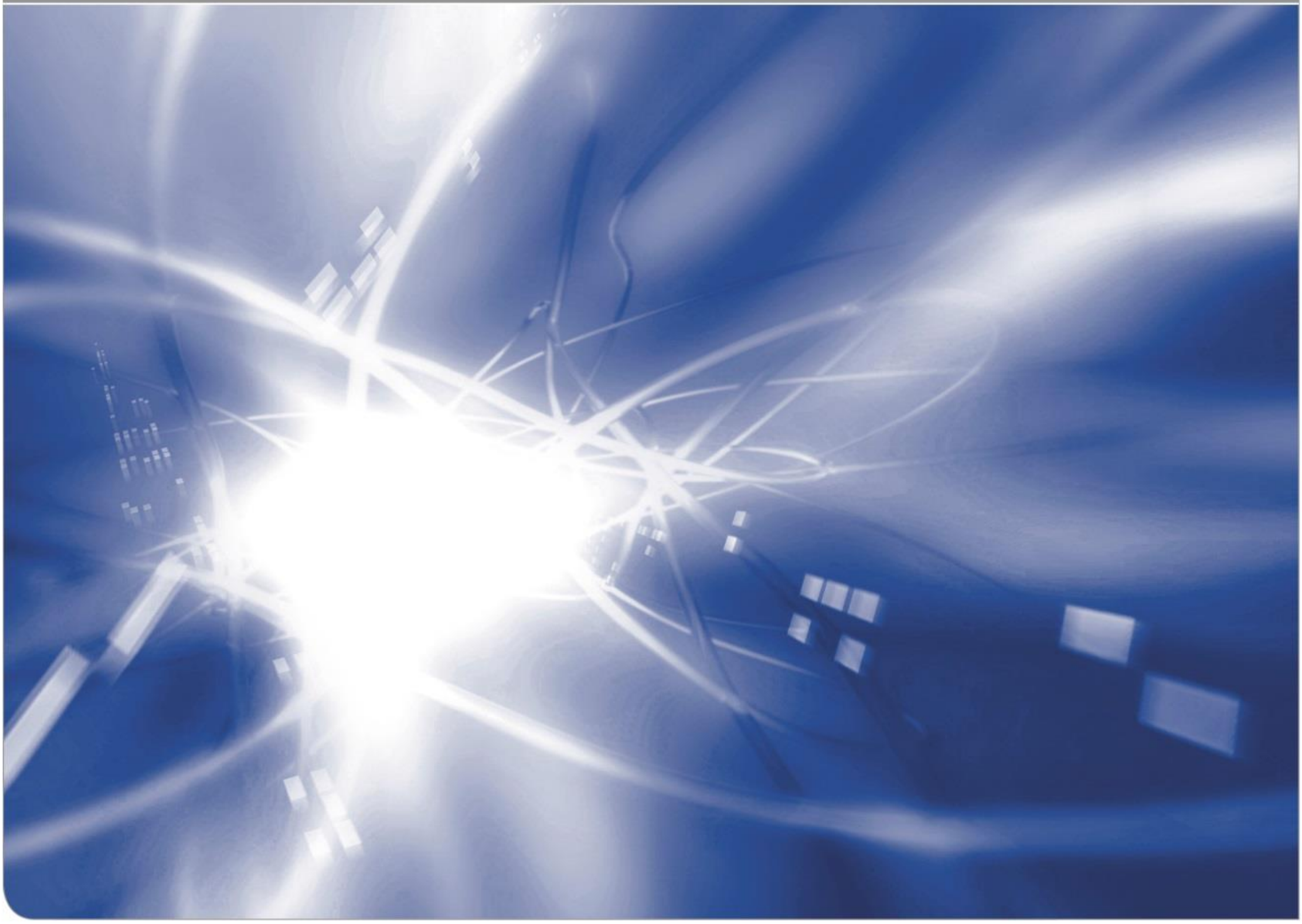
Karlsruhe Institute of Technology

Supporting early robust design for different levels of specific design knowledge – An adaptive method for modeling with the Embodiment Function Relation and Tolerancing model

by Jiahang Li[1], Dennis Horber[2], Patric Grauberger[1], Stefan Goetz[2], Sandro Wartzack[2], Sven Matthiesen[1]

KIT SCIENTIFIC WORKING PAPERS 246



[1] Karlsruhe Institute of Technology (KIT), IPEK – Institute of Product Engineering
[2] Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), KTmfk – Engineering Design

IPEK – Institute of Product Engineering

Kaiserstraße 10
76131 Karlsruhe
www.ipek.kit.edu

Impressum

Karlsruher Institut für Technologie (KIT)
www.kit.edu

2024

ISSN: 2194-1629

Abstract
Early robust design can lead to significant cost savings in the later stages of product development. In order to design systems that are insensitive to various sources of deviation in the early stages, specific design knowledge (SDK) plays a crucial role. Different design situations result in higher or lower levels of derivable SDK, which leads to different activities to achieve the development goal. Until now, no method has been developed to consider the different levels of derivable SDK from various design situations for early robustness evaluation. To address the problem, this paper outlines an adaptive modeling method using the Embodiment Function Relation and Tolerance (EFRT) model. The method is developed in two contrasting design situations, each with a high and low level of derivable SDK, and evaluated in another two corresponding case studies. It has a consistent structure with five stages and gates. At each stage, the derivable SDK is taken into account and the individual modeling steps are adapted. This method provides design engineers with concrete support for early robustness evaluation of a product concept in different development scenarios.

1. Introduction

Today's market demands fast development cycles with high product functionality. During a product’s lifecycle, its design parameters are prone to deviation, e.g., geometric deviations caused by manufacturing, wear during usage, or deviations in environmental conditions. Despite these deviations, technical systems must reliably complete their tasks throughout their entire life cycle. Robust Design (RD) aims to design systems that are insensitive to various sources of deviation, which originates from the quality engineering framework according to Taguchi et al. (2005). Implementing RD in the early stages can reduce costly iterations during later product development (Jugulum and Frey, 2007). The challenge for early robustness evaluation of a product concept is that often only qualitative data is available, which makes simulation or experimentation for robustness optimization difficult. One possibility for evaluating the product concept without quantitative data is qualitative modeling, which represents the properties and characteristics of a system without detailed parameters (Grauberger et al., 2022). An example of a qualitative model is the Embodiment Function Relation and Tolerance (EFRT) model developed by Horber et al. (2022). Using the EFRT model, the robustness of the product concept can be evaluated with appropriate criteria in a given design situation (Li et al., 2023).

In order to perform effective qualitative modeling of a technical system, specific knowledge is required. This knowledge of describing the technical systems in the design process is defined by Hubka and Eder (1990) as specific design knowledge (SDK). Different design situations result in higher or lower levels of derivable SDK at the beginning of the development process, which leads to different activities to achieve the development goal (Ponn and Lindemann, 2005.).

Until now, no method has been developed to consider the different levels of derivable SDK from various design situations for early robustness evaluation. To successfully evaluate the robustness of a product concept through qualitative modeling, it is essential to investigate how the existing SDK affects the modeling activities. This paper aims to deal with the lack of consideration given to different design situations by developing a modeling method, which incorporates the EFRT model. This method provides design engineers with concrete support for the evaluation of the product concept through modeling in different development scenarios.

1.1 Early Robust Design

Existing conventional RD methods mainly focus on robustness optimization based on controlled experiments (Phadke, 1989; Taguchi et al., 2005). These methods need detailed product data in a well-defined shape, which is only available in the later stages of design (Hasenkamp et al., 2009). The conceptual design has a significant influence on product performance and cost (Ullman, 2010). Robust conceptual design is often ignored in practice, yet it is crucial to robust product development (Eifler and Schleich, 2021; Jugulum and Frey, 2007).

Several approaches have been developed to address this challenge. A common practice for early RD is to use design principles such as reducing load paths or avoiding overdetermination. (Andersson, 1997; Ebro et al., 2012; Ebro and Howard, 2016). The application of existing design principles is often difficult as they are not linked to the concrete product concept. To evaluate the concrete product concept, qualitative models such as Bond Graphs (Gawthrop and Bevan, 2007), Design Structure Matrices (Eppinger and Browning, 2012), or Characteristics Properties Models (Weber, 2014) are used to analyze the system structure behavior. A qualitative modeling approach that has already been applied to RD tasks is the Contact and Channel Approach (C&C²-A) (Grauberger et al., 2019; Matthiesen and Ruckpaul, 2012). It supports the understanding of the embodiment function relation (EFR). EFRs describe how the design parameters of a system in its embodiment affect its behavior and, through that behavior, the desired functional requirements that are fulfilled. In addition to analyzing the system behavior, other approaches focus on optimizing the system structure by appropriately allocating functions to different components of the system (Suh, 1998). In the early stages, the graph-based approach facilitates the representation of the structure of the product concept despite limited product information (Ballu et al., 2010). An example of such an approach is the tolerance graph. It enables the analysis of the geometric relations in a product concept, and therefore an initial robustness evaluation of the product concept in its early stages (Goetz et al., 2018). As the tolerance graph does not support the analysis of the system behavior, Grauberger et al. (2020) propose combining the tolerance graph and C&C²-A in a holistic model to support design engineers with extended modeling aspects for the product concept.

1.2 Embodiment Function Relation and Tolerance (EFRT) model

The EFRT model developed by Horber et al. (2022) integrates the benefits of both tolerance graph and C&C²-A by combining the two approaches. The EFRT model offers great potential for early robustness evaluation by modeling a wide range of qualitative information, such as the system structure for early tolerance design, or the EFR for analyzing the system behavior (Li et al., 2023). The models and the result of the robustness evaluation can be used in subsequent approaches (Horber et al., 2023), e.g., for the automated generation of a preliminary Computer-Aided Design (CAD) model (Goetz et al., 2022). Figure 1a) illustrates the combination of both approaches and the core elements in an EFRT model (Li et al., 2023). This figure also shows the relationships between the individual elements and how they can represent the information in the early stages of product development.

The initial workflow proposed by Horber et al. (2022) covers the main modeling steps for building the model, as seen in Figure 1b). This workflow is a general approach without consideration of design situations, and the level of detail in the initial workflow is limited. To better support design engineers, the workflow has to be further detailed and adapted to the design situation (Horber et al., 2022), since different processes are expected from different design situations in product development. In practice, design engineers often choose inappropriate design processes or methods due to a lack of specific knowledge (Wilmsen et al., 2019). Therefore, it is important to examine how the specific knowledge from different design situations affects the modeling activities.

