INTERNATIONAL DESIGN CONFERENCE - DESIGN 2010
Dubrovnik - Croatia, May 17 - 20, 2010.

DESIGN 2010

AVOIDANCE OF UNNECESSARY DESIGN ITERATIONS BY MONITORING THE PRODUCT´S DEGREE OF MATURITY

H. Krehmer, H. Meerkamm and S. Wartzack

Keywords: design iterations, product´s degree of maturity, properties, characteristics

1. Introduction

Increasing customers´ requirements on technical products and their functionality are mainly responsible for those products getting more and more complex. Amongst this, the market demands an increasing degree of individualisation and diversification, which results in increased use of electrical, software-based and electronic components. This individualisation and diversification means a rising complexity, which can only be handled by splitting up development among specialised workgroups. This shared development takes place spread out across different departments or even companies. The higher product complexity results in an increased complexity of the entire product development process and causes a higher necessity for communication between the disciplines. These challenges result in an increased likelihood of unneccessary design iterations. This is contradictory to the decreasing duration of technical products beeing competitive in the market and conflicts with innovation cycles getting shorter and shorter. This demonstrates the demand for a framework that enables developers to avoid unnecessary and time-consuming detours by assessing the effects on the overall product of each decision they made. If developers got an insight in all consequences of their actions, there wolud result the opportunity to prevent avoidable design iterations. This opportunity promotes the purposeful increase of the product´s degree of maturity, which finally means a straight course of the development process and thus helps to save cost and money.
Hence, aim of this contribution is to identify key factors for the avoidance of unnecessary iterations and to present approaches that were developed for coping with these challenges. Furthermore, a framework for the avoidance of unnecessary iterations based on a continuous monitoring the product's degree of maturity is introduced. For this purpose section 2 dwells on design iterations in general, section 3 identifies some key factors for the successful avoidance of design iterations and introduces promising approaches to cope with these requirements. Finally, sections 4.1 and 4.2 depict fundamentals and preparatory work for the framework for the avoidance of unnecessary iterations, which is introduced in section 4.3 by means of a simple example.

2. Design iterations in product development

The product development process is not foreseeable in detail and develops iteratively in its progression. Due to the fact that the solution for complex products cannot be found in one step, design iterations are an integral part of product development. Increasing requirements as well as growing applications of mechatronical components result in higher product complexity and thus in steadily increasing challenges in product development. As mentioned above, the complexity of products and product development processes can be handled only by splitted development work taking place

separated in different domains. Due to the network of dependencies between individual work steps the increased complexity of modern technical systems also results in an increased complexity of the product development process: The high degree of division of labour increases the demand for a sufficient communication of information. In case of lacking communication unnecessary design iterations and thus jumping back to previous sections of the development process can get necessary.
Design iterations can be a result of unknown boundary conditions that are becoming concrete in the course of the development because a data basis completion is reached only in the progression of the process. Furthermore iterations are induced due to faulty decisions made on basis of unclear or uncertain assumptions or by optimisation potentials being recognised in late process steps. Further triggers for iterations can be for example functional improvements, elimination of faults, changes in the customers´ requirements, changes of the information basis, changed demands of the market, unclear requirements in the beginning of the process or late completion of the data basis [Wynn 2007], [Krehmer 2008a]. This list of possible triggers demonstrates the risk of unnecessary design iterations during the development of complex technical systems: Jumping back to previous sections of the development process can take place, which means that earlier process steps are to be run through again: Results of earlier steps have to be checked to assure the fulfilment of the new requirements. If results of some partial steps diverge from the modified requirements, the product development process has to start again from this point. In worst case, design iterations can result in a jump back to the start of the product development process. So, interdisciplinary product development gets considerably more and more difficult by iterative progression of the product development process. The development period extends and the traceablility of product development processes get worse. On the other side, design iterations cause a higher level of information and thus they can be considered not just as unnecessary detours but also as learning processes. This overview points the importance of a purposeful support for developers in situations in which design iterations get necessary.

3. Key factors for the avoidance of design iterations and related approaches

As demonstrated above, to provide a purposeful support for the avoidance of design iterations, such a framework has to meet some demands regarding the product and the development process. Section 3 introduces some challenges that are seen as key factors for a successful avoidance of unnecessary design iterations and presents different approaches that were developed to cope with the identified key factors and thus can help to avoid unnecessary iterations.

