Complexity Costs Evaluation in Product Families by Incorporating Change Propagation

Wolfgang Bauer
Institute of Product Development
Technische Universität München
München, Germany
bauer@pe.mw.tum.de

Patrick Bosch, Nepomuk Chucholowski, Fatos Elezi, Sebastian Maisenbacher, Udo Lindemann, Maik Maurer
Institute of Product Development
Technische Universität München
München, Germany

Abstract—Platform-based product families have become an important strategy in many industries as a wide range of products can be offered to the customers while achieving economies of scale in design and manufacturing. During the life cycle of a product family, the amount of derived product variants increase due to numerous internal and external driven reasons. This leads to additional complexity within the product family as variety and the dynamics raise. This paper presents an approach to evaluate the additional complexity costs, originating from changes within the product family. The approach combines change-propagation methods with cost calculation methods. The approach differentiates between costs based on changes of existing variants and costs based on changes leading to substituting or additional variants. Variety-inducing change drivers are assigned to affected product components and functions. All affected components, especially the indirectly changed-ones, are identified by domain-spanning change propagation. This incorporates geometrical and functional dependencies between components. Running as well as one-time expenses are determined by a process-based costing system taking into account the required additional processes and their duration to handle the added complexity. The approach is implemented into a software tool, using data of an industrial product family. An industrial case study is conducted for evaluation of the approach and the tool. Exemplary changes showed that the amount of cost caused by indirect changed components represents about half of the total complexity costs.

Keywords— complexity costs; engineering changes; change propagation; product family;

I. INTRODUCTION

Due to globalization and competitive advantages, companies are forced to offer their products in many different markets. In these global markets, various local legislations, standards, and approvals, as well as diverse customer needs have to be satisfied by different product functionalities. This requires high external variety and results in multifacteted product families. Meanwhile, the units produced decrease and the internal complexity increases. Accordingly, the development and manufacturing costs rise due to product variety. To offer external variety resulting from product differentiation while reducing the internal complexity and costs the application of product platforms are proposed [1-3].

Product platforms realize economies of scale [2] by using synergies like commonalities of product elements. The product platform builds a constant core of elements, which are common for all variants [4], and should be stable and unchanged during the product life cycle [5]. The product differentiation is achieved by flexible modules attached to the platform. The planning and development of such platform architectures requires a high investment to define and design both, the robust platform and the flexible elements [6]. Examples for platform-based products include aircrafts, cars or cell phones.

Moreover, the companies’ context is constantly and rapidly changing due to changing markets, competitors and evolving technologies [7]. These dynamic changes occur constantly during the product life cycle. Especially during the market cycle, new variants are requested by sales and distribution as a reaction to changes in market demands, customer preferences or changes in the competitive context. Also new technologies, developed internally or externally, can also lead to new variants in the market phase.

When following a platform strategy – the sharing of core elements among different products in the product family [8] – the necessary changes to derive new product variants has to be examined carefully. On the one hand, the components or functions to be changed have to be identified, followed by an change impact analysis to determine if further changes emerge [9]. It has to be ensured that common elements are not going to be changed as this effects all product family members and is very costly and time-consuming [5]. On the other hand, the benefit of the new variant must be verified in terms of cost-benefit ratio. The benefit is usually determined by expected sales volumes and the product variant price. The costs of a new variant are hard to calculate as not only change or switching costs for direct changes have to be considered, but also cost effects over the whole product family. These indirect costs can comprehend all occurring costs over the whole life cycle [10].

As new variants cause an increase of complexity by a higher variety and dynamics [11], the costs for new variants can be interpreted as complexity costs. Olbrich and Battenfeld [12] define complexity costs as the additional costs with regard to the current complexity which emerge due to a raised complexity. This definition aptly describes the costs caused by the mentioned product family variety-influencing changes.

To evaluate these complexity costs, three research questions (RQs) were derived: (1) which parameters and which methods are suitable for the determination of product-related
change impacts? (2) Which costs types have be considered for calculation of complexity costs and which methods exist? (3) How has a support for the evaluation of complexity costs within product families to look like?