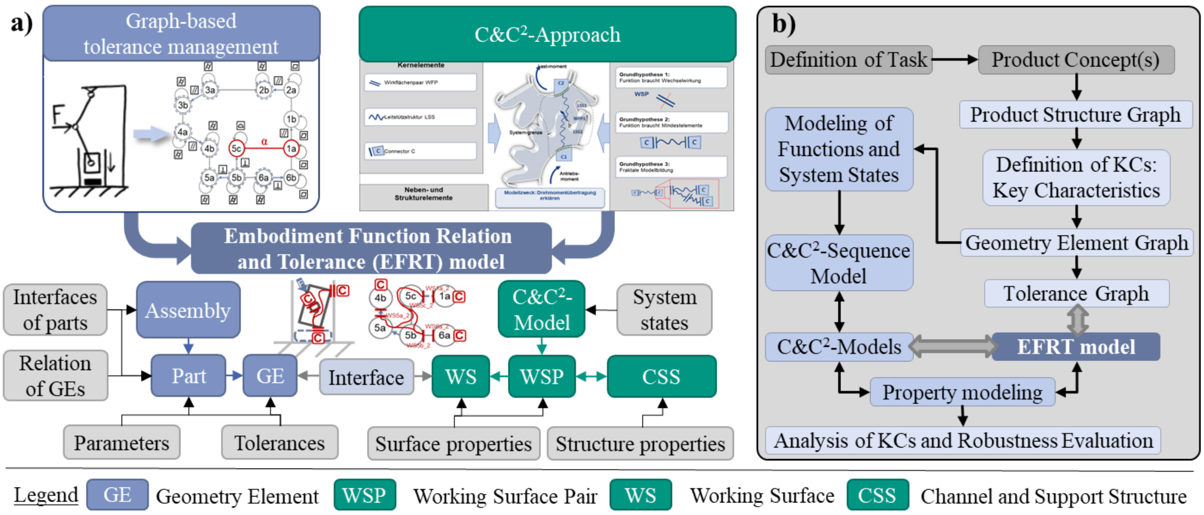


Figure 1. a) Summarized content and information of the EFRT model according to Li et al. (2023), b) Initial workflow according to Horber et al. (2022)

1.3 Design situations and Specific Design Knowledge (SDK)

In product development, design engineers must deal with different types of design situations that require different activities in the design process (Ponn and Lindemann, 2005.). At the start of development, there may be some differences in the design situation, such as how novel the design task is, what information can be used, and how mature the present concept is. These differences lead to different levels of derivable SDK. Derivable SDK means the SDK that can be derived from the data available at the start of development. In this contribution, we will explore three particular indicators of the design situation - the **level of innovation**, the **usability of data**, and the **design phase** - and their influences on the SDK. Other indicators could be considered, but the ones presented are sufficient to demonstrate this influence.

The first indicator of the design situation we explored is the **level of innovation**. Henderson and Clark (1990) distinguish between four types of product innovation: incremental innovations, architectural innovations, modular innovations, and radical innovations, where the innovation level increases. Pahl et al. (2007) categorize the design tasks into original design, adaptive design, and variant design, and suggest different strategies for each. Albers & Rapp (2022) state that every development of a new system is based on references, and they divide the development into principle variation, attribute variation, and carryover variation. The authors have differentiated the design tasks according to the level of innovation. This is because different difficulties are to be expected and therefore different design processes are required.

Higher levels of innovation result in lower levels of derivable SDK. Developing a new product concept involves more uncertainty than refining an existing one, which benefits from prior information in the reference systems (Pahl et al., 2007). The reference systems can vary in their level of innovation, such as references from the market, from the company, or previous products. Low derivable SDK occurs with absent or abstract references, e.g., in principle variation. High derivable SDK arises from concrete or carryover references, e.g., in an adaptive design.

The second indicator of the design situation we explored is the **usability of data**. In addition to considering the art of the reference system, data availability can affect design activities (Ponn and Lindemann, 2005.). Information becomes valuable only when it is used, as it is a resource that gains its worth from its utilization (El Hani et al., 2012). Therefore, it is necessary to check the usability of data. Such data can be qualitative, e.g., requirements specifications, lists of requirements, sketches, technical drawings, detailed CAD data, or bills of materials. It must be distinguished whether existing information can be reused or whether the information must first be prepared. Data processing effort distinguishes between abstract and concrete data. It determines the amount of information that can be obtained from the system.

A higher level of data usability indicates a higher level of derivable SDK, as more sources and evidence can be used to support and validate the product concept. Higher information usability also enables more analysis and simulation of the system design, which can enhance the SDK for further development steps.

The third indicator of the design situation we explored is the **design phase**. In product development, design activities are typically divided into different phases. For general design tasks, Pahl et al. (2007) divide the design into three phases: conceptual design, embodiment design, and detail design. Haik and Shahin (2011) supplement this division with a solution concept phase. Other divisions of design phases also exist for different development goals (Andreasen et al., 2015; Taguchi et al., 2005). Although there is no generalized separation of design phases, it can be said that early stages can be distinguished by the parameterization of the product. During these phases, the level of maturity of the concept is continuously increasing, and the difference in the concept maturity should be considered during modeling.

The level of derivable SDK increases as the design phase progresses and the concept becomes more mature. Niu et al. (2022) collected the key information items at different design phases in an explorative study, showing how the information content grows during the design phase. Capturing and storing the essential information generated throughout the product development process is the task of a design model (Eisenbart et al.). The EFRT model is proposed to support the early stages from the initial idea to the early embodiment design phase before the final elaboration of the geometric parameters. The challenge for the modeling is how to deal with the different maturities of the product concept.

The design situation determines which SDK is derivable at the start of the development, but this is not identical to the required SDK for the development goal. The relationships between the design situation, derivable SDK, modeling activities, required SDK, and development goal are illustrated in Figure 2.

As shown in Figure 2, different design situations lead to different levels of derivable SDK. Comparing the derivable SDK at the beginning of the development process and the required SDK for the development goal, different sequences of activities can be derived. Two contrasting cases deserve special attention. In the first case, the development project starts with a low level of experience and only references that are far away from the product to be developed are available. Therefore, the derivable SDK is insufficient and less than the required SDK. It is necessary to generate information and detail the desired concepts. In the other case, the development project is based on a previous product and starts with extensive data. The derivable SDK is larger than the required SDK. In this case, the information content and the derivable SDK are excessive. The information needs to be filtered. To efficiently model the product concept and obtain the required SDK for early robustness evaluation, this difference must be considered and the different modeling activities must be examined in different workflows.

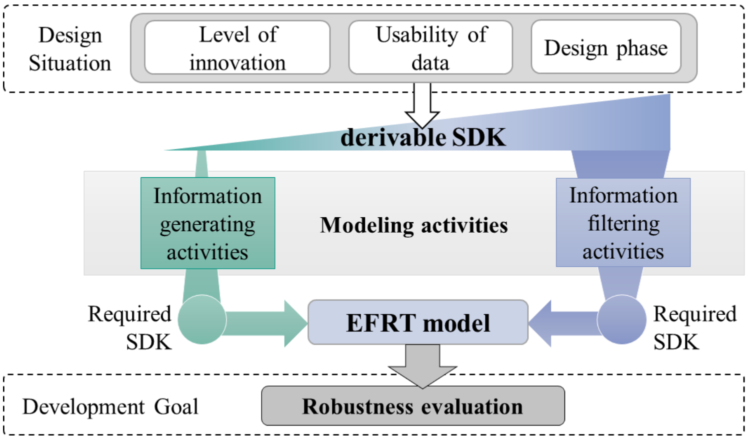


Figure 2. Derivation of modeling activities for the EFRT model from different specific design knowledge (SDK)

1.4 Research question

Due to the lack of detailed product information, e.g., a well-defined shape, the robustness evaluation of a product concept in the early stages of product development is challenging. Multiple approaches address this challenge by modeling different aspects of the product concept. Among these approaches, the EFRT model enables modeling a wide range of qualitative information in a product concept for the analysis of the robustness of the product concept. The combination of early tolerance design and EFR modeling offers a high potential for early robustness evaluation. Derived from the state of the art described in section 1.3, various design situations result in higher or lower levels of derivable SDK. Different modeling activities are required to achieve the development goal, as seen in Figure 2. However, the initial approach by Horber et al. (2022) was a general workflow without considering design situations. It remains unclear, how the design situations affect the modeling activities to build the EFRT model for the early robustness evaluation. Without properly adapting to the design situation, the model would be prone to becoming overly detailed due to the large amount of possible information and content, increasing the possibility that it fails to achieve its purpose through additional and unnecessary effort. To enable successful early robustness evaluation of a product concept with the EFRT model, it is essential to investigate how the derivable SDK affects the modeling activities. As a result, this paper addresses the following research question:

How can an EFRT model be built whilst taking different levels of derivable specific design knowledge for early robustness evaluation into account?

2. Methodical approach for developing the modeling method

To address the research question, the methodical approach begins with the formalization of the EFRT model to facilitate the development of the modeling method. Then, the boundary conditions for the modeling method are defined. Next, we develop the modeling method by investigating two different design situations. Finally, we evaluate the developed modeling method with two case studies. The main steps are illustrated in Figure 3 and described below.

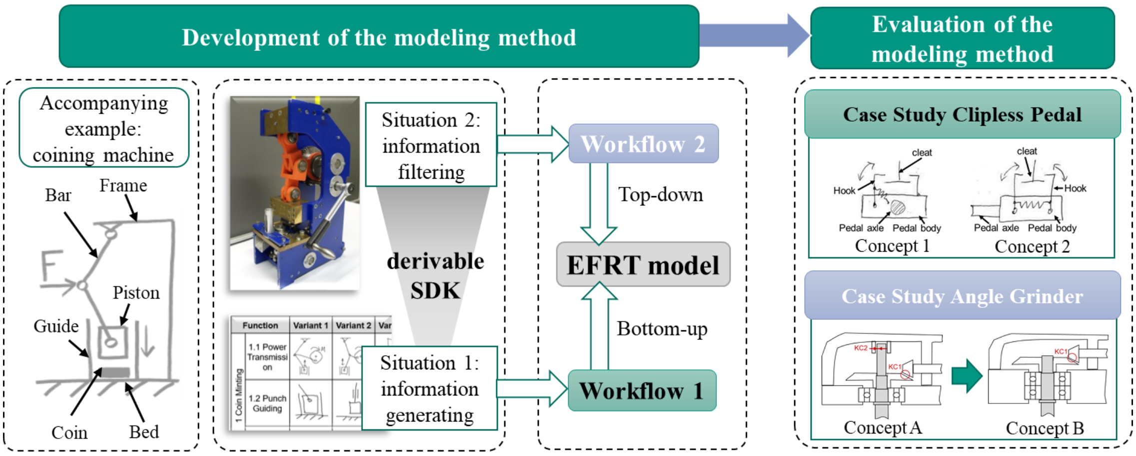


Figure 3. Methodical approach for developing the modeling method

Step 1: **Formalization of the EFRT model**

To develop a modeling method with the EFRT model, it is imperative to provide a sufficiently formal description of the EFRT model, which is currently lacking. In this step, we first provide a formal description of the EFRT model, including its core elements. Then, we investigate which aspect should be modeled with the EFRT model to evaluate the robustness of a product concept. To illustrate the model, we use a hand-operated coining machine as an example (see Figure 3). This mechanism is described in detail in Horber et al. (2022). The result of this step is a formalized description of the EFRT model and its modeling aspects. This formalization forms the basis for the development of the modeling method.

Step 2: **Framework for the modeling method**

After formalizing the EFRT model, we develop a framework that complies with some boundary conditions for the modeling method. Considering the differences in the design situations, the modeling method must facilitate the adaptation of different modeling activities within it. In this step, it should be explored how the modeling activities for the modeling aspects from Step 1 vary from different design situations. Despite the different activities, the modeling method should have a consistent structure. This will make it easier for users to follow the modeling method.

