Towards Cross-domain Release Engineering - Potentials and Challenges for Automotive Industry

David Inkermann, Tobias Huth, Thomas Vietor

Institute for Engineering Design
Technische Universitaet Braunschweig
Langer Kamp 8, 38106 Braunschweig, Germany
Email: {d.inkermann, tobias.huth, t.vietor}@tu-braunschweig.de

***Abstract*—Product Development (PD) is facing fundamental challenges since the proportion of hardware and software-based functions realized to create innovative products is changing. Performance and enthusiastic attributes of products are more and more based of software providing new functionalities and services to the user. Short innovations cycles of software-based functions result in a decreasing life time of the overall products. Also, it is a trend that products are taken out of operation due to availability of products with new or enhanced functionalities and performance. However, from a technical viewpoint the products retired still provide full functionality. Release Engineering (RE) provides a concept to handle the different innovation cycles of subsystems and maintain or improve the functionality of a product within PD and during the whole life cycle. However, there is a divers understanding and focus regarding RE in the different engineering domains. This contribution discusses the basic concepts in the domains of software and mechanical engineering and highlights the challenges and potentials of RE in the field of automotive engineering. As a basis for further research, different fields of actions are highlighted.**

Keywords–Release Engineering; Release Planning; Innovation Cycles; Cross-Domain System Modelling; Automotive Engineering.

I. Introduction

There is an increasing trend to retire products before they reach their end of technical life time. Major reasons for this are that customer's decisions to acquire or substitute existing products are mainly based on enthusiastic attributes like comfort and entertainment functions or connectivity functionalities. These functions are often based on software and are driven by short innovation and technology cycles. However, hardware components are required to fulfil the software functions. As a result of the differing innovation and technology cycles of the hardware and software subsystems, there is an increasing gap between the technical and value life time of products [1]. This fact is highlighted by a study of the German Federal Environment Agency [2]. With focus on consumer products like laptops during the years from 2003 to 2013, there is a clear trend to replace products that are completely functioning because products with better performance and increased functionality are available; see Fig. 1.

On the one hand, this trend leads to a waste of resources since a big amount of material and energy used to produce the hardware is no longer used [1]. On the other hand, shortened product life cycles put challenges to product development. Albers et al. [3] formulates this dilemma by stating *"for economic and risk-minimizing reasons, as few subsystems as possible should be newly developed. Nevertheless, an innovative product with good performance and new enthusiasm attributes has to be developed"*. To address this challenge different approaches are proposed. Albers et al. highlight the need for a consequent product generation engineering by consequently reusing existing product concepts and perfuming variations (principle or geometrical variation) [3] [4]. Other approaches propose upgrades of products during the use phase by introducing new functionalities based on the concept of modular product concepts [5] [6]. Aside from this approaches in the mechanical domain, there are established methods of Release Management (RM) in the field of software engineering. Objective of these methodologies is to plan, develop and deploy releases to provide new features with minimal disruption of the existing product [7].

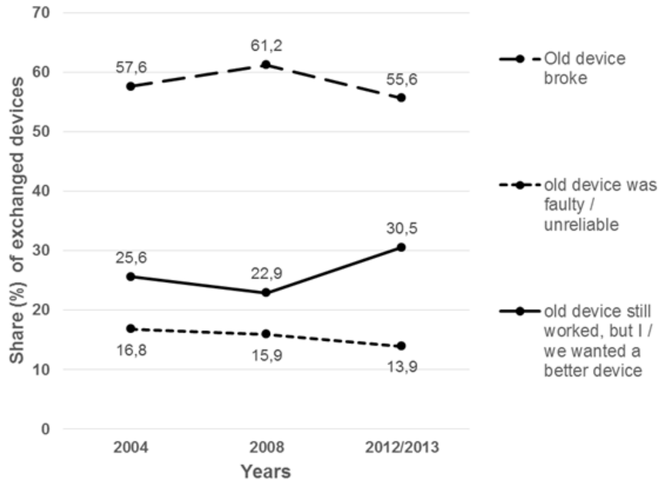

Figure 1. Reasons for the replacement of products and the increasing trend to substitute products with full functionality [2].