The approach resulting from RQ 3 shows how to determine these complexity costs for changes in a product family. The approach is grounded on the combination of two research fields: by the application of change-propagation methods [9, 13], all affected components are identified by a systematic navigation through the networks of geometric and functional component dependencies (RQ 1). This knowledge is then used as an input for an activity-based costing system (ABC) [14-16] to calculate the direct and indirect costs for implementing and verification of the changes. For the identification of complexity cost it is distinguished between running costs and one-time costs, dependent if an existing variant is modified, a variant is substituted by variant or a new variant is introduced (RQ 2).

The answers to the three research question also provide the structure for the paper. After introducing relevant definitions in the next section, a literature review of existing methods for change propagation in technical systems and complexity cost evaluation is given (RQ 1 and 2). Based on this, the developed approach as described above is presented in section V. The approach is implemented in software tool and is demonstrated in a case study from the home appliance industry. After a discussion, the paper will be conclude with an outlook.

II. THEORETICAL BACKGROUND

This section provides theoretical background and definitions regarding platform-based product families, engineering changes and complexity costs.

A. Platform-based product families

A platform is defined as a set of common assemblies, modules and parts, which form a mutual basis [4]. Robertson and Ulrich expand the component view by three further categories: processes, knowledge, people and relationship [17]. The objective of incorporating a platform is to achieve synergies by using same elements (e.g. components, functions, technologies) for several products. Based on product platforms, product families are derived. Product families group related products which can be derived from a product platform to satisfy a variety of markets niches [17]. Product families can be further sub-divided in basic devices which serve as instance between the product family and the product variants. The basic devices are distinguished from each other by the variation of a relevant specification of characteristics, for example performance- or dimension-related [18]. The single representatives of a product family respectively basic unit are called product variants. Product variants are referred to as products with the same purpose which are distinguished in at least one characteristic [19, 20].

B. Engineering changes in technical systems

In the field of engineering, various definition for technical changes are given. Wright defines an engineering change (EC) as a modification to a component of a product, after the product has entered production [21]. This definition is extended by Jarratt et al. [9] by alterations also made to drawings or software that have already been released during the product design process. The change can be of any size or type and can involve any number of people and take any length of time [9]. Köhler [22] provides the most comprehensive definition of an EC. He describes an EC as “all additional definitions of new (instead of present) states of already released results within an interconnected technical development and manufacturing process, following the objective of ensuring or improving the ability of the product to be successfully sold and increase revenue.” So, an EC serves not as an end in itself but should create additional value in terms of ensuring / improving the competitiveness or raise of sales revenue. Similar to Jarratt et al. [9], Köhler states that ECs comprise changes of the technical documentation respectively data basis but also of products and processes [22]. Moreover, ECs can occur at any time during the product life cycle [22].

In the context of this paper, the definition of Köhler [22] is used for ECs as it provides the widest range of affected entities, relates to economical values as well as caused process changes.

C. Complexity costs as result of change costs

The concept of change costs refers to the monetary impact of EC [23] and comprehends all costs along the value chain as well as the required consumption of competition-relevant resources, especially the necessary time [10]. EC provoke both, direct and indirect costs. Direct costs can be assigned source-specific to the EC, while the indirect costs can only be assigned to all ECs. This comprehensive process perspective allows the identification of the whole range of change costs and serves as support for the identification and allocation of the costs caused by an EC [23].

As stated in the introduction, complexity costs are defined as the additional costs with regard to the current complexity which emerge due to a raised complexity [12]. The raised complexity originates from a higher variety by due to new product variants and the subsequent dynamics within the product families. Complexity costs are therefore related to the direct and indirect change costs caused by the required EC to implement the product variant. Complexity costs can be categorized in direct complexity costs and opportunity costs [24] which comprise exemplary cost types shown in Fig. 1.
Based on the relationship between variety-induced changes and the complexity, complexity costs can be considered as change costs across the value chain. In the context of this paper, complexity costs are defined as followed: complexity costs emerge as related to a zero-variant scenario and comprise the direct and indirect costs for the development and maintenance of product variants. These costs are composed of direct and indirect variety-driven activities of all organizational units along the value chain.