3.1 Early completion of the list of requirements

To avoid design iterations resulting from unexpected and late changes of requirements, from new constraints or from changed boundary conditions, an early completion of the list of requirements is essential. To assure these early completion of the list of requirements, developers have to focus on two sources of requirements: On the one hand they have to focus on the desired product's behaviour to satisfy the customers´ needs, and on the other hand they have to consider different requirements resulting in boundary conditions from the product's lifecycle phases like e. g. purchasing, manufacturing, production, assembly, transport, logistics or recycling. The requirements in this contribution are seen as a combination from two sources: Firstly, they consist of those properties that are required to fulfil the desired product’s behaviour; secondly they contain those properties that are resulting from demands of the product's lifecycle phases. The following section will dwell on these two sources that both have to be considered in the completion of requirements.

(1) The product's behaviour as a measurement for the fulfilment of customers´ requirements

Customers will be up to pay for the product only if the product's behaviour meets their requirements. These requirements made on a technical system determine its main function. Starting from the description of this main function a certain desired behaviour is defined, which is considered to be able to fulfil the customers´ needs. In the following process steps developers define structural details of the product for the achievement of the desired behaviour. But to be able to evaluate the fulfilment of customers´ requirements based on the product’s behaviour, there is to define a way to describe the

product's behaviour. In [Krehmer 2009], the authors presented a framework, in which the behaviour is seen as based on the product's properties, and thus can be analysed from the defined characteristics. Hereby, the terms "characteristics" and "properties" are used analogue to the CPM/PDD-framework developed by Weber [Weber 2005]: Developers use the characteristics in form of specifications of geometry and material and shape to configure the product and thus to influence the product's properties. Thus characteristics are the developers´ set screws to achieve customers´ requirements. According to [Chakrabarti 2001], characteristics here are distinguished in two groups: "physio-chemical form" (material) and "geometrical form" (geometry & structure). Analogue to [Weber 2005], the product's properties cannot be defined directly, but result from definition of characteristics. Properties are measurable or quantifiable product properties like weight, stiffness or cost, as well as merely qualitative assessable properties like security, producibility, environmental friendliness or aspects regarding aesthetics. In this contribution, properties are divided in "intensive properties" and "extensive properties" [Chakrabarti 2001]. "Intensive properties" are the result from the choice of "physio-chemical form" (material) and thus are specific properties (e.g. density, specific weight, specific stiffness…). The product's "extensive properties" result from the combination of "intensive properties" with the "geometrical form": By choosing a certain material (physio-chemical form), the developers define also a certain density (intensive property). By defining the product's geometrical form, developers also define the volume. Thus, in combination with the material's density (physio-chemical form) the weight of the device can be determined as an "external property". These relations represent a framework for the complete assessment of the profile of product properties, which can be deduced from the defined characteristics by conducting analyses. The "behaviour" specifies the system's interaction with the environment. Boundary conditions of later product application as well as the whole product's lifecycle have influence on this behaviour and thus must be regarded during product development. Thus the product's behaviour can be considered as outcome of the properties under influence of the "mode of conditions and use" (figure 1), and properties are emerging from the definition of the characteristics "physio-chemical form" and "geometrical form".

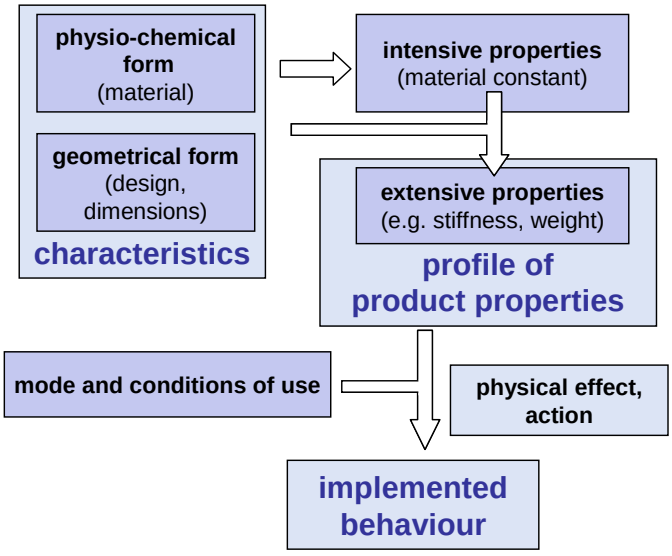

Figure 1. Definition of the product's behaviour based on characteristics and properties

By using these definitions, developers have to achieve such a profile of implemented product properties that there results a product's behaviour that fulfils the customers´ requirements on examination of the given mode of condition and use. When evaluating the degree of fulfilment of customers´ requirements, the properties that are used for monitoring have to be identified and selected at the beginning of the product development process.