As the workflows aim to keep the modeling process simple, a stage-gate process based on the Quality Gates approach is used. This approach divides a process chain into phases and periodically checks the quality of the process. The process moves to the next stage when the results are mature enough, i.e., when a gate is reached (Hammers and Schmitt, 2009). The modeling activities in this paper are named after the summarized activities from Cash and Kreye (2017), partially adapted from Goetz et al. (2018) and Matthiesen et al. (2019) The modeling method to be developed can be considered as a sequence of information action and representation action according to Cash and Kreye (2017). The result of this step should be a framework that has a consistent structure but still allows for different modeling activities.

Step 3: **Elaboration of two workflows in the modeling method**

Within the framework of the modeling methodology developed in step 2, this step aims to explore different modeling activities depending on the derivable SDK. As derived from section 1.3, two cases are interesting to follow: one, where the design engineers do not have enough information and have to generate it, and second, where they have too much information and need to filter it. These result in different modeling activities and therefore, different workflows. A workflow can be understood as a specific sequence of activities. Based on the differences in the two opposite cases, two workflows should be developed.

The coining machine described above is used here as an accompanying example to illustrate the solutions. The development scenarios are defined as follows: a company receives an order to develop a coining machine for a customer. The robust and high-quality minting of coins is the primary goal of a coining machine. However, the company has no experience in this area, and the level of derivable SDK is low. In this scenario, a bottom-up method is needed to integrate the collected information into concepts. Later, a concept is selected, detailed, and produced. In the second scenario, a problem occurs during mass production, where the coin is partially minted crooked. The company wants to solve this problem in the next product generation of the coining machine. Since the previous product and its documentation already exist and various data can be collected, the level of derivable SDK is high. This scenario requires a top-down method due to the amount of data.

In these two cases, the EFRT model cannot be built in the same way because the derivable SDK is different. Using the coining machine, it is important to examine which activities are necessary to build the EFRT model for its robustness evaluation. The aim is to first integrate different modeling activities into the framework developed in Step 2 and then to investigate the differences between these two cases. The result of this step should be a unified modeling method with two workflows.

Step 4: **Evaluation of the modeling method**

In this step, the evaluation of the developed modeling method takes place using two technical systems, the clipless pedal and the angle grinder in respective development situations. In addition to checking the transferability of the developed modeling method, the differences between the two workflows derived from the development will be analyzed and evaluated with the case studies. With this analysis, the need for action for the further development of the modeling method can be derived.

The two technical systems used for the case studies are depicted in the overview in Figure 3. The first case study involves a company that is new to the design of clipless bicycle pedals and is entering the market due to the growth of the bicycle market. Since the development team has no access to references besides buying other click pedals from the market, the level of derivable SDK is low. The second case study is set in the concept phase of the development of the next generation of an angle grinder. A challenge is the high rejection rate of output shafts in the current generation, as very tight tolerances are required to mount them in the bearings. Due to the large amount of data from the previous development project, the level of derivative SDK is considered high.

3. Development of the modeling method

This section describes the results from the first three steps outlined in section 2. It is structured as follows: in section 3.1, the EFRT model is formalized. In section 3.2, the framework for the modeling method is derived. In section 3.3, the details of the developed modeling method are elaborated.

3.1 Formalization of the EFRT model

The formalization starts with a formal definition of the EFRT model. An EFRT model is a combined model derived from the tolerance graph and the C&C^2-A. It models the qualitative information in a product concept and aims to evaluate its robustness. In this paper, we construct the EFRT model into the EFRT graph and the EFRT sketch. In the EFRT graph, the product concept is decomposed into Geometry Elements (GE) and their geometric relations. In the EFRT sketch, the product concept is visualized with a proper sketch, supplemented with the key elements of the C&C^2-A.

For a more detailed introduction to the core elements of an EFRT model, Figure 4 on the left shows the EFRT graph and EFRT sketch with the example coining machine. The product concept of the coining machine and its main parts for the coining process are shown in Figure 4 on the left. To evaluate the fulfillment of functional requirements of a product concept, the concept of Key Characteristic (KC) is used in the EFRT model. A KC serves as a quantifiable product specification, whose deviation has a major impact on the fulfillment of functional requirements (Thornton, 2004). In an EFRT model, a KC can be integrated into the EFRT graph between GEs, or it can be drawn directly in the EFRT sketch (see Figure 4 – EFRT model).

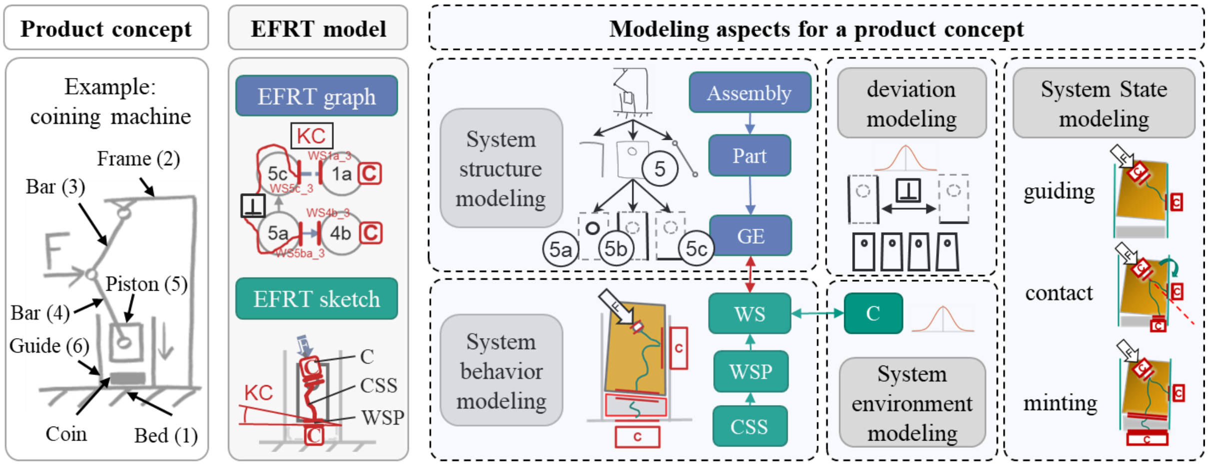


Figure 4. Formalization of the EFRT model (GE: Geometry Element, WS: Working Surface, WSP: Working Surface Pair, CSS: Channel and Support Structure, C: Connector)

To build the EFRT graph, the assembly is first divided into several parts, e.g., a piston. The next step is to divide a part into GEs, i.e., interacting surfaces. For example, the part piston has surfaces such as the hole, skirt, and crown that serve as GEs, as seen in Figure 4 in the middle (system structure modeling). In the EFRT graph, the GEs are represented as nodes, for example, 5c is the crown and 5a is the skirt. Their relations are labeled on the edges, for example, the perpendicularity of the crown surface to the skirt surface is labeled on the edge between 5c and 5a (see Figure 4 – EFRT graph). The graph is supplemented with key elements of the C&C^2-A. This supplementation has several advantages, such astracking of load path in the graph.

In the EFRT sketch, a certain area in the product concept, which is considered to be important for function fulfillment, is visualized with the elements from C&C^2-A. Three key elements are needed to describe a function: the Working Surface Pair (WSP), the Channel and

Support Structure (CSS), and the Connector (C) (Matthiesen et al., 2019). Beginning with the WSP, it describes, where exactly an interaction between the components occurs. A WSP is the place of the interface, where parts of the system connect while it fulfills its function, it consists of two working surfaces (WS). The path for the information transmission is defined as CSS. A CSS runs through parts in the system and connects two WSPs. Finally, the information on the system boundary is stored in the Connector (Matthiesen et al., 2019). In the accompanying example coining machine, a function for applying the minting force through the part piston on the coin is depicted in the EFRT sketch (see Figure 4).

An EFRT model can be used to model different aspects of a product concept. As described above, the process of building the EFRT graph serves to model the **system structure**, while the EFRT sketch aims to visualize the **system behavior** by finding the active WSP and CSS in the current system state. For a comprehensive view of a system, the **system environment** must be considered. The Connector is used to model the system environment, which includes information beyond the system boundary, such as external loads.

For robustness evaluation, various **deviations** can be modeled in an EFRT model and their influence on function fulfillment can be investigated. Geometric deviations in the system are modeled in EFRT graph and visualized in the EFRT sketch. Non-geometric deviations, such as material properties or changes in the system environment, can be stored in model elements such as WSP, CSS, and Connector. To evaluate the influence of these deviations on the fulfillment of functional requirements, the relation between deviations and KCs can be then analyzed using EFRT graph and EFRT sketch.

The analysis is often not sufficient in only one **system state**. Therefore, it is imperative to evaluate the robustness of the product concept in different system states. To model system states, the C&C^2 sequence models are used, which can be categorized into temporal or spatial states (Matthiesen et al., 2019). Figure 4 on the right shows three temporal states and the system behavior of a coining machine relevant to the coin minting function.

3.2 Framework for the modeling method

Considering the boundary condition of facilitating the adaptation of modeling activities while keeping a consistent structure, we developed a framework for the modeling method corresponding to a stage-gate process. An overview of the framework is shown in Figure 5. The structure comprises five stages: 1. *Define*, 2. *Sketch*, 3. *Structure*, 4. *Model*, and 5. *Decide*. The gates for each stage also follow the common structure: 1. defined task and data, 2. sketch, 3. product structure graph, 4. EFRT model, and 5. decision. Two workflows are integrated in this common framework, while the modeling steps of the workflows in each stage are different.

Stage 1 *Define* is the initial processing of the derivable SDK in a database to support the next stages. In *Stage 2 Sketch*, the product concept with its main parts for fulfilling the functional requirements are outlined in an appropriate and simplified sketch. In Stage 3 *Structure*, the parts in the product concept and their geometric relations are structured and integrated in the product structure graph. In Stage 4 *Model*, the EFRT model is built with a sufficient resolution. By Analyzing the influences of deviations on the function fulfillment, the robustness of the product concept is evaluated in Stage 5 *Decide*.