III. METHODS FOR THE DETERMINATION OF CHANGE IMPACTS

In order to determine the variety-induced complexity costs, all components affected by the change must be identified (RQ1). Hereafter, the most relevant methods to do so are given.

The Change Prediction Method (CPM) [13] uses a product model which represents change dependencies between components in a Design Structure Matrix (DSM). Based on this matrix, two numerical DSMs for showing the likelihood and impact of change propagation are established. The combined risk of change propagation is calculated by multiplying the direct and indirect risk. Indirect changes are considered via change propagation trees.

The Change Propagation Analysis (CPA) [25] serves the analysis of change requests to visualize the changes and the affected components. Based on the Change Propagation Index (CPI), the components are classified as absorbers, carriers, and multipliers. The CPA serves the analysis after changes are executed but the CPI and the classification can also be used for determining the change impact upfront.

CPM/PDD (Characteristics-Property Modeling/Property-Driven Development) [26] is an approach to illustrate and compare the impact of changes. The change impacts, which are linked by product characteristics/properties to the product model, are visualized by a matrix representation. Another method, combining the CPM/PDD-approach and FMEA, is the Change Impact and Risk Analysis (CIRA) [27]. Possible solutions for changes are analyzed and assessed regarding risk and impact. The risk evaluation uses the Change Classification Number (CCN). The change impact is determined by analyzing the underlying product structure.

The Change Modes and Effects Analysis CMEA [28, 29] assesses the current flexibility of products towards future changes by combining three figures to the Change Potential Number (CPN): design flexibility (F) comprehends cost of a change of a current product and is determined by the change-to-Function Ratio (amount of changed components at the ratio of all product functions); occurrence (O) delineates the likelihood of a change occurring and the Readiness (R) the easiness of manufacturing the change.

In the context of the paper, the change propagation should apply the interactions between component as well as components and functions, as many of the changes during the life cycle address functions [30]. Therefore, CPM/PDD and CIRA are not suitable due to their property-indicated view but provide a domain-spanning change propagation. For determining the CPN in the CMEA method, the change impacts have to be known already. CPA and CPM uses the dependencies between components and consider indirect change propagations but do not provide a domain-spanning analysis, e.g. for the integration of the functional domain. Expanding CPA and CPM by using a Multiple-Domain Matrix [31], allows the consideration of various domains for determining the change impacts, as proposed in [32]. Most of the presented methods use risk-based metrics such as likelihood or impact. Their significance can vary due to both, the subjectivity and the fact that different changes can propagate in a different way via the same interface between the same two components.

IV. METHODS FOR COMPLEXITY COST CALCULATION

To answer the second research question, methods suitable to determine the cost types in section II. are presented here.

Activity-based Costing (ABC) is a common method to trace costs to the activities which caused the occurring costs [15, 33]. It can also be used to determine project cost and cost savings. In ABC-systems, “an activity is an event, task or unit of work with a specified purpose, for example, designing products, setting up machines, operating machines and distributing products [15].” The three-step approach proposed by [33] translates physical parameters to specific costs. The costs are categorized into four categories: part- and material related process related, operational, and financial. This approach can be found similarly in [14] and [16] with a stronger focus on modularity than on activities. For example, Park & Simpson [16, 34] portray ABC as a production cost estimation framework for four objectives: communality decisions, platform leveraging decisions, process commonality decisions and delayed production decisions. They show that ABC means that the costs of resources are linked to activities, then the activity costs are linked to components and products.

Time-Driven Activity-Based Costing (TD-ABC) [35], a further development of ABC, is based on the estimation of two parameters: cost per time unit of capacity (cost per delivered capacity / practical capacity of the resources) and unit times of activities. The unit times of activities describes the time required to fulfill a task and is estimated by observations or interviews. The cost driver rate is calculated by multiplying the two described variables. The cost driver rate multiplied by the consumed cost rates leads to the final object costs.