A. Basic Concept of Release Engineering

The basic concept of Release Engineering (RE) is to maintain and improve the performance and enthusiasm attributes of products by providing additional functionalities during the development and use phase of a product. Major drivers and objectives are (1) the bundling of development, testing and implementation activities during the development phase [7], the (2) consolidation of changes as well as the adaptation of variants [8] and the (3) implementation of innovations for product enhancement and life-cycle-accompanying updates [8]. These objectives are realized by planning and providing release units for the product. The term *release item* or *release unit* is defined in software engineering as the collection of one or more new or changed configuration items deployed into the live environment as a result of one or more changes [7]. Thus, a main task of RE is to define suitable release units in order to address the above mentioned drivers and aims. This highlights the strong interaction with the task of product architecture design [9], configuration management and change management [10]. Based on the evaluation of impact and weakening of releases Schuh [8] introduces a basic categorization of releases units; see Fig. 2. This categorization helps to define a suitable product architecture, determine appropriate release cycles and manage development to achieve the required degree of innovation over the product life cycle.

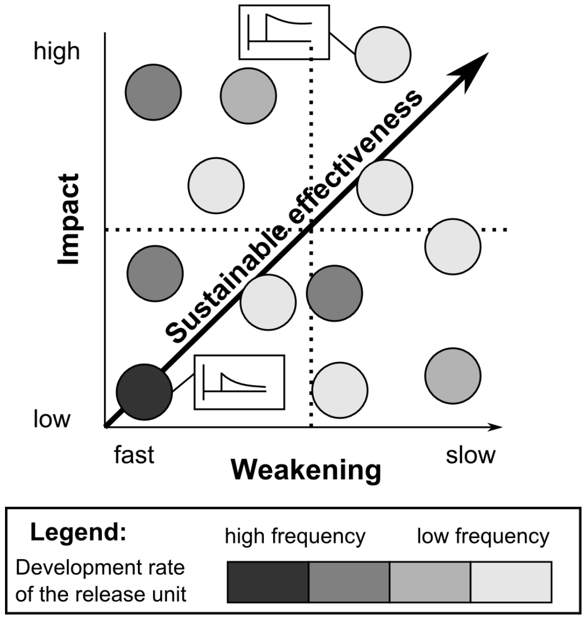

Figure 2. Categorization of Release Entities based on their Impact and Weakening for the Innovation of the overall Product [8].

In the field of software engineering the task of release unit definition is allocated to Release Planning [11]. Based on an extensive literature review Svahnberg et al. [12] highlight the diversity of objective and restriction to be considered when planning release units and frequencies. Aside from customer feedbacks, defects of previous releases, market factors and new customer demands, technical factors like the existing system architecture, interdependencies between requirements and the features to be included have to be incorporated.

To set the focus of this contribution in the following the term Release Engineering (RE) is used and defined following Aleksic [13]: *Release Engineering (RE) is a part of Release Management and defines release units, which are understood as assemblies or modules that can be assigned to specific release cycles*. The tasks and activities covered by this definition match with the common understanding of Release Planning in software engineering, however, the term Engineering is used to highlight the perspective of the mechanical engineering domain.

In order to apply the concept of RE to products inclosing subsystems of different engineering domains like automobiles, it becomes obvious that the definition of release units and cycles is a task involving all domains. To provide support for this interdisciplinary task in this contribution it is analysed how this task is supported in the different domains and which methods and approaches are used. Furthermore, it is discussed which challenges and potentials exist to apply cross-domain RE in automotive industry.

B. Research Focus and Outline

This contribution aims at introducing a basic understanding of cross-domain RE by analysing and comparing existing models and methodologies in the domains of mechanical and software engineering. The research is guided and structured by the following questions:

- Which approaches and methods for RE exist in mechanical and software engineering and what are their main objectives and principles?
- What are challenges in cross-domain RE and which information and product models are needed to support cross-domain RE?
- What are relevant fields for further research to support cross-domain RE?

In order to answer these questions in, Section 2, an analysis of existing approaches of RE is conducted focusing software and mechanical engineering. Based on this analysis, a brief comparison is presented. This comparison serves as a basis to formulate requirements and implications for cross-domain RE in general. In Section 3, potentials and challenges to apply cross-domain RE in automotive industry are discussed. With reference to existing methods and tools for architecture design and change management relevant fields for future research are derived and described in Section 4. Section 5 concludes the contribution with a discussion and brief outlook.

II. Existing Approaches for Release Engineering

Release Management is an established process in software engineering. The common understanding already emphasizes the importance of considering the interrelations between software and hardware systems when planning and developing releases [14]. However, there are different obstacles to overcome when introducing consequent RE into industrial practice. In the following paragraphs a brief overview of existing approaches is given in order to highlight their main differences regarding the objectives, principles and relevant product information and models. Based on a comparison requirements to support cross-domain RE are derived.