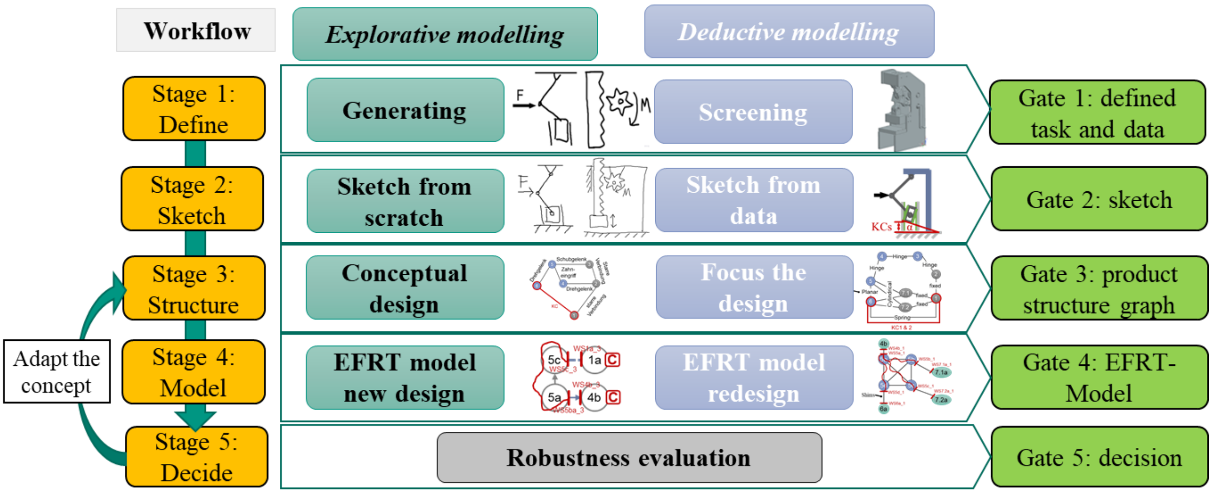


Figure 5. Overview of the framework for the modeling method with two workflows. *Explorative modeling* is shown on the left side and *Deductive modeling* is on the right side

As described in section 1.3, modeling activities are different due to the level of the derivable SDK. With a low level of derivable SDK, it is difficult to define the task accurately at the first time, and the system structure must be conceptualized first. While the system environment and system states are mostly unknown, the deviations in the system should be estimated based on prior knowledge and experience. Information such as material properties must be defined or collected for a more detailed analysis of the system behavior. With a high level of derivable SDK, it is imperative to first filter the information to a sufficient amount. Meanwhile, some modeling aspects described in section 3.1 can be found or derived directly from the existing data. The required modeling activities for different levels of derivable SDK are investigated with two cases described in Section 2. As a result, the modeling method for the EFRT model is divided into two workflows (see Figure 5):

- *Explorative modeling*: low level of derivable SDK
- *Deductive modeling*: high level of derivable SDK

3.3 Elaboration of two workflows in the modeling method

This section presents the modeling activities through five stages in the two workflows within the modeling method. The steps in the workflows and a first analysis of the differences between the workflows are described in the following subsections 3.3.1-3.3.5.

3.3.1 Stage 1 Define

Figure 6 shows the steps of the workflows in the Stage 1 *Define*. In this stage, the task should be clarified and information should be gathered and documented. At the end of this stage, sufficient data should be available for analysis in the next stages. Gate 1 of the stage is then the defined task and data that will serve as a database for the later modeling process. The letter L in the steps stands for low level of derivable SDK and H stands for high level.

Explorative modeling is shown in Figure 6 on the left. At the start of the development, the level of derivable SDK is considered to be low. In Step L1.1 the task is defined. In Step L1.2, generating information, the state of the art and the possible principle solutions from the market are researched, and principle solutions can be created using ideation methods such as brainstorming (Hatcher et al., 2018). The principle solutions to the tasks are generated methodically, for example in a morphological box (Zwicky, 1967). With Step L1.3, documentation of gathered data, Gate 1 is reached.

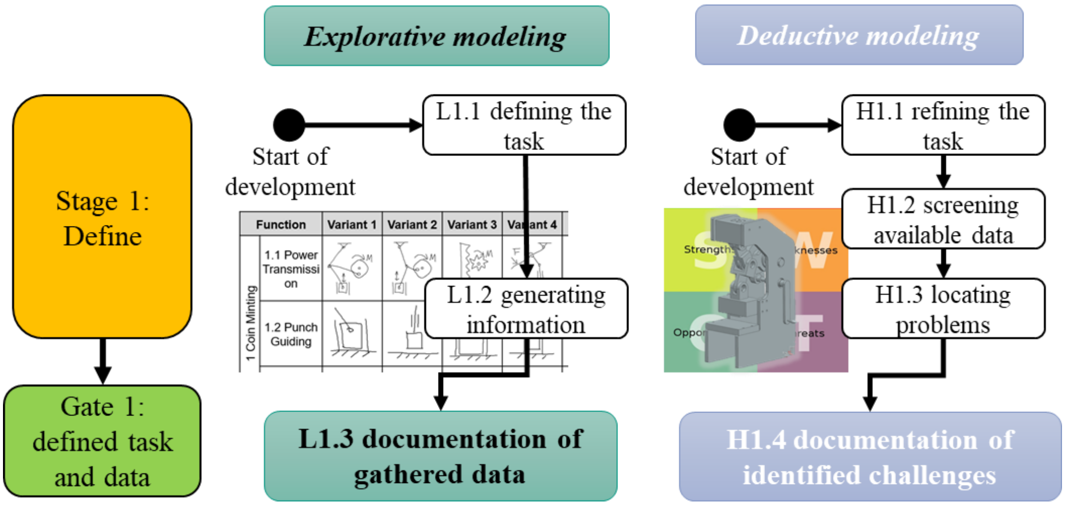


Figure 6. Overview of Stage 1 Define

Deductive modeling is shown in Figure 6 on the right. At the start of the development, the level of derivable SDK is considered to be high. In step H1.1, refining the task, problem areas are identified, resulting in a need for further development. In Step H1.2, screening available data, relevant data from references are screened. References such as CAD data from the previous product generation can be used, as well as external references, e.g., competitor products. Step H1.3 is to locate "problem spots" in existing data. This can be done using problem analysis methods such as SPALTEN (Albers et al., 2016) or SWOT (Strengths, Weaknesses, Opportunities and Threats) analysis. Step H1.4 is the documentation of the identified challenges.

For the accompanying example coining machine, in *Explorative modeling*, basic information about the system to be developed is gathered. In Step L1.1, the task is defined as minting the coin consistently, which results in a constant height and as the target variables. This is also the basis for the later evaluation. In Step L1.2, different principle solutions are collected or developed in a morphological box. In *Deductive modeling*, a situation analysis and problem definition are carried out in this stage, the problem from the previous product generation is that the coin was minted crooked. This means that the actual function (crooked minting) deviates from the target function (high minting quality). Therefore, the task is refined to improve the parallelism from both sides of the coin in Step H1.1. In Step H1.2, the CAD data of the previous coining machine is screened and analyzed. As a result of Step H1.3, the problem lies with the coining mechanism in the guide area.

The primary difference between *Explorative modeling* and *Deductive modeling* is how the knowledge is structured from the different levels of derivable SDK. The Information content of *Deductive modeling* is higher than that of *Explorative modeling* at the start of development. During task defining, *Explorative modeling* requires defining the task if it is unclear, whereas *Deductive modeling* requires refining the task under the consideration of current problems. In the accompanying example, *Explorative modeling* aims to mint a coin, but the solution is not yet known. Conversely, with *Deductive modeling*, the task is more concrete: improving coin quality. In *Explorative modeling*, identifying problems can be difficult due to a lack of information. In *Deductive modeling*, it can be challenging to identify causes of known problems. To reach Gate 1, *Explorative modeling* requires gathering information, while *Deductive modeling* requires screening data and selecting relevant information.

3.3.2 Stage 2 Sketch

Figure 7 shows the workflow steps in Stage 2 *Sketch*. In this stage, the documentation from Stage 1 is used to sketch the product concept. Gate 2 consists of sketches that represent the principle solutions and have sufficient resolution for analysis in subsequent stages.

Figure 7 on the left illustrates the process of *Explorative modeling*. In Step L2.1, selecting solutions, the collected principle solutions are selected with appropriate methods considering advantages and disadvantages, e.g., the weighted sum model (Pahl et al., 2007). Bad solutions can also be initially eliminated with a rough estimate based on experience. Step L2.2 involves defining the temporal or spatial system boundary. When defining the spatial system boundary, it is necessary to consider the connecting components outside the system boundary that are essential for concept realization. Finally, Step L2.3 involves combining the principle solutions in a product concept and drawing it into a sketch.

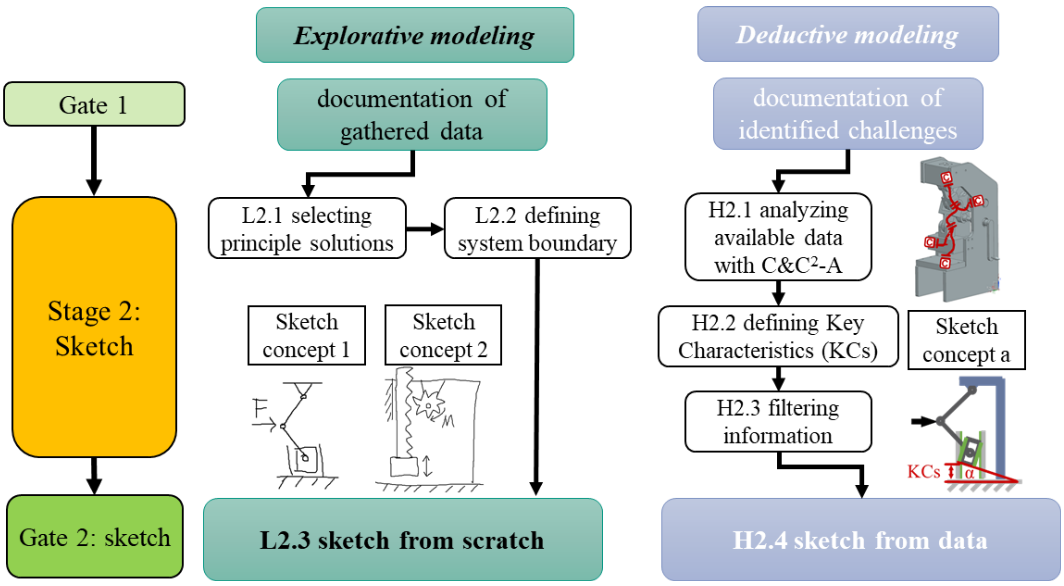


Figure 7. Overview of Stage 2 Sketch

Figure 7 on the right illustrates the process of *Deductive modeling*. In Step H2.1, the available data is analyzed using C&C^2-A, and the components that are required for the fulfillment of the functional requirements are identified by finding active WSPs and CSSs. If only 3D data is available, it is necessary to identify suitable intersections to visualize the core elements of the C&C^2-A. Taking in to account the functional requirements, the KCs are then defined in the existing system in Step H2.2. In Step H2.3, filtering information is performed to retrieve the principle solution from existing data, such as CAD. Finally, Step H2.4 involves drawing the existing concept that includes the KCs.

For the accompanying example coining machine, in *Explorative modeling*, the collected information of the principle solutions leads to different concepts for the coining mechanism. In Step L2.1, the different principle solutions are first evaluated based on company-specific criteria, and suitable solutions are selected. In Step L2.2, the spatial system boundary is limited to the area through the coining load path without the drive mechanism. The temporal system boundary is one minting cycle. In *Deductive modeling,* the function-relevant parts that are responsible for the crooked minting of the coins should be identified from the previous product generation. For this purpose, the reference product is first analyzed to identify the function-relevant parts using the C&C² approach in Step H2.1. In Step H2.2, two KCs are defined as the angle α and the height h of the coin, which can be determined between the bed surface and the piston crown surface during the minting process. These contact surfaces have a significant influence on the minting quality. In Step H2.3, The identified parts and the defined KCs are then integrated into a simplified product concept that still meets the functional requirements. This concept is then drawn in a sketch in Step H2.4.

