

Procedural model to ensure consistency and validity of complex, variant-oriented product portfolios

Felix Braun^{1,2}, Matthias Kreimeyer², Kristin Paetzold¹

¹*Institute for Technical Product Development, Universität der Bundeswehr München
Felix.braun@unibw.de, Kristin.paetzold@unibw.de*

²*MAN Truck & Bus AG, München
Felix.braun@man.eu, Matthias.kreimeyer@man.eu*

Abstract

Product portfolios with a large variance can be found – amongst others – in the commercial vehicle industry. The variance in the portfolio is characterized through various combinations of portfolio items, for example technical components or customer-oriented properties, documented in different product structures. To represent any constraints or prohibitions in the combination of portfolio items, e.g. technical constraints regarding the combinability of engines and gear boxes, sales and pricing constraints, weight or legal restrictions, large quantities of Boolean rules are used. These different sets of rules are defined manually by various employees and departments of an organisation.

Therefore, the key challenge is to ensure a consistent and valid portfolio definition throughout all product structures and across all divisions working on the portfolio and its rules. These types of complex, variant-oriented product portfolios consist of thousands of elements which can be combined in uncountable ways ($\sim 10^{300}$ combinations in this industrial example). So, automated algorithmic tools have been developed to help detecting inconsistencies.

This research is based on a literature study on the validity of data sets and empirical experiences from the industrial application supported by a case study with a global truck manufacturer.

As only isolated approaches to solve certain aspects of portfolio validation have been described yet, this research proposes a procedural model to help establish and maintain a valid product portfolio definition. The procedural model describes a set of validation tasks and related algorithmic validation services to automatically detect errors and inconsistencies. By executing this procedural model, it is ensured that the entire portfolio definition, including all rules across different structures, is valid and consistent. This approach has already been largely implemented with the project partner, a global truck manufacturer. In consequence, it has become an essential process step to discover any errors or inconsistencies and to build and maintain a valid and consistent product portfolio across the entire organisation.

Keywords: *Variant-oriented product portfolios, Boolean rules, Algorithmic validation tools, Portfolio validation*

1 Introduction

In large and complex product portfolios, different product structures are defined to fulfil the specific needs of related departments, e.g. for Sales, Engineering and Production. To represent the variance in the portfolio, i.e. the combinability of portfolio items, large quantities of Boolean rules are defined manually across an organisation. Therefore, the key challenge is to ensure a consistent and valid portfolio definition throughout all product structures across the entire organisation. This research is part of a company-wide project for a leading global truck manufacturer to implement a comprehensive, variant-oriented product portfolio within an entire Product Data Management (PDM) ecosystem. In this context, the project partner is facing the challenge of validating a complex product portfolio.

1.1 Key challenges to ensure a consistent and valid product portfolio definition

The commercial vehicle industry, which this research is based upon, is a sector characterized by a high technical variance yet small quantities at the same time (Kreimeyer, Förg, & Lienkamp, 2013). To handle modularity and its complexity accordingly, variant-oriented product portfolios are often in use in the industrial application. Figure 1 for example shows a small part of the product portfolio as used by the industrial partner of this research.

























Category	Engines	Cabins	Gear boxes	Frame rails	Rear crossbeams	Steered axles	Driven axles	Brakes
Variants								
								
								

Figure 1: Parts of a modular, variant-oriented product portfolio of a global truck manufacturer

The product portfolio is described from different perspectives, whereby most-often a Sales perspective (“what is sold to the customer”) and an Engineering perspective (“what is developed / built”) are implemented. This is due to both different structures and “languages”, as well as use cases for the portfolio. For example, in a sales / configuration perspective customers order an *automatic air-conditioning*. In an engineering view, the different components of the A/C-system are relevant, e.g. the *cooling compressor*. Within large product portfolios, not all portfolio items can be combined freely. This is due to constraints originating from various sources, e.g. buildability can be restricted from a technical perspective, sales or pricing restrictions can exist or load restrictions or legal constraints may apply. In Figure 1 for example, the largest engine in the top left-hand corner only fits into wide cabins because of geometrical constraints. Also, the larger engines require stronger gear boxes to handle the additional torque. These constraints are documented through sets of Boolean rules in the form of constraints and prohibitions:

$$\text{Engine: large} \rightarrow \neg \text{Cabin: small}$$

To account for all types of restrictions, different sets of rules are developed manually by a large number of employees sitting in different divisions of an organisation, especially within Sales, Engineering and Production.

The key challenge is now to ensure the consistency and validity of these different sets of rules established manually by various stakeholders across the entire product portfolio. Consequently, all configurations have to be buildable, rules must be error-free and the actual truck manufacturing has to be possible. As stated above, product portfolios can become very large with uncountable numbers of variants (around 10^{300} different variants with the project partner).

Therefore, automated validation tools for different validation aspects need to be developed and applied, as shown by Roth, Gehrlicher, & Lindemann (2015).

To ensure consistency and validity, existing validation tools have been adapted and new ones have been created together with the project partner. Based on these tools, a new procedural model has been developed and implemented in the project context to ensure a consistent and valid portfolio definition across all sets of rules, different product structures and for all the departments working with portfolio rules. In this paper, the procedural model is explained in detail and validated against the requirements based on the industrial application with the global truck manufacturer.

1.2 Structure of this research

This research is organized in four main sections. Firstly, the context for assessing validity of portfolio definitions is defined by focusing on general validation criteria of data sets, as well as existing approaches that only partially cover validation aspects of product portfolios. On this basis, the current gap in literature is shown and requirements for the solution are outlined. Secondly, the proposed procedural model is outlined in theory. Thirdly, the implementation of the model with the global truck manufacturer is presented. Hereby, the fulfilment of the requirements is reviewed and the validity of the approach is examined. Lastly, further research needs based on the experiences from the industrial application are pointed out.

2 State of the art

In the first part, this section gives an overview of existing definitions on the validity of data sets in general. In the second part, present approaches on how to handle inconsistencies in portfolio definitions as well as their shortcomings in the industrial application are outlined and discussed. In the last part, the requirements for the proposed solution are explained, based on the findings in literature and current state of research.

Before that, three terms used frequently in the context of this research need to be clarified. The inconsistency handling in these sets of rules is considered as a *validation*. This is in line with the IEEE Standards Coordinating Committee definition (1990) where *validation* is defined as “*the process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements.*” In our case, the specified requirements for the system, i.e. the portfolio definition, are to allow a consistent and error-free configuration as well as to ensure buildability – the validation is to check whether these requirements are met by the developed portfolio model. Second, all portfolio objects that are chosen by customers, e.g. within a sales configurator, are described as *customer-required differentiating properties* (or short: *customer-required properties*), as outlined by Eilmus & Krause (2012) and Eilmus, Ripperda, & Krause (2013). Opposite to that, the actual technical modules are considered as *components* and *component variants* as explained by Kreimeyer, Baumberger, Deubzer, & Ziethen (2016).

2.1 General requirements on the validation of data sets

To prove the consistency and validity of variant-oriented product portfolios, a set of certain requirements has to be met. In general, Wand & Wang (1996) outline four key dimensions of data quality issues that need to be addressed for ensuring a valid data set:

Completeness: all necessary values are included

Unambiguity: the data cannot be interpreted in more than one way

Meaningfulness: data interpretation in a meaningful way is possible

Correctness: data can be counted on to convey the right information

Another definition is found with Askham et al. (2013) who identified six different measures to define and evaluate data quality: *Completeness, Uniqueness, Timeliness, Validity, Accuracy, Consistency*. Other aspects mentioned include the flexibility of data, confidence in data or the value of data (Askham et al., 2013). These aspects will not be further considered when discussing validity in this research, as they rely on quality views that are not in focus here. As outlined by Wand & Wang (1996), the four data quality dimensions are on an abstract and generic level. Therefore, specific requirements for the validity of product portfolios need to be derived and will be explained in detail in section 2.3.

2.2 Existing approaches to inconsistency handling and validation of product portfolios

Only few authors have dealt with the validation of variant-oriented portfolio definitions. In the following section, a comprehensive overview on the existing work is given.

Walter, Felfernig, & Kuchlin (2017) describe three main requirements for the validation of a set of rules within automotive configurations. Their focus is on the unique representation of a configuration from a sales structure into the Bill-of-Materials (BoM):

Test on redundant parts: test that any part within the technical product structure has at least one use case in which it can be integrated into a vehicle configuration

Test on overlap error (double hit): it is checked that no ambiguities or double-hits are possible within the Bill-of-Materials

Test on incomplete position (no-hit): it is checked that any configuration on the properties level results in a representation within the Bill-of-Materials

Their approach regarding the derivation of a consistent Bill-of-Materials is promising, yet their focus is only on generating well-defined Bill-of-Materials, no other business perspectives and use cases, e.g. from a customer configuration perspective are considered.

For matrix-based product definitions, for instance a product representation in a properties-characteristics matrix, algorithms can be used for clustering and partitioning of relations between portfolio elements (Luft, Ewringmann, & Wartzack, 2014). On this basis, an analysis of portfolio elements is possible on a graphical level, but still an automated algorithmic detection of errors in the network has not been described yet. Therefore, this approach cannot serve complex portfolios with large numbers of variants. This also applies for the use of feature-oriented domain analysis methods, e.g. as found with Weilkiens, Lamm, Roth, & Walker (2015) or Pohjalainen (2008), as they also often rely on graphical representations which are too complex to handle for large portfolios.

Configit developed a software solution to test, refine and validate a set of portfolio rules (Cimdata, 2006). However, their focus is more directed towards a correct and valid set of configuration rules so that down-stream processes (e.g. sales configuration) are secured. Other business perspectives, e.g. the derivation of a consistent Bill-of-Materials as described with Walter et al. (2017) are not considered.

Not only Roth et al. (2015) but also Herfeld, Fürst, & Braun (2007) have identified the increasing need for tools and methods to manage risks like inconsistencies in complex portfolios, e.g. by considering circular paths. Nevertheless, they only outline the relevance of this topic without giving precise solutions for the industrial application.

All in all, relevant literature shows that isolated approaches to solve certain aspects of portfolio validation have been described and already existing software algorithms developed by information scientists can be used in an engineering context. Still, no comprehensive solution to ensure a consistent and valid product portfolio definition, especially for the manufacturers of complex, mechatronic products with many variants, has been found. Therefore, this research proposes a new procedural model to help establishing and maintaining a valid product portfolio definition in variant-oriented product structures.

2.3 Refined requirements for the solution

In chapter 2.1, general requirements for valid data sets have been examined. In particular, the definition of the four requirements *Completeness*, *Unambiguity*, *Meaningfulness* and *Correctness* (Wand & Wang, 1996) is applicable as it gives a brief, yet detailed classification of validation challenges. Therefore, based on these general data quality criteria, refined requirements for the specific use-case within truck development have been derived:

Requirement R.1 (Completeness): All customer-oriented properties of the sales product structure have to be selectable in at least one vehicle configuration

Requirement R.2 (Unambiguity): No Boolean rule must contradict any other rule

Requirement R.3 (Correctness): For every complete sales configuration, a complete and consistent Bill-of-Materials has to be derived

Requirement R.4 (Meaningfulness): All possible configurations have to be buildable in the technical product world

The adaption of Wand & Wang's definition leads to four refined requirements for the solution. After theoretically outlining the solution in section 3, the proposed procedural model will be validated based on the example from the industrial application in section 4.2

3 A procedural model to ensure consistency and validity

To respond to the challenges regarding validity and consistency, a new procedural model was developed as part of the project with the industrial partner. The procedural model is necessary to sort all validation use cases, to help not to forget any tasks and to correctly structure the validation process. In the following, the key elements of the procedural model are explained.

3.1 Levels and key elements within the proposed procedural model

For structuring the portfolio validation process within the procedural model, the St. Gallen Management Model (Rüegg-Stürm, 2005; Schwaninger, 2001) consisting of a normative level (aims / targets), a strategic level (approach / procedures) and an operative level (tools) has been chosen. It suits this application well as it provides a framework for structuring and executing large, complex tasks by breaking them down into smaller sub-problems. Therefore, this model is adapted for the use in terms of portfolio validation.

First, a normative level, which consists of validation tasks, is defined. As explained earlier, different requirements for the validity of portfolio rules are present and users are spread over the entire organisation. Therefore, the problem of *validation* needs to be broken down into smaller sub-problems that can be worked on by different user groups. For these sub-problems (e.g. "how to find contradicting rules"), separate *validation tasks* have been defined. Second, the strategic level provides *validation services*, which are computing algorithms tailored to solve the individual validation tasks. Finally, the operative level is formed by so-called *elementary services*, which are algorithmic basic functions recombined to form a special validation service. Figure 2 outlines the three levels and key elements of the model.

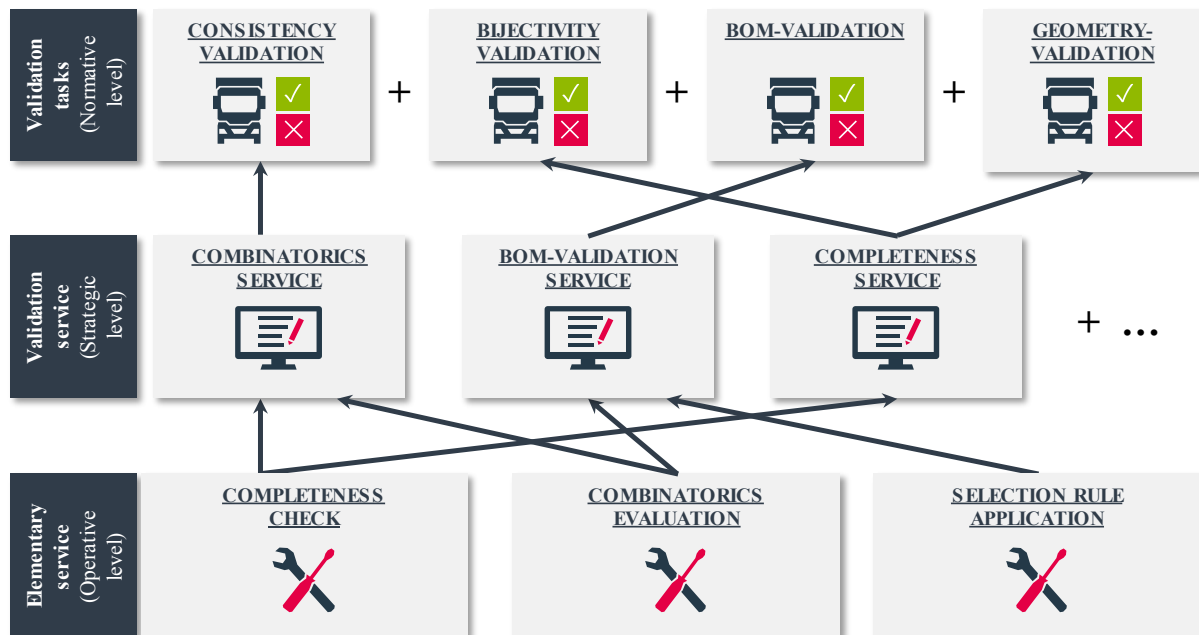


Figure 2: Levels and elements of the procedural model to ensure consistency and validation

The procedural model is now executed by performing two process steps: First, one validation task is performed repeatedly. With the help of validation services, inconsistencies are detected. Then the concerning rules are changed to eliminate the inconsistency. After that, the validation service is executed again to check that the inconsistencies have been eliminated. Second, this process of inconsistency detection, inconsistency fixing by altering rules and revalidation is gone through for every validation task in Figure 2. When all validation tasks have been performed and no inconsistencies can be found any more within any validation task, a consistent and valid portfolio definition has been achieved.

3.2 Elements of the procedural model on a detailed level

To cover all requirements concerning validity, the problem is broken-down into sub-problems. In the industrial application, four validation tasks have been identified in Table 1 in combination with a classification of rules in combinatorics rules and selection rules, as shown in Figure 4.

Table 1: Overview of validation tasks as part of the procedural model

Validation task	Description	Example
Consistency validation	This task proves that no constraints or prohibitions contradict each other. Also, non-selectable items shall be identified	$A \Rightarrow B$ $A \neg B$ Both rules are contradicting
Bijection validation	This task proves that bidirectional rules lead to the same result in every configuration	$A \Rightarrow (B \wedge C)$ $(B \wedge C) \Rightarrow A$ Both rules have to have the same effect
Bill-of-Materials validation	This task proves that for any possible (part) configuration, exactly one component variant is found within every compound.	$A \neg B$ $(A \wedge B) \Rightarrow \text{Component Z}$ Comp. Z is never chosen as the selection criterion $(A \wedge B)$ is forbidden
Geometry validation	This task proves that for any selected component variant, an exact position within the vehicle context is found.	$A \neg B$ $(A \wedge B) \Rightarrow \text{Position 12}$ Pos. 12 is never chosen as the selection criterion $(A \wedge B)$ is forbidden

To work on one of the validation tasks, a specific set of computational algorithms was developed and implemented as one of five specific validation services: the *combinatorics validation service* helps to identify inconsistencies in Boolean rules defined in the form of constraints and prohibitions. The *BoM-validation service* is targeted for the execution of component selection rules, i.e. to identify the right component variants for a given sales configuration. The *completeness service* is able to check whether in every category (properties category in the sales product structure, component in the engineering product structure) exactly one variant has been selected. The *bijection service* is used to analyse the reversibility of Boolean rules and the *error report service* lists all rules involved, when inconsistencies have been found. The following Figure 3 outlines which validation service is used to fulfil which validation task.

Validation task \ Validation service	Combinatorics validation service	BoM-validation service	Completeness service	Bijection service	Error report service
Consistency validation	x		x		x
Bijection validation				x	x
Bill-of-Materials validation		x	x		x
Geometry validation		x	x		x

Figure 3: Execution of validation tasks through a set of implemented validation services

Each specific validation service, which is the actual automated algorithm, is executed as a recombination of three different elementary services, which describe basic algorithmic tasks.

Completeness check: it is analysed whether always one unique variant is chosen within its category for any set of elements in the sales and engineering product structure

Combinatorics evaluation: the actual variance (number of combinations) that is allowed by the set of Boolean rules regulating the combinability within the sales product structure (constraints and prohibitions) is calculated

Selection rule application: the selection rules are confronted with a choice of customer-oriented properties to find out whether the selection condition is true or false

A more detailed example on the execution of the elementary services is found in section 4.1.

By executing this procedural model, it can be ensured that a consistent and valid product portfolio definition is found. This procedural model has been largely implemented and validated by the industrial partner of this research and will be shown in section 4.

4 An industrial example from a global truck manufacturer

In this section, the implementation of the proposed procedural model with the project partner, a global truck manufacturer, is explained in detail and a validation of the requirements from section 2.3 based on the practical application is described.

4.1 Implementation of the procedural model in the industrial application

When implementing the new variant-oriented product portfolio definition, two product structures were defined. The sales product structure containing the customer-oriented properties is used mainly for the sales configurator. The engineering product structure with components and component variants is used by the engineering and production division. These structures need to be always consistent to fulfil the defined requirements, e.g. regarding buildability or unambiguity. As the portfolio is changing constantly based on new projects and product variants, consistency and completeness need to be managed. Therefore, the procedural model including automated validation algorithms was developed, based on the portfolio model

described by Kreimeyer et al. (2016). Figure 4 below shows an adapted version with focus on the elements relevant to ensure consistency and validity.

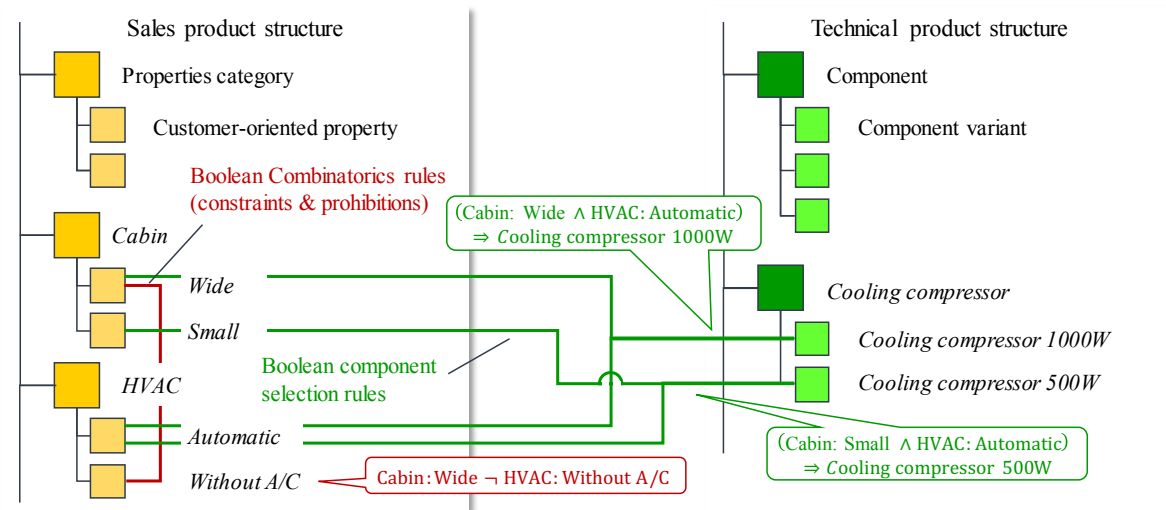


Figure 4: Product portfolio definition with the project partner, adapted from Kreimeyer et al. (2016)

The key challenge is to ensure consistency across both product structures as well as within each structure itself. One of the most relevant validation tasks is the BoM-validation to obtain an unambiguous and complete BoM for any configuration based on the sales product structure. The application of the procedural model as implemented with the project partner is shown in this example: First, one specific component within the BoM is selected by the user. Here, the component “Cooling compressor” with its variants “Cooling compressor 500W” and „Cooling compressor 1000W“, as shown in Figure 4, has been chosen. The two related properties categories are: *Cabin: Wide; Small* and *HVAC: Automatic; Without A/C*. Second, the BoM-validation service is applied for the component „Cooling compressor“. On this basis, the elementary services perform the actual validation activities in a specified order:

Selection rule application: The selection rules of the two chosen component variants are loaded and the included configuration objects (the customer-chosen properties) are extracted:

$(\text{Cabin: Wide} \wedge \text{HVAC: Automatic}) \Rightarrow \text{Cooling compressor 1000W}$

$(\text{Cabin: Small} \wedge \text{HVAC: Automatic}) \Rightarrow \text{Cooling compressor 500W}$

As seen above, the properties categories used within these rules are: *Cabin* and *HVAC*

Combinatorics evaluation: all allowed combinations of the relevant properties are formed along the set of constrains and prohibitions. For this case, one prohibition is present:

$\text{Cabin: Wide} \rightarrow \text{HVAC: Without A/C}$

Therefore, three possible use cases exist based on the two properties categories:

$(\text{Cabin: Wide} \wedge \text{HVAC: Automatic}),$ ~~$(\text{Cabin: Wide} \wedge \text{HVAC: Without A/C})$~~

$(\text{Cabin: Small} \wedge \text{HVAC: Automatic}),$ $(\text{Cabin: Small} \wedge \text{HVAC: Without A/C})$

Selection rule application: Then, the selection rules for all component variants are evaluated against all identified combinations of properties, as shown in Table 2 below:

Table 2: Selection rule application for the BoM-validation

Use case	$(\text{Cabin: Wide} \wedge \text{HVAC: Automatic})$	$(\text{Cabin: Small} \wedge \text{HVAC: Automatic})$	$(\text{Cabin: Small} \wedge \text{HVAC: Without A/C})$
Selected component variant	Cooling compressor 1000W	Cooling compressor 500 W	*

Completeness check: Finally, a completeness check is performed in three directions: a) if a component variant has been selected, is it exactly one? $\rightarrow \checkmark$ b) is there no component variant

left that has not been selected for any properties' combination? → ✓ c) is there no properties' combination left where no component variant has been found? → ✗

When these three checks are error-free for one component, the whole component is “*BoM-clean*”, which means that for any configuration a consistent and complete variant selection is possible for this component. In the example above, there is no variant found for the use case (Cabin: Small \wedge HVAC: Without A/C). Therefore, an error within the portfolio definition exists: either one component variant is missing (e.g. *Without cooling compressor*) or constraints / prohibitions to forbid this use case are missing. The right solution now has to be discussed with the relevant departments (here: Sales and Engineering) and the rules need to be changed. When performing this check for all components of the portfolio, the whole product portfolio can be validated to ensure a correct representation of the sales product structure into the engineering product structure.

4.2 Validation of the refined requirements

The procedural model has been largely implemented within the project partner. The defined requirements shown in section 2.3 can now be evaluated against the proposed procedural model, as outlined in Table 3 below.

Table 3: Validation of the refined requirements against the defined validation tasks

Validation task	Fulfilled requirement
Consistency validation	By showing that no constraints or prohibitions contradict each other and all variants are selectable, <i>R.1 (Completeness)</i> and <i>R.2 (Unambiguity)</i> can be fulfilled
Bijectivity validation	Through validating that all bidirectional rules lead to the same result in every configuration, <i>R.2 (Unambiguity)</i> can be fulfilled
Bill-of-Materials validation	By showing that for any possible (partly) configuration, exactly one component variant is found within every compound, <i>R.3 (Correctness)</i> is fulfilled
Geometry validation	Through validating that for any selected component variant, an exact position within the vehicle context is found, <i>R.4 (Meaningfulness)</i> is fulfilled

As the portfolio validation process with the project partner is still on-going, no final statistical proof can be given yet. Nevertheless, many validation errors and inconsistencies could be identified and fixed already with the help of this procedural model. Furthermore, large subparts of the portfolio already entirely fulfil the requirements concerning validity. Therefore, this procedural model delivers an essential contribution to building a stable and reliable product portfolio definition for the industrial partner.

5 Conclusion and next steps

In large and complex product portfolios, the key challenge is to build and maintain a valid and consistent product portfolio definition across different product structures and thousands of Boolean rules. Yet, only isolated approaches of how to validate entire portfolio definitions have been found. The new procedural model proposed here helps to build and maintain a valid set of portfolio rules across an entire organisation by structuring and outlining all necessary validation tasks. The implementation of the model in the industrial application has already shown the usefulness of this approach. Therefore, the procedural model delivers a valuable approach to ensuring consistency and validity for complex product portfolios.

Still, different levers to further improve and refine this model based on the implementation with the global truck manufacturer were identified. In particular, the optimal sequence of going through the different validation tasks, e.g. to only start every BoM-validation after performing a consistency validation has to be further evaluated. Also, future changes in the portfolio due to new products and lifecycle / change management aspects have not been examined in detail yet. Finally, the best process of how to handle and fix identified inconsistencies needs to be further evaluated. To refine the procedural model, work on these aspects will continue and further research will be carried out in the future.

Citations and References

- Askham, N., Cook, D., Doyle, M., Fereday, H., Gibson, M., Landbeck, U., Schwarzenbach, J. (2013). *The six primary dimensions for data quality assessment*. DAMA UK Working Group.
- Cimdata. (2006). *Configit Software "A Powerful Approach to Managing Complex Configurations."* Ann Arbor.
- Eilmus, S., & Krause, D. (2012). An Approach for reducing Variety across Product Families. In *DS 71: Proceedings of NordDesign 2012, the 9th NordDesign conference, Aalborg University, Denmark. 22-24.08. 2012*.
- Eilmus, S., Ripperda, S., & Krause, D. (2013). Towards the development of commonal product programs. In *Proceedings of iced13. 19th International Conference on Engineering Design (ICED13), Design for Harmonies (Vol. 4, pp. 209–218)*.
- Herfeld, U., Fürst, F., & Braun, T. (2007). Managing complexity in automotive safety development. In *DSM 2007: Proceedings of the 9th International DSM Conference, Munich, Germany, 16.-18.10. 2007 (pp. 271–286)*.
- IEEE Standards Coordinating Committee. (1990). *IEEE Standard Glossary of Software Engineering Terminology (IEEE Std 610.12-1990)*. Los Alamitos, CA: IEEE Computer Society (Vol. 169). Los Alamitos.
- Kreimeyer, M., Baumberger, C., Deubzer, F., & Ziethen, D. (2016). An Integrated Product Information Model for Variant Design in Commercial Vehicle Development. In *DS 84: Proc. of DESIGN 2016, the 14th Int. Design Conf., Dubrovnik, Croatia (pp. 707–716)*.
- Kreimeyer, M., Förg, A., & Lienkamp, M. (2013). Multi-level modular kit development for commercial vehicles. In *VDI Fachtagung NFZ 2013 (pp. 99–112)*. Celle: VDI-Verlag.
- Luft, T., Ewringmann, N., & Wartzack, S. (2014). Application and Validation of the Matrix-based Product Description in a Case Study by Using the Software Loomeo. *Procedia CIRP, 21*, 479–484.
- Pohjalainen, P. (2008). Feature oriented domain analysis expressions. In *InNordic Workshop on Model Driven Software Engineering (NW-MoDE'08), Reykjavik, Iceland*.
- Roth, M., Gehrlacher, S., & Lindemann, U. (2015). Safety of Individual Products-Perspectives in the Context of Current Practices and Challenges. In *DS 80-3 Proc. of the 20th Int. Conf. on Eng. Design (ICED 15) Vol 3: Org. and Mgt., Milan, Italy, 27-30.07. 15*.
- Rüegg-Stürm, J. (2005). *The New St. Gallen Mgt. Model*. London: Palgrave Macmillan UK.
- Schwaninger, M. (2001). Intelligent organizations: An integrative framework. *Systems Research and Behavioral Science, 18(2)*, 137–158. <https://doi.org/10.1002/sres.408>
- Walter, R., Felfernig, A., & Küchlin, W. (2017). Constraint-based and SAT-based diagnosis of automotive configuration problems. *Journal of Intelligent Inf. Sys., 49(1)*, 87–118.
- Wand, Y., & Wang, R. Y. (1996). Anchoring data quality dimensions in ontological foundations. *Communications of the ACM, 39(11)*, 86–95.
- Weilkiens, T., Lamm, J. G., Roth, S., & Walker, M. (2015). *Model-based system architecture*. John Wiley & Sons.