Variety-Driven Activity-based Costing (VD-ABC) [36] distributes the indirect costs to the product variety dependent on the according effort. The approach represents a model-based evaluation of variety-driven complexity by considering the resulting costs and capacity influences from variety. The capacity consumption is determined by a zero-variant scenario and the variety-induced scenarios result from the additional consumptions. VD-ABC consist of three capacity equations: the “process time equation” assigns the required time to each process, adjusted to the degree of the handled product complexity, i.e. variety. The “inventory equations” assigns individual inventory levels to variants based on the inventory
strategy and product variety. The third equation “floor space” assigns the required space for each variants’ processes.

The complexity costs, driven by additional complexity due to new variants, can only be determined by a reference complexity, the so called zero-variant scenario. Neither ABC nor TD-ABC consider such a scenario, whereas VD-ABC determine such a scenario and adds up the complexity costs by the additional required capacity.

V. APPROACH FOR DETERMINING VARIETY-INDUCED CHANGES AND ACCORDING COMPLEXITY COSTS

The approach, developed based on the previous background insights, applies domain-spanning change propagation for identifying all affected components, links them to the required additional process and uses VD-ABC for determining the required processes time and consequently their costs. The overall approach consists of four consecutive steps, see Fig. 2. The steps will be described in more detailed in the following sub-sections.

Fig. 2. Approach for determining variety-induced complexity costs

A. Step 1: Identification of change drivers

For the monetary evaluation of complexity costs caused by ECs, the most common changes leading to new variants during the life cycle, here defined as variant-inducing change drivers (VICD), must be identified. Interviews with experts from sales and distribution as well as historical variant data sets are valuable sources. In the first step for evaluating the impact of a planned variants on complexity costs, one or more of the predefined VICDs have to be chosen. It is distinguished between the types of variant, as this has significant impact on the complexity costs. Three different type of product variants are proposed, based on the specific case:

- Existing variant: such a variant already exists and is updated by exchanging existing components. As no new components or features have to be designed and manufactured, such a variant only cause additional running complexity costs.
- Substituting variant: such a variant replaces an existing variant by incorporating new components. It produces one-time complexity costs.
- Additional variant: such a variant supplements the current product family. It cause one-time and running complexity costs.

B. Step 2: Direct Link to Product Architecture

Once the VICDs and a list of generic components and functions which represents the platform-based product family, a Domain-Mapping Matrix (DMM) [31] is applied to map the domains. As this link represent the transition of the VICDs to the product for the later change impact analysis, this step is crucial for the determination of the complexity costs. Therefore, the acquisition of the relations between the VICDs and the generic components and functions must be supported by experienced engineers. Only directly affected components and functions must be linked as the indirectly affected ones will be identified in the next step. The set dependencies can differ for one VICD, depending on its exact specification. For example, a VICDs “change of color” can be directly influence different components of a car, depending if the VICD refers to the outer or inner color. In such a case, the corresponding VICD must be further specified and the list of VICDs is extended (iteration indicated by the dashed line in Fig. 2).

C. Step 3: Domain-Spanning Impact Analysis

As stated in section III, an expansion of existing change propagation methods such as CPA and CPM by Multiple-Domain-Matrices [32] are applied here. The objective is to identify all components that are indirectly affected by the considered change. Therefore, geometrical as well as functional dependencies between components are incorporated in this analysis step (see Fig. 3). First, the change propagation is executed by a neighborhood analysis of the directly affected components with regard to their geometrical interfaces. In case of an indirect change, the change path is followed until no other component is changed. This step is iteratively run through until no more indirectly changed components can be identified. The link between components and functions and vice versa allows also the domain-spanning identification of further affected components due to functional relationships. In can be initiated by both, the affected components form the first iteration or by the direct link of VICDs to function. This step is also run iteratively until all affected components are identified. In contrary to the presented methods in section III, no assumed impact or likelihood for the change propagation is used, only direct geometrical and functional relationships. In this way, the user is supported in identifying all affected components by the possible propagation paths in the networks. Depending on the specific VICD, it can be decided which change propagates via a geometrical or functional dependency and which stops there.

