified example from the automotive domain.

3.1. Requirements and Concerns for Structuring the Variability Perspective

ISO/IEC Std. 15504 (cf. [II06]) and domain specific derivations like automotive SPICE require the definition of specific responsibilities and authorities in an engineering process (cf. [Aut10]) to manage and develop engineering artifacts. Responsibilities are represented by roles in an engineering process (cf. [So11]) and hence essential to each engineering process. In the setting of variable software for embedded system, the responsibility for an engineering artifact also covers its variability. Hence, this responsibility causes e.g. the role EE-System Architect to retrieve a certain subset of the variability information, resulting in role-based variability-concerns. Consequently, we consider role-based variability-concerns as a key driver for structuring the variability information and hence the variability model. Based on the experiences made in several development projects of the automotive industry we derive the following requirements for structuring the variability perspective:

(R1) Reduce the variability model complexity by neglecting irrelevant variability information for a determined SPES viewpoint, granularity layer or role.

(R2) Define role-specific variability views to support the communication between different roles (e.g. requirements engineers and software designers).

These requirements address the required capabilities of an approach for structuring the variability information. In other words R1 and R2 are necessary to realize a variability-

related concern based structuring of the variability model. The structuring of the variability model is based on role-specific variability-concerns, which we identified, when working in projects that applied automotive SPICE.

(C1) Which variability information is required by a specific role?

(C2) Which variability information is required by a specific role w.r.t. a specific granularity layer e.g. of a subsystem?

(C3) Which variability information is the required by a specific role w.r.t. a specific SPES viewpoint?

The three concerns (C1 – C3) need detailing to support development, maintenance, change impact analysis and software defect detection activities of an engineering process. C1 is the most generic concern. Hence, we understand C2 and C3 as more detailed concerns. C2 potentially covers the variability information of multiple SPES viewpoints, whereas C3 potentially covers the variability information of multiple granularity layers. From the requirements and concerns it follows that a comprehensive view concept on variability models, is necessary (cf. [MSR13]).

3.2. Structuring the Variability Perspective by Variability Viewpoints

Views on instances of variability models focus on specific variability-related concerns, which are documented in the model. Hence, views allow for analyzing a concern in isolation to get a deep understanding of this concern (cf. [GJM03]). This notion led us to introduce variability viewpoints based on the core concepts of IEEE Std. 42010: *stakeholder-specific concern* and *viewpoint* as a specification for a *view*. A *variability view* can be understood as a role-specific excerpt of information from the variability perspective. A *variability viewpoint* that addresses a role-specific variability-concern specifies which information of the variability perspective is needed by the corresponding role to be able to fulfill its responsibilities. Typically, different variability viewpoints share the same ontology for modeling variability information namely a variability model. In accordance to the requirements R1 and R2, and the concerns C1 – C3 we propose using role-based variability-concerns.

These concerns are the conceptual fundament for specifying variability viewpoints. Thus, we explicitly relate the concerns of roles to variability viewpoints and thereby allow a role-based structuring of the variability perspective. This structure can be independent from the structure, which is realized by the SPES viewpoints and their corresponding concerns. In doing so, we provide a structuring mechanism for the variability perspective. Essentially, the variability viewpoints provide role-based projections, inspired by relational database theory (cf. [EN11]), on the relation of the variability information *VI* (see figure, 1 top layer circular elements).

