PRODUCT MODEL OF THE AUTOGENETIC DESIGN THEORY

Konstantin Kittel (1), Peter Hehenberger (2), Sándor Vajna (1), Klaus Zeman (2)
(1) Otto-von-Guericke University Magdeburg, Germany, (2) Johannes Kepler University Linz, Austria

ABSTRACT
Product development plays the key role in defining all product properties and benefits. There is a need for appropriate supporting methods that are able to serve and to satisfy multi-criteria and multi-domain requirements. These requirements usually go beyond function fulfilment. Developing mechatronical systems raises the requirements for the applied product development methods, because of the high number of linked elements from different technical und physical domains. Integrating and combining solutions from different domains does not only increase the number of possible solutions, but also the complexity of such systems. Product development methods need to be able to handle these requirements appropriately in order to support the designer within the product development process.

Keywords: Product Development, Evolution, Mechatronics

1. INTRODUCTION
The Autogenetic Design Theory (ADT) uses analogies between natural evolution and product development to ensure that the best possible solution can be found within given requirements, conditions, and boundaries [1][2][3]. These requirements, conditions, and boundaries can also contradict each other and can change over time, i.e. "best possible" has always to be interpreted in relation to the actual situation. The ADT describes the development of products as a continuous improvement and optimisation process. The ADT is called “autogenetic” due to its similarity to processes of autogenesis of the natural evolution.
The ADT is not another variety of Bionics (where results of an evolution, e.g. the structure of trees, are transferred to technical artefacts). Rather, the ADT transfers procedures from biological evolution to accomplish both a description and broad support of product development with its processes, requirements, boundary conditions, and objects (including their properties).

2. THE AUTOGENETIC DESIGN THEORY
The Autogenetic Design Theory (ADT) applies analogies between biological evolution and product development [4] by transferring the methods of biological evolution (and their advantageous characteristics) to the field of product development. Such characteristics are for example the ability to react appropriately to changing environments (requirements and boundary conditions), so that new individuals are in general better adapted to the actual environment as their ancestors.
The main thesis of the ADT is that the procedures, methods, and processes of developing and adapting products can be described and designed as analogies to the procedures, methods, and processes of biological evolution to create or to adapt individuals. Main characteristics of biological evolution (with the underlying principle of trial and error) are continuous development and permanent adaptation of individuals to dynamically changing targets, which in general have to be accomplished in each case at the lowest energy level and with the minimal use of resources, i.e. the evolution process runs optimised in terms of energy consumption and resource employment. The targets can change over time because of (unpredictable) changing requirements, resources, conditions, boundaries, and constraints, and they can contradict each other at any time.
The result of a biological evolution is always a set of unique solutions being of equal value but not being of similar type. Consequentially, the result of the ADT is for the very most part a set of equivalent, but not similar unique solutions that all fulfil the actual state of requirements and conditions best.
Furthermore, biological evolution doesn't have prejudices. This means that new individuals (described by their chromosomes) will not be discarded because they are different. Each individual has to prove

itself in its natural environment. If it turns out that an individual with a new chromosome set is superior to already existing individuals, then this individual gets a better chance for reproducing. By transferring this behaviour of impartiality to product development, new concepts would not be discarded because they were totally different than former concepts but only if their properties were proven not to be superior.
This suggests that both evolution and product development can be described as a continuous but not straightforward improvement process or as a kind of multi-criteria and continuous optimisation [2][4].
One may argue that a weakness of the ADT is the processing (creation, evaluation) of a high number of individuals for reaching a certain product progress due to the evolutionary based approach. Compared to other methods, the number of solutions, which need to be evaluated, is in fact much higher. But one has to keep in mind that, by exploring this high number of possible solutions (individuals) within a solution area, the chance of finding the really best set of solutions to a set of requirements, conditions, boundaries, and constraints is much higher than with traditional approaches that continuously delimit the solution area and thus result only in a single "next best" solution [5].
The analysis of product development from an evolutionary perspective leads to the following insights [1]:

- In every phase of the product development process, various alternatives are developed and compared. These alternatives are in competition with each other, because only the actually best were selected for further processing.
- The processes of searching, evaluating, selecting, and combining are also typical approaches of biological evolution.
- Regardless of the phase of product development or of the complexity level of the emerging product, always similar patterns of activities can be identified that modifies existing or that generates new solutions. This patterns can be compared to the TOTE-Scheme [6][7]. Self-similarity can be found at all levels of complexity of product development as well as in all stages of the emerging product [2].
- According to chaos theory small changes or disruptions in the system can cause unpredictable system behaviour [8]. The fact that the result of the development of a product usually can't be predicted definitely because of the influence of the creativity of the product developer leads to the assumption that the product development process also contains elements of a chaotic system or at least shows a chaotic behaviour in some aspects.

At the present state, three major components of the ADT have been researched. First, a process model describing how the ADT works and what the steps are, which the product developer has to perform (see section 2.1). Secondly, the solution space model, which shows how the space, in which product development takes place, is structured (see section 2.2). Thirdly, the underlying product model that holds the description of how product information is structured and used (see section 3).
The development of the ADT is content of a research project running right now. This means that the concepts presented in this paper are not in a final state. Especially the practical appliance will be in the focus of further work.

2.1 Process Model

There are numerous ideas and concepts aiming on describing the complex and often chaotic process of developing. Caused by the complexity of most products, the focus within the development process isn't mostly on the complete product, but rather on a specific part of the product. This means that each development step is focused on improving or modifying only one specific product property or a specific set of properties by varying certain design parameters.
A challenge in the development of the process model is to define a model, which is not only able to describe the development process in the later stages of the development process but also in the early stages. FAN, SEO, ROSENBERG, HU and GOODMAN describe an approach, wich provides an automated system-level design for mixed domain systems [17]. But as most other methods [9][10] this approach uses a set of predefined elements which can be used by the algorithm to create new solutions. Depending on the available predefined elements the achievable solution quality is more or less limited (see section 2.2).
The ADT process model currently under research aims on providing a holistic development process model that is also able to describe the processes of partial improvements/modifications, which

normally do not follow a predefined pattern. The ADT process model describes the development process on two levels.

Level one

Level one provides an overall look on product development within the ADT. It starts with the definition and description of the target function[1] and the solution space based on requirements, starting conditions, boundary conditions, conditions of the environment of the solution space, and (internal and external) constraints. Within the solution space, possible solution patterns are searched, combined, and optimised in a random order. To evaluate the actual state of development, the particular fitness[2] is determined. Because requirements, processes, and both internal and external influence factors are all dynamic due to unforeseeable changes, it is clear that it is only possible to describe (rather small) process patterns, which can be used randomly, instead of specifying a sequence of steps or any predefined "way" the designer has to follow.

To nevertheless support the process of partial improvements or modifications (and allow the designer to address only a limited set of properties) without loosing both consistency and the overall picture, different views can be applied (see section "Product Model"). These views act like filters that ensure that only a specific set of product properties are considered. Thus, the development process becomes a set of activities, each containing a product improvement or modification under a specific view.

Level two

Level two describes under a certain view the activities of improving or modifying a set of properties that are determined by design parameters. The ADT uses the steps creation, evaluation, and updating to modify and to improve a product.

Creation

In analogy to biological evolution, the creation step consists of the four sub-steps selection – recombination – duplication – mutation.

- **Selection** The parent solutions for the next generation are designated.
- **Recombination** The design parameters of two already existing solutions are combined to create (in most cases) a more evolved solution. This is the usual way to create new solutions.
- **Duplication** Creation of an identical copy (duplicate, clone). Creating clones is recommended if solutions with superior properties exist, which should be inherited to the next generation.
- **Mutation** A random change of the design parameters of a solution that was created by recombination. Mutation is necessary to ensure dynamics in the evolution process.

Evaluation

In this step each new solution is evaluated to determine the fulfilment of the optimisation criteria described in the target function. Based on this information the actual fitness (representing the quality of the actual solution) is calculated.

