16

A Design Process Model based on Design Working Spaces

H. GRABOWSKI, R.-S. LOSSACK, C. WEIS
Institute for Computer Application in Planning and Design (RPK),
University of Karlsruhe (Germany)
Kaiserstraße 12
D-76128 Karlsruhe
email: {gr, lossack, weis}@rpk.mach.uni-karlsruhe.de

Abstract

The german design methodology demonstrated its usefulness for solving design problems by being applied in the enterprises' design departments during the years. Design methodologists as Roth, Pahl, Beitz and Hubka developed an instrument for a systematic approach to solve design problems. This approach is strongly *process oriented* and describes together with the fundamentals of design, *general strategies* for solving design problems. Another approach followed by the researchers of CAD systems was concentrated on *information modeling*. Until now there was no approach combining these two different approaches consequently. In this article we introduce *a concept which integrates these different views*. In the first part we give an overview of the systematic approach of the german design methodology especially describe what they have in common. After that we introduce the statical fundamentals of the design process for technical systems. This part is conform with the information modeling approach. In the third part we describe the fundamentals of the process oriented aspects of the german design methodology with the conceptual model of the so called *modeling space of design*. Then we introduce so called *design working spaces to structure and administer design solutions*. Based upon the fundamental statical information model and the dynamical view of the design process we will finally give a proposal of an architecture and present the concept of process patterns for an intelligent CAD-System supporting the designer in the whole design process. We always illustrate the ideas with the help of a product example from the area of mechanical engineering.

Keywords

Design Process, Design Methodology, Design Process Model, Functional Model, Physical Principle Model, Product Requirements, Dynamic Model

T. Tomiyama et al. (eds.), *Knowledge Intensive CAD*

1 Introduction and Overview of the design process

A general design process describes the characteristic and fundamental steps in the process of solving multiple and diverse design tasks in a product-independent manner by giving a methodology to reach the intended design goal. Many systematic approaches has been developed by design methodologists during the years. Common to all is that they start the going forward in the design process at *clarifying the product requirements*, going on by describing the *product functions* to find *principle solutions* and end at the *embodiment* stage (Koller 1985, Pahl and Beitz 1994, Roth 1994) a. o. According to this the basics of a general design process can be distinguished and defined by four modeling layers (see Figure 1). The design starts at modeling the product requirements and ends at the embodiment layer; in doing so the result (output) of a modeling layer serves as a requirement (input) for the following layer. Depending on the design task the modeling layers will be processed *completly*, *particularly* or *multiple iteratively*.

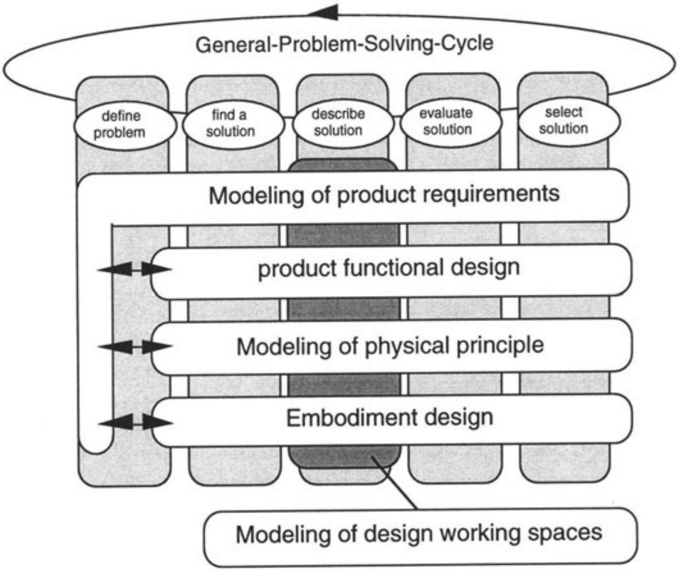


Figure 1 State of the art of the design process in mechanical engineering

The modeling layer of requirements serves for the computational projection of the results won by the clarification of the design task. This modeling layer contains the *preconditions of the design*, the *to-be properties of the future product* and is the starting point for the transition to the functional modeling layer.

Functional modeling layer serves for describing the product function and for finding design solutions. The functional modeling includes the specification of a *single product function*, the interrelationships between functions (called *function structure*) and the *functional flow* within a product.