For each role-based variability-concern *rc* the corresponding variability view is defined as a projection Π_{rc} on the variability information (*VI*) within the variability perspective:

$$\prod_{rc}(VI) = \left(VP_{rc}, VA_{rc}, \mathcal{R}el_{VP\otimes VP_{rc}}, \mathcal{R}el_{VP\otimes VA_{rc}}, \mathcal{R}el_{VA\otimes VA_{rc}}\right)$$

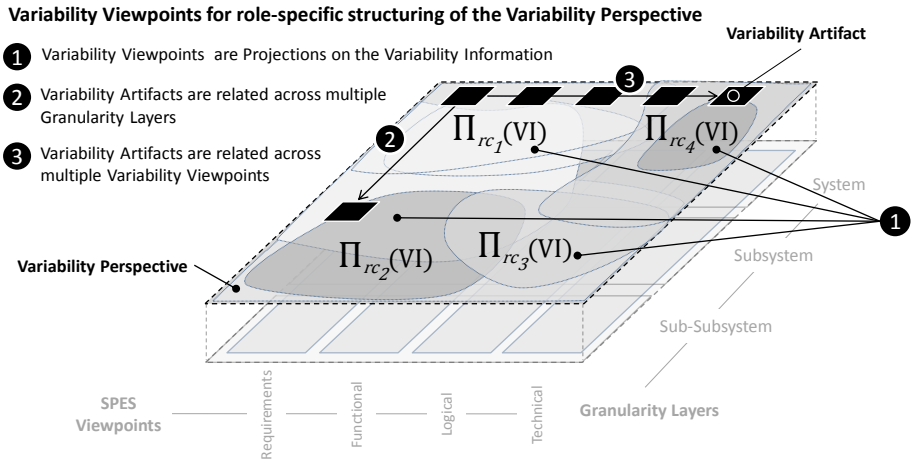

Figure 1: Variability Viewpoints are projections on the Variability Perspective

Here *VI* is defined as the set of all explicitly documented variability information within the variability perspective:

$$VI = \left(VP, VA, \mathcal{R}e\ell_{VP\otimes VP}, \mathcal{R}e\ell_{VP\otimes VA}, \mathcal{R}e\ell_{VA\otimes VA}\right) | \mathcal{R}e\ell_{A\otimes B} \subseteq A \times B$$

In the expression above *VP* is defined as the set of all variation points (a variation point is the manifestation of a variability subject), *VA* is defined as the set of all variants (a variant is the manifestation of a variability object) and $\mathcal{R}e\ell_{A\otimes B}$ is the set of all relationships between the elements of the sets *A* and *B*. The relation $\mathcal{R}e\ell_{A\otimes B}$ is defined based on the set of different *semantics of relationships* $\mathbb{s}$ between elements of the sets *A* and *B*. Given *n* different semantics $\mathbb{s}_1, \ldots, \mathbb{s}_n$ otf relationships between the elements of *A* and *B* then $\mathcal{R}e\ell_{A\otimes B}$ is defined as:

$$\mathcal{R}e\ell_{A\otimes B} = \bigcup_{j=1}^{n} \mathcal{R}e\ell_{A\otimes B}^{\mathbb{s}_j}$$

For instance, let the *requires*-relation between a set VP of four variation points *vp1*,*vp2*,*vp3*,*vp4* be defined as:

$$\mathcal{R}e\ell_{VP\otimes VP}^{requires} = \left((vp1, vp3), (vp2, vp4)\right)$$

The relation above states that the variation point *vp*1 *requires* the variation point *vp*3 and variation point *vp*2 *requires* variation point *vp*4. Thus, vp1 *requires vp*3 means that when deciding upon the variants (i.e. the binding of variants) of *vp*1 also a corresponding decision concerning the variants (i.e. the binding of variants) of *vp*3 has to be made.

3.3. Industrial Example

The following example represents a small part of an Advanced Driver Assistant System (or short: ADAS) variability model. The complete variability model comprises several hundred features, which is a common size of variability models for automotive systems like engine control or electric drive. The ADAS supports a car driver in usual traffic scenarios to increase comfort and safety. Thus, it provides multiple functions (e.g., cruise control or brake assist, cf. figure 2) with individual dependencies (e.g., adaptive cruise control requires signals of the high-end EE-architecture). The ADAS is offered in different vehicles classes (e.g., mid-size or luxury) and multiple markets (e.g. Europe or NAFTA-North American Free Trade Agreement). Figure 2 visualizes the ADAS variability information *VI* using the Orthogonal Variability Model variability modeling language (cf. [PBL05]).