Updating

The update of both solution space and target criteria is necessary to take dynamic requirements into account. Often requirements change within the development process. Such a change influences the solution space (also with the result that a specific solution is not allowed any further) as well as the target criteria (with the result that a specific target criterion gets less important or that another criterion should have a bigger influence on the product fitness).

2.2 Solution Space

In general, the term "solution space" is understood to be a set of all feasible solution elements, which can be used within product development. This includes all elements that a product developer may use for the evolution of a solution or several solutions, on the basis of requirements, inner and outer

[1] The target function calculates the fitness of a solution based on the optimisation criteria of this specific solution.

[2] The fitness represents the actual degree of development of a specific solution. The fitness is used to compare two or more solutions.

conditions, ecological/environmental conditions and others. This definition of a solution space can be compared with the mathematical term “domain”.

Every product development method has its solution space, which usually is spanned by the requirements and limited by both starting and boundary conditions. The inner structure is influenced by constraints. Some approaches, e.g. TRIZ [9] and Gene Engineering [10], use a solution space with a particularly structured dataset. This dataset contains the solution elements for the emerging solutions. It is common to all such solution spaces that the product developer is offered only a limited amount of possible solution elements. However, the more limited the quantity and possible configurations and combinations are the lower is the achievable solution diversity and quality.

Thus, to improve both solution diversity and quality, it is necessary to not artificially limit the quantity of solution elements, but rather to include permissible elements for all concrete tasks. Thus follows, however, the task of holding on to all permissible elements in the solution space description. As the diversity of existing solution elements (materials, manufacturing methods, operating principles, etc) is immense, a complete solution space description at reasonable time and costs is in most cases impossible. Since the optimal configuration and combination of solutions elements are not available under these circumstances, the maximal possible solution quality can't be achieved.

In order not to limit the product developer and to permit the maximal possible number of allowable solution elements, the ADT applies an inverted solution space. The only limitations of such an inverted solution space are the laws of natural science, i.e. the space is virtually infinite. The inverted solution space contains prohibited areas (taboo zones). Taboo zones are formed by those solution elements of which the use for possible solutions is explicitly forbidden. This inverted solution space is referred to in the following as Prohibition Space.

Another advantage of the Prohibition Space is evident in the early phases of the development process. Due to the lack of knowledge about the relationships between requirements and solution elements, it isn't often possible to determine the forbidden criteria, based on the forbidden requirements. The Prohibition Space therefore contains too few taboo zones at the beginning of the product development process. At this time, the product developer is able to use product criteria, for example, that should actually be forbidden. If, for example, impermissible solution elements are used, it will be noticed upon evaluation that the resulting solution possesses impermissible product criteria. This newly obtained information can be used to refine the Prohibition Space.

A traditional solution space would in the same case be incomplete as well (due to the lack of knowledge about the coherences). Here, however, "incomplete" means that permissible elements are missing, thereby restricting the product developer’s solution possibilities. Since a traditional solution space consists by definition of only permissible solution elements, only solutions with permissible criteria will be generated (exceptions could be certain combinations of permissible elements). This means that the existence of an incomplete solution space (some permissible solution elements are missing) is difficult to detect. The definition of the Prohibition Space is based on the requirements and the various starting and boundary conditions, constraints, and the environment, which can arise from different sources. Through the inversion, requirements and conditions turn into appropriate bans that can be (quite easily) formulated and the solution elements that are forbidden can be derived.

The Prohibition Space dynamically changes whenever an external event (for example a requirement modification, a change of a condition, etc.) occurs during the evolution, because, as a result of this external event, changed possibilities for the evolution can arise or existing ones have to be omitted. To reflect these, taboo zones within the Prohibition Space have to be redesigned, which may result in changing taboo zones, in omitting existing, or in adding new zones (figure 1).