After defining the product function the solution principle of a design will be determined. This stage covers all informations describing the *product's physical solution principles.* These informations contain the *physical effects* and its *structure,* the principle idea of the used *materials* and the *mathematical equations* describing the nexus. *Effektive geometry* as effective-lines, -faces and -spaces serve for the principle geometrical relationship. This stage is completed by assigning one physical principle to one already modelled product function of the functional modeling layer.

The shape modeling (or embodiment stage) is the most concrete of the product modeling layers ending up the design process by completing all geometrical design features, determins the three dimensional parts and structures it to asseblies.

Within every modeling layer of the design process a *general problem-solving-cycle* has to be performed (Figure 1). The problem-solving-cycle consists of basic and *elementary solution steps* which have been derived from the psychology of problem solving (Rutz 1985). First the designer is *confronted* with the given problem. Afterwards the *definition* of the essential problems is performed by fixing the objectives, main constraints and the environment for the intended solution. The next step is *finding* and *representing (representing)* a solution for the defined problem (this is the creative part of design). After that the solution has to be *evaluated* followed by making a *decision* for one alternative. Finally the problem-solving cycle is reiterated either in the same modeling layer or in the following one. In doing so the established solution serves as a requirement for the next problem. In this way, the intended design proceed from the *qualitative* to the *quantitative,* from the *abstraction* to the *concretion,* from the *uncomplete* to the *complete* and from *possible alternatives* to the *optimal solution.* This general problem solving cycle is together with the methodological and systematic approach of the design process building the basis for the development of our process model of design.

A *design-working-space* is an *Euclidean space* available for a designer *to solve* a given *design task.* The design-working-space is defined by a geometric system boundary and its constraints (in-/outputs). The fundamental idea of modeling design-working-spaces comes from system theory and therefore design-working-spaces are not limited to the geometric layer. A design-working-space groups design objects of all design stages (Figure 1), reduces the complexity of subtasks and *structures design process knowledge* to fix a special *design state.* The *main purpose* of *design-working-spaces* in this context of modeling the design processes is *to fix a special design state* (Grabowski, Rude, Lossack, 1995) so that new design states will be derived by applying so called process patterns (Grabowski, Lossack, Weis 1995); in doing so getting stepwise to the intended solution. In addition to this design-working-spaces support not only the design process but the solution finding of products by so called solution patterns (Grabowski, Rude, Lossack 1994) and the distributed problem solving (Lossack 1995).

Before stepping into more details of the fundamentals of the design process we direct some interest on a design (see Figure 2) serving as an example to explain our abstract design process model. It shows a robot gripper designed for handling small parts, for durability and for low maintenance costs. A standard connection to the robot arm was given as well as the space in which the gripper has to fit. The working method of the gripper is as follows:

The force with which the handled part is gripped is generated by an pneumatic energy source. The resulting force is then transmitted through a pistion rod to a wedge splitting the force into two resulting forces which are applied to the gripper´s jaws. The applied pressure causes a movement of the piston rod towards the jaws and therefore the wedge causes a turning motion of the jaws which results in the gripping force of the robot gripper. The detachment of the handled product is relized by a spring (not depicted) and by reducing the pressure applied on the piston. So the spring pushes the piston back to it's original position.