A. RE in Software Engineering

Importance of Release Management (RM) in software engineering is highlighted by a number of guidelines and standards for instance described in the IT Infrastructure Library (ITIL) [15]. The process of RM strongly connected to the software engineering process [16] and tasks of service management [14] and covers the superior tasks *defining*, *developing*, *implementing* and *operating* releases. Furthermore, a strong interrelation is given to the processes of configuration and change management since the release units have to be planned based on and extending the existing system architecture.

Different approaches and models exist to plan releases, differing between *ad-hoc* and *systematic* planning strategies. Ad-hoc planning is common in industrial practice [12] and has limited planning scope of one or two releases while systematic release planning considers a number of releases planned for the future. In order to time future releases two basic models stand out, namely time-based and feature-based releases [17]. Time-based release processes aiming at introducing major versions of the software with regular intervals, following a strict schedule. These time related release plannings are often applied in highly modular projects [18]. In contrast feature-based release processes are focussing on delivering a predefined set of features in one release; see Fig. 3. Because of the high percentage of 80% [19] of features being dependent in industrial software systems, it is vital to analyse their interdependencies when planning a release. According to Ruhe eight types of dependencies between features have to be distinguish [11]. In order to assign features to releases different methods exist, taking for instance into account the (subjective) priorities given to features by the different stakeholders and the estimated amount of resources consumed by the features. Established methods are described and reviewed by Ruhe [11] and Svahnberg et al. [12] including the EVOLVE II procedure [20] or the greedy planning algorithm [21] as common approaches. Like highlighted by Ruhe, these methods differ in their objectives spanning from value based to the maximizing the financial value function, the stakeholder involvement as well as the consideration of feature dependencies [11].

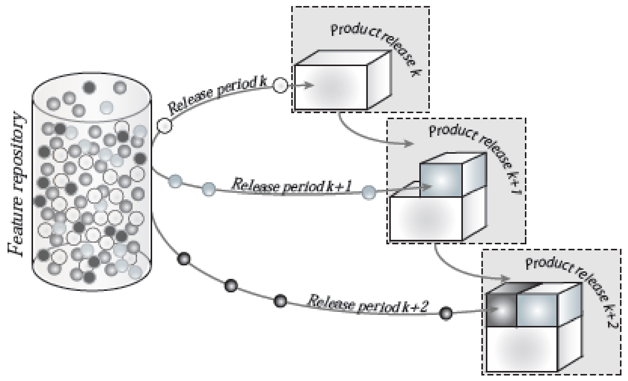

Figure 3. Schematic Illustration of Release Planning as the Selection and Assignment of Features to Releases [11].

Of special interest with regard to cross-domain RE are the models used to define and evaluate the release scope as well as the factors considered. This is often done by pre-selecting and prioritizing of features using methods like the Multiscore method or the voice of the customer [11]. Thus, main preconditions for feature-based release planning are project information including feature set, their description as well as stakeholders nominated for prioritization. Moreover, different soft and hard factors have to be considered during release planning. According to Svahnberg et al. [12], hard constraints include technical constraints, budget and cost constraints, resource constraints, effort constraints, and time constraints. Soft constraints cover stakeholder influence factors, risk factors, value factors, and resource consumption factors [12].

The brief description of RE in software engineering and the related methods on the one hand highlights that there are established approaches and detailed procedures like the EVOLVE II available. On the other hand it becomes clear that features are frequently used to describe the ”selling units of a product” and form the additional functionalities to be delivered by a release.