Different activities to create a sketch can be identified between the two workflows due to the different databases. Compared to *Explorative modeling*, *Deductive modeling* facilitates a more detailed analysis of the concept due to its high information content. In *Deductive modeling*, analysis with C&C^2-A can help filter the data, while in *Explorative modeling*, many terms still need to be defined to reach Gate 2. Since the problem is assumed to be known in *Deductive modeling*, the KCs can be derived directly from the task definition and defined in the existing product concept. In *Explorative modeling*, the product concept need to be detailed to define the KCs in the next stage. To reach Gate 2, *Explorative modeling* requires sketching from limited data, while *Deductive modeling* requires extracting sketches from a larger amount of data.

3.3.3 Stage 3 Structure

Figure 8 illustrates the workflow steps in Stage 3 *Structure*. Based on the created sketch from Stage 2, the principle solutions can be decomposed into function-relevant parts and their relations in the product structure graph, which forms Gate 3. A detailed introduction to the product structure graph is described by Goetz et al. (2018).

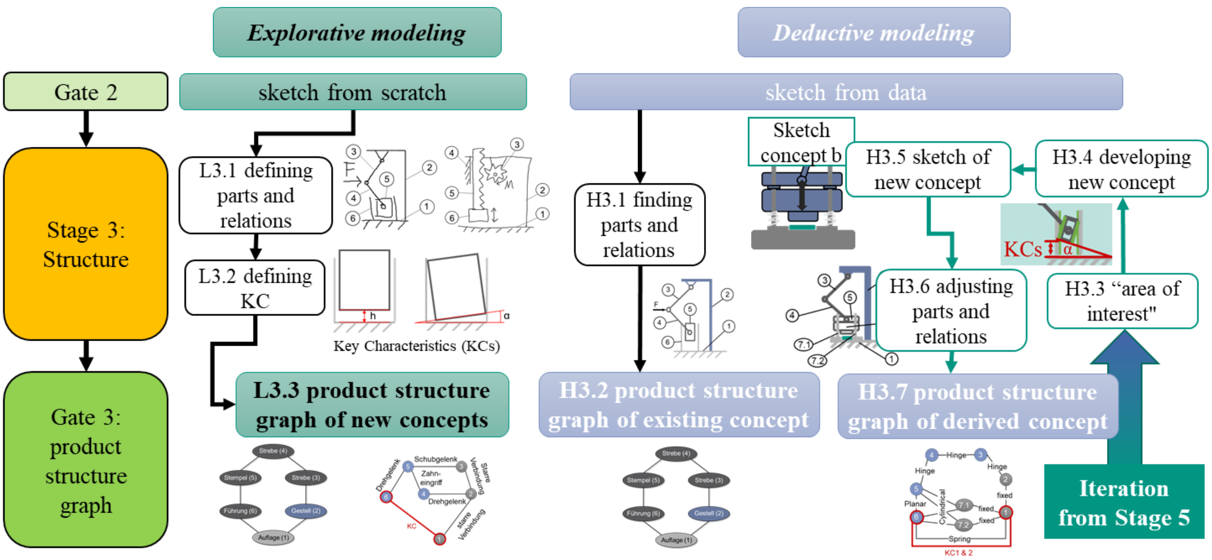


Figure 8. Overview of Stage 3 Structure

Figure 8 on the left illustrates the process of *Explorative modeling*. In Step L3.1, the sketched product concept is detailed with independent and function-relevant parts or subassemblies. This step also involves defining the geometric relations between the parts, i.e., types of contact. The types of contact are limited to a few basic interface relations, such as fixed contact, prismatic joint, or cylindrical joint between two parts (Chase et al., 1996). In Step L3.2, KCs are defined and related to the interface relations between the parts. Finally, Step L3.3 involves creating a product structure graph in which each node is a part or subassembly, and each edge is an interface relation between them.

Figure 8 on the right illustrates the process of *Deductive modeling*. In Step H3.1, the required parts and their relations already exist in the previous product generation, and they can be found from available data. Based on that, the product structure graph for the existing concept is derived in Step H3.2. At this stage an iteration takes place, as Step H3.3 is continued after the complete evaluation of the existing concept in Stage 5. The existing concept will be further developed in steps H3.3-H3.7.Step H3.3 defines the “area of interest”. This is related to the localized problem in Step H1.3. In Step H3.4, the existing concept is adapted in the identified area of interest, and a new concept is developed. The new concept aims to solve the localized problem, but its robustness has not yet been assessed. The adapted concept is then sketched in Step H3.5. In Step H3.6, the defined parts and relations are adjusted from the existing ones. This result in a new product structure graph in Step H3.7.

For the accompanying example coining machine, in *Explorative modeling*, the parts that are relevant for coin minting and their relation are first defined in Step L3.1. Two KCs are identified in L3.2 as the angle α and the height h of the coin. Then the product structure graph is created with this information. In *Deductive modeling*, the relevant parts and their relations are found from the existing data in Step H3.1. Then the product structure graph of the existing concept is created and evaluated in further stages. After the evaluation, an iteration is carried out to improve the minting quality by function separation. The improvement takes place in the “area of interest” in Step H3.3. A new concept with an extended guide is developed in Step H3.4 and sketched in Step H3.5, taking into account the refined task from Stage 1. Based on the sketch, the defined parts and their geometric relations from the previous concept can be adjusted in Step H3.6. The product structure graph of the new concept is then created in Step H3.7.

Differences between the two workflows can be identified. In *Explorative modeling*, the new concepts must be specified with parts and their geometric relations, deriving new product structure graphs. In *Deductive modeling*, the first product structure graph is derived from the existing concept, where parts and relations already exist. Then, a new concept is derived and the product structure graph is adapted in the area of interest. In this stage, *Explorative modeling* extends the information content constantly, while *Deductive modeling* first limits the information content for the existing concept and then extends it for the new concept.

3.3.4 Stage 4 Model

Figure 9 shows the steps of the workflows in Stage 4 *Model*. Based on the created sketches from Stage 2 and the product structure graphs from Stage 3, the EFRT model is built in this stage. Different states are also taken into account to derive state-dependent models. Gate 4 represents the EFRT model, including the EFRT graph and the EFRT sketch.

Figure 9 on the left illustrates the process of *Explorative modeling*. L4.1 is the state modeling of the system. Based on the sketch, different system states are investigated with the C&C^2 sequence model, which involves variable WSPs. Step L4.2 involves defining GEs and their geometric relations. The relations encompass the interaction of two parts that were defined in Stage 3. In addition, the relative location and orientation of the GEs in a part, such as parallelism or perpendicularity, must be specified in this step. In step L4.3, the system states are visualized with EFRT sketches. Step L4.4 involves deriving the geometry element graph, from which the functional tolerance chain is determined. Each node represents a GE, while the edges show the geometric relation between them. A detailed introduction to the creation of a geometry element graph is described by Goetz et al. (2018). Step L4.5 involves assigning C&C^2 elements in the geometric element graph to derive the EFRT graph.

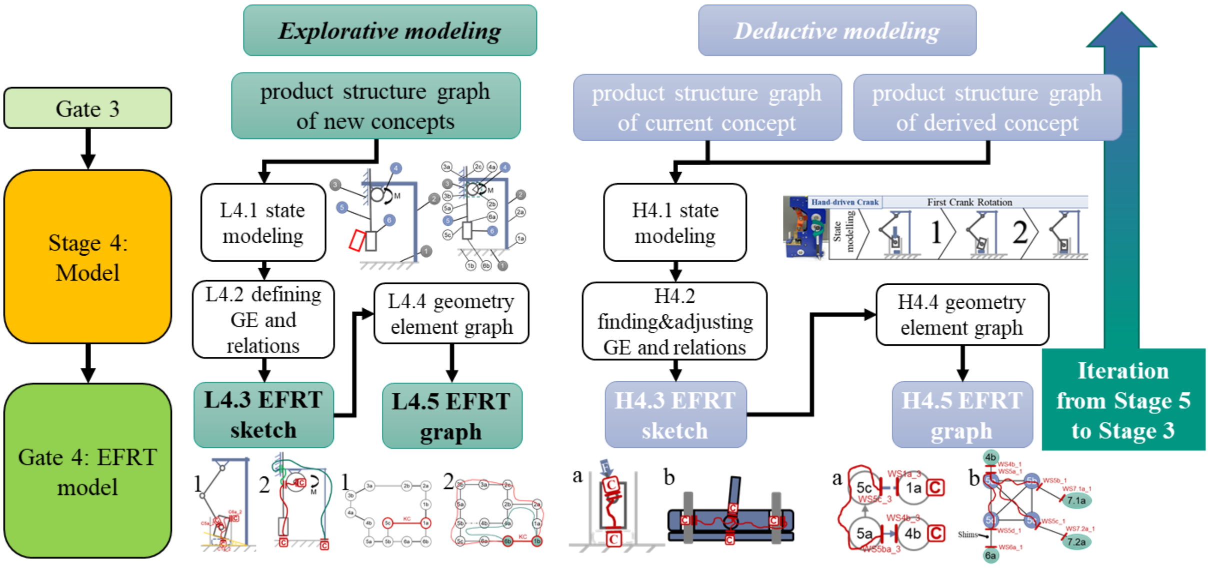


Figure 9. Overview of Stage 4 Model

Deductive modeling is shown in Figure 9 on the right. In Step H4.1, state modeling, the previously identified problems in Stages 1-3 should be given special attention. In step H4.2, GEs and their relations can be found in the existing concept, and they are adjusted for the adapted concept. The description for H4.3-H4.5 is identical to L4.3-L4.5.

For the accompanying example coining machine, in *Explorative modeling*, different system states around the minting process are modeled in Step L4.1. Relevant GEs and their geometric relations are then defined in Step L4.2, e.g., the piston skirt and crown as well as the perpendicularity between these two GEs. The minting process ist then visualized with the EFRT sketch in Step L4.3. After the derivation of the geometry element graph in Step L4.4, the EFRT graph for the coining machine is created for the further analysis. In *Deductive modeling*, the process can be repeated with the previous and adapted concept.

Although the steps in Stage 4 are similar between the two workflows, differences still exist. While the system states have to been first identified in *Explorative modeling*, knowledge about system states already exists in *Deductive modeling*. In addition, existing tolerance information can be used in *Deductive modeling* to help derive function-relevant GEs and their geometric relations. According to Goetz et al. (2021), a faster derivation of the geometry element graph is possible with available CAD data.

3.3.5 Stage 5 Decide

In Stage 5 *Decide*, the EFRT model built from Stage 4 is used for robustness evaluation. Gate 5 incorporates the design decision made from this evaluation. Since the steps up to the robustness evaluation are identical in both workflows, no distinction is made between L and H in this stage. The steps of the workflow are shown in Figure 10.