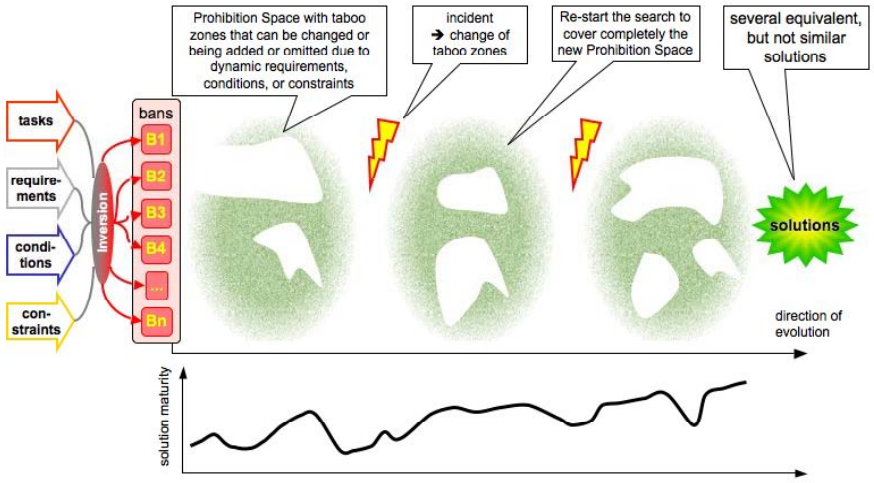

Figure 1: Inversion of requirements and changes of the Prohibition Space due to external events

3. PRODUCT MODEL

The aim of a product model is to provide a framework to capture all product data, which are necessary to describe the product and its life cycle in a structured way. This framework shall be stable for the whole development process, so that data can be completed step by step. In "traditional" product development, different data structures arise along the sequential phases of the product development process.

The design of the product model shall regard the requirements of the modeling of multidomain systems. Mechatronic systems differ from conventional technical systems in the higher number of coupled elements from different technical and physical domains. By combining and integrating solutions from various domains the number of possible solutions increases together with the complexity of such systems. This raises the need for a structured approach for designing and modelling.

An incontestable precondition for successful product development is the interdisciplinary definition, description and presentation of product information. The following issues and requirements for the model representation need to be realised with the product model.

3.1 General Requirements

- Description of product information from all phases of the product life cycle: Taking into account the lifespan of a product is necessary when storing the product relevant information within a single structure.
- Coupled analysis of different physical product properties: Caused by the increasing integration of different applications and functionality in technical systems, the demand arose to find a uniform description of the various product properties, to define, describe, dimension, calculate the product.
- Taking into account the view on an application domain: Different development phases and mechatronic disciplines require different views on the object to be analysed. It follows for a general description of the requirement that model parameters should be provided in various combinations and granularity for a further (more detailed) processing. The influence of the system should be possible on the need for the individual points of view characteristics.

3.2 Mechatronical Requirements [12][13][14]

- Description of divergent aspects, which are related to the complex characters of and dependencies between the involved domains.
- Support product information from the design phase.
- The product model has to represent the physical behaviour of the technical system, which depends on the properties of the subsystems and their interactions. A sub-model can be a model of a single mechanical component or of a complex system e.g., a model of an integrated

mechatronic system including an embedded control system.

- The product model has to represent the interface model between sub-models form different disciplines.

3.3 Types of properties

For the description of the ADT the product properties are divided into required properties, resulting properties and definable properties. The required properties can be retrieved from the requirements on the product (customer requirements, boundaries, conditions). The required properties describe the goal state of the product. The actual state of the product is described by the resulting properties. The resulting properties are set by the definable properties (according to physical relations). The definable properties can directly be determined by the product developer. At the end of the development process, the definable properties need to define a product, whose resulting properties match the required properties. A complete matching of resulting properties and required properties at the beginning of product development process need not be given. Along the product development process, the properties are expanded and detailed.

3.4 Concept

The structure of the ADT product model aims on storing the product information from the entire product lifecycle. This requires a structure, which can already be used in the early phases and which does not change along the product development process. Normally, product information is structured according to the geometry of the product or according to the assembly structure of the product. This results in problems in the early phases of the product life cycle, because the geometry or assembly structure is a result of the development process and neither exists in the early phases nor is it stable during the development process.

The “object” which is most stable during product development is the set of resulting properties, which defines the product. As stated before, requirements can change over time, but when being precise, changed requirements mean to development a new product. Trying to find a product information structure which is suitable for every set of requirements needs to result in a structure which is identical for every product development process.

