Transdisciplinary Lifecycle Analysis of Systems
R. Curran et al. (Eds.)

doi:10.3233/978-1-61499-544-9-194

Model-Based Variant Management with v.control

Christopher JUNK [a] and Robert RÖßGER [a] and Georg ROCK [b] and Karsten THEIS [c] and Christoph WEIDENBACH [d] and Patrick WISCHNEWSKI[a,1]
[a] *Logic4Business GmbH, Saarbrücken, Germany*
[b] *Hochschule Trier, Germany*
[c] *PROSTEP AG, Darmstadt, Germany*
[d] *Max-Planck Institute for Informatics, Saarbrücken, Germany*

Abstract. Manufacturers of products that are instances of variants out of a complex product portfolio have learnt that a rigid process management is mandatory to meet today's standards of quality. An important part are processes that aim at mastering variant complexity. v.control supports these by providing for the first time both a complex product model able to represent detailed engineering, manufacturing, logistics, finance and marketing data in the very same model and a workbench of provably mathematically correct and rigid analysis tools.You want to know whether product changes performed by different engineers are compatible? Press a button and v.control guarantees consistency of all product variants. You want to know whether all your products suggested by marketing can actually be build? Press a button and v.control checks your portfolio and detects problematic variants. You are searching for a product meeting partial customer requirements and being optimal in profit? Press a button and v.control provides the optimal product cash cow. You want to make sure that your product portfolio meets future environmental regulations? Press a button and v.control identifies opportunities. You want to engineer shared parts of your product line to meet manufacturing inventory requirements? Press a button and v.control designs an optimal solution. This paper presents a detailed overview of the functionality of v.control as well as typical industrial applications successfully conducted with the help of v.control. It addresses current research in the field of complexity management, variability management and SAT-solving and their functional integration within v.control.

Keywords. Variant Management, PLM, Product Mangement

Introduction

Managing complex product portfolios is one of the major challenges of manufacturers today. This is because there is a range of requirements for each individual product concerning, for example, quality, safety, market demands, compliance with legal requirements, time-to-market and product costs. This is even more challenging because the products are instances of huge product portfolios and, therefore, it is not possible to build and test each product manually if it meets the requirements. In particular market

[1] Corresponding Author, Mail: Patrick.wischnewski@logic4business.com

demands for individual products have increased in recent years and are still increasing. As a consequence, for the manufacturers this means that they need to build a diversity of products in order to address their customers and keep/gain market shares.

Therefore, mastering the product complexity and meeting all the requirements for each of the individual products is the key for obtaining competitive advantages in the future. As a result, it is essential to develop methods and tools that support manufacturers in achieving these goals today [14].

On the background of the increasing demands for individual products, identifying and reducing complexity from the products and processes [8,7,9] can only by one part of a solution. Even after removing "useless" variants there is still a huge complexity remaining that needs to be dealt with in the product life cycle. In [10] feature models are used to model product variability. Based on feature models there exist commercial software tools that aim at managing products and their respective artifacts [1,3]. In addition, there has been research done on developing automatic analysis procedures for feature models [12,5]. However, feature models are restricted to structural relations and cannot represent general product build rules as they are induced, e.g., by engineering requirements or marketing strategies. Furthermore, product model analysis techniques need to be extended to these richer models. In particular, requirements are

- a combination of detailed product building rules from different areas such as engineering, manufacturing, logistics, finance, marketing into a single model (single source of truth)
- support for feature attributes and values enabling a detailed modeling
- analysis and optimization procedures

The model-based variant management method supported by the software tool v.control fulfills all these requirements. It provides, for the first time, both, the ability to model complex products together with their detailed product information in the very same product model and a workbench of efficient, provably mathematically correct [13,11,6] and rigid analysis and optimization procedures. v.control is able to express a diversity of product properties in its model and contains efficient analysis and optimization procedures for it.

This paper presents Model-Based Variant Management from a user and application perspective and is structured as follows: Section 2 briefly presents the Model-Based Variant Management method. Section 3 gives an overview of our software tool v.control and Section 4 presents case studies where we have successfully applied the Model-Based Variant Management method together with v.control.

1. Model-Based Variant Management

This section presents the Model-Based Variant Management approach together with its properties in order to master the complexity of today's variant products. The central part of this method is the product model that is introduced in the first part of this section. The second part describes the steps necessary to implement this method for a product portfolio.