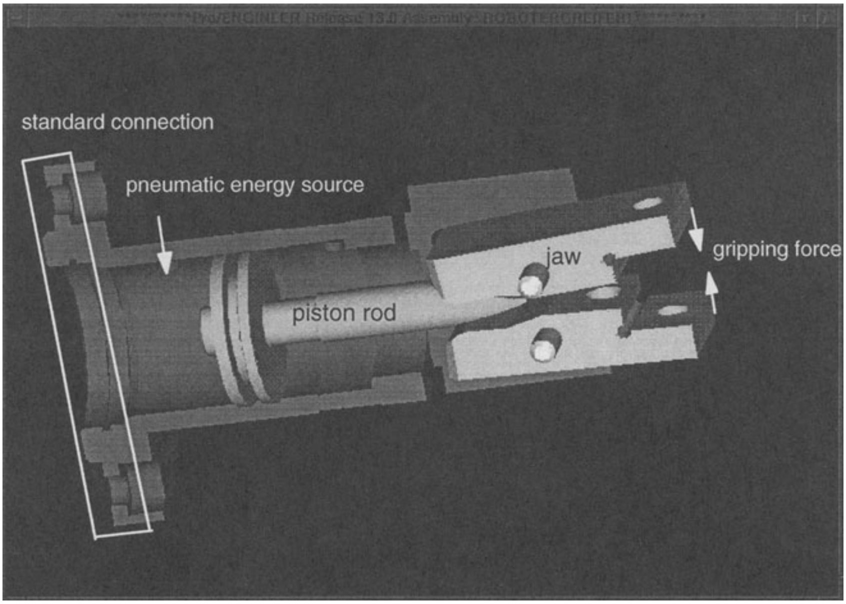


Figure 2 Robot gripper serving as a product example for the intended design

2 Statical fundamentals of the Design Process for technical systems

With the three fundamental magnitudes of design, *matter*, *energy* and *information* every technical system or artifact can be described on an abstract physical level (Roth 1994, Pahl and Beitz 1994). Information determines what has to be done to fulfill a certain purpose. Only with energy technical systems are able to perform any change in nature or in itself. Matter is the stuff, a technical system consists of. It is also the medium in which every process takes place. The human himself is the best example for a technical system. In the first instance he used tools to integrate it in his own technical system, later on he created tools themselves (that means he controlled the matter), in the last centuries he learned to control the energy and in our century, he ruled over the information processes (cybernetics). With respect to being so fundamental every design theory has to be based on this three categorical magnitudes, matter, energy and information.

Technical artifacts are connected to the environment by means of *inputs* and *outputs* and can be treated like a system. A *system* can be divided into *sub-systems*. What belongs to a particular sub-system is determined by the *system boundary*. With this approach it is possible to describe every technical system at every stage of *abstraction*. Describing a proposed technical artifact[1] by means of a system consisting of elements which are grouped by the system boundary related with each other by input and output we use the term *function* or *product function*. If the product function is described on the basis of matter, energy and information as inputs and outputs then we use the term *general function (GF)*. If the inputs and outputs represent physical magnitudes like force or torque and the relationship between input and output is not yet known, then we use the

---

[1] The solution is not yet known

term *special function (SF)*. If the inputs and outputs represent physical magnitudes and the relationship between input and output is given by a physical law, then we use the term *physical principle (PP)*. In the case of a GF the relationship between the in- and output is expressed by a limited number of so called *function verbs*. The function verbs describe the proposed transformation between the in- and output. With reference to (Roth 1994) we use the set of function verbs *Change*, *Connect*, *Channel* and *Store* for the GF. All technical artifacts are complex constructions, so this complexity of an artifact has to be modelled in a network of *GFs and SFs*; in doing so we talk about *general, special function and physical principle structures (GFS, SFS, SPP)*. Because of introducing the above function types and its particular structure the fundamental working principle of *abstraction* is applicable and therefore a *top down approach to the design process* is possible.

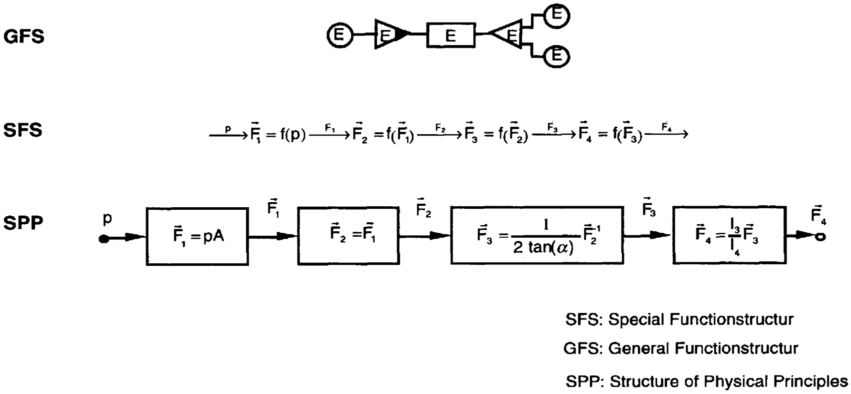


Figure 3 Established General Function Structure, its derived Special Function Structure and Structure of Physical Principles

Figure 3 shows the established General Function Structure (GFS) of the robot gripper, its derived Special Function Structure (SFS) and Structure of Physical Principles (SPP). First there is stored energy depicted by the symbol of a circle containing the character "E". This energy will be changed in another form of energy. On the SFS level there is shown that the energy type of pressure will be changed in the energy type of force. After that the energy will be channeled, distributed and amplified. At the bottom of the figure the physical principles which perform this process are shown.