(2) Additional requirements resulting from the product's lifecycle phases

Due to the existence of e.g. established manufacturing methods, existing supply chains as well as applicable laws and provisions there results a multitude of impacts affecting varied aspects of product development. These influences constitute several boundary conditions on product development, which are not expressed explicitly in the customers´ requirements. This means that the merely concentration on the customers´ requirements is not sufficient. The consideration of the influences from these different sources in the product development means the appliance of approaches regarding the field of "Design for X" [Bauer 2009]. Each product development has to meet these further requirements that are not directly demands on part of customers: These could be for example the demand for the ease of manufacturing and assembly or for example the constraint for the use of certain manufacturing methods that are easily available or notably advantageous. Independent from the kind of product that is being developed there is a general set of properties that has to be monitored to ensure a holistic safeguarding of the product. The DfX-approach provides a multiplicity of guidelines that help developers to consider aspects regarding these additional requirements. As mentioned above, these additional requirements emerge from boundary conditions resulting from the whole product's lifecycle. Table 1 gives an overview about different perspectives of possible stakeholders and the resultant diversity of properties that have to be monitored. For each of these stakeholders there is to define a number of key properties as they are suggested exemplified in table 1. By consideration of these key properties developers can achieve a more reliable completion of the requirements that are to be complied with by their product.

Table 1. Stakeholder perspectives and associated product's properties

Stakeholder	**Monitored properties / design for…**
Purchase	obtainment time for outsourced items, price for outsourced items, number of possible suppliers, reliability of possible customers...
Manufacturing	scrap rate and process reliability of chosen manufacturing method, manufacturing and processing time, durability of tools…
Assembly	storage, handling, positioning, testing, processing time, unambiguousness…
Marketing and sales	weight, dimensions, storage life, classification as hazardous good…
Laws, standards, regulations	laws, norms, engineer standards…
Utilisation	initiation, advice and documentation, customers´ requirements, ergonomics, maintenance, service, abandonment…
Recycling	recycling rate, compatibility of materials, disassembly time…

3.2. Appropriate sequence of development process steps

Besides the knowledge of requirements another important factor for avoidance of unnecessary iterations is the appropriate sequence of the development process steps. With every choice of certain solutions there are emerging new boundary conditions that have to be respected in further development. Detailing a chosen solution always results in new and so far unknown boundary conditions for other parts of the system which means the need for extensive adaption and thus for conducting design iterations. To ensure a development process proceeding without unnecessary design iterations developers have to consider existing dependencies between certain solution elements when defining the product. By neglecting these dependencies and by defining product details that entail many boundary conditions for other part systems in late process steps, developers can produce extensive chain reactions which finally result in time consuming and thus expensive detours. Consequently, a framework that aims at the avoidance of design iterations has to identify and handle those dependencies and has to provide the possibility to solve these chain reactions.

For the definition of purposeful process sequences the authors presented a method based on the Design-Structure-Matrix (DSM) in [Roelofsen 2008]. Aim of this approach was to introduce an approach for the determination of a process sequence in which given process steps are to carry out in that way that information loops and design iterations can be prevented. To do so, in the first step any existing connection between given process steps have to be identified by means of the DSM. Those

connections may result because one process step is dependent on the results of other process steps or because there exists a high demand for communication between different process steps. The process sequence derived from the DSM is based on the idea to start with the most active element because this element influences other more passive elements and thus provides information for other process steps. By planning the sequence of process steps according to the activity of elements a straight information flow without circular connections can be achieved, which helps to prevent unnecessary design iterations [Wynn 2007], [Grebici 2008]. By means of the DSM it is also possible to identify clusters of process steps that should be carried out concurrently to guarantee short communication cycles. If it gets possible to identify existing connections between properties and characteristics as well as dependencies between the chosen characteristics, it would be possible to identify the most active property and thus to start the product development with the definition of the related most active characteristic. As mentioned before, from each of these early definitions emerge new boundary conditions. Due to the fact that the development process proceeds from the most active to the most passive characteristic a possible late change of boundary conditions and thus also design iterations can be prevented widely. By determination of the influence of each characteristic on each other this DSM-based approach can avoid unnecessary design iterations preventatively and thus reduces cost and shortens development time.