The ADT product model at present is inspired on the extended feature model of the FEMEX. The FEMEX model provides a unified structure to store all product data and information from the whole life cycle [15]. The ADT product model uses the basic matrix structure of the FEMEX model and the method of classifying the product information according to product properties. Contrary to the FEMEX model the ADT product model does not use the product life cycle phases, but multiple views[3] as a second way to classify the product information. In the actual ADT product model a product is described by a certain number of definable properties (represented by the different coloured boxes in Figure 2). To organise all the definable properties, a structure is derived from the existing resulting properties on the one hand and from the different views on the other side.

[3] Views are like filters on the product. They include a subset of definable properties as well as a subset of resulting properties.

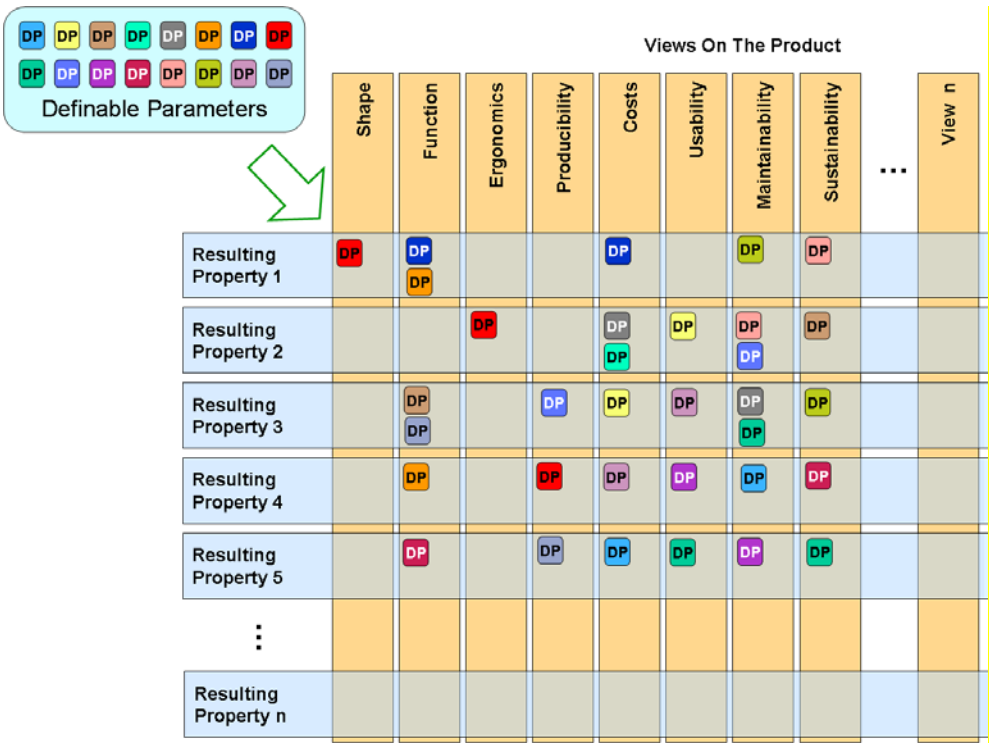

Figure 2: The ADT product model

It has to be mentioned that a definable property is not equivalent to a geometric parameter. Basically, a definable property can be any property the designer defines directly (e.g. materials, manufacturing parameters, geometry, …). The totality of all definable properties defines the complete product with all its properties and its behaviour.

The resulting properties are used to structure the generally high number of definable properties. This is done, by assigning each definable property to the resulting properties, which were influenced by this definable property. As mentioned before, it is possible that a definable property influences more than one resulting property (for example: the definable property “material” influences the resulting properties “maximum weight” and “maximum stress”). A second level of structuring is achieved by assigning the definable properties to the different views. Each definable property can appear in a single view or in multiple views (for example: the definable property “material” appears in the view “producibility” and “costs”). In order to classify the definable properties in the matrix, a meta information (tag) can be assigned to each of them. This meta information can be the view or the product property influenced by the definable property.