Step 5.1 identifies the critical states selected from the state modeling in Stage 4. In Step 5.2, the EFRT graph is simplified by deleting longer redundant loops for the considered KC. The method to derive an extracted functional tolerance chain of a KC is described in detail by Goetz et al. (2018). In Step 5.3 critical deviations are identified. They are assigned in the nodes and edges of the EFRT graph for analysis of their influence on KC and then visualized in the relevant parts. Step 5.4 is analysis with EFRT sketch. Due to the identified critical states, a failure mode can now be predicted with deviations in the design parameters. Using the evaluation criteria according to Li et al. (2023), a first robustness evaluation can be carried out in Step 5.5. In addition to the robustness evaluation, the design decision in Gate 5 should also consider criteria from other disciplines, e.g., economy.

For the accompanying example coining machine, Step 5.1 identifies the state “minting” as a critical state, since the lateral force from the lever leads to a crooked position of the piston. In Step 5.2, a shorter tolerance chain in the EFRT graph is selected for the KCs. In Step 5.3, geometric deviation in the parallelism of the part piston is identified as critical because it affects the WSP between the piston and the guide and thus the angle of minting. Another critical deviation can occur in the perpendicularity between the piston crown surface and the bed. The system behavior is then analyzed in Step 5.4, where the tilting moment is identified as the cause for the crooked coin and compared in the concepts. This analysis brings new insight for the concept adaption, ideas for reducing the tilting moment can be used to adapt the previous concept.

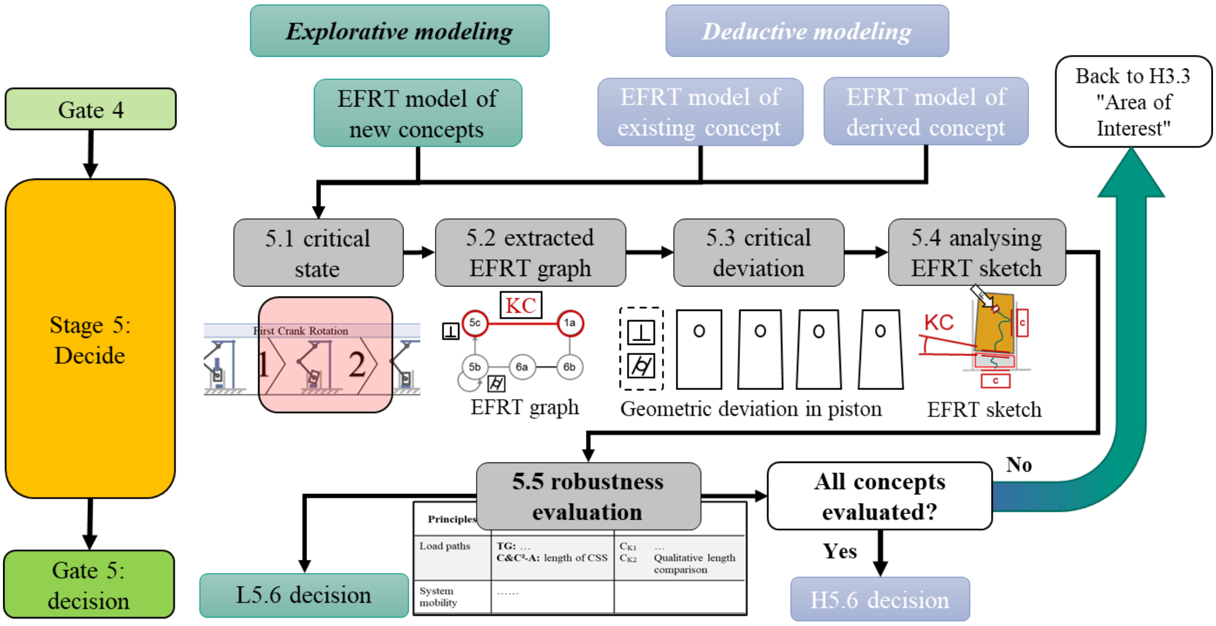


Figure 10. Overview of Stage 5 Decide

In this stage, the robustness evaluation follows identical steps in both workflows. The main difference lies in the information content. The information content in *Deductive modeling* is higher than that of *Explorative modeling*. Another difference lies in the iteration in *Deductive modeling* between Stages 3 and 5, aiming to adapt the existing concept after its initial evaluation.

4. Evaluation of the modeling method with case studies

The developed modeling method was implemented and evaluated with two different case studies described in Section 2. In both case studies, the workflows enabled the EFRT models to be built systematically and used as a basis for robustness analysis of the respective product concepts. The implementation of the modeling method with selected steps is illustrated in Figure 11 and briefly summarized below.

The first case study with bicycle clipless pedals evaluates *Explorative modeling* for a low level of derivable SDK. Two concepts were created in the project. One is based on an existing concept from a competitor in the clipless pedal market and the other is a new idea (see Figure 11). The first concept has one rotating and one fixed hook, requiring riders to tilt the tip of their foot down to insert the cleat into the mechanism. The second concept has two rotating hooks that hold the pedal cleat in place. For this brief introduction, only the first concept is presented in Figure 11 from Stage 2. In Stage 1, the first step is to define the task. The mechanism must enable the clip-in and clip-out of the pedal plate while ensuring a secure grip of the shoe on the pedal. In stage 2, sketches are created in the area of pedal axle, pedal body, and pedal plate. In stage 3, product structure graphs are derived. The gap between the cleat upper surface and the shoe is defined as a KC, as this is important for the function click-in and ergonomics. The EFRT graph and sketch are then built in Stage 4 and used for the robustness evaluation in Stage 5. In this case, a critical state can be identified when considering the clip-in function. The gap between the top of the cleat and the shoe (KC1) must be large enough to allow the hook to close unhindered. Otherwise, there may be a penetration between the hook and shoe by clip-in if there is a deviation in the hook length. This penetration results in undesired WSP and can be visualized with the EFRT sketch. It should be noted that this evaluation is based on only one critical state, and multiple states must be considered for an overall evaluation.

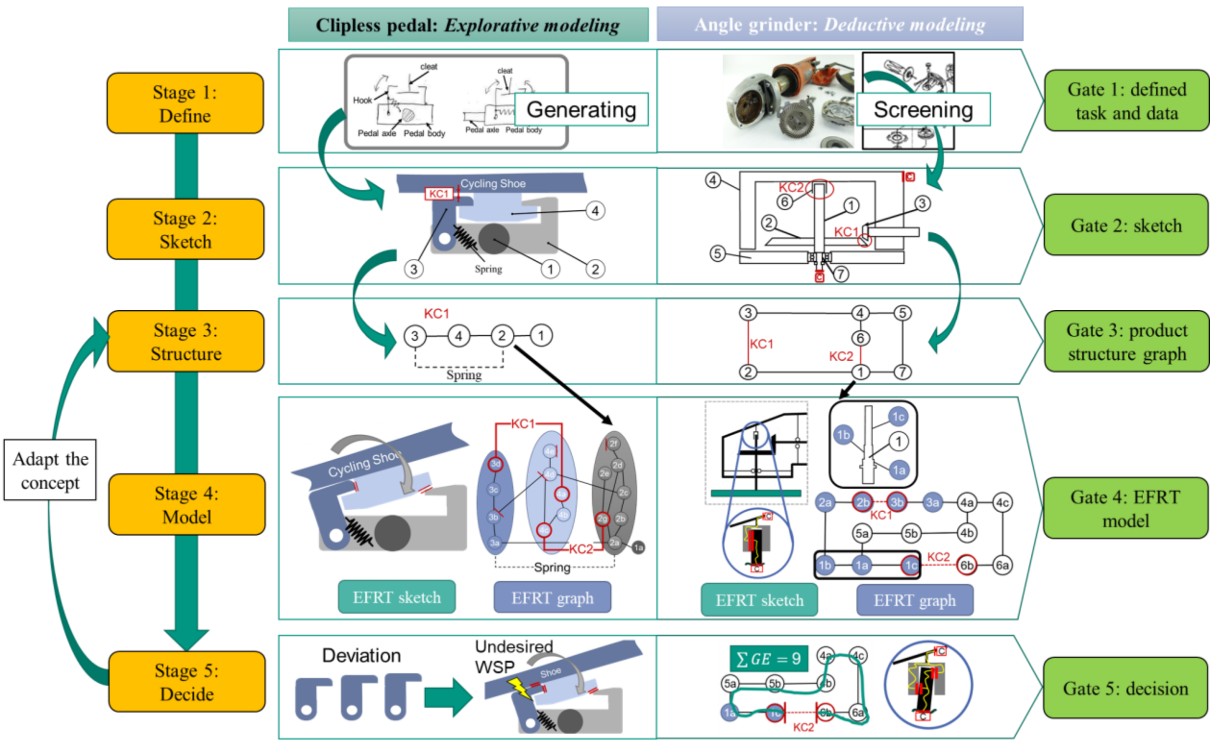


Figure 11. Selected steps for the implementation of the modeling method in the case studies

The second case study with an angle grinder evaluates *Deductive modeling* for a high level of derivable SDK. Figure 11 shows the current generation of the angle grinder, where both shafts connect the bevel gear and pinion in a perpendicular arrangement. The bearing concept for the output shaft includes a deep groove ball bearing in the bearing cap and a plain bearing in the housing. In Stage 1, the task is defined as reducing the rejection rate in the assembly. In Stage 2, the angle grinder is simplified in a sketch with the components relevant to the task. Derived from the task, two KCs are defined. KC1 is the perpendicularity between the input and output shaft as it affects the gear function. KC2 is the gap in the plain bearing in the housing, as the problem usually occurs here during assembly. Then, in Stage 3, the relationships of these components are represented in the product structure graph. In Stage 4, the EFRT model of the angle grinder is built. In Stage 5, the extracted EFRT graph for KC is derived. It becomes clear that the number of GEs for KC2 can be reduced. This insight leads to an adaptation of the concept, where the main shaft is supported only by two deep groove ball bearings in the bearing cap. It should be noted that this adaptation only addresses the assembly problem. Other functional requirements must also be considered for an overall evaluation.

Of particular interest in evaluating the modeling method are the differences between the two workflows. The differences are initially identified in the development of the modeling method with the accompanying example of a coining machine. They are summarized and illustrated in

Table 1 and are examined using the two case studies. An evaluation of the differences leads to different required actions, which are also summarized in

Table 1.

Table 1. Analysis of differences in the workflows and required actions for each difference

Identified differences in the workflows	**Required actions**
Tasks and problems	Suggestion of ideation and analysis methods
Information content in the gate	A clear description of the required information content in each gate
Definition of KC	Iterative process to define KCs
Focus of modeling	Method to limit the system boundary
Sequence of steps	A knowledge-based software to support selection in modeling steps
Modeling of system states	Considering different resolutions for modeling rough and detailed states
Modeling of deviation	Method to select different deviations for modeling
Focus of evaluation	A multi-criteria robustness evaluation method

The workflows differ in **tasks and problems**. In *Explorative modeling*, the task is to design a functional product and the problem is mostly not clearly defined, while in *Deductive modeling*, the product is already designed, and the goal is to refine it with the given problems or requirements. The case studies confirm this difference. In case study 1, the task is defined as enabling the clip-in and clip-out of the pedal plate while ensuring the shoe securely grips the pedal. A problem is first identified when a critical state is found in the state model. In case study 2, reducing the rejection rate to solve the known assembly problem refines the task. To clearly define the task and problems, ideation or analysis methods can be helpful.