Fig. 3. Procedure for domain-spanning change propagation
D. Step 4: Determination of Complexity Costs

In the last step, the complexity costs are determined. The complexity costs are calculated by multiplying the sums the running and one-time additional costs with the hourly rate of the direct and indirect affected departments. The duration of each process required to implement the change of all identified components must be considered for each type of VICD. In interviews with experts from every department, the process time is acquired by learnings from former change projects or change requests. Also the effort for the maintenance of the new variant must is acquired this way. Running cost as well as one-time costs must be identified for all relevant organizational units, dependent on the VICD. With this information, any change leading to a new variant during the product life cycle can be evaluated regarding its complexity costs, depending on the type of variant, the VICD and the direct and indirect affected components. Cannibalization effects between the product variants are not considered yet here; interesting insights are given in [37].

VI. TOOL-BASED CASE STUDY AND EVALUATION

The proposed approach was implemented in a software tool and evaluated by a case study. The case study was conducted at an European home appliances company.

During step 1, 67 representative VICDs were collected by interviews with engineers and implemented into a software. These VICDs have been mapped to 108 components which have been again mapped to 13 main and 56 sub functions. The change propagation layer is implemented four times, so four iterations can be executed per change propagation dependency type. The geometrical and functional network are visualized in the tool, allowing a neighborhood analysis, and affected components can be marked. Common or standardized components which are not allowed to change due to vast impacts on all product family members are color-coded. In total, 136 processes, consisting of 593 activities, of 7 main departments have been identified and evaluated according their duration for additional variants per VICD. Based on the effort of previous variants and change projects, the effort of future variants is estimated by the experts. The point of reference is represented by a zero-variant scenario. For a plausibility check, the one-time and running costs are summed up per department and compared to actual operational effort. The result of running through the process, as shown in Fig. 2, comprises all components to be changed, affected departments, required processes and activities per department and their duration [h], one-time and running effort [h] and costs [€], as well as the total cost [€] for the change. The running efforts and costs are related to one year and can also be calculated for any longer time period.

The approach and the tool were evaluated regarding their support, applicability and success by industrial examples. First, different VICDs typical for this industry were tested. The

<table>
<thead>
<tr>
<th>Component</th>
<th>Type of Variant</th>
<th>Direct Changed Costs (per year)</th>
<th>Indirect Changed Costs (per year)</th>
<th>Running Costs (per year)</th>
<th>Total Complexity Costs (TCC)</th>
<th>Costs of Ind. Changed Comp. (% of TCC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example 1</td>
<td>Change of a component's geometry and print</td>
<td>2</td>
<td>1</td>
<td>7.0</td>
<td>8.0</td>
<td>3.6</td>
</tr>
<tr>
<td>Example 2</td>
<td>Change of key performance attribute</td>
<td>1</td>
<td>0</td>
<td>0.0</td>
<td>7.1</td>
<td>8.1</td>
</tr>
<tr>
<td>Example 3</td>
<td>Change of a component's geometry and print</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7.0</td>
<td></td>
</tr>
</tbody>
</table>

Example 4 | Light and Housing Component | 2 | 5 | 4.4 | 5.4 | 2.6 | 48.1% |

\[\text{Example Type of Variant} \quad \text{Direct Changed Costs (per year)} \quad \text{Indirect Changed Costs (per year)} \quad \text{Running Costs (per year)} \quad \text{Total Complexity Costs (TCC)} \quad \text{Costs of Ind. Changed Comp. (% of TCC)}

\[\text{Change of one component} \quad \text{existing} \quad 1 \quad 0 \quad 1.0 \quad 0.0 \quad 1.0 \quad \text{none} \]
\[\text{Change of key performance attribute} \quad \text{additional} \quad 1 \quad 0 \quad 1.0 \quad 7.1 \quad 8.1 \quad |

\[\text{Change of a component's geometry and print} \quad \text{additional} \quad 2 \quad 1 \quad 1 \quad 7.0 \quad 8.0 \quad 3.6 | 45.3% |

\[\text{Light and Housing Component} \quad \text{additional} \quad 2 \quad 5 \quad 1 \quad 4.4 \quad 5.4 \quad 2.6 \quad 48.1% \]

All monetary values in Table I are normalized to the running costs of the corresponding example.