1.1. Product Model

As depicted in Figure 1, a product model Φ is the central part in the Model-Based Variant Management method that has to fulfill three requirements.

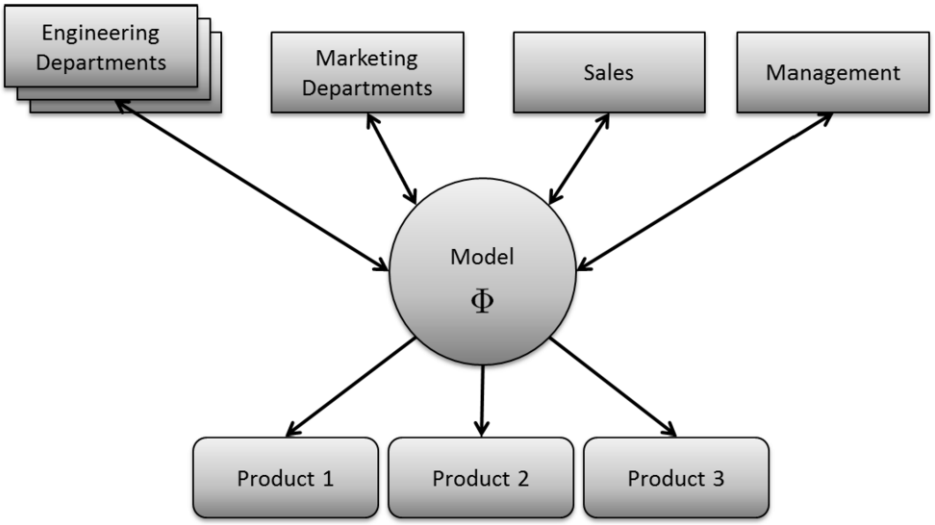

Figure 1. Product Model: Collective Composition that defines the product portfolio

Firstly, the product model Φ combines the requirements, properties, interfaces and dependencies of all stake holders involved in the product life cycle, i.e. engineering, marketing, construction, management, after-sales.

Secondly, the product model Φ is rigid in the sense of being expressed in a precise formal language with a unique semantics which represents the above constituent parts of the product.

Thirdly, the product model Φ enables push button analysis and optimization procedures.

As a consequence, the product model Φ is composed of relevant product information from all involved departments. It is a comprehensive representation of the product portfolio containing all relevant properties of the products. The above specified requirements for the product model are achieved by a formal logic, where a logic is simply a formal and precise language with a fixed, unique semantics. This means that Φ is a set of formulas of a logic. More precisely, any "instantiation" of the product model Φ represents a product in terms of the specified properties. So Φ represents all eventual products in a compact form.

The product model does not only represent all products of the portfolio, it is the basis for a diversity of analysis and optimization operations which can be performed push button. These operations give valuable properties and information about the products back to the stake holders that cannot be obtained otherwise. Important properties of the product model are:

- Is the composition of all product requirements and properties consistent, i.e., can these all be fulfilled at the same time for a real product?

- What are properties of the products that are not explicitly specified but are consequences of the specified properties?
- Are all parts of the product model eventually used at least in one product? (dead feature)
- Are there product configurations with respect to predefined properties?
- Are there redundant rules in the product model, i.e., rules that are not needed for any eventual product?
- What is the optimal product family according to attributes such as cost, profit contribution, time to build with respect to profit, regulations, customer orientation?
- What is an optimal product portfolio with respect to customer requirements?

1.2. Implementing the Method

Implementing this method for a product portfolio involves the following steps:

- Definition of an adequate logic that can express the products, properties, attributes.
- Definition of a translation from available product data to the formal product model.
- Integration into PLM processes and IT-Infrastructures.

Because of the fact that products can be composed of several thousand parts and the analysis and optimization operations are computationally expensive, in general, the definition of the right product model is essential for a successful implementation of the Model-Based Variant Management method. In addition, the product model is defined in terms of the specific properties of the products of a particular customer and adapted to its specific requirements.

1.3. Summary

This section has presented the Model-Based Variant Management approach that combines all product relevant information from all stake holders into one collective model. Because of the fact that this model contains all these information, it can perform sophisticated analysis and optimization operations on the product that give valuable insights to the products that could not be obtained with other methods. In addition, this section has presented the necessary steps in order to implement this method for a product portfolio. The following Section 3 describes our tool v.control that implements the Model-Based Variant Management method and section 4 presents industrial use cases where we have successfully used this method together with v.control.