Another advantage of this representation form is that the influence of definable properties can be quickly determined. It is obvious, which definable property influences which resulting property. Resulting properties that depend only on a single definable property can be determined at the very beginning of the development process without taking into account dependencies with other resulting properties. This dependence can be checked very simple by analysing the tags of the definable properties.

The ADT product model uses chromosomes to cluster the product information: A chromosome contains a resulting property and all the definable properties on which it depends on (marked with the purple line in Figure 3).

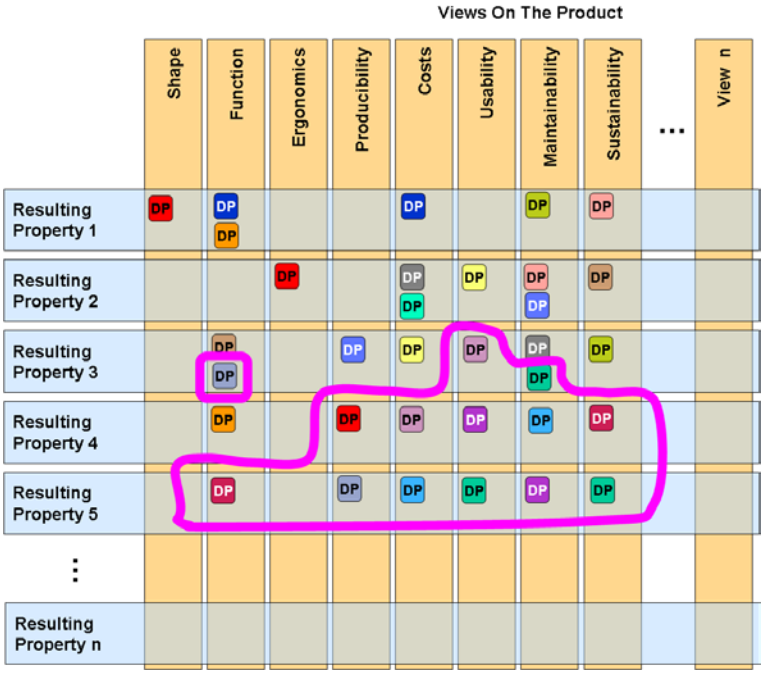

Figure 3: Chromosome model

The values of the included definable properties of the chromosome can change along the development process. Such a change can be triggered by different events. Possible types of events are:

1. Optimisation of a definable property. This is the most common case. The product developer adjusts a definable property to achieve an improvement of a resulting property.
2. Changes due to dependencies. When a definable property changes, it is possible that a thereof dependent definable properties must be adjusted, e.g. if a definable property dictates a material and another definable property dictates the wall thickness. In this constellation it can occur that the definable property wall thickness has to change when the definable property material changes, because not all former values were permitted any longer.
3. External event. By a change in the requirements new or adjusted taboo zones arise. This new circumstance can create the need that definable properties need to be adjusted, because certain values are not permitted any longer.