The third example comprises a change of print and geometry of one component, resulting in two further changed components because of geometrical relationships. In the last example, change in light and its housing, a total of five components were affected in an indirect way: three for geometrical dependencies on the 1st level and one on the 2nd level; one for functional dependencies. In this example, complexity costs caused by indirect changed components have a stake of around 50% in the total complexity costs, showing the importance of change propagation integration into cost calculation.

For the second part of the evaluation, two responsible experts for new variant orders (since 5 and 12 years) and future users of the tool were given four exemplary VICDs to determine the complexity costs and questionnaires for evaluation. After two examples, they got used to the tool and its user surface and could process the other two examples notably faster. The resulting complexity costs for each case were rated reasonable and serve as a meaningful benchmark value. Also the breakdown of the total costs into the running and one-time costs per unit and their processes the identification of the main cost drivers. The implemented networks give good assistance in identifying the affected components. Especially the functional change propagation was seen as extremely helpful as this change propagation is not obvious and intuitive. Here, the saving of already identified change paths per VICD was missing as a feature. By the navigation through the networks by the users themselves, a high transparency and consequently a high traceability and reliability is achieved, especially compared to the previous applied cost calculation tools without change propagation.

VII. DISCUSSION

The proposed approach combines change propagation methods with VD-ABC cost calculations to determine the complexity costs for new variants during the product family.
life cycle. The approach was implemented in a software tool and was evaluated by an industrial case study. Especially, the domain-spanning change propagation, expanding the geometrical dependencies between components by functional ones, enriches the reliability of the results regarding affected components. In contrary to risk-based change impact assessments, which are rather subjective, the change impact can be identified by geometrical and functional dependencies. Also the clear distinction between the one-time and running efforts and costs, in combination with the three different types of variants and very detailed required activates per department, results in a high reliability of the resulting complexity costs. This was not only confirmed by the case studies, but also in several discussions with managers from controlling. These complexity costs can be used as an objective input for cost-benefit analysis and support the decision making regarding new variants during the product family life cycle. Moreover, the data input – VICDs, architectural networks, processes, their duration and the hourly rate – are flexible and allow different scenarios, besides future variant developments, to be evaluated by their impact on complexity costs. The effects of process changes or changes in organizational structures as well as in different production sites can be assessed. The authors assume that also break-even analysis for single variants and product families can be calculated as the running costs can be calculated for any given time period. Moreover, comparisons of different product family respectively product platform concepts can be evaluated regarding their flexibility (in terms of the ability for easy changes) and their monetary benefit. These assumptions have to be proven in further studies.

On the other hand, the approach is quite knowledge intense. All product architecture related dependencies have to be acquired in workshops. Moreover, the identification of all processes and activities in all departments takes a lot of time as well as the commitment of the departments’ employees. Consequently, the maintenance of the information is a crucial issue for industrial implementation. As the user selects the affected components by himself, the results can be manipulated, depending on the desired outcome. So the resulting complexity costs have to be double-checked before using it for decision-making but this is rather easy because of the possible breakdown of all costs.

VIII. FUTURE WORK

On the one hand, the approach must be evaluated by further experts at the industry partner and the software tool must prove its applicability in the daily management of the examined product family. To maximize its value, the product architecture of other product families should be included. Then, commonality as well as cannibalization effects across the product families must be considered as this will raise the complexity of the scope. On the other hand, the approach should be applied to other products as well to demonstrate if it works out in different environments and on different levels of product and product family complexity. Another aspect is the level of detail of the interface specifications: the geometrical and functional dependencies can be further specified, e.g. by interface characteristics and their relationship. In combination with already multiple times analyzed VIDCs and their change paths, this could even allow to automatize the change propagation process. But the trade-off between user integration and the consequent traceability and reliability and a fully automatized complexity cost calculation should be determined by further studies.

REFERENCES