The **information content in the gate** shows a different direction of information processing in both workflows. In *Explorative modeling*, the information in the gates is constantly being expanded, while in *Deductive modeling*, the information in the gates is increasingly being precise. Comparing the two workflows, the SDK changes in different directions. The higher the stage, the less difference of the SDK exists in the gates. At the end of stage 4, the SDK is very similar in both workflows. This flow of information through gates in both workflows is also confirmed by the case studies. In case study 1, information such as parts or relationships is not available, less information can be used directly. More details are added from the original hand sketch in order to build the EFRT model. In case study 2, various information can be reused, such as CAD data for finding GE and their relations. The analysis removes unnecessary information at each stage. This reveals the need for a clear description of the required information content in each gate.

The **definition of KC** also varies between the two workflows. In *Deductive modeling*, KC can be directly relocated in the given embodiment of the existing concept, considering the key function of the system. In *Explorative modeling*, the concept has to be structured first before KC can be defined. The case study partially supports this difference. In the case study 2, two KCs, the gearing and the gap in the sliding bearing, are defined in Stage 2. They are derived from the problem of a high rejection rate in assembly. In case study 1, the first KC is identified as the gap between the pedal plate and the pedal body, but these parts are not defined until step 3, so the KC is also defined later. During state modeling, another KC, the gap between the clear upper surface and the shoe, is identified as a critical parameter for the clip-in function. This means that new information may emerge during state modeling, which may lead to a new definition of KCs. An iterative process is required to define KCs in the modeling method.

The **focus of modeling** differs between the two workflows. *Explorative modeling* concentrates on the whole system, while *Deductive modeling* targets the subsystems that are modified during development. Besides that, *Deductive modeling* allows for a more comprehensive analysis of the tolerance chain, as it includes more defined elements, such as connecting elements or seals. *Explorative modeling* often omits these elements in the concepts. The case studies confirmed this difference. In case study 1, the whole click pedal systems are modeled, but only with the essential parts for function fulfillment. In case study 2, the modeling is carried out in a subsystem around the bearing and relevant connecting elements, but in more detail. This difference is addressed in *Explorative modeling* in step L2.2 “defining system boundary” and in *Deductive modeling* in step H3.3 “area of interest”. A Method to limit the system boundary for the analysis should be useful.

The two workflows have different **sequences of steps**. In *Explorative modeling*, the concepts are first generated, then analyzed and modeled. In *Deductive modeling*, the existing system is analyzed and modeled first, and then a new concept is derived. The case studies confirmed this difference, and the proposed steps in the modeling method are generally applicable to both case studies. However, some steps are optional for the evaluation, e.g., the product structure graph can be omitted if the GEs and their relations are easy to find from the existing concept. Still, the product structure graph as a gate stores intermediate information, aiming at automated information processing. It requires knowledge and experience to reduce the effort by selecting modeling steps, a knowledge-based software should support the design engineer in selecting the necessary modeling steps and their sequence.

The **modeling of system states** differs between the two workflows. In *Explorative modeling*, the system states are unknown and need to be investigated, while in *Deductive modeling*, they are given by the previous product. The case studies partially confirmed this difference. In case study 1, three states are identified: clip-in, operation, and clip-out. Further analysis shows that there are sub-states that need to be explored through modeling. In case study 2, only one state addresses the assembly problem. It is necessary to select sufficient resolution for state modeling, in order to find the right problem. Rough system states can be determined through experience. For dynamic systems, it may be necessary to model detailed sub-states.

The **modeling of deviation** differs between the two workflows. In *Explorative modeling*, the deviation modeling is based on assumptions, and the aim is to find the influence of the assumed deviations on the function fulfillment. In *Deductive modeling*, the problem is given, it must be investigated what is the relationship between the given problem and the deviations. The case study confirms this difference. In case study 2, the existing assembly problem and the tolerance information allow an early analysis with deviations, while in case study 1, the relevant deviations must be identified through the EFRT graph and sketch. Functional fulfillment is affected by more than a single deviation. Therefore, a method to select different deviations in terms of their impact and frequency of occurrence is required for sufficient modeling.

The **focus of evaluation** varies in the two workflows. In *Explorative modeling*, the entire concepts are assessed, with an emphasis on modeling the critical states in the EFRT sketch. In *Deductive modeling*, the evaluation focuses on the modified parts relative to the unchanged parts, using the EFRT graph as the main tool for comparison. The case studies confirm this difference. In case study 1, the critical states in the clip-in are identified using the EFRT sketch. In case study 2, the evaluation is based on the EFRT graph, which shows the contribution of GEs to the KC and therefore the rejection rate. The evaluation so far is based on single criteria. A multi-criteria evaluation method, e.g., the robustness evaluation matrix by Goetz et al. (2020), should be taken into account.

5. Discussion

Based on the results, the research question "*How can an EFRT model be built whilst taking different levels of derivable specific design knowledge for early robustness evaluation into account?*" can be answered as follows.

Depending on the design situation at the start of the development, the derivable SDK is often different. Therefore, different modeling activities can be derived to obtain the required SDK for robustness evaluation. The proposed modeling method for the EFRT model takes these differences into account when compared to the previously common qualitative modeling methods, such as FBS (Gero and Kannengiesser, 2014), C-K theory (Hatchuel and Weil, 2009), and modeling method with C&C^2-A (Matthiesen, 2021). It allows the sequence of modeling activities to be adapted within a stage-gate process. Investigation using two contrasting cases results in two situation-specific workflows in the modeling method for building an EFRT model for early robustness evaluation. With two different workflows, modeling is now closer to the design process in practice. Compared to the RD methods whose description is based on one use case (Goetz et al., 2018; Göhler and Howard, 2015; Mathias et al., 2011), the modeling method proposed in this paper can support the design engineer in a wider range of design situations.

Motivated by the high potential of the EFRT model for early RD and recognizing the insufficient methodical support to build it, a detailed description of the modeling steps is provided in this paper. With a special focus on the individual product concept, the modeling method developed in this paper provides a guideline for early robustness evaluation, complementing the existing experience-based RD approaches in the state of the art (Andersson, 1997; Ebro et al., 2012; Ebro and Howard, 2016). Therefore, the design engineers can now be better supported throughout the design process.

Another new insight is the possibility to model deviations in the product concept using the modeling method, which is introduced for the first time in the formalization of the EFRT model. As proposed by Hasenkamp et al. (2009), awareness of deviations is emerging as a fundamental step in robust design, but effectively dealing with such deviations is still an ongoing challenge for early RD methods in concrete design practice. With the EFRT graph and EFRT sketch, deviations can be modeled initially and their impact on functional fulfillment can be systematically analyzed in the modeling method proposed in this paper. As emphasized in section 4, it is important to know which deviations are relevant for modeling. For this purpose, the robustness ratios proposed by Mathias et al. (2011) can be considered.

The real design situation is mostly a mixed situation, therefore the two workflows developed in the modeling method don't cover all the design situations. With the evaluation of the differences between both workflows in section 4, new insights can be identified for the further development of the modeling method. For example, a supporting guide can be helpful for the selection of modeling steps within the stages, which is still missing. Special situations require further investigation, for example, *Explorative modeling* should be chosen in the case of a large amount of data with a low level of usability. Such investigation shows that the derivable SDK, not only the available data, determines the modeling activities. In many design situations, certain steps in the workflows can be skipped, e.g., when an EFRT model from the previous product is available. Another example is the adaptation of the concept in *Deductive modeling*, which is not necessary when comparing the robustness of two existing product concepts. These special situations also reveal the need for flexible adjustment of modeling activities between the two workflows in response to changing situations during modeling.

The modeling method in this paper links other appropriate methods for different design situations. For example, in Stage 2 Sketch, ideation methods can be integrated into *Explorative Modeling*, e.g., TRIZ (Altshuller, 1998), the 6-3-5 method (Petersson and Lundberg, 2018), and brainstorming (Hatcher et al., 2018). Analysis methods can be integrated in *Deductive modeling*, e.g., SPALTEN (Albers et al., 2016). This potential to link the other methods can be further investigated in the proposed stages in the modeling method.

A limitation of this research is that the development is based on the illustrative examples, which leaves open the question of applicability in real cases in the industry. It also remains unclear, whether the distinction between low and high levels of derivable SDK serves its purpose in practice, where modeling steps for the EFRT model may be influenced by additional factors.

Neither the EFRT model nor the developed workflows could be evaluated regarding any of their success factors, such as the applicability or efficacy for another method user. However, the formalization of the EFRT model and the development of the two workflows lay the foundation for investigating success factors of the modeling method in the future. Since the theoretical considerations of these design situations are literature-based, it is recommended to investigate the modeling steps and the success criteria in empirical studies within the industry.

6. Conclusion

Different design situations yield higher or lower levels of the derivable SDK at the start of development. Recognizing the lack of consideration given to different design situations in the current RD methods, this paper proposes a novel modeling method with the EFRT model, which takes the differences in SDK into account. Different modeling activities can be derived by comparing the derivable SDK and the required SDK for early robustness evaluation. Thus, the developed modeling method is integrated into a consistent stage-gate process, which facilitates the individual adaption of modeling activities. In this paper, two ways of dealing with information, i.e., information generating and information filtering, are investigated with the accompanying example of a coining machine. Considering the high and low levels of the derivable SDK, two workflows, *Explorative modeling* and *Deductive modeling*, are developed in the modeling method. Evaluating the developed modeling method with two further case studies confirms the differences between the workflows and the need for further development of the modeling method. The early robustness evaluation can now be carried out with the developed modeling method using the EFRT model, which takes different design situations into account. This provides an initial basis for adapting RD methods to design situations. The modeling method presented in this paper enables an alternation of the modeling steps and thus reduces unnecessary tasks in the modeling process, especially in scenarios with substantial differences in the level of derivable SDK.

7. References

Albers, A. and Rapp, S., 2022. Model of SGE: System Generation Engineering as Basis for Structured Planning and Management of Development, in: Krause, D., Heyden, E. (Eds.), Design Methodology for Future Products. Springer International Publishing, Cham, pp. 27–46.

Albers, A., Reiß, N., Bursac, N. and Breitschuh, J., 2016. 15 years of SPALTEN problem solving methodology in product development, in: 12th Biennial Norddesign 2016 Conference "Highlighting the Nordic Approach", NordDesign 2016. Trondheim, Norway, pp. 411–420.

Altshuller, G., 1998. 40 principles, TRIZ keys to technical innovation: TRIZ Keys to Technical Innovation, 1st ed. Technical Innovation Center, Worcester, Mass., 135 pp.

Andersson, P., 1997. On Robust Design in the Conceptual Design Phase: A Qualitative Approach. Journal of Engineering Design 8, pp. 75–89. DOI:10.1080/09544829708907953.

Andreasen, M.M., Hansen, C.T., Cash, P., 2015. Conceptual Design: Interpretations, Mindset and Models. Springer International Publishing, Cham, Switzerland.