3.3. Continuous monitoring of the achievement of objectives

To be able to avoid unnecessary design iterations, it is necessary to detect possible triggers for these iterations as early as possible. As shown above, design iterations are always induced by the non-fulfilment of at least one requirement. So, to prevent avoidable iterations, developers must be able to detect the non-fulfilment as early as possible. This requires continuous monitoring of achievement of objectives, which - as explained in section 3.1 - also considers the fulfilment of additional requirements being not formulated explicitly in the customers´ requirements.
For the efficient control of important dates and costs, knowledge about the progress of development processes and about the actual product's degree of maturity are fundamentals. Thus, aim of assessing the product's degree of maturity is to provide reliable statements regarding the outcome of development projects. In case of unsatisfying progress, monitoring of the development process and of the product enables the management to interfere and thus to stop negative trends as early as possible. Advantageous is the resulting transparency and the opportunity to avoid aberrations as early as possible. Furthermore, monitoring the degree of maturity allows forecasting of progressions and trends, earlier recognising of risks and enables developers to determine the real state of product development. Due to the fact that more properties than just the customers´ requirements are relevant when safeguarding the product being developed, in this contribution the product's degree of maturity is understood as being based on the products properties that determine the behaviour as well as on the properties necessary for the fulfilment of additional requirements resulting from the product's lifecycle phases. As shown in [Krehmer 2009], assessment of the product's degree of maturity can be conducted by using specific values by comparing the time-dependent actual realised state of product properties with their also time-dependent reference (scheduled) state:

$$achievement\ of\ property_i\ (t) = \frac{actual\ \ value\ \ of\ \ property_i(t)}{scheduled\ \ value\ \ of\ \ property_i(t)} \quad (1)$$

By specific weighting it gets possible to express the overall product’s degree of maturity at the considered moment. Weighting allows to blind out, to dilute or even to amplify the influence of each property if necessary:

$$product's\ degree\ of\ maturity\ (t) \quad = \quad \frac{\sum_{i=0}^{n} (achievement\ of\ \ property_i(t) \cdot weighting\ \ G_i)}{\sum_{i=0}^{n} weighting\ \ G_i} \qquad (2)$$

In cases where there is no possibility to quantify the achievement of properties as shown in formula (1), it may be the only way to check solely the existence of the property without any quantification. In those cases the overall-value in formula (2) for example can be used to compare the number of realised properties with the number of required properties. An approach for the evaluation of the product's degree of maturity as well as some requirements that a holistic safeguarding based on indicators has to cope with was outlined by the authors in [Krehmer 2009] and [Krehmer 2008b].
A framework that wants to prevent avoidable iterations has to be able to satisfy the contributing stakeholders´ different information requirements: Each perspective has to be enabled for detailed assessment of certain properties. This requirement can be satisfied by the possibility to accumulate the information in so called partial degrees of maturity. For this purpose, depending on what the developer wants to evaluate, three partial degrees of maturity could be helpful:

(1) Stakeholder oriented view on the degree of maturity

To enable all contributing stakeholders to evaluate the fulfilment of certain requirements resulting from their own specific perspectives, the accumulation of a stakeholder-oriented view is indispensable. By this stakeholder-oriented report it gets possible to prove perspective-specific aspects of the product (here e.g. the product's suitability for manufacturing/Design for manufacturing).

(2) Product structure oriented view on the degree of maturity

To be able to evaluate the fulfilment of perspective-specific requirements is not enough. Developers need the possibility to safeguard single parts and to evaluate the degree of fulfilment of requirements that the part has to meet. For this purpose it takes a product-structure oriented view on the degree of maturity: This allows safeguarding of each single part regarding to its compliance with requirements resulting from all existing stakeholders like e.g. manufacturing, assembly, and recycling. This partial degree of maturity allows the holistic safeguarding of each single part of the system.