2. v.control

v.control [2] is a tool supporting the Model-Based Variant Management method. v.control supports a variety of logics for building product models together with the respective analysis and optimization operations. v.control is very flexible and can be easily modified and integrated into existing IT-landscapes and infrastructures.

There are two versions of v.control: A server version that performs scheduled analysis and optimization operations on specified product data and generates respective reports. This version is particularly useful for maintaining a high quality of product data in the development process where the product data is frequently changing.

The second version is a desktop application that provides functionality for interactively exploring the product data, understanding and fixing defects. This version is used to perform a *What-If* analysis on the product data and understanding relationships of the individual parts and their properties. Figure 2 shows a screenshot of the interactive product explorer of v.control.

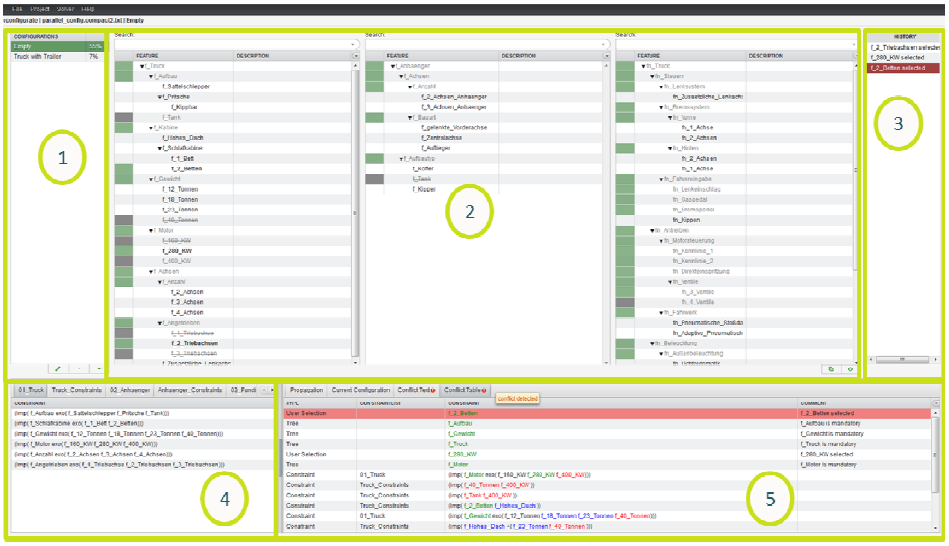

Figure 2. v.control: Interactive Product Explorer

The product explorer is divided into five areas:

1. The list of configurations: A configuration is a temporal modification of the product model. The following modifications are possible:
   - selection/deselection of options
   - activating/deactivating of rules
   - changing of rules
   - adding rules

   These changes do not change the product model itself, it rather stores a temporal change of the product model in the configuration. Consequently, this has no effect to the other configurations. Note, the selection/deselection of options leads from a 150% model that does not represent a complete model to a 100% model that represents a concrete product. This is in contrast to the modification of rules, that change the product model and, therefore, the set of valid products represented by the product model.
2. The product parts arranged in the structure of the product (150% BOM). Because of the fact that v.control has a central product model for all aspects of

the product, there are multiple of such structures possible, for example: One structure for the engineering structure, one for electronics structure and another for the marketing structure.

3. This area is an action history that shows the exact temporal actions performed in a particular configuration on the product model in terms of the actions described above. Removing, reordering and temporarily disabling actions provide interactive exploration possibilities for configuration changes.
4. The area 4 contains all rules of the product model arranged with respect to their source.
5. The area 5 is the output area. In the case of an error the reason for the error is shown here. Otherwise, all consequences and effects to the product parts are indicated. This presents the result of a *What-If* analysis.

v.control can be either used to manage the master data for the product model or it supports the process of changing the data in another master data system by allowing to export a change report. Aside of the product the standard desktop application of v.control allows to perform the following operations:

- Dead-Feature detection, i.e. parts that can never occur in any valid product
- Consistency check, i.e. is the product model itself consistent
- Product Optimization with bounds

In addition to these analysis and optimization operation, v.control supports a variety of product specific analysis operations. These have to be adapted specifically to the structure of the product data.