B. RE in Mechanical Engineering

Release Engineering is not a well-established concept in mechanical engineering. Research works around the group of Schuh propose to transfer the principles of RM from software domain to the domain of mechanical engineering. In analogy to the processes in software engineering they emphasize the strong interrelation to the processes and methods of modular product architecture design [5] and technical change management [10]. In addition the importance of anticipating innovations during the whole life cycle and the settlement of product variants are mentioned as major objectives [8]. Essential tasks of RE in mechanical engineering cover the definition of components to be substituted or added in order to provide additional functionalities and value to the customer. To support this task there are numerous methods described in literature addressing the definition of modules with regard to the functional and/or physical structure of the product. For instance the modular function deployment (MFD) proposed by Ericsson and Erixon uses so called *module drivers* to define suitable modules to build up the product [22]. With focus on modular products this approach uses a matrix for mapping functional requirements to certain modules. The Design for Variety introduced by Martin [23], focusses on creating robust platforms for modular products and reducing interdependencies between system elements. By introducing the Coupling Index (CI) that specifies the strength of the connection between the components of a product the importance of the physical system structure to define suitable modules (release units) and evaluate the design effort is highlighted. To represent and analyse interdependencies between components Design Structure Matrices are often used [24]. Here, the system is decomposed into single parts or subsystems and the different interactions for instance *spatial*, *energy*, *material*, and *information* are denoted within the cells. By analysing these interactions clusters with strong interrelations between parts and subsystems can be identified that are suitable to bundle development effort or changes [25]. Other works provide principles to enable changes in systems throughout the life cycle. These principles focus on suitable system architectures to implement changes required for upgrading or releasing new derivate with small impact on the existing product [26]. Major objective of the concept of design for changeability is to increase the changeability of products with regard to the dynamic of marketplaces, technological evolution, and varying environments.

Although, there are numerous methods to support system architecting and technical change management in mechanical engineering there are less approaches that address the timing of releases based on the consolidation and bundling of changes and consideration of innovation gaps. A basic concept to define release cycles and synchronize changes is proposed by Aleksic [13]. He introduces the module change flexibility classification number (MCF) to support future planning of changes. The MCF is a result of the evaluation of the dimensions *market driven requests caused by customer demands*, *production costs*, *one-time complexity costs*, *running complexity costs*, and *module interaction*. Each dimension is evaluated by the classification number D_{MCF_i} with a maximum value of nine. The factor g_i is used to weight the dimensions and adapted the importance of each dimension to the boundary conditions of different companies and targets. The product of the classification number D_{MCF_i} and the weighting factor g_i is normalized by the sum of the g_i and x_i, which is consistently given the maximum value of nine [27].

$$MCF_g = \frac{1}{\sum_{i=1}^{5} g_i \times x_i} \times \sum_{i=1}^{5} g_i \times D_{MCF_i} \tag{1}$$

According to the given equation, the MCF ranges between zero and one. This value is used to locate each module within an onion peel model visualizing the incensement of the MFC from the inside to the outside; see Fig. 4. Modules placed in the inner shell are thus less flexible than these placed at the outer shell.

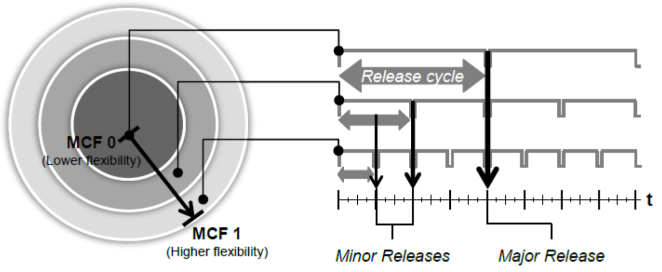

Figure 4. Onion Peel Model to Visualize the Module Change Flexibility Classification Number (MCF) and Resulting Release Cycles [27].

The MCF and the onion peel model help to identify modules that can be changed with small efforts and those that can hardly be changed without affecting other modules. Furthermore, the onion peel model can be used to plan release cycles systematically by defining suitable release frequencies with regard to the MCF. Here, high flexibility modules are chosen to by changed more frequently since they cause lower change effort. Furthermore, the onion peel model gives advices of how to structure the product architecture in order to enable releases.

From the brief discussion of the methods it becomes clear that the content of release in the field of mechanical engineering is defined by modules. These modules contain different parts of the mechanical structure and are defined using established methods considering for instance life cycle and technology aspects. However, there is now established procedure or method covering all activities required for RE.

C. Comparison of RE Approaches

Based on the domain specific methods and approaches for RE described beforehand, this paragraph introduces a brief comparison. The comparison presented in Table I highlights the main differences according to the considered *time horizon*, *main objectives and drivers*, *required product information and preconditions*, *content of releases*, *release cycle definition*, and *considered restrictions*.

From the comparison, it becomes clear that main differences of existing RE approaches in software and mechanical engineering concern the product information used to define releases as well as the content of the releases and the restriction considered. While RE in software engineering is based on features and their relations within the software system, in mechanical engineering modules are used to define release units. This points out that on the one hand in both cases the representation and analysis of the system structure and the included interdependencies between the elements (components or function) is essential when defining release units and on the other hand (changing) requirements and user needs are important for suitable planning of releases. A main deficit of RE related approaches in mechanical engineering is the missing support of adequate release cycle definition. Most of the existing methods focus on the initial definition of the product architecture but do not consider the possibilities of consequently delivering new or additional functionalities by releases. One main obstacle to do this in industrial practice can be found in the high efforts and costs caused by production of hardware parts.