(3) Indicator oriented view on the degree of maturity

In cases where developers want to safeguard certain properties of the product, an indicator oriented view on the product’s degree of maturity is needed: This view allows to pick a certain indicator and to check its different values for each part. This view allows developers for example to asses the weight or durability of every part of the product.

3.4 Communication and warning in case of unintended side affects of changes

As shown above besides consideration of existing dependencies and possible chain reactions and the continuous monitoring of the product's degree of maturity the early communication between affected developers and the inter-domain circulation and distribution of relevant information is one of the most important factors for the prevention of unnecessary iterations and thus absolute necessary. Sometimes developers may choose certain solutions without paying attention to the creation of unintended boundary conditions to other parts of the system. If it got possible to communicate those unintended but inevitable impacts as early as detectable, developers could be informed about new and changed requirements in time. This enables developers to respect these constraints as soon as possible and helps to prevent avoidable chain reactions and promotes the early completion of list of requirements.
In [Krehmer 2008a] the authors presented the so called property-driven-parameter-exchange to cope with the demand for the identification and communication of unintended changes and inevitable impacts as early as possible. In this approach, information is exchanged in form of external and internal parameters. External parameters influence other partial systems and therefore constitute

boundary conditions for other part systems and thus need to be considered in further development. Internal parameters have no direct influence on other partial systems and are only to be considered within this partial system. Firstly, the realised properties have to be classified in internal and external parameters (at the bottom level of the hierarchic product structure), before external parameters are communicated and delivered to the next higher level of hierarchic product structure where parts are combined to modules or partial systems. On each new level of the hierarchic product structure this classification of parameters has to take place again, and external parameters are to communicate to the next higher level of the product structure. There are three possibilities for the progress of external or internal parameters: Firstly, external parameters of one level can be internal parameters of the superior level. Secondly, external parameters also can stay as external parameters at higher levels, and thirdly, internal parameters do not reappear in higher levels. By consistent application of this approach developers get a framework on hand that helps to recognise and to evaluate unintended changes by paying attention to the fact that each decision creates constraints and boundary conditions to other parts of the product. Furthermore the framework promotes the often lacking communication between development partners as early as possible, whereby unnecessary iterations can be prevented.

4. Avoidance of unnecessary design iterations based on the monitoring of the product's degree of maturity

Section 3 depicted some approaches for the prevention of avoidable design iterations. The following two subsections introduce fundamentals for the framework for the avoidance of unnecessary design iterations, before the framework is explained by means of a simple example in section 4.3.

4.1. Connecting the product's degree of maturity and design iterations

The framework is based on an approach for the connection between design iterations and the degree of maturity which was presented by the authors in [Krehmer 2009]. According to that, monitoring the progress of the product's degree of maturity helps to detect the necessity of design iterations, allows evaluating the outcome of design iterations and determines gained influences on the product: Due to the fact that in case of new requirements or boundary conditions the reference value is changing, a decreasing product's degree of maturity can be the result of defining this degree as a time-dependent value comparing actual and scheduled state of properties. The example in figure 2 shows a possible progress for the degree of maturity and clarifies the connection between the need for design iteration and the progress of the degree of maturity: In the case of changing requirements or new cognisance the degree of maturity decreases (1). This decrease can be seen as a trigger for the execution of design iteration (2). Result of successful design iterations is an increased degree of maturity (3).