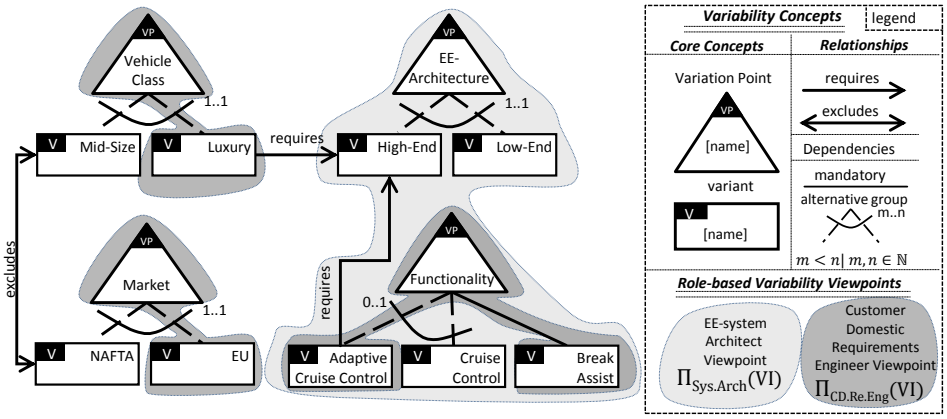

Figure 2: Advanced Driver Assistance System Variability Model

Multiple roles are involved in the engineering process and marketing of the ADAS. Two of them are the *EE-System Architect* (or short: Sys.Arch) and the *Customer Domestic Requirement Engineer* (or short: CD.Re.Eng). The role Sys.Arch is responsible for designing and maintaining the architecture of an EE-system and takes responsibility for the internal characteristics and variability of an EE-system. The set of role-based variability-concerns that is associated with the role Sys.Arch is named Sys.Arch. The variability viewpoint is denoted as $\prod_{Sys.Arch}(VI)$ in figure 2. In contrast, the role CD.Re.Eng manages the requirements of specific markets and defines the market individual characteristic of an EE-System (e.g., in scope of after sales). The set of role-based variability-concerns that is associated with the role CD.Re.Eng is named RC.CD.Re.Eng. The variability viewpoint is denoted as $\prod_{CD.Re.Eng}(VI)$ in figure 2.

According to IEEE Std. 42010, viewpoints are required to be documented explicitly. Thus, based on the concerns of the role Sys.Arch and CD.Re.Eng two different variability viewpoints can be defined. Each has a *unique name* (e.g. vv.Sys.Arch and vv.CD.Re.Eng) and focuses on *role-specific concerns* such as the technical variability of the EE-System of the ADAS (vv.Sys.Arch) and the market specific variability in terms of different functionality and behavior of the ADAS (vv.CD.Re.Eng). Thus, both

viewpoints use a role-specific subset of the variability information documented in the variability model (e.g. Feature Model, Orthogonal Variability Model) and use the same model for representing the variability information. These variability viewpoints can be used to derive the views visualized in figure 2.

4. Discussion

The proposed approach (cf. section 3) impacts the engineering artifacts. Thus, we discuss possible impacts and challenges for a required evaluation.

Relation to Engineering Artifacts: The proposed view concept for variability model impacts the related engineering artifacts, because information documented by these artifacts is automatically tailored according to the tailored variability view. This is due to the relation of the variability model and the corresponding engineering artifacts. Today it is not clear whether the tailoring of variability information can be transferred to the engineering artifacts. Further research in this area is necessary.

Evaluation: As our work is in an early stage, it requires evaluation. The effects of overlapping viewpoints need to be studied in detail. One reason for overlapping variability viewpoints is the overlap of responsibilities in an engineering process, which can lead to discussions on the variability information of interest to multiple roles.