Ballu, A., Mathieu, L. and Legoff, O., 2010. Representation of Mechanical Assemblies and Specifications by Graphs, in: Villeneuve, F., Mathieu, L. (Eds.), Geometric tolerancing of products, vol. 21. ISTE; John Wiley & Sons, London, Hoboken, NJ, pp. 87–110.

Cash, P. and Kreye, M., 2017. Uncertainty Driven Action (UDA) model: A foundation for unifying perspectives on design activity. Des. Sci. 3. DOI:10.1017/dsj.2017.28.

Chase, K.W., Gao, J., Magleby, S.P. and Sorensen, C.D., 1996. Including Geometric Feature Variations in Tolerance Analysis of Mechanical Assemblies. IIE Transactions 28, pp. 795–807. DOI:10.1080/15458830.1996.11770732.

Ebro, M., Howard, T. and Rasmussen, J., 2012. The foundation for robust design: Enabling robustness through kinematic design and design clarity. Proceedings of International Design Conference, DESIGN, Ebro, M. and Howard, T.J., 2016. Robust design principles for reducing variation in functional performance. Journal of Engineering Design 27, pp. 75–92. DOI:10.1080/09544828.2015.1103844.

Eifler, T. and Schleich, B., 2021. A robust design research landscape - review on the importance of design research for achieving product robustness. Proceedings of the Design Society 1, pp. 211–220. DOI:10.1017/pds.2021.22.

Eisenbart, B., Gericke, K. and Blessing, L. A Framework for Comparing Design Modelling Approaches across Disciplines, in: DS 68-2: Proceedings of the 18th International Conference on Engineering Design (ICED 11), Impacting Society through Engineering Design, Vol. 2: Design Theory and Research Methodology, Lyngby/Copenhagen, Denmark, 15.-19.08. 2011, pp. 344–355.

El Hani, M.A., Rivest, L. and Maranzana, R., 2012. Product Data Reuse in Product Development: A Practitioner's Perspective, in: Product Lifecycle Management. Towards Knowledge-Rich Enterprises, Berlin, Heidelberg. 2012. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 243–256.

Eppinger, S.D., Browning, T.R., 2012. Design Structure Matrix Methods and Applications. The MIT Press, 334 pp.

Gawthrop, P.J. and Bevan, G.P., 2007. Bond-graph modeling. IEEE Control Systems Magazine 27, pp. 24–45. DOI:10.1109/MCS.2007.338279.

Gero, J.S. and Kannengiesser, U., 2014. The Function-Behaviour-Structure Ontology of Design, in: Chakrabarti, A., Blessing, L.T.M. (Eds.), An Anthology of Theories and Models of Design: Philosophy, Approaches and Empirical Explorations. Springer London, London, pp. 263–283.

Goetz, S., Kirchner, P., Schleich, B. and Wartzack, S., 2021. Integrated approach enabling robust and tolerance design in product concept development. Des. Sci. 7, e14. DOI:10.1017/dsj.2021.13.

Goetz, S., Lechner, T. and Schleich, B., 2022. Computer-aided tolerance specification of preliminary designs based on variation analysis. Procedia CIRP 114, pp. 203–208. DOI:10.1016/j.procir.2022.10.028.

Goetz, S., Schleich, B. and Wartzack, S., 2018. A new approach to first tolerance evaluations in the conceptual design stage based on tolerance graphs. Procedia CIRP 75, pp. 167–172. DOI:10.1016/j.procir.2018.04.030.

Goetz, S., Schleich, B. and Wartzack, S., 2020. Integration of Robust and Tolerance Design in early stages of the product development process. Research in Engineering Design. DOI:10.1007/s00163-019-00328-2.

Göhler, S.M. and Howard, T.J., 2015. The Contradiction Index (CI): A New Metric Combining System Complexity and Robustness for Early Design Stages, in: 27th International Conference on Design Theory and Methodology, Boston, Massachusetts, USA. 2015. DOI:10.1115/DETC2015-47255.

Grauberger, P., Eisenmann, M., Windisch, E. and Matthiesen, S., 2022. Investigating the generation of specific design knowledge: experimental validation of a qualitative modelling method. Journal of Engineering Design 33, pp. 870–895. DOI:10.1080/09544828.2022.2151788.

Grauberger, P., Goetz, S., Schleich, B., Gwosch, T., Matthiesen, S. and Wartzack, S., 2020. A Conceptual Model Combination for the Unification of Design and Tolerancing in Robust Design, in: Proceedings of the 16th International Design Conference DESIGN. Dubrovnik, Croatia.

Grauberger, P., Wessels, H., Gladysz, B., Bursac, N., Matthiesen, S. and Albers, A., 2019. The contact and channel approach – 20 years of application experience in product engineering. Journal of Engineering Design 81, pp. 1–25. DOI:10.1080/09544828.2019.1699035.

Haik, Y., Shahin, T.M.M., 2011. Engineering design process, 2nd ed. Cengage Learning, Stamford, CT, 287 pp.

Hammers, C. and Schmitt, R., 2009. Governing the process chain of product development with an enhanced Quality Gate approach. CIRP Journal of Manufacturing Science and Technology 1, pp. 206–211. DOI:10.1016/j.cirpj.2008.09.005.

Hasenkamp, T., Arvidsson, M. and Gremyr, I., 2009. A review of practices for robust design methodology. Journal of Engineering Design 20, pp. 645–657. DOI:10.1080/09544820802275557.

Hatcher, G., Ion, W., Maclachlan, R., Marlow, M., Simpson, B., Wilson, N. and Wodehouse, A., 2018. Using linkography to compare creative methods for group ideation. Design Studies 58, pp. 127–152. DOI:10.1016/j.destud.2018.05.002.

Hatchuel, A. and Weil, B., 2009. C-K design theory: an advanced formulation. Res Eng Design 19, pp. 181–192. DOI:10.1007/s00163-008-0043-4.

Henderson, R.M. and Clark, K.B., 1990. Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms. Administrative Science Quarterly 35, pp. 9–30. DOI:10.2307/2393549.

Horber, D., Brand, O., Li, J., Goetz, S., Matthiesen, S. and Wartzack, S., 2023. Digitale Durchgängigkeit im Toleranzmanagement am Beispiel der ganzheitlichen Robustheitsbewertung in frühen Phasen, in: Tagungsband der 5. Summer School Toleranzmanagement 2023, Erlangen. 03.07.2023, pp. 69–76. DOI:10.25593/978-3-96147-665-7.

Horber, D., Li, J., Grauberger, P., Schleich, B., Matthiesen, S. and Wartzack, S., 2022. A model-based approach for early robustness evaluation – Combination of Contact and Channel Approach with tolerance graphs in SysML, in: Proceedings of the 33rd Symposium Design for X. DFX 2022, Hamburg. DOI:10.35199/dfx2022.18.

Hubka, V. and Eder, W.E., 1990. Design Knowledge: Theory in Support of Practice. Journal of Engineering Design 1, pp. 97–108. DOI:10.1080/09544829008901646.

Jugulum, R. and Frey, D.D., 2007. Toward a taxonomy of concept designs for improved robustness. Journal of Engineering Design 18, pp. 139–156. DOI:10.1080/09544820600731496.

Li, J., Horber, D., Keller, C., Grauberger, P., Goetz, S., Wartzack, S. and Matthiesen, S., 2023. Utilizing the Embodiment Function Relation and Tolerance Model for Robust Concept Design. Proceedings of the Design Society 2023, 3, pp. 3771–3780. DOI:10.1017/pds.2023.378.

Mathias, J., Kloberdanz, H., Eifler, T., Engelhardt, R., Wiebel, M., Birkhofer, H. and Bohn, A., 2011. Selection Of Physical Effects Based On Disturbances And Robustness Ratios In The

Early Phases Of Robust Design, in: Proceedings of the 18th International Conference on Engineering Design. ICED'11, Lyngby/Copenhagen, Denmark, pp. 324–335.
Matthiesen, S., 2021. Gestaltung – Prozess und Methoden, in: Bender, B., Gericke, K. (Eds.), Pahl/Beitz Konstruktionslehre: Methoden und Anwendung erfolgreicher Produktentwicklung, 9th ed. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 397–465.
Matthiesen, S., Grauberger, P. and Schrempp, L., 2019. Extended Sequence Modelling in Design Engineering – Gaining and Documenting Knowledge about Embodiment Function Relations with the C&C²-Approach, in: International Conference on Engineering Design. Amsterdam, The Netherlands.
Matthiesen, S. and Ruckpaul, A., 2012. New Insights on the Contact&Channel-Approach - Modelling of Systems with Several Logical States, in: International Design Conference: DESIGN 2012. Dubrovnik, Croatia, pp. 1019–1028.
Niu, X., Wang, M. and Qin, S., 2022. Product design lifecycle information model (PDLIM). Int J Adv Manuf Technol 118, pp. 2311–2337. DOI:10.1007/s00170-021-07945-z.
Pahl, G., Beitz, W., Feldhusen, J., Grote, K.-H., 2007. Engineering design: A systematic approach, 3rd ed. Springer, London, 617 pp.
Petersson, A.M. and Lundberg, J., 2018. Developing an ideation method to be used in cross-functional inter-organizational teams by means of action design research. Res Eng Design 29, pp. 433–457. DOI:10.1007/s00163-018-0283-x.
Phadke, M.S., 1989. Quality Engineering using Design of Experiments, in: Dehnad, K. (Ed.), Quality Control, Robust Design, and the Taguchi Method. Springer US, Boston, MA, pp. 31–50.
Ponn, J. and Lindemann, U., 2005. Characterization of design situations and processes and a process module set for product development, in: DS 35: Proceedings ICED 05, the 15th International Conference on Engineering Design, Melbourne, Australia. 15.-18.08. 2005.
Suh, N.P., 1998. Axiomatic Design Theory for Systems. Research in Engineering Design 10, pp. 189–209. DOI:10.1007/s001639870001.
Taguchi, G., Chowdhury, S., Wu, Y., Taguchi, S., Yano, H., 2005. Taguchi's quality engineering handbook. John Wiley & Sons, Hoboken, N.J., Livonia, Mich., 1662 pp.
Thornton, A., 2004. Variation risk management: Focusing quality improvements in product development and production. Wiley, Hoboken, 320 pp.
Ullman, D.G., 2010. The mechanical design process, 4th ed. McGraw-Hill Higher Education, New York, 433 pp.
Weber, C., 2014. Modelling Products and Product Development Based on Characteristics and Properties, in: Chakrabarti, A., Blessing, L.T.M. (Eds.), An Anthology of Theories and Models of Design: Philosophy, Approaches and Empirical Explorations. Springer London, London, pp. 327–352.
Wilmsen, M., Dühr, K. and Albers, A., 2019. “A context-model for adapting design processes and methods”. Procedia CIRP 84, pp. 428–433. DOI:10.1016/j.procir.2019.04.243.
Zwicky, F., 1967. The Morphological Approach to Discovery, Invention, Research and Construction, in: Zwicky, F., Wilson, A.G. (Eds.), New Methods of Thought and Procedure: Contributions to the Symposium on Methodologies. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 273–297.

KIT Scientific Working Papers
ISSN 2194-1629