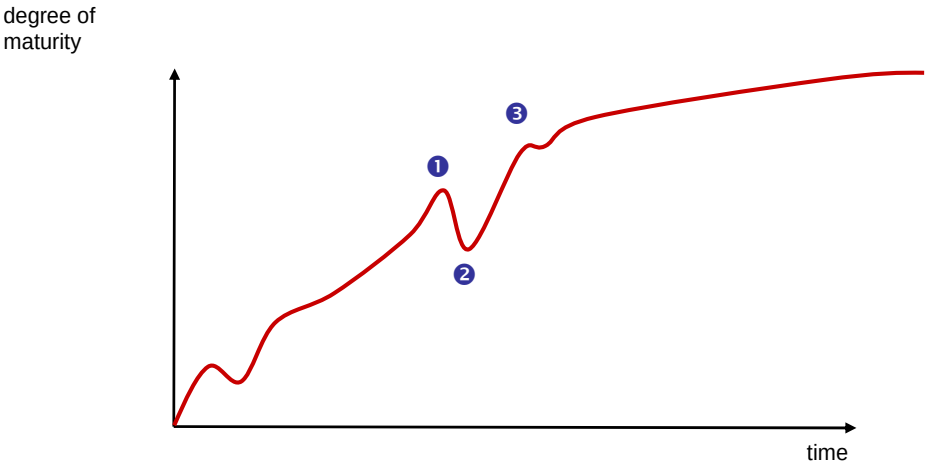

Figure 2. Connection between the product's degree of maturity and iterations

Furthermore, this connection enables developers to evaluate design iterations by assessing their effects on the product's degree of maturity and thus allows avoiding unpromising design iterations that do not have any positive influences on the product's degree of maturity.

4.2. Matrix based approach for representing the product's behaviour

Thinking about aspects regarding the product's degree of maturity means always to think about the product's properties, whereas conducting design iterations always means changing the product's characteristics. Due to the fact, that product's properties are result of the characteristics, the connection between characteristics and resulting properties is an appropriate possibility to connect the product's degree of maturity and design iterations. Based on the approach for a change impact analysis with the CPM/PDD-framework according to [Köhler 2008] an advanced variation of this approach based on the "House of Quality" was developed by the authors to provide an appropriate method to represent the overall product's behaviour [Krehmer 2009].

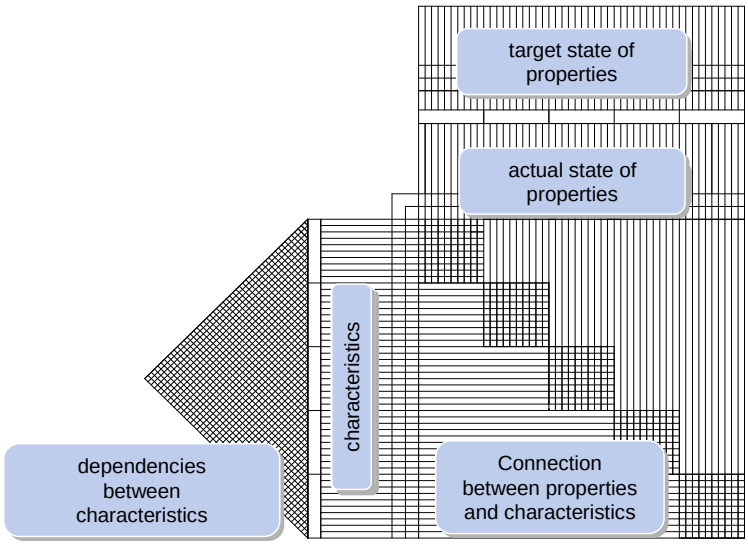

Figure 3. Matrix based approach for the representation of the product's behaviour

As the properties of a product are seen as the relevant measurement for the product's degree of maturity, by arranging the characteristics and the properties in form of a matrix and by signing certain characteristics as being connected with certain properties, it gets possible to evaluate the effects of a considered design iteration on the overall product's degree of maturity [Krehmer 2009].

4.3. Framework for the avoidance of unnecessary design iterations

In this section the constituents depicted in sections 4.1 and 4.2 as well as the approaches depicted in section 3 are integrated to a framework for the avoidance of unnecessary design iterations. This framework is based on the matrix based approach for the representation of the product's behaviour shown in figure 3. In the first step, by considering both sources of required properties (customers´ and additional properties regarding to DfX, c.f section 3.1 part (1) and (2)), the developer has to make sure that the list of required properties is supplemented by the additional properties shown in table 1. The developers have to assure that there exists a complete requirement list as basis for the further ongoing development. This complete list of required properties is to be filled into the field "target state of properties" of the matrix shown in figure 3. In case of an unachieved target property, this framework provides a method to identify all subsequently necessary changes before conducting any change of characteristics. Thus the framework helps to avoid unnecessary iterations by depicting connections between properties and characteristics.
In the example shown in figure 4, the developer may identify the value "weight" (signed with ①) as an unachieved property. The arrow directed downwards (↘) indicates that the aim of the design iteration is to minimise the weight. In the next step (signed with "A") the developer will identify all

characteristics that have influence on the property "weight". That may be the characteristics "height" (signed with ❶), "width" (❷) or "material" (signed with❸) in this example. Then, in step "B" the developer may elect the value "height" (❶) as the characteristic to be changed in design iteration. The consequences of the change of "height" and thus the impact on the properties are detected in step "C": Besides a minimised weight, the design iteration in this example results in a decreased stiffness (signed with ②) as well as in increased costs (③).