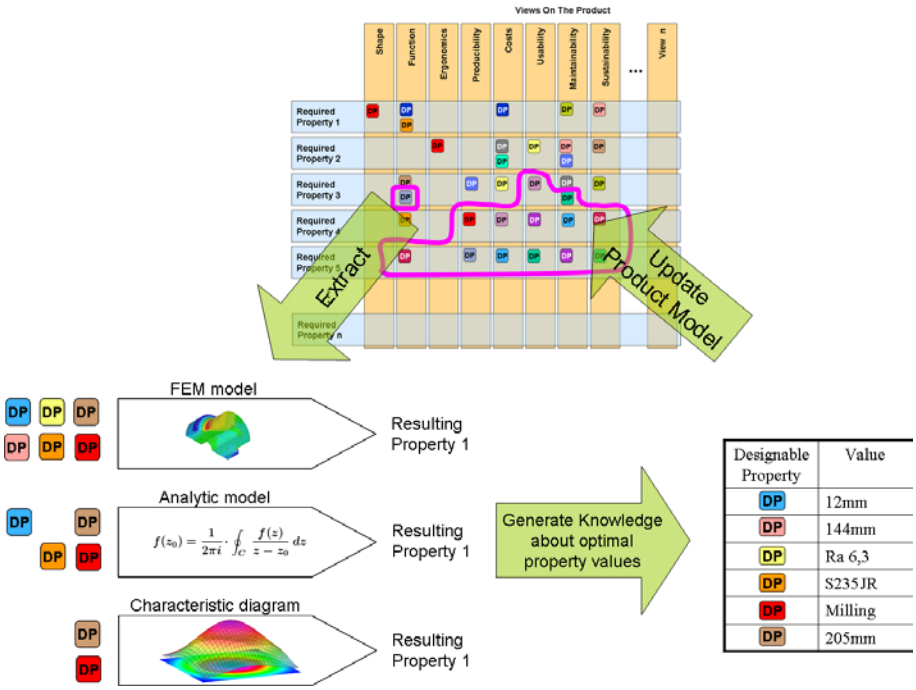

Figure 4: Subsidiary models

Subsidiary models

When working with the product model, the product developer will never work on all definable properties parallel. In most cases, he will focus on the properties clustered by one chromosome (a resulting property and all definable properties it depends on) or a few chromosomes. Subsidiary models are used to describe the behaviour of the product reduced on the interrelation between definable properties and resulting properties within the chromosome (see Figure 4). Normally there is more than one subsidiary model available to describe this interrelation. The subsidiary models differ in the model aim and the method which describes the interrelation (FEM, analytic, diagram). The model aim is defined by the resulting properties, which shall be described and the effects, which were considered or were explicitly not considered (e.g friction, inhomogeneity).

Different subsidiary models use the information from product model. These models differ in their complexity, which depends on:

- Considered definable properties
- Simplifications (considered effects)
- Method to describe the interrelation (numeric, analytic, graphical)

In most cases more than one subsidiary model can be used for the description of a specific resulting property, whereas the less complex subsidiary models were used in the early phases of the product development process, when there is only little information about the product. During the product development process more and more complex subsidiary models can be used, to describe the interrelation between definable properties and resulting properties.

The knowledge about the interrelation between definable properties and resulting properties, gathered together from the subsidiary models can be used to set the values of the definable properties, such as geometric dimensions, material type, properties of the production processes. For example the product designer uses a fem model to determine the occurring stresses. This subsidiary model than provides him information about the relation between definable properties (geometric parameters and material parameters) and resulting properties (maximum stress, maximum deformation). Due to the information from this subsidiary model, the product developer is than able to choose the right values for the definable properties in order to match resulting properties and required properties.

Property dependencies

Increasing the number of definable properties is going along with an increased number of dependencies between these properties. In order to ensure the consistency of the product model the dependencies between the definable properties need to be managed. In [16] a meta model is used to describe the dependencies between the different domains of the product. A similar approach shall be used to ensure the consistency within the ADT product model.

The goal of this system of rules is to support the designer in generating permissible solutions, by providing information about depended definable properties. The system of rules displays possible consistency problems, which can arise by changing a specific definable property.

The dependencies are stored as meta information for every definable property. At the moment the system of rules is designed to provide the following information:

- Which other definable properties depend on the regarded definable property (e.g. a material forces a specific manufacturing process).
- Which values were forced for the dependent property, if the regarded property takes a specific value (e.g. if definable property “material crossbar” takes the value “aluminum”, a value of “>0,5mm” is forced for the definable property “wall thickness crossbar”).
- Which values were forbidden for the dependent property, if the regarded property takes a specific value (e.g. if definable property “surface roughness” takes the value “<Ra 0,5”, the value “milling” is forbidden for the definable property “manufacturing process”).

4. CONCLUSION & OUTLOOK

The research on the ADT has shown that both complexity and disciplinarity of products increase towards incorporating domains beyond mechanical engineering. This overview over the different components of the ADT shows the concepts that have been developed so far.

The ADT product model offers an approach to describe a product detached from its actual physical structure. This allows the application of the product model even in the early phases of the development process. The system of rules supports the product developer by providing important information about definable properties.
Future research work on the ADT will deal with process model of the ADT to describe what the product development process looks like and what steps the product developer has to take to develop product with the ADT.

ACKNOWLEDGMENTS

We gratefully acknowledge that this work has been supported by the German Research Foundation (DFG) and has been supported in part by the Austrian Center of Competence in Mechatronics (ACCM), a K2-Center of the COMET/K2 program (which is aided by funds of the Austrian Republic and the Provincial Government of Upper Austria)

REFERENCES

[1] Vajna, S., Clement, St., Jordan, A., Bercsey, T., "The Autogenetic Design Theory: an evolutionary view of the design process". Journal of Engineering Design 16(2005)4 pp. 423 – 444
[2] Wegner, B., "Autogenetische Konstruktionstheorie – ein Beitrag für eine erweiterte Konstruktionstheorie auf der Basis Evolutionärer Algorithmen", Dissertation Universität Magdeburg 1999
[3] Vajna, S., Clement, St., Jordan, A., Bercsey, T., "The Autogenetic Design Theory: an evolutionary view of the design process". Journal of Engineering Design 16(2005)4 pp. 423 – 444
[4] Bercsey, T., Vajna, S., "Ein Autogenetischer Ansatz für die Konstruktionstheorie". CAD-CAM Report 13(1994)2, pp. 66-71 & 14(1994)3, pp. 98-105
[5] Clement, S., " Erweiterung und Verifikation der Autogenetische Konstruktionstheorie mit Hilfe einer evolutionsbasierten und systematisch-opportunistischen Vorgehensweise", Dissertation Universität Magdeburg 2005
[6] Miller, G.A., Galanter, E., Pribram, K.H.: Strategien des Handelns. Pläne und Strukturen des Verhaltens (2. Auflage). Klett-Cotta Stuttgart 1991
[7] Ehrlenspiel, K.:, "Integrierte Produktentwicklung". Carl Hanser Verlag München 2007
[8] Briggs, J., Peat, F. D., "Die Entdeckung des Chaos". Carl Hanser Verlag München 1990
[9] Altshuller, G., "40 Principles – TRIZ Keys to Technical Innovation" (translated and edited by L. Shulyak, S. Rodman). Technical Innovation Center Worcester MA (USA) 2003
[11] Chen, K.Z., Feng, X.A., "A Framework of the Genetic-Engineering-Based Design Theory and Methodology for Product Innovation", in: Norell, M. (editor): Proceedings of the 14th International Conference on Engineering Design ICED03, presentation 1745, Design Society 31, Stockholm 2003, Abstract p. 671
[12] Avgoustinov, N. Modelling in Mechanical Engineering and Mechatronics, Springer Publishing Group, UK, 2007
[13] Bishop, R.H. Mechatronic Fundamentals and Modeling (The Mechatronics Handbook), CRC Press Inc, New York, 2007
[14] Alvarez Cabrera AA, Foeken MJ, Tekin OA, Woestenenk K, Erden MS, De Schutter B, van Tooren MJL, Babuska R, van Houten FJAM, Tomiyama T. Towards automation of control software: A review of challenges in mechatronic design, Mechatronics ISSN 0957-4158. http://dx.doi.org/10.1016/j.mechatronics.2010.05.003
[15] Ovtcharova, J., Weber, C., Vajna, S., Müller, U., "Neue Perspektiven für die Feature-basierte Modellierung" VDI-Z 140(1997)3, pp. 34-37
[16] Hehenberger P., Egyed A., Zeman K.: Hierarchische Designmodelle im Systementwurf mechatronischer Produkte, VDI - Tagung Mechatronik 2009, Komplexität beherrschen, Methoden und Lösungen aus der Praxis für die Praxis, 12.- 13. Mai 2009, Wiesloch bei Heidelberg, Deutschland
[17] Z. Fan, K. Seo, R. Rosenberg, J. Hu, E. Goodman, “A Novel Evolutionary Engineering Design Approach for Mixed-Domain Systems”. Engineering Optimization, Vol. 36, No.2, April 2004, pp127-147