INTERNATIONAL DESIGN CONFERENCE – DESIGN 2020
https://doi.org/10.1017/dsd.2020.90

USING A CROSS-DOMAIN PRODUCT MODEL TO SUPPORT ENGINEERING CHANGE MANAGEMENT

R. Wilms [1,2,✉], P. Kronsbein [1], D. Inkermann [3], T. Huth [1], M. Reik [2] and T. Vietor [1]

[1] Technische Universität Braunschweig, Germany, [2] Volkswagen AG, Germany, [3] Technische Universität Clausthal, Germany

✉ r.wilms@tu-braunschweig.de

Abstract

Engineering changes (ECs) and engineering change management (ECM) are crucial for successful product design processes (PDP). Due to the increasing complexity of today's products (like vehicles) and the interaction of different engineering domains (mechanics, electric/electronics, software) involved in the PDP, cross-domain EC impact assessments as well as processes are required. To better support engineers in assessing change propagation across domains and products, existing approaches for ECM product models are analyzed in this paper and an enhanced product model is derived using MBSE.

Keywords: engineering change, change prediction, complex systems, systems engineering (SE), model-based engineering

1. Introduction

Engineering change management (ECM) is crucial in product design, since engineering changes (ECs) are for instance required to react to changing markets or legal conditions as well as quality issues in the design process. Driven by the complexity of today's products (such as vehicles) involving numerous interlinked components, the impact assessment of ECs is a challenging task. Taking automotive engineering as example, insufficient ECM can even result in significant financial risks. On the one hand, this is due to high numbers of required ECs (see e.g. Yildirim and Campean, 2013) in the design process. On the other hand, identical components are used in different vehicles, so that change propagation can even occur across the product portfolio. To face these risks, engineers have to be supported in assessing potential impacts of ECs they plan to implement. As Jarratt et al. (2004a) state, "[…] the key issue in any proposed method or tool to support engineering change management is the development of an accurate and appropriate product model." The ECM approach for automotive engineering presented in this paper aims at providing a functional, structural and change analysis view for EC impact assessments of complex products. Therefore, it integrates (1) product functions (as customer perspective on the product), (2) components designed in different engineering domains realizing complex functions, (3) different component linkage types indicating required domain-specific or cross-domain interaction in case of an EC, (4) change propagation data from previous projects, (5) different products to assess potential change propagation across the product portfolio as well as (6) additional information for the EC process in one consistent model. In section 2 of this paper, the applicability of established product models in ECM with reference to these aspects and cross-domain

ECM of complex products is discussed. Then, section 3 describes how Model-based Systems Engineering (MBSE) can enable adequate product models to support ECM. Objective of this paper is to present (see section 4) and apply (see section 5) the product model for cross-domain ECM of complex products outlined before. Finally, section 6 provides a conclusion of this paper and outlines future work.

2. Product models as a key element for ECM

In order to assess and predict potential impacts of ECs, engineers need a detailed understanding of the product they are designing. This understanding often can be built up and expanded over several product generations, since many products are not designed from scratch, but based on predecessor or also competitor's products (Albers et al., 2015). However, transferring and providing knowledge in product representations offers opportunities for several reasons. For instance, technical discussions can be facilitated, an overview on entire systems or products can be provided and the loss of knowledge can be limited or avoided in case someone leaves the department or even company. For (cross-domain) ECM in particular providing an overview on the entire product is important. In this context Jarratt et al. (2004a) state that "[…] engineers and designers found it hard to appreciate fully the complexity of linkages between parts that could cause changes to propagate." and "We miss the other things that the components are doing because most components are doing several jobs." Considering that for instance today's vehicles provide more and more complex functions (e.g. parking assistants, adaptive cruise control) based on interacting components designed in different engineering domains (mechanics, electric/electronics, software), to understand and overview this "complexity of linkages" and all components' "jobs" becomes even more challenging. Or as Clarkson et al. (2005) stated: "[…] current change management depends heavily on individual designers' typically limited product overview. For complex products, this approach is error-prone [...]."As indicated adequate product or system models are extremely relevant to be able to predict potential impacts of ECs in complex products. To what extent established models in ECM are suitable in this context will be discussed in section 2.1 and 2.2.

2.1. DSMs, DMMs and MDMs in ECM

Design Structure Matrices (DSMs) - also called Dependency Structure Matrices (Lindemann et al., 2009) - are often used in ECM to illustrate element dependencies or interactions within a specific system. DSMs are square matrices showing elements of the respective system along the rows and columns of the matrix. If for instance a change can propagate from component 1 to component 2 and 3, this is represented by a cross in cell C12 and C13 in the component DSM (see Figure 1). Besides crosses in the cells of the matrix, numerical values can be used to illustrate the likelihood and potential impact of change propagation between components (Clarkson et al., 2001). By multiplying likelihood and impact values the overall risk of change propagation can also be calculated and presented in a DSM (see Figure 1 for the different types of DSMs).

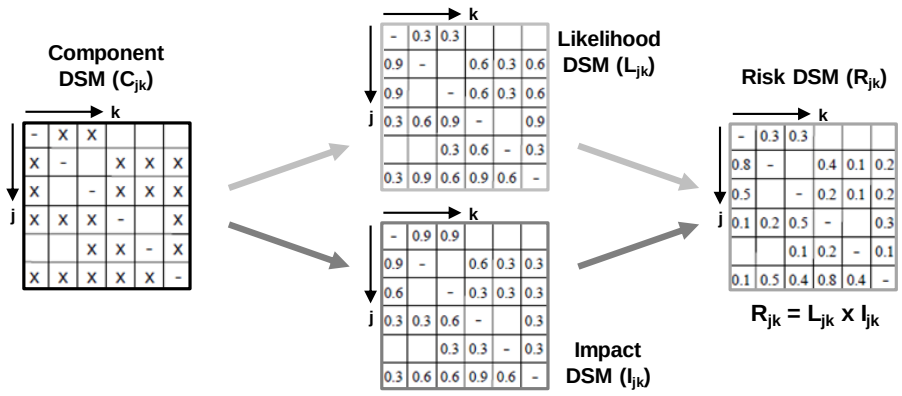

Figure 1. Component, likelihood, impact and risk DSMs (based on Clarkson et al., 2001)

Although these kind of DSMs easily illustrate, where and with which likelihood, impact and risk change propagation can occur, they do not show the interface or linkage causing changes to propagate. This missing information is for example provided by Yildirim and Campean (2013) using capital letters - M for material,

https://doi.org/10.1017/dsd.2020.90 Published online by Cambridge University Press

E for energy, I for information and P for physical relation - in the cells of the matrix. Besides these types of component DSMs, different applications and purposes of square matrices exist, so that DSMs are not only used to represent component interaction. For instance Flanagan et al. (2003) create function-function matrices to illustrate, whether changing one function demands a change to another function. Additionally, component-function matrices are derived by Flanagan et al. (2003) to represent, if components are linked due to a specific function. Koh et al. (2012) use matrices to illustrate to what extent change options have an influence on product requirements and components. To represent dependencies between different kinds of elements (e.g. components and functions) or different domains (e.g. product, process and organization), DSMs cannot be used, because different types and amounts of elements in the rows and columns of the matrix have to be represented. This led to non-square matrices, which are also called Domain Mapping Matrices (DMM) in literature (Danilovic and Browning, 2007) due to different domains that can be mapped in one matrix. Another type of matrix used in ECM is the Multidomain Matrix (MDM). MDMs aggregate or combine DSMs and DMMs, so that intra- (DSM) and inter-domain (DMM) relationships can be modelled together (Browning, 2016). Further explanations with reference to and applications of DSMs, DMMs and MDMs are for instance provided by Browning (2016) and Lindemann et al. (2009).

Although matrix representations are widely used in product design and also ECM, they have significant limitations with reference to large and complex products (Keller et al., 2005; Koh et al., 2012). These products consist of many elements interacting via different linkage types or interfaces. On the one hand, this diversity of linkages is difficult to show in one matrix. On the other hand, matrices representing many elements (and their diverse linkages) are difficult to read because of their size (Jarratt et al., 2004a). Additionally, matrices are suitable to show direct links (A to B) between elements, but indirect links (A to C to B) are hard to represent (Keller et al., 2005). Apart from that, additional information required for ECM with reference to product elements (e.g. name of the responsible engineer) or element linkages can hardly be integrated in one manageable matrix and keeping matrices updated is challenging.

2.2. Graph representations in ECM

Regarding the representation of indirect links and potential paths for change propagation, in particular graph (or network) representations are more suitable than matrices (Keller et al., 2005). In a graph nodes represent product elements, whereas dependencies are depicted by edges between these nodes. Following all possible edges in a graph from node A to B, direct and indirect links can be (re)traced, so that extensive change propagation analyses can be conducted. In general, element dependencies and propagation paths are easier to visualize in graphs, since these representations provide more degrees of freedom than matrix representations (Keller et al., 2005). Besides large graphs representing all product elements and their (direct as well as indirect) linkages, these additional degrees of freedom can for instance be used to focus a specific element and therefore place it's node in the centre of the graph, while all connected nodes are arranged around it (Jarratt et al., 2004b). Additionally, the risk of change propagation between the focused node and other nodes can be represented by their distance to the centre or different propagation paths from the centre to specific nodes can be highlighted using graph representations (Keller et al., 2005).

As indicated, graph representations offer advantages in the context of ECM compared to matrix representations in particular with reference to the visualization of dependencies. How advantages of graph and matrix representations can be exploited and for instance additional information required for ECM integrated in a product model using MBSE approaches, is discussed in the following sections.

3. Potential of system models in MBSE to support ECM

Systems Engineering (SE) is an interdisciplinary approach providing methods and processes to design and realize complex technical systems across (engineering) domains (Walden et al., 2017). While SE is mainly document-based (i.e. text documents, calculation sheets, presentation slides) leading to challenges for instance with reference to consistency and up-to-date information in and across documents, Model-based Systems Engineering (MBSE) uses system models as central repository within the design process. Following Delligatti (2013) to create system models three so-called MBSE pillars are needed: a modeling language, a modeling tool and a modeling method (see Figure 2).

https://doi.org/10.1017/dsd.2020.90 Published online by Cambridge University Press

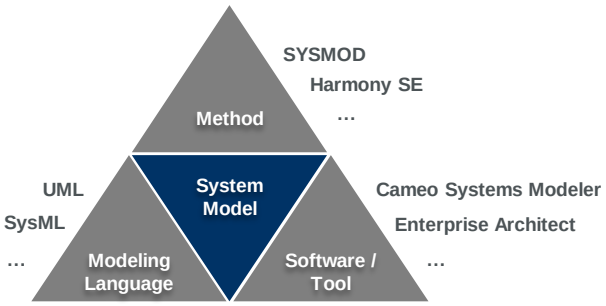

Figure 2. MBSE pillars (following Delligatti, 2013)

As **modeling language** the Systems Modeling Language (SysML) - a derivative of UML - is widely used in MBSE. SysML is a graphical modeling language providing and defining in particular the elements, element relationships and diagrams that can be used to create a model. To build the model, a software or **tool** is needed (e.g. Cameo Systems Modeler, Enterprise Architect). In contrast to diagramming tools (e.g. Visio), modeling tools provide consistency of all diagrams, since all diagrams are updated automatically, if one diagram is modified. How to create the model and it's purpose has to be defined by the modeling method. The **modeling method** therefore in particular defines the content and scope of the model (e.g. product requirements, behavior, structure), how the model is supposed to support (specific tasks in) the design process and how it can or has to be used. (Delligatti, 2013)
Since (MB)SE is an interdisciplinary approach to design complex systems, it also provides advantages for ECM of complex products. On the one hand, complex systems consisting of many elements with diverse linkages can be represented and stored in one model without graphical limitations (see section 2.1). On the other hand, differing views on the model can be used to provide and highlight specific (ECM) aspects. As a consequence, view (1) might show component linkages and therefore potential paths for change propagation, whereas view (2) represents within which products of the product portfolio the components are used. Additionally, the views of the system model are kept updated and consistent automatically (based on the functionalities of the modeling tools described before) and additional data required for ECM (e.g. responsible engineers, suppliers) can be integrated and shown in the model. Although all information are kept in the model and for instance presented in different views, also matrix representations (see section 2.1) can be generated from the model, so that the interaction with people not being used to these kind of models can be facilitated.
Since system models can be derived for diverse applications, a modeling language as well as tool has to be chosen and an appropriate method has to be defined to derive a model adequately supporting ECM. From the author's point of view, in this context the modeling method is the key pillar, because it defines the purpose and scope of the model and thereby influences its complexity, applicability and benefit for the user. Considering this, an approach to support ECM of complex technical products is presented in section 4. The approach is then applied using SysML and Enterprise Architect in section 5.

4. Concept of a cross-domain product model for ECM

The concept presented in this section is supposed to support engineers to better analyze and assess potential (cross-domain) impacts of ECs they plan to implement. In consequence, the general purpose of the product or system model is defined, while the exact scope of the model has to be clarified. On the one hand, product components have to be represented in the model, since these components can be the starting point of or affected by an EC. By representing components (and technical systems they belong to) in the product model, the product structure becomes part of the model. On the other hand, engineers working with the model should be able to analyze, which (customer) functions might be affected by a component they plan to modify, since these functions are directly perceived by customers. In consequence, (high-level) functions (e.g. automatic windshield cleaning in wet conditions) and function clusters (e.g. comfort and safety) these functions belong to have to be represented in the product model. Besides this (high-level) functional structure and the product structure, the product portfolio structure has to be represented in the model to be able to indicate, in which products (i.e. vehicles) modified components or functions might cause problems. Based on the functional, product and

https://doi.org/10.1017/dsd.2020.90 Published online by Cambridge University Press

product portfolio structure that have to be represented, the general scope of the product model is defined. How these structures, their elements and different element linkages are implemented and used in the model to provide it's defined purpose is explained in the following sections.

4.1. Views of the ECM product model

Views provided within a product model represent the product based on specific interests. In product design, in particular a functional and physical description of the product, together leading to the product's architecture, are relevant (Feldhusen and Grote, 2013). Here, functions are seen as an external (customer) perspective in particular determining the product's purpose and value (Haberfellner et al., 2019). The **functional view** provided in the model (see Figure 3) contains a function derivation view (a), a functional structure view (b), a component function allocation view (e) and a function technical linkage view (h). While the function derivation view and the functional structure view just represent, how (customer) functions within the product model were derived (a) and to which function clusters they belong to (b), the component function allocation view represents all components realizing one specific (customer) function. How these components interact to provide this function and therefore possible paths for change propagation are represented in the function technical linkage view (h). Besides the functional view, the **structural view** - representing in particular the product's physical composition (Feldhusen and Grote, 2013) - is the second main view within the product model (see Figure 3). This view contains the product structure view (c), product portfolio structure view (d), component product allocation view (f), function product allocation view (g), product technical linkage view (i) and the product historical linkage view (j). While the product structure view shows components, the technical systems they belong to and therefore the product hierarchy, the product portfolio structure view represents different products within the product portfolio. Which components or functions are used in which products of the portfolio is represented in the component (f) and function (g) product allocation view. The product technical linkage view (i) represents component interaction (based on various technical linkage types) for a specific product. In contrast to that, the product historical linkage view (j) represents component dependencies indicated by historical data from predecessor projects. To provide detailed EC analyses, the **change analysis view** (see Figure 3) is the third main view of the product model. This view consists of the change prediction view (k), the change path view (l) and different layer views (m). Based on the change prediction view a specific component and it's dependencies to other components indicated by historical data from predecessor projects can be analyzed. On the contrary, using the change path view, this component can be analyzed with reference to it's technical linkages to other components defined in the current project. Additionally, analyses on different abstraction levels of the function and product structure can be conducted based on different layer views (m). How the different views are created and used for ECM is presented in section 5. In section 4.2 important elements and linkages within the model (and it's views) are described. Please note, that change analyses with reference to (the alignment of) requirements are not discussed within this paper. Nevertheless, requirements could be integrated in the product model as additional view, so that impact analyses with reference to requirements can be enabled based on the product model defined in this paper.

4.2. Elements and linkages within the ECM product model

To represent the functional structure of the product (illustrated in the function structure view), in particular **functions** (e.g. automatic windshield cleaning in wet conditions) need to be represented as elements of the ECM model. Functions can be categorized and are therefore linked to function clusters (e.g. comfort and safety), so that a hierarchy within the function structure can be represented. In order to represent, how functions are realized from a technical point of view, elements illustrating **product components** are part of the ECM model. These elements contain additional information using element attributes, so that for example the component supplier can be shown in the model (see Figures 4 and 5). Since components realize functions, functions are allocated to components. Additionally, the component interaction to provide a specific function is documented in the model using so-called technical linkage types. These **technical linkage types** represent different types of component interaction (e.g. material transfer, information or control signals), that are at the same time potential paths for domain-specific and

https://doi.org/10.1017/dsd.2020.90 Published online by Cambridge University Press

cross-domain change propagation (see Wilms et al., 2019). Besides technical linkages also so-called **historical linkages** can be implemented between components. These unidirectional linkages (see Figure 4) show change propagation information derived from predecessor projects. As a consequence, historical data e.g. regarding the likelihood of change propagation from component A to B can be integrated and used in the ECM model. Components are not only linked to functions and other components, they are also part of the product structure (view) and therefore belong to **technical systems** also represented in the model. Technical systems can be integrated in the model on different levels of abstraction (e.g. steering system, chassis, vehicle), so that the hierarchy within the product structure can be represented. To represent the product portfolio and therefore products using the defined functions and components, also elements representing **products** (e.g. VW Tiguan) or even **product segments** (e.g. SUV-class) and **brands** (e.g. VW) are part of and linked in the ECM model. How the ECM model and it's different views (functional, structural and change analysis view) are created using the defined elements and linkages and how EC analyses can be supported with the model is presented in section 5.

5. Realization and application of the ECM product model

For the efficient and purposeful realization and application of a product model, a methodical procedure must be defined as part of the modeling method. In the present case, the methodological procedure is documented in a so-called ECM process. This process consists of different main phases and subphases. Tasks within these phases determine which aspects (elements and element linkages) should be modeled to create a particular view (see section 4.1) of the product model. Figure 3 shows the connection between the ECM process and the ECM product model. In the main phases (e.g. (1) Derive) and subphases (e.g. (1.1) Function Derivation) of the ECM process, model views (e.g. (a) Function Derivation View) are created that belong to the main views (functional, structural and change analysis view) of the product model. All views are represented and modeled using SysML diagrams (one diagram represents one view). Besides SysML also the Object-Process Methodology (OPM) can be seen as leading standard or framework in MBSE (Dori, 2016). However, since (nowadays) SysML is more widespread in particular in (the automotive) industry, SysML is used for the approach presented in this paper.

The main phases (1) Derive, (2) Allocate and (3) Link are used to create (i.e. realize) the product model, while the main phase (4) Analyze describes the model's application and therefore the use of information contained in the model to evaluate changes. To better understand the realization and application of the product model, an exemplary (customer) function is modeled in this section following the ECM process (see Figure 3). Here, the focus is on the basic tasks and contents in order to show "what" has to be done to build up the ECM product model. At the end of this section, possible analyses for ECM are presented, that can be carried out using the product model. A detailed description of the model's implementation (with SysML) in the modeling tool is omitted in this paper for reasons of space.

The exemplary (customer) function focused in this section is a function used in automotive engineering to detect precipitation on the windshield in order to adapt the frequency of the windshield wipers. In simplified terms, the function is realized by the following components: rain sensor, rain sensor software, electrical control unit (ECU), power supply, wiper motor, wiper, windshield and the vehicle body. In the product model, the software code is treated and implemented as component, since changes in the software can also lead to changes in hardware or other software components.

The **first main phase** of the ECM process is phase (1) Derive, which is divided into three subphases: (1.1) Function Derivation, (1.2) Component Derivation and (1.3) Product Derivation. In subphase (1.1) the aspects of the (a) Function Derivation View are presented on a Block Definition Diagram (BDD). The exemplary function "automatic windshield cleaning in wet conditions" focused in this section, is derived from the vehicle configurator and therefore linked to the respective configurator item. Since functions might e.g. also be derived from requirements, view (a) is used to show how a specific function was derived and became part of the product model. After creating view (a), the aspects of the (b) Function Structure View are modeled on a BDD. The aim of this view is to represent the function structure, i.e. the decomposition of all functions of the product. The considered function "automatic windshield cleaning in wet conditions" belongs to the function cluster "comfort and safety". On the BDD, this hierarchical structure of functions can be represented by means of a composite association between the functions. In the (1.2) Component Derivation subphase the BDD of

https://doi.org/10.1017/dsd.2020.90 Published online by Cambridge University Press

the (c) Product Structure View is created. The aim of this view is a logical decomposition of the product. Therefore, the product is subdivided into systems and subsystems until a decomposition level is reached, so that components that are used to realize functions can be defined and assigned to (sub)systems. Using the rain sensor system as an example, components on this level are the rain sensor, ECU, wiper etc.. On the BDD, the hierarchical structure of systems, subsystems and components is modeled by means of composite association between blocks. In consequence, view (c) can be used to analyze (sub)systems being affected by component ECs. The (1.3) Product Derivation subphase is used to derive and show products from the company's product portfolio. In the product portfolio structure, products are grouped according to their characteristics. Using the example of a car company, vehicles can be assigned to different vehicle segments (e.g. SUV-class). These vehicle segments might be assigned to a specific brand (e.g. VW), whereas all brands are assigned to the car company. These dependencies and assignments are represented on a BDD of the (d) Product Portfolio Structure View by means of a composite association between the blocks of products, segments and brands.

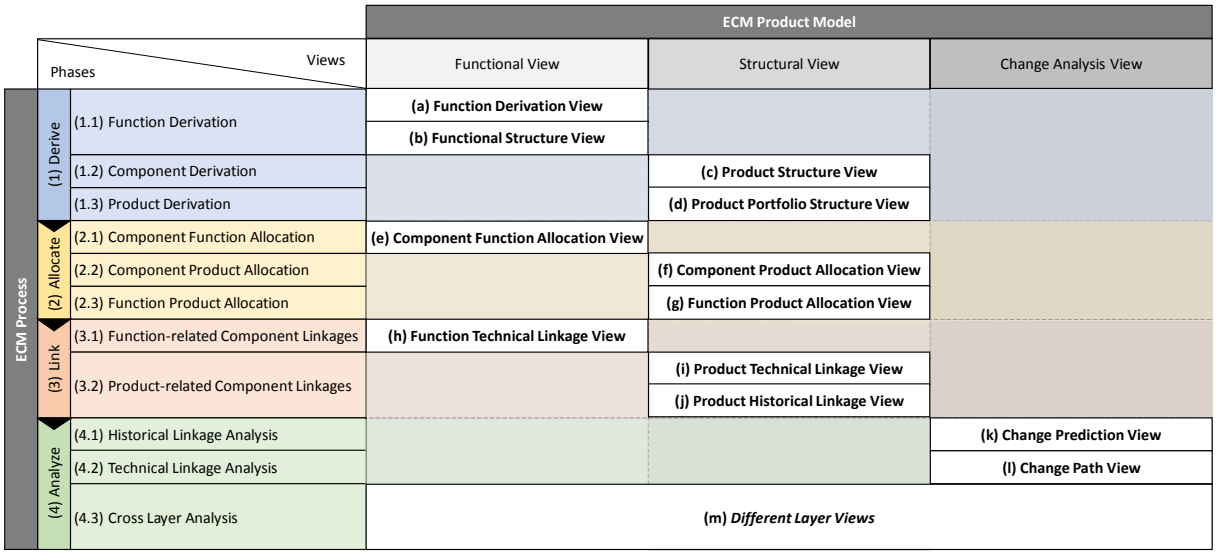

Figure 3. ECM product model and process

The **main phase (2)** Allocate consists of the subphases (2.1) Component Function Allocation, (2.2) Component Product Allocation and (2.3) Function Product Allocation. The subphase (2.1) Component Function Allocation - leading to the (e) Component Function Allocation View - has the aim to represent components realizing one specific function on an Internal Block Diagram (IBD). The components and functions on the IBD are instances (i.e. properties in SysML) of the function and component blocks on the BDD of view (b) and (c). The assignment of components to a function is modeled on the IBD using an allocation linkage. Using property values, additional information (e.g. responsible engineer or component supplier) can be added to the component instances. This facilitates the identification and integration of relevant stakeholders in the EC process in case of a change request. A similar procedure is used in the (2.2) Component Product Allocation subphase. All components - as instances of the component blocks - are assigned to products - as instances of product blocks from view (d) - they are used in. This relation is modeled using an allocation linkage between the instances of the components and products on the IBD of the (f) Component Product Allocation View. In the following subphase (2.3) Function Product Allocation, an IBD of the (g) Function Product Allocation View is created showing which functions are used in which product. Again, allocation linkages between the elements (i.e. instances of functions and products) are used on the IBD.

The **main phase (3)** Link is divided into two subphases: (3.1) Function-related Component Linkages and (3.2) Product-related Component Linkages. In phase (3.1), technical linkages between components that are required to realize one specific function are set. The IBD of the (h) Function Technical Linkage View shows how the components interact in order to realize the function. For this purpose, different technical linkage types (see section 4.2) are set between the instances of the component blocks. These technical linkages also represent potential paths for change propagation and

https://doi.org/10.1017/dsd.2020.90 Published online by Cambridge University Press

therefore enable detailed change propagation analyses (in particular regarding the focused function). The (i) Product Technical Linkage View derived in subphase (3.2) displays all technical linkages, that exist between components used in the same product. The IBD of this view can be generated almost automatically, since the product model knows based on previous subphases, which components realize specific functions (view (e)), which functions are used in which products (view (g)), and how the components are technically linked to realize the associated function (view (h)). In consequence, to create view (i) only all components realizing the functions of the product and their technical linkages have to be displayed on the IBD. Besides technical linkages between components of the product, historical linkages (see section 4.2) can be set between these components, which leads to the (j) Product Historical Linkage View. Due to the unidirectional historical linkages (see e.g. Figure 4), change predictions based on historical values for change propagation can be made. These values derived from analyses of change executions in previous projects can be set as tagged values (included in the model for change likelihood, impact and risk) on the historical linkages in the IBD. How to derive adequate likelihood and impact values (change propagation risk = likelihood x impact) is a company specific issue, which is not part of this paper. By completing view (j), the realization of the product model for the selected function is completed and the application of the model can be started in main phase (4) Analyze.

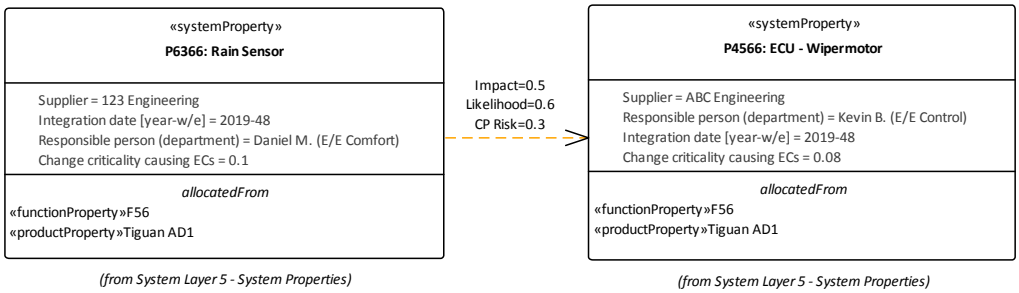

Figure 4. IBD of the change prediction view (excerpt, other components not illustrated)

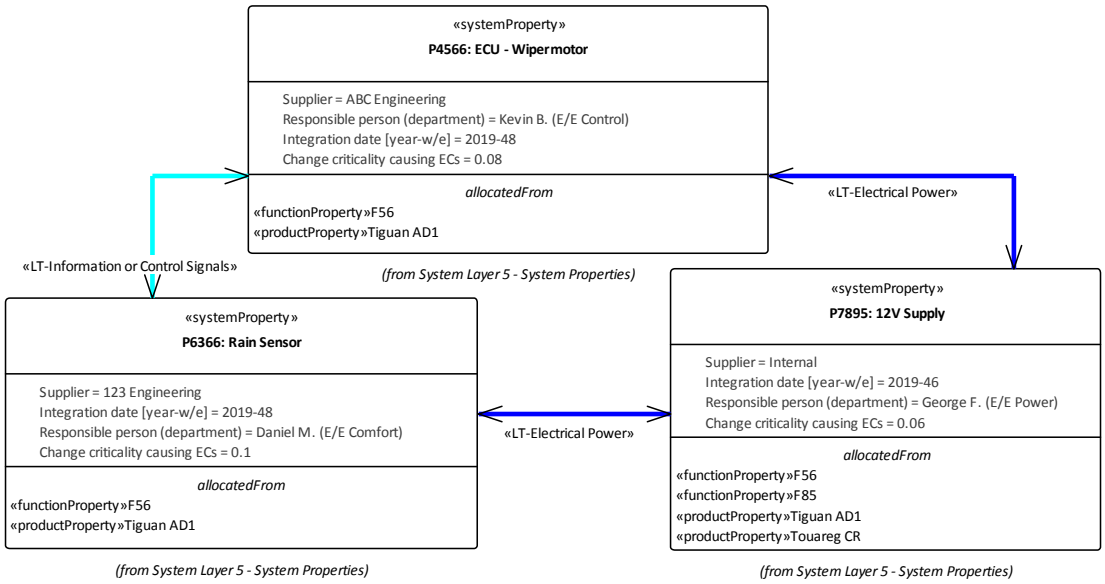

Figure 5. IBD of the change path view (excerpt, other components not illustrated)

In addition to the function "automatic windshield cleaning in wet conditions", another function from automotive engineering called "Adaptive Cruise Control (ACC)" was integrated in the product model in order to demonstrate the cross-functional and cross-product change propagation analyses in main phase (4) Analyze. For this purpose, the function "automatic windshield cleaning in wet conditions" is randomly named "F56", the "ACC"-function "F85" in the model. Additionally, both functions are used in different products, i.e. in subphase (2.3) Function Product Allocation the "F56"-function was assigned to the product "Tiguan AD1", whereas "F85" now is assigned to the product "Touareg CR".

https://doi.org/10.1017/dsd.2020.90 Published online by Cambridge University Press

The **main phase (4)** Analyze consists of three subphases: (4.1) Historical Linkage Analysis, (4.2) Technical Linkage Analysis and (4.3) Cross Layer Analysis. In case of an EC for one specific component, the component can be analyzed in subphase (4.1) on an IBD of the (k) Change Prediction View showing historical linkages with likelihood, impact and risk values for change propagation from this component (e.g. the rain sensor) to other components (e.g. the ECU) (see Figure 4). Using the historical linkages set on the IBD of view (j), the diagram of view (k) is easily created in the modeling tool. For the engineer of the rain sensor, view (k) on the one hand provides the ability to predict probable (i.e. historical linkages with high likelihoods) ECs of affected components and take countermeasures. On the other hand, the element attributes of these potentially affected components offer additional information, so that for instance responsible engineers or suppliers can be contacted to discuss the planned EC. In addition to view (k), the (l) Change Path View (see Figure 5) derived in subphase (4.2) displays all components on an IBD, that are in the current design project technically linked to a focused component (e.g. the rain sensor). ECs can propagate via technical linkages, so that this view enables the engineer to analyze different interfaces and therefore potential paths for change propagation from the selected component into the overall system. The "allocatedFrom" section of the component instance automatically displays all functions and products the component is used in (see Figure 5), so that EC impacts can be evaluated across functions and products. As illustrated in Figure 5, an EC of the rain sensor can lead to an EC of the component "12V Supply" via the technical linkage <<LT-Electrical Power>>. In this case, the functions "F56" and "F85" are affected. Additionally, the EC can propagate across products (from "Tiguan AD1" to "Touareg CR"). On the one hand, components, which contribute to many functions and are used in many products, lead to economies of scale. On the other hand, they can become a significant risk, because undetected change propagation might lead to EC efforts and costs in various products. Therefore, view (l) provides - especially in case of high complexity - a useful overview of potentially affected components, functions and products, so that EC analyses can be enhanced and EC implementation planned more carefully to reduce follow-up costs of ECs.
Looking at the IBDs of view (k) and (l) together (see Figure 4 and Figure 5), analyses can be conducted to determine, which technical linkages may have led to (the data for) historical linkages. With reference to the components "Rain Sensor" and "ECU", the historical linkage could have arisen due to the direct technical linkage <<LT-Information or Control Signals>> or/and due to an indirect linkage via the component "12V Supply" based on the technical linkages <<LT-Electrical Power>>. In any case, historical linkages with high likelihood values should lead to detailed analyses of the technical linkages between components. Additionally, a historical linkage between components not having technical linkages, might indicate, that a technical linkage of the current project was overlooked while creating the product model. The last subphase of main phase (4) is the (4.3) Cross Layer Analysis. Using (m) Different Layer Views in this phase, function- and product-specific analyses as well as cross-functional and cross-product analyses can be conducted on different hierarchical levels of the functional and product portfolio structure. For a car manufacturer, for instance, all components and their technical linkages that belong to a selected product can be displayed on an IBD representing the lowest layer (the product level) of the product portfolio structure. This analysis is called product-specific analysis. On a higher layer in the product portfolio structure different products were e.g. assigned to the SUV-class. At this level, an IBD can be created, that for instance represents all components and their technical linkages of products that belong to the SUV-class. In this case, a cross-product analysis of change propagation is possible, since components of different vehicles and their linkages are displayed in one diagram. Similar cross-functional analyses, e.g. across functions of the same cluster, can also be conducted in phase (4.3).

6. Conclusion and future work

Due to the amount and diversity of component dependencies in complex products, adequate product models are extremely relevant for ECM. In order to support engineers in assessing ECs and their impacts on other components, (customer) functions and products, different views and information on the product as well as it's elements and linkages have to be provided in one consistent ECM product model. With reference to these requirements, shortcomings of established product models in ECM are identified in this paper. Additionally, advantages of MBSE regarding ECM are discussed and a concept

https://doi.org/10.1017/dsd.2020.90 Published online by Cambridge University Press

for ECM of complex products (like vehicles) is presented. This concept consists of an approach using MBSE to create an ECM product model as well as a process describing how to generate the ECM model following different process phases. Based on these phases, the ECM model is created for an exemplary (customer) function from automotive engineering and resulting possibilities for ECM analyses of complex products are explained. Since the product model and process presented in this paper have not been evaluated in a regular product design process yet, the application of the approach in an industrial context is part of future work. Additionally, shortcomings of the ECM product model will be analyzed to identify possible improvements and increase the model's applicability and usability.

Disclaimer: The results, opinions and conclusions expressed in this paper are not necessarily those of Volkswagen Aktiengesellschaft.

References

Albers, A., Bursac, N. and Wintergerst, E. (2015), "Produktgenerationsentwicklung: Bedeutung und Herausfor derungen aus einer entwicklungsmethodischen Perspektive", *Stuttgarter Symposium für Produktentwicklung 2015*, Stuttgart.

Browning, T.R. (2016), "Design Structure Matrix Extensions and Innovations: A Survey and New Opportunities", *IEEE transactions on engineering management*, Vol. 63 No. 1, pp. 27-52. https://doi.org/10.1109/TEM.2015.2491283

Clarkson, P.J., Keller, R. and Eckert, C.M. (2005), "Multiple Views to Support Engineering Change Management for Complex Products", *3rd International Conference on Coordinated & Multiple Views in Exploratory Visualization - CMV 2005*, London. https://doi.org/10.1109/CMV.2005.20

Clarkson, P.J., Simons, C. and Eckert, C.M. (2001), "Predicting Change Propagation in Complex Design", *2001 ASME Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, Pittsburgh.

Danilovic, M. and Browning, T.R. (2007), "Managing complex product development projects with design structure matrices and domain mapping matrices", *International Journal of Project Management*, Vol. 25 No. 3, pp. 300-314. https://doi.org/10.1016/j.ijproman.2006.11.003

Delligatti, L. (2013), *SysML Distilled: A Brief Guide to the Systems Modeling Language*, 1st ed, Addison-Wesley.

Dori, D. (2016), *Model-Based Systems Engineering with OPM and SysML*, Springer New York, New York, NY. https://doi.org/10.1007/978-1-4939-3295-5

Feldhusen, J. and Grote, K.-H. (2013), *Pahl/Beitz Konstruktionslehre*, Springer, Berlin Heidelberg, https://doi.org/10.1007/978-3-642-29569-0

Flanagan, T.L. et al. (2003), "A functional analysis of change propagation", *International Conference On Engineering Design - ICED 2003*, Stockholm.

Haberfellner, R. et al. (2019), *Systems Engineering*, Springer International Publishing, Cham., https://doi.org/10.1007/978-3-030-13431-0

Jarratt, T., Eckert, C.M. and Clarkson, P.J. (2004a), "Development of a product model to support engineering change management", *International Symposium on Tools and Methods for Concurrent Engineering 2004*, Lausanne.

Jarratt, T. et al. (2004b), "Visualization Techniques for Product Change and Product Modelling in Complex Design", *International Conference on Theory and Application of Diagrams 2004*, Cambridge. https://doi.org/10.1007/978-3-540-25931-2_47

Keller, R. et al. (2005), "Visualising Change Propagation", *International Conference On Engineering Design - ICED 2005*, Melbourne.

Koh, E.C.Y., Caldwell, N.H.M. and Clarkson, P.J. (2012), "A method to assess the effects of engineering change propagation", *Research in Engineering Design*, Vol. 23 No. 4, pp. 329-351. https://doi.org/10.1007/s00163-012-0131-3

Lindemann, U., Maurer, M. and Braun, T. (2009), *Structural Complexity Management: An Approach for the Field of Product Design*, Springer-Verlag, Berlin Heidelberg. https://doi.org/10.1007/978-3-540-87889-6

Walden, D.D. et al. (2017), *INCOSE Systems Engineering Handbuch*, 1st ed., GfSE Verlag, München.

Wilms, R. et al. (2019), "Identifying Cross-Domain Linkage Types to Support Engineering Change Management and Requirements Engineering", *29th CIRP Design Conference*, Portugal, Póvoa de Varzim. https://doi.org/10.1016/j.procir.2019.04.224

Yildirim, U. and Campean, F. (2013), "An Enhanced Interface Analysis Method for Engineering Change Management", In: Abramovici, M. and Stark, R. (Eds.), *Smart Product Engineering*, Springer-Verlag, Berlin Heidelberg, pp. 191-200. https://doi.org/10.1007/978-3-642-30817-8_19

https://doi.org/10.1017/dsd.2020.90 Published online by Cambridge University Press