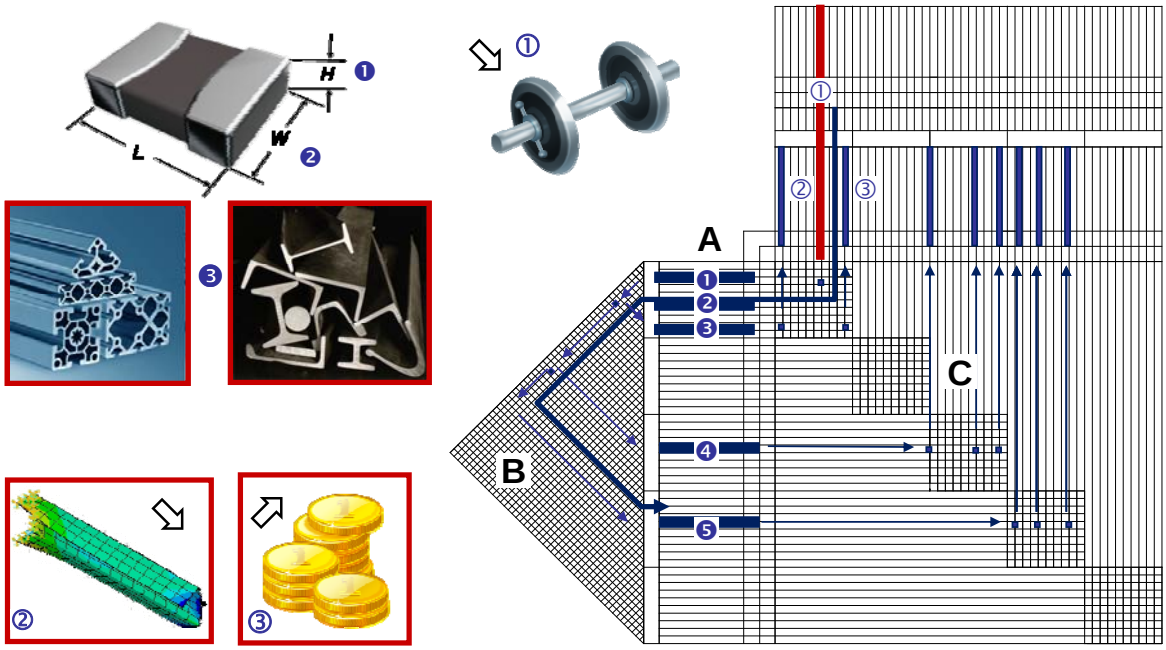

Figure 4. Identification of responsible characteristics and affected properties

As shown in section 3.3, a framework that wants to help developers to avoid unnecessary iterations has to promote a continuous monitoring of the achievement of objectives. By showing the developers all consequences of their decisions quite plainly, this framework enables continuous monitoring of the degree of fulfilment of each required property of the product. Another requirement for the framework was to enable the developers to determine an appropriate sequence of development process steps, and furthermore to support the early communication between developers in case of inevitable impacts. By detecting complex networks between properties and characteristics, this framework allows identifying certain characteristics that have influence on many others (as they are very active characteristics) and helps developers to start product definition by the determination of the correct "set screws". By preventing late appearance of new boundary conditions the framework prevents unnecessary and avoidable design iterations. Furthermore, due to the fact that it allows an early recognition of unintended impacts, early communication and even a warning system for negative impacts gets possible.