Based on the given comparison the following requirements can be formulated to support cross-domain RE:

- A cross-domain linking between features and components or functions is needed to define suitable release units and provide new and additional functionalities to the users.
- The different interdependencies between features and components of initial and existing systems architectures have to be modelled to support planning for instance of product upgrades.
- An interdisciplinary requirement and innovation management is required to ensure value oriented planning and definition of release units across domains.

While the first two requirements pertain to the activities of interdisciplinary system modelling, the third one is concerning assisting processes and information needed for planning and evaluation activities within RE. Based on these findings and requirements, in the following section, potentials and challenges of RE in automotive industry are briefly discussed.

III. Towards Cross-Domain RE in Automotive Industry

Automobiles are complex mechatronic systems. Due to increasing value creation based on software-based functions like comfort or assistance functionalities there is a trend towards decreased use phases of cars like discussed in Section 1. This results in increasing pressure to shorten development times and coordinate the cross-domain development activities. To cope with the resulting complexity of highly linked organisational structures, requirements, development documents and product structures and processes different approaches exist including RM. Fig. 5 illustrates the hierarchical structuring of systems and organization as well as the correlations between data and processes using electrical and electronic systems as an example. Also here RM processes on different system levels are represented. Major objective of these RM processes is to

TABLE I. Basic Comparison of RE Approaches in Software and Mechanical Engineering.

Characteristics	*Software Engineering*	*Mechanical Engineering*
Time Horizont	Next release (ad-hoc planning) and overall life cycle (systematic planning)	Overall life cycle of the product and single modules
Main Objectives and Drivers	Change requests (errors or upgrades), Bundling of development, testing and implementation activities (temporal and functional)	Implementation of "innovations" for product enhancement, life-cycle-accompanying updates, consolidation of changes and innovations, adaptation of variants
Product Information and Preconditions Required	Features sets of the software system, interdependencies between features, requirements	Functions and components of the product, dependencies between components, requirements
Related Activities	Configuration management, change management, requirement management, service & quality management	Technical change management, product architecture design, requirement management, life cycle management
Content of Releases	Features as value units for customers	Modules as changeability units of the product
Release Cycle Definition	Considered as essential part of RE	Not consequently considered
Restrictions Considered	Technical constraints, budget and cost constraints, resource constraints, effort constraints, and time constraints, stakeholder influence factors, risk factors, value factors, and resource consumption factors	Market driven requests, production costs, complexity costs, innovation cycles, module interactions

release subsystems at specific points of time during the PD process like quality gates or product starts [28]. Necessity and complexity of these RM processes results from the different hardware and software versions as well as for instance electrical control units' development during the PD. The introduced RM aims on ensuring proper functioning of all variants as well as the total car system by initiating tests and changes.

The described understanding and process of RM in automotive industry is basically different from the understanding introduced before since it focusses to support the efficient development a predefined configuration of subsystems and functions. The releases to be delivered in this context are solutions or variants of subsystems of the overall car system. At the same time the predefined point of times serve to consolidate and coordinate required cross-domain changes for following development activities. With regard to the understanding of RE introduced in Section 3 the following basic concepts to handle releases in PD have to be distinguished:

- Releases as *development units* to be delivered at predefined points of time in the PD process in order to handle process and product complexity based on a defined configuration.
- Releases as *value and innovation units* to enable upgrading of the product during use phase including changes and expansions of the system configuration.

Independent from the applied concept of releases, it becomes clear that there is a strong interrelation to the structuring of the automobile system since the content of the releases in both cases is modules of the system including hardware and software. Thus, the illustrated hierarchical structuring of processes, systems and related requirements and data have to be considered when discussing potentials and challenges of RE in context of automotive industry.

In the following paragraphs, essential potentials and challenges of cross-domain RE in automotive industry are described focusing on the concept of releases as *value and innovation units*.

A. Potentials of Cross-Domain RE in Automotive Industry

Based on the introduced understanding of RE the following potentials can be formulated for automotive industry with regard to the vehicle system:

- Consequent RE enables to continuously provide new and additional functionalities to customers by upgrading existing vehicle systems.
- Value oriented RE supports diversification of vehicle systems and innovation leadership of by integration by efficient market launches of new technologies.
- Life cycle oriented RE contributes to the reduction of resources needed for manufacturing of hardware components.