5. Related Work

Regarding the realization of variability viewpoints *annotative* approaches augment variability model elements with additional information, which are used to create views on the variability model. In SCHROETER et al. [SLW12] feature models consist of attributed features. Viewpoints are defined based on these attributes. In contrast to *annotative* approaches, *descriptive* approaches specify sets of variability information and use them to define which variability information is part of a view. The work of FEY et al. [FFB02] proposes using feature sets to group features based on the needs on domain experts. But this so called *feature set plane* disregards the dependencies between features. In HUBAUX et al. [Hu13] a slicing operator based on sets of features for feature models is proposed to create multiple different views on a feature model. The approach focuses solely on feature-based configuration and proposed three different visualizations of not accessible features. THOMPSON and HEIMDAHL [TH03] use a set based approach to structure the multi-dimensional product lines allowing for different views on the software product line under development, but the approach is not explicitly related to role-specific variability-concerns. CZARNECKI et al. [CHE05] propose a staged configuration approach in which the three dimensions *time* (cf. engineering stages), targets (cf. subsystems) and roles (cf. responsibilities) can be used to define successive configuration stages. Moreover, this approach focuses solely on feature-based configuration. In contrast to this approach, variability viewpoints do not necessarily cover specific configuration stages. MUTHIG and SCHROETER [MS13] describe an approach, which uses

role-specific views to filter and manage access to feature information to support feature life cycle management, which is concerned with the documentation and evolution of features.

6. Conclusion and Future Work

In this paper we explained why it is not sufficient to rely on using SPES viewpoints as the structuring mechanisms for the variability perspective (cf. [HKW13]) in section 1 and explained the conceptual foundations in section 2. We introduced an approach to structure the variability perspective based on role-based variability-concerns. The applicability of this concept was demonstrated by an industrial example (cf. section 3). In section 4, we discussed the impact of variability views on the related engineering artifacts and discussed that additional studies are required to evaluate the approach. In future work, we plan to examine existing view-building techniques and evaluate the proposed approach in scenarios close to industrial practice to get deeper insights into their benefits and shortcomings. Inconsistencies between overlapping variability viewpoints (cf. [MSA09]) need to be also targeted by future work.

Acknowledgement

This paper was partially funded by the BMBF project SPES 2020_XTCore under grant 01IS12005C and the DFG project KOPI grant PO 607/4-1. We would like to thank André Heuer and Vanessa Stricker for their fruitful comments regarding this paper.

References

[Aut10] Automotive SPICE® Process Assessment Model Version 2.5, The SPICE User Group, 2010

[Br12] Broy, M. et al.: Model-Based Engineering of Embedded Systems – The SPES 2020 Methodology, Springer, Berlin, Heidelberg, 2012; pp. 31-48

[CHE05] Czarnecki, K.; Helsen, S.; Eisenecker U.: Staged configuration through specialization and multilevel configuration of feature models. In (van Ommering, R.; Weiss, D. Eds.): Journal of Software Process: Improvement and Practice, 2005, 10; pp. 143-169

[CN02] Clements, P.; Northrop, L.: Software Product Lines – Practices and Patterns. Addison-Wesley, Boston, 2002

[EN11] Elmasri, R.; Navathe S.: Fundamentals of database systems. Boston: Addison-Wesley, 2011

[FFB02] Fey, D.; Fajta, R.; Boros, A.: Feature modeling: A meta-model to enhance usability and usefulness. In (Chastek, G. J. Ed.): Proc. 2nd Software Product Lines Conference, San Diego, 2002. Springer, Lecture Notes in Computer Science, 2002; pp. 198-216

[GJM03] Ghezzi, C.; Jazayeri, M.; Mandrioli, D.: Fundamentals of software engineering. Upper Saddle River, N.J: Prentice Hall. 2003; pp. 44-49

[HHB08] Hubaux, A.; Heymans, P.; Benavides, D.: Variability modeling challenges from the trenches of an open source product line re-engineering project. In (Geppert, B.; Pohl, K.