5. Conclusion and Outlook

Conducting design iteration always means a change of the product's configuration. Frequently, those changes activate further product changes and thus they can generate chain reactions and unnecessary iterations. Aim of this contribution was to introduce a framework for the avoidance of unnecessary iterations which is based on the monitoring of the degree of maturity. For this purpose, section 2 dwelled on of design iterations in general and in section 3 some key factors for the successful avoidance of unnecessary design iterations were identified. Furthermore, section 3 introduced some promising approaches that were developed by the authors to cope with these requirements to avoid unnecessary detours. In section 4, the fundamentals and constituents for the framework were presented as well as the framework for the avoidance of unnecessary design iterations based on the monitoring of the product's degree of maturity was introduced and clarified by means of an example. By

demonstrating the changes of the product's properties this framework enables developers to consider unintended influences of design changes and to evaluate subsequent necessary changes and their resulting influence on the product's properties. Thus the framework helps to conduct only those design iterations that are manageable and have predominantly useful impacts. Thereby the framework establishes the basis for a purposeful and more transparent product development with faster increasing of the degree of maturity and less unnecessary and time consuming loops. Thus the framework is fundamental for saving cost and money.
An important topic of further research will be the application, the enhancement and validation of the introduced framework on an example of a reasonable degree of complexity. Other objectives of further work will be the research of possibilities of capturing interdependencies between characteristics and properties and the dissolving of circular dependencies. Furthermore mechanisms for the detection and recognition of necessary design iterations and their evaluation by means of the changed degree of maturity will be in focus of further work.

References

Bauer, S. "Entwicklung eines Werkzeuges zur Unterstützung multikriterieller Entscheidungen im Kontext des Design for X", 978-3-18-340401-8, Lehrstuhl für Konstruktionstechnik, Erlangen, 2009.

Chakrabarti, A., "Engineering Design Synthesis - Understanding, approaches and tools", Springer Verlag, London 2002.

Grebici, K.; Goh, Y. M.; McMahon, C., "Uncertainty and risk reduction in engineering design embodiment process", Proceedings on the 10th International Design Conference (DESIGN 2008), D. Marjanović (Ed.), FMENA, Zagreb, 2008, pp. 143-156.

Köhler, C. Conrad, J., Wanke, S. "A matrix representation of the CPM/PDD approach as means for a change impact analysis", Proceedings on the 10th International Design Conference (DESIGN 2008), D. Marjanović (Ed.), FMENA, Zagreb, 2008, pp. 167-174.

Krehmer, H.; Meerkamm, H, "Approach on the control of iterations in the multidisciplinary development of technical systems", Proceedings of the 10th International Design Conference (DESIGN 2008), D. Marjanović (Ed.), FMENA, Zagreb, 2008, pp.1303-1310.

Krehmer, H.; Paetzold, K., "Eine Betrachtung zur ganzheitlichen Abschätzung des Produktreifegrades auf Basis des Verhaltens", Design for X - Beiträge zum 19. Symposium, Meerkamm, H., Neukirchen, 2008, pp. 67-78.

Krehmer, H.; Meerkamm, H.; Wartzack, S, "The product´s degree of maturity as a measurement for the efficiency of design iterations", eProceedings of the 17th International Conference on Engineering Design 2009 (ICED´09), Palo Alto, USA, 2009.

Roelofsen J.; Krehmer, H.; Lindemann U.; Meerkamm, H. "Using the Design-Structure-Matrix for the avoidance of unnecessary iterations", Proceedings of the 10th International Design Structure Matrix Conference DSM´08, Stockholm, Sweden, 2008, pp.. 209-211.

Weber, C., "CPM/PDD - An extended theoretical approach to modelling products and product development processes", Proceedings of the 2nd German-Israeli Symposium on advances in methods and systems for development of product and processes, TU Berlin / Fraunhofer-Institut für Produktionsanlagen und Konstruktionstechnik (IPK), Fraunhofer-IRB-Verlag, Stuttgart 2005, pp.159-179.

Wynn, D.; Eckert, C. M. and Clarkson, P. J., "Modelling iteration in engineering design". Proceedings on the International Conference on Engineering Design (ICED´07), Paris, 2009, Paper No. 561.

Dipl.-Ing. Hartmut Krehmer
Research Associate
University Erlangen-Nuremberg/Department for Engineering Design
Martensstraße 9, D-91058 Erlangen, Germany
Tel..: +49-(0)9131-8523215
Fax.: +49-(0)9131-8523223
Email: krehmer@mfk.uni-erlangen.de
URL: http://www.mfk.uni-erlangen.de