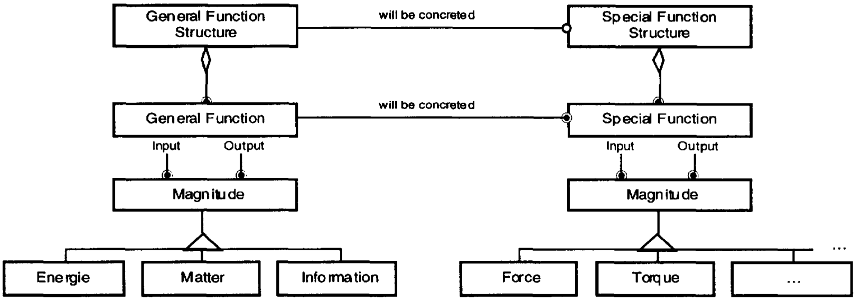


Figure 4 Conceptual Object Model of the design objects General and Special Function and its Structures

The knowledge about the general functions and special functions and the interrelationship between the different levels of abstraction (design stages) is modeled in a conceptual *object model.* Figure 4 is intended to give an idea about the relationships between the design objects of the conceptual model. In this object model the design knowledge is instantiated. This means the model contains all information which the designer describes for the intended product. For that reason the model contains information belonging to all design stages. If all the information described above is contained in the product model on which a design system is based any technical product can be modeled by this system.

3 The modeling space of design

So far we talked essentially about statical facts, statical object modells or resp. about information modeling in generall. These informations are necessary because we have to know about the *what* the intended product consists of. For describing and respresenting the design process statical object information is not enough. So dynamic informations have to be added and represented in a so called dynamic design model (Grabowski, Rude, Lossack 1995).

Building up a dynamic design model the first step is to collect possible design activities. In the past we have had found 3 fundamental pairs of design activities and structured it in the *modeling space of design* (Grabowski, Lossack, Weis 1995). So the modeling space of design includes all *possible design directions (activities) to design any technical artefact* (Figure 5). The modeling space of design is called *space* because every design direction is represented in a 3-D coordinate system and we say design *direction* because we begin at any starting point in the design process going in one of the six directions; in doing so we navigate a designer through the design process. The pairs of design (solution) directions are *abstraction-concretion*, *combining-detailing* and *variation-limiting*.

Based on any solution state SS_i, different solution directions can be followed in order to reach a following solution state SS_{i+1}. In the following we understand by the term "*solution state SS_i*" the instantiation of the product model belonging to the intended product after the i-th design step. This contains all informations in the product model acquired by performing i design steps. Figure 5 shows this procedure. Starting at any solution state of a design task the solution directions showed can be followed. These solution directions derived by Birkhofer (1980), Krumhauer (1975) and Rude (1991) describe a possible way to transform a solution state SS_i into a following one SS_{i+1}. In general this means to be one step closer to the intended solution.

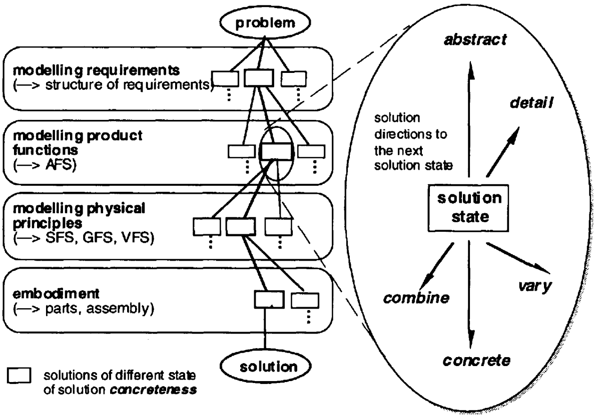