Eds.): Proc. 12th Software Product Lines Conference, Limerick, 2008. IEEE Computer Society, 2008; pp. 55-64

[Hu13] Hubaux A. et al.: Supporting multiple perspectives in feature-based configuration. In (France, R.; Rumpe, B. Eds.): Journal of Software & Systems Modeling, Volume 12, Issue 3, Springer, New York, 2013

[HKW13] Heuer, A.; Kaufmann T.; Weyer, T.: Extending an IEEE 42010-compliant Viewpoint-based Engineering-Framework for Embedded Systems to Support Variant Management. In (Schirner, G. et al. Eds.): Proc. 4th International Embedded Systems Symposium, Paderborn, 2013. Springer, IFIP Advances in Information and Communication Technology, 2013.

[II06] ISO/IEC International Standard Information Technology – Software Process Assessment, ISO/IEC TR Standard 15504:2006, 2006

[III11] ISO/IEC/IEEE Systems and Software Engineering – Architecture description. ISO/IEC/IEEE Standard 42010:2011, 2011

[Ka98] Kang, K. C.; Kim, S.; Lee, J.; Kim, K.; Kim, G. J. & Shin, E.: FORM: A feature-oriented reuse method with domain-specific reference architectures. In (Frakes, W. Ed.): Annals of Software Engineering, 5, 1998; Springer, Berlin, Heidelberg, 1998; pp. 143-168

[LKK04] Lee, J.; Kang, K. C.; Kim, S.: A feature-based approach to product line production planning. In (Nord, R. L. Ed.): Proc. 3rd Software Product Lines Conference, Boston, 2004. Springer, Lecture Notes in Computer Science, 2004; pp. 183-196

[MS13] Muthig, D.; Schroeter, J.: A framework for role-based feature management in software product line organizations. In (Kishi, T.; Jarzabek, S. ; Gnesi, S. Eds.): 17th International Software Product Line Conference, Tokyo, 2013. ACM 2013; pp. 178-187

[MSA09] Mannion, M.; Savolainen, J.; Asikainen, T.; Viewpoint-oriented variability modeling. In (Ahamed, S. I. et al. Eds.): Proc. 33rd Annual IEEE International Computer Software and Applications Conference, Seattle, 2009. IEEE Computer Society, 2009; pp. 67-72

[MSR13] Manz, C.; Stupperich, M.; Reichert, M.: Towards Integrated Variant Management in Global Software Engineering: An Experience Report. In: Proc. 8th IEEE Int. Conf. on Global Software Engineering, Bari, 2013; IEEE 2013

[PBL05] Pohl, K.; Böckle, G.; van der Linden, F.: Software Product Line Engineering: Foundations, Principles and Techniques. Springer, Berlin, Heidelberg, 2005

[RW12] Rozanski, N.; Woods, E.: Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives. 2nd edition, Addison-Wesley, Upper Saddle River, 2012

[SLW12] Schroeter J.; Lochau M.; Winkelmann T.: Multi-Perspectives on Feature Models. In (France R. B. et al. Eds.): Proc. 15th Model Driven Engineering Languages and Systems Conference, Innsbruck, 2012. Springer Berlin, Heidelberg, 2012; pp. 252-268

[So11] Sommerville, I.: Software engineering. Boston: Pearson, 2011

[TH03] Thompson, J. M., Heimdahl M.P.E.: Structuring product family requirements for n-dimensional and hierarchical product lines. In (Loucopoulos, P.; Mylopoulos, J. Eds.): Requirements Engineering Journal, Volume 8, Issue 1, Springer, Berlin, Heidelberg, 2003; pp. 42-54.

[Yu08] Yu, Y. et al.: From goals to high-variability software design. In (An A. et al. Eds.): Proc. 17th International Symposium Foundation of Intelligent System, Toronto, 2008. Springer, 2008 Lecture Notes in Computer Science; pp. 1–16