3. Case Studies

This section presents two case studies that have successfully implemented the Model-Based Variant Management approach together with v.control extended by product specific analysis operations

3.1. Consistency of Marketing and Engineering Product Data.

In this case study, the goal was the implementation of a method that ensures the consistency of the engineering product data with the marketing product data for a globally operating manufacturer.

Figure 3 depicts the implementation of the Model Based Variant Management into the customer process. The engineering product data consists of rules that describe the dependencies between the parts of the product. Likewise, the marketing product data describes the products from a marketing point of view. This means, the marketing departments define rules with respect to their marketing strategy, for example that $part_1$ shall only be sold in combination with $part_2$ and with color red.

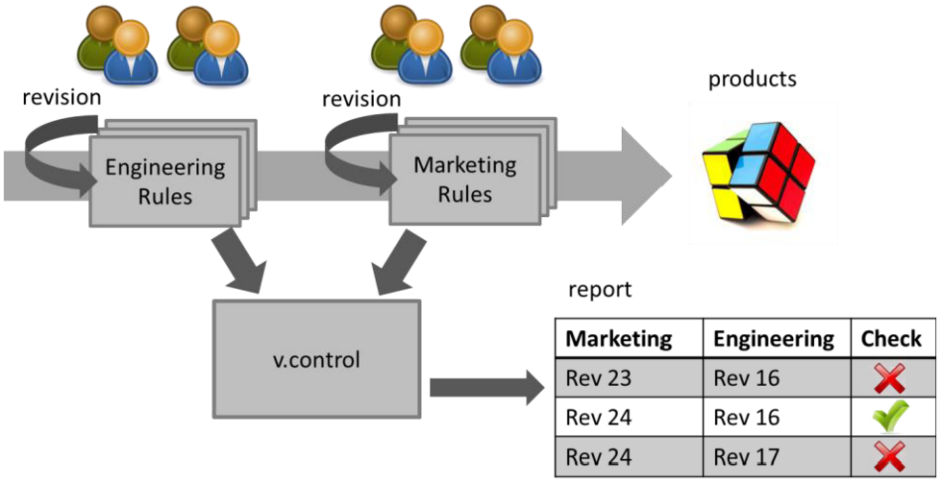

Figure 3. Consistency of engineering and marketing product data

Both, the engineering data as well as the marketing data, are changing daily. Consequently, every day they produce new revisions of the respective data. Because of the fact, that many employees from different areas in the company are involved this is an error prone process. For example, it regularly happened that marketing created offers that actually could not be build. For this reason, they wanted to implement an automatic check verifying that no invalid product was offered and integrate this into their process.

Both, the engineering data as well as the marketing data, are changing daily. Consequently, every day they produce new revisions of the respective data. Because of the fact, that many employees from different areas in the company are involved this is an error prone process. For example, it regularly happened that marketing created offers that actually could not be build. For this reason, they wanted to implement an automatic check verifying that no invalid product was offered and integrate this into their process. The first step towards this goal was the definition of a product model

$$\Phi_{i,\,j} = \Phi_{Ei} \bigcup \Phi_{Mj}$$

Where Φ_{Ei} is created from the engineering data in revision i and Φ_{Mj} from the marketing data in revision j, respectively. Example rules from Φ_{Ei} are "$part_1$ requires $part_2$", "either $part_1$, $part_2$ or $part_3$ is contained in the product", "if the attribute value x of $part_1$ is larger than 5 then $part_3$ is needed". Example rules from Φ_{Mj} are "red products all have $part_1$", "$part_1$, $part_3$ and $part_5$ can only be ordered as a group and then lead to a reduction in price". All these rules have their formal logic counterparts in Φ_{Ei} and Φ_{Mj}, respectively. After the definition of the logical model and the definition of the automatic translation of the product data, the actual analysis operations were implemented. The first check verifies the consistency between the engineering data and marketing data, i.e., all potential products according to the two rule sets can actually be build. In terms of the combined logical product model $\Phi_{i,\,j}$ this means verifying if the product model is consistent:

$$\Phi i, j \nvDash \bot$$

In addition, we implemented another check that verifies if every offered option occurs in at least one valid product. In terms of the combined logical model $\Phi_{i,j}$ that means for all options opt there is an instance of $\Phi_{i,j}$ containing opt:

$$\Phi_{i,j} \cup \{\text{opt}\} \nvDash \bot$$

By appropriate algorithm design this does not need to consider each option separately. After implementing this check, verifying if the engineering data and marketing data are consistent to each other is just the push of a button in v.control or, alternatively, is checked automatically via a scheduled analysis task.

3.2. Car Optimization based on CO2 Emission

The goal of the car optimization based on the CO_2 emission is to find the cars with the best profit by complying with the CO_2 budget regulation of the European Union [4]. The goal of this regulation is the reduction of the average CO_2 emission of passenger cars. If a manufacturer does not comply with the specified budget, they have to pay penalties. Figure 4 depicts the integration of an optimization check with v.control into a PLM system that contains all master data.

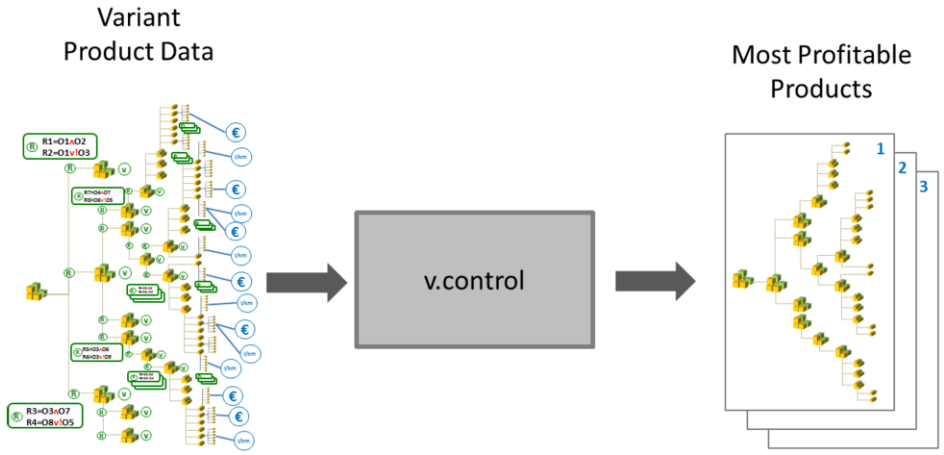

Figure 4. Compute most profitable products with a given CO_2 emission bound

In order to perform this optimization operation based on a part level an extended product model is required. In addition to parts and their relations, attributes of parts like weight, fuel, gas consumption and price is required. The non-trivial objective function defines for each individual product the CO_2 emission and, therefore, the respective optimization procedure is operating on the whole product portfolio. Table 1 shows an example of parts extended with attributes and respective values as they may be stored in a PLM system. The function for computing the CO_2 emission is a function with the following signature which is the objective function for finding the products with the least CO_2 emission: E_{CO_2}:fuel × consumption → ℕ.

Table 1. Example: Parts with attributes

part	weight	fuel	consumption	price
$Engine_1$	90	Petrol	5.1	4000
$Engine_2$	120	Petrol	7.5	6000
$Engine_3$	110	Diesel	3.6	5000
$Extra_1$	30		0.3	250
$Extra_2$	50		0.6	280

In order to compute the CO_2-emission of a particular product (car) with this function, the product must contain an engine because this is the only part that has the attribute *fuel*. v.control verifies for each product if it satisfies the signature of the specified objective functions. Consequently, this defines valid products. The fleet optimization with respect to cost and a given CO_2 emission budget B_{em} is the following operation:

$$\min_{\Psi \sqsubseteq \Phi} \sum_{P \in \Psi} cost(P) \quad \text{with} \quad \frac{\sum_{P \in \Psi} E_{CO_2}(P)}{|\Psi|} \leq B_{em}$$

Note, Ψ is interpreted as a multi-set of products defined by the product model Φ. From 2020 the regulation of the European parliament [4] defines a CO_2 bound for B_{em} of 95 g CO_2/km.

In addition, other non-trivial bounds can be used during an optimization operation. An example for such a bound is the computation of a CO_2 label [4] which relates the CO_2 emission to the weight of a vehicle.

Because of the tight CO_2 regulations, it is crucial to respect the emission individually for each product instead of an emission per vehicle class. This section has depicted an approach based on the model-based variant management method in order to implement a solution for this requirement.

4. Conclusion

This paper has presented a comprehensive approach for rigid process and product management for mastering the complexity of huge variant product portfolios. This approach consists of the method *Model-Based Variant Management* and a respective workbench of of provably mathematically correct and rigid analysis tools.

The method Model-Based Variant Management aims at defining a detailed product model that contains all product relevant data from a diversity of business departments such as engineering, manufacturing, logistics, finance and marketing in the very same model.

This model builds the basis for the rigid analysis operations that are implemented in our tool v.control. It performs these operations push button on the whole product model and returns valuable information back to the respective stake holders which could not be obtained otherwise. These results build the foundations for informed decisions and actions concerning business strategies, corrections of defects, changes in marketing strategies, etc.

On the basis of two use cases we have presented in this paper the potential of the Model-Based Variant Management approach. The first use case implements the approach for the marketing and engineering aspects of the product in order to verify the

consistency between these two. The second use case combines product properties with costs and aims at developing business strategies with respect to CO_2 regulations of the European Parliament.

We have presented model-based variant management from a user and application perspective. Because of the fact that the products of one particular customer have specific properties and requirements, the product model has to be designed and adapted for each customer individually. The specific reasoning techniques that are implemented in v.control and enable the shown detailed analysis are beyond the scope of this paper. In general, almost any rigid analysis of an arbitrary rich product model requires exponential time in the size of the model. However, the real-world product models that we have studied so far enjoy additional structure. In v.control, we explore this additional structure by specific algorithms resulting in a "push-button" behavior for all use cases presented in this paper. With respect to all use cases we considered so far, we have been able to guarantee a response timing of v.control that meets the requirements of the respective use case.

References

[1] Biglever software inc., http://www.biglever.com.
[2] Logic4business gmbh, http://www.logic4business.com.
[3] pure-systems gmbh, http://www.pure-systems.com.
[4] Regulation (ec) no 443/2009 of the european parliament and of the council.
[5] D. Benavides, S. Segura, and A. Ruiz-Corts. Automated analysis of feature models 20 years later: A literature review, *Information Systems*, 35(6):615 – 636, 2010.
[6] D. Dhungana, C.H. Tang, C. Weidenbach, and P. Wischnewski. Automated verification of interactive rule-based configuration systems. In: E. Denney, T. Bultan, and A. Zeller (eds.) *28th IEEE/ACM International Conference on Automated Software Engineering, ASE 2013*, Silicon Valley, CA, USA, November 11-15, 2013, pp. 551–561. IEEE, 2013.
[7] H. ElMaraghy, A. Azab, G. Schuh, and C. Pulz, Managing variations in products, processes and manufacturing systems, *CIRP Annals - Manufacturing Technology*, 58(1):441 – 446, 2009.
[8] H. ElMaraghy, G. Schuh, W. ElMaraghy, F. Piller, P. Schönsleben, M. Tseng, and A. Bernard, Product variety management. *CIRP Annals - Manufacturing Technology*, 62(2):629 – 652, 2013.
[9] W. ElMaraghy, H. ElMaraghy, T. Tomiyama, and L. Monostori, Complexity in engineering design and manufacturing, *CIRP Annals - Manufacturing Technology*, 61(2):793 – 814, 2012.
[10] K.C. Kang, S.G. Cohen, J.A. Hess, W.E. Novak, and A.S. Peterson, *Featureoriented domain analysis (foda) feasibility study*, Technical report, DTIC Document, 1990.
[11] J. Larrosa, R. Nieuwenhuis, A. Oliveras, and E. Rodriguez-Carbonell, A framework for certified boolean branch-and-bound optimization, *Journal of Automated Reasoning*, 46(1):81–102, 2011.
[12] M. Mendonca, A. Wasowski, and K. Czarnecki, Sat-based analysis of feature models is easy. In: D. Muthig and J. D. McGregor (eds.) *SPLC, volume 446 of ACM International Conference Proceeding Series*, pp. 231–240. ACM, 2009.
[13] R. Nieuwenhuis, A. Oliveras, and C. Tinelli, Solving SAT and SAT modulo theories: From an abstract davis–putnam–logemann–loveland procedure to dpll(T), *Journal of the ACM*, 53(6):937–977, 2006.
[14] G. Rock, K. Theis, P. Wischnewski, Variability Management, in: J. Stjepandić et al. (eds.): *Concurrent Engineering in the 21st Century: Foundations, Developments and Challenges*, Springer International Publishing Cham, 2015, pp. 491–520.