These potentials highlight the use of RE from the external (customer) and internal (manufacture) viewpoint related to the product. At the same time, positive implications of RE can be expected with regard to the PD process:

- Definition of technology and innovation oriented release units enable to prioritize development activities. Release units with short technology and innovation cycles can be developed and implemented late in the PD process while those with longer cycles should be part of the long term development.
- Function and value oriented definition of release units supports cross-domain engineering and coordination of change and integration activities.
- Consequent RE helps to shorten PD time by release a basic configuration of the vehicle system that is continuously extended by further releases.

It is obvious that realization of these potentials requires the definition of release units across domains since there are strong interrelations between the subsystems needed to realize the intended functions. Thus, it is essential to analyse and modify the interrelations between subsystems of the different engineering domains both on the level of system structure and requirements. However, the existing structure of systems and document based development activities established in automotive development projects, as shown in Fig. 5, often hinder to identify and represent domain crossing links needed for release planning. In the following paragraph the major challenges hindering cross-domain RE are described.

B. Challenges of Cross-Domain RE in Automotive Industry

Consequent implementation of the principles in industrial practice of automotive development is hindered by organizational, process and product related issues. On the one hand there are established – in most cases hierarchical– approaches to decompose systems and structure processes in order to handle complexity. Up to now, systems and processes are structured with regard to the engineering domain (mechanical, electric/electronical, and software). This also results in domain specific models and documents to hold product

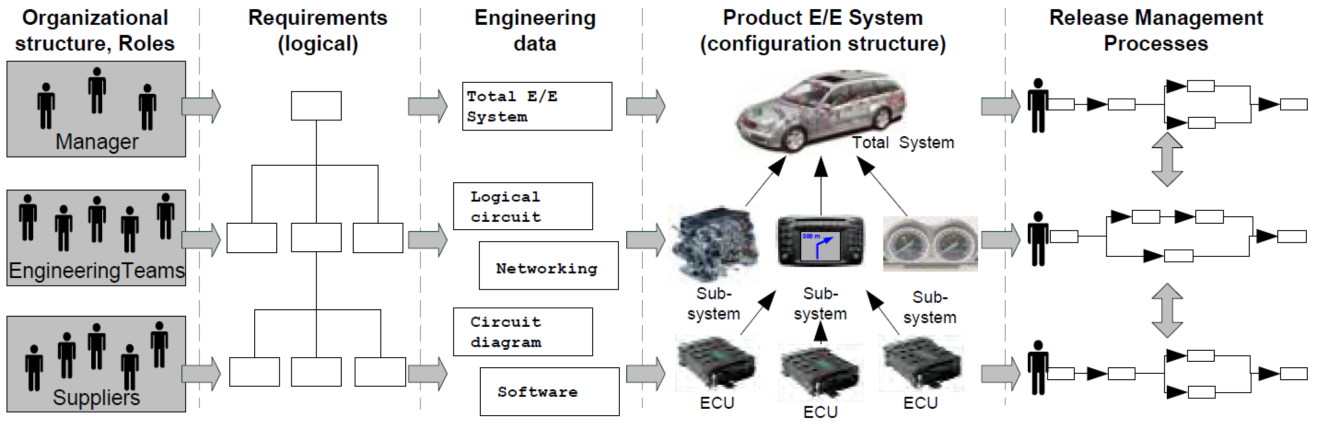

FIGURE 5. Schematic illustration of the structure and interrelation of organization, requirements, documents, product structure and processes in E/E development [29].

relevant information. In consequence there are less models and documents representing interrelations between subsystems across domains. With regard to RE this leads to the following challenges:

- Processes, documents and models of automotive development are structured with regard to components and systems and not value or function oriented.
- Configuration management is often done on the level of components and functions in the single domains using specific documents and models without linking information.
- Definition of suitable release units is complicated since technical constraints cannot be elaborated.
- A consistent requirement management addressing all involved domains and system levels is missing but needed for release planning and definition.

These boundary conditions highlight challenges that have to overcome when introducing cross-domain RE. It becomes clear that required changes to support cross-domain RE address processes as well as the way and structure the emerging products are described by models and documents. In order to overcome challenges and support the changes required in the following section fields of research are introduced.

IV. Fields for Future Research to support Cross-Domain RE

Based on the discussed potentials and challenges to introduce cross-domain fields for further research can be derived. The fields named in the following were identified based on observations in industrial practice and the analysis of existing approaches and principles of RE in the domains of mechanical and software engineering:

- Development of value and innovation oriented descriptions of systems and subsystems (related to requirement management).
- Development of modelling techniques to represent interrelations between components (mechanical domain) and functions (electric/electronical and software domain) on different level of aggregation and with regard to different kinds of relations like geometrical, logical or functional constraints.
- Development of methods to support cross-domain definition of release units with regard to different innovation cycles.

The formulated fields of research highlight the most relevant areas to work on. In order to establish cross-domain RE approaches and principles interdisciplinary research is essential to integrate viewpoints and methods of software and hardware engineering. Thus, the fields of research are closely connected with the area of systems engineering and address the fields of configuration management, requirement management, change management, and life cycle engineering.

V. Discussion and Conclusion

Objective of this contribution was to introduce a basis understanding of cross-domain RE based and the analysis of methods and principles in software and mechanical engineering. Moreover, it aims on pointing out potentials and challenges of cross-domain RE for automotive industry.

The analysis and comparison presented in Section 2 represent a first overview on existing and related methods used to support RE. It points out the main differences between software and hardware engineering for instance with regard to information of the product and preconditions needed to define release units. However, it is limited in its focus and conclusions to be derived because of the small number of methods analysed. The potentials and challenges formulated in Section 4 are based on observations in automotive industry and derived from the general potentials of RE described in literature. In further works potentials have to be analysed in more detail for instance by analysing examples from other industries. The challenges also have to be clarified based on the specific boundary conditions of industry partners.

Further work will focus on the fields defined in Section 5 as well as case studies to refine requirements for modelling techniques and methods to support cross-domain RE. Short-term work aims on applying existing and adapted methods of RE for an interior subsystem of an automobile. Furthermore, research will be conducted to analyse the interrelations and impact of intelligent manufacturing approaches and technologies like Internet of Thinks or flexible production concepts to the aspects of Release Engineering. Here, the focus will be to investigate new possibilities for dynamic planning and agile development concepts in PD.

REFERENCES

[1] Y. Umeda, T. Daimon, and S. Kondoh, "Life Cycle Option Selection Based on the Difference of Value and Physical Lifetimes for Life Cycle Design," in Proceedings of the International Conference on Engineering Design (ICED), August 28–31, 2007, Paris, France, J.-C. Bocquet, Ed. The Design Society, 2007, pp. 145–146.

[2] Umweltbundesamt, "Texte 11/2016: Einfluss der nutzungsdauer von produkten auf ihre umweltwirkung: Schaffung einer informationsgrundlage und entwicklung von strategien gegen obsoleszenz," 2016.

[3] A. Albers, N. Bursac, and E. Wintergerst, "Produktgenerationsentwicklung - Bedeutung und Herausforderungen aus einer entwicklungsmethodischen Perspektive," in Tagungsband Stuttgarter Symposium fuer Produktentwicklung "Entwicklung smarter Produkte fuer die Zukunft", June 19, 2015, Stuttgart, Germany, H. Binz, B. Bertsche, W. Bauer, and D. Roth, Eds. Fraunhofer IAO.

[4] A. Albers and G. Moeser, "Modellbasierte Prinzip- und Gestaltvariation," in Tagungsband 14. Gemeinsames Kolloquium Konstruktionstechnik, October, 6-7 Oktober, 2016, Rostock, Germany, K. Broekel, J. Feldhusen, K.-H. Grote, F. Rieg, R. Stelzer, P. Koehler, N. Mueler, and G. Scharr, Eds. Shaker Verlag, Aachen.

[5] K. Ulrich and S. Eppinger, Eds., Product Design and Development. Irwin/McGraw-Hill, Boston, 2011, ISBN: 978-0073404776.

[6] W. Bauer, "Planung und Entwicklung Aenderungsrobuster Plattformarchitekturen," Ph.D. dissertation, Technische Universitaet Muenchen, 2016, ISBN: 978-3-8439-2778-9.

[7] "Ieee standard - adoption of iso/iec 20000-2:2012, information technology - service management - part 2: Guidance on the application of service management systems," 2013.

[8] G. Schuh, Ed., Produktkomplexitaet managen: Strategien - Methoden - Tools. Carl HanserVerlag, Muenchen, 2005, ISBN: 978-3446400436.

[9] T. Richter, D. Inkermann, and T. Vietor, "Product Architecture Design as a Key Task within Conceptual Design," in Proceedings of the International Design Conference (DESIGN), May 16–19, 2016, Cavtat-Dubrovnik, Croatia, D. Marjanovic, M. Storga, N. Pavkovic, N. Bojcetic, and S. S., Eds. The Design Society.

[10] U. Lindemann and R. Reichwald, Eds., Integriertes Aenderungsmanagement. Springer-Verlag, Berlin, Heidelberg, 1998, ISBN: 978-3-642-71958-5.

[11] G. Ruhe, Ed., Product Release Planning: Methods, Tools and Applications. CRC Press, Taylor & Francis Group, Boca Raton, London, New York, 2010, ISBN: 9780849326202.

[12] M. Svahnberg, T. Gorschek, R. Feldt, R. Torkar, S. Saleem, and M. Shafique, "A systematic review on strategic release planning models," Information and Software Technology, vol. 52, no. 3, 2010, pp. 237–248.

[13] S. Aleksic, "Nachhaltige Weiterentwicklung von modularen Produktarchitekturen durch Release-Management," Ph.D. dissertation, RWTH Aachen, 2015, ISBN: 978-3-86359-397-1.

[14] "Iso/iec 20000 part 1: Service management system requirements. geneva,switzerland: International organisation for standardisation," 2011.

[15] C. Wischki, Ed., ITIL V2, ITIL V3 und ISO/IEC 20000. Gegenberstellung und Praxisleit-faden fr die Einfhrung oder den Umstieg. Carl Hanser Verlag, Muenchen, 2009, ISBN: 9783446419773.

[16] "Iso/iec/ieee 15288: Systems and software engineering - system life cycle processes," 2015.

[17] M. Michlmayr, "Quality improvement in volunteer free and open source software projects - exploring the impact of release management," Tech. Rep., 2007.

[18] H. Wright, "Release Engineering Processes: Their Faults and Failures," Ph.D. dissertation, University of Texas at Austin, 2012.

[19] P. Carlshamre, K. Sandahk, M. Lindvall, B. Regnell, and J. Nattoch Dag, "An industrial survey of requirements interdependencies in software release planning," in Proceedings of the Symposium on Requirements Engineering, August 27–30, 2001, Toronto, Canada, F. Titsworth, Ed. IEEE Computer Society Washington, DC, USA.

[20] D. Greer and G. Ruhe, "Software release planning: An evolutionary and iterative approach." Information and Software Technology, vol. 46, no. 4, 2004, pp. 243–253.

[21] T. Cormen, C. Leiserson, and R. Rivest, Eds., Introduction to Algorithms. McGraw Hill, New York, 2009, ISBN: 978-0262533058.

[22] A. Ericsson and G. Erixon, Eds., Controlling Design Variants: Mudular Product Plattfroms. ASME Society of Manufacturing Engineers, Dearborn, 1999, ISBN: 978-0791801505.

[23] M. Martin, "Design for Variety - A Methodology fpr Development of Product Plattform Architectures," Ph.D. dissertation, Stanfort University, 1999.

[24] T. Browning, "Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions," IEEE Transactions on Engineering Management, vol. 48, no. 3, 2001.

[25] U. Lindemann, M. Maurer, and T. Braun, Eds., Structural Complexity Management - An Approach for the Field of Product Design. Springer-Verlag, Berlin, Heidelberg, 2009, ISBN: 978-3-540-87888-9.

[26] W. Fricke and A. Schulz, "Design for Changeability (DfC): Principles to enable Changes in Systems Throughout their Entire Lifecycle," Systems Engineering, vol. 8, no. 4, 2005, pp. 279–341.

[27] G. Schuh, S. Aleksic, and S. Rudolf, "Module-based Release Management for Technical Changes," in Progress in Systems Engineering - Proceedings of the Twenty-Third International Conference on Systems Engineering, August, 2014, Las Vegas, NV. Springer International Publishing, 2014, pp. 293–298.

[28] D. Mueller, J. Herbst, M. Hammori, and M. Reichert, "IT Support for Release Management Processes in the Automotive Industry," in Proceedings of the International Conference on Business Process Management (BPM 2006), September 5–7, 2006, Vienna, Austria. Springer-Verlag Berlin Heidelberg, 2006, pp. 368–377.

[29] K. Rouibah and K. Caskey, "A workflow system for the management of inter-company collaborative engineering proces," Engineering Design, vol. 14, no. 03, 2003, pp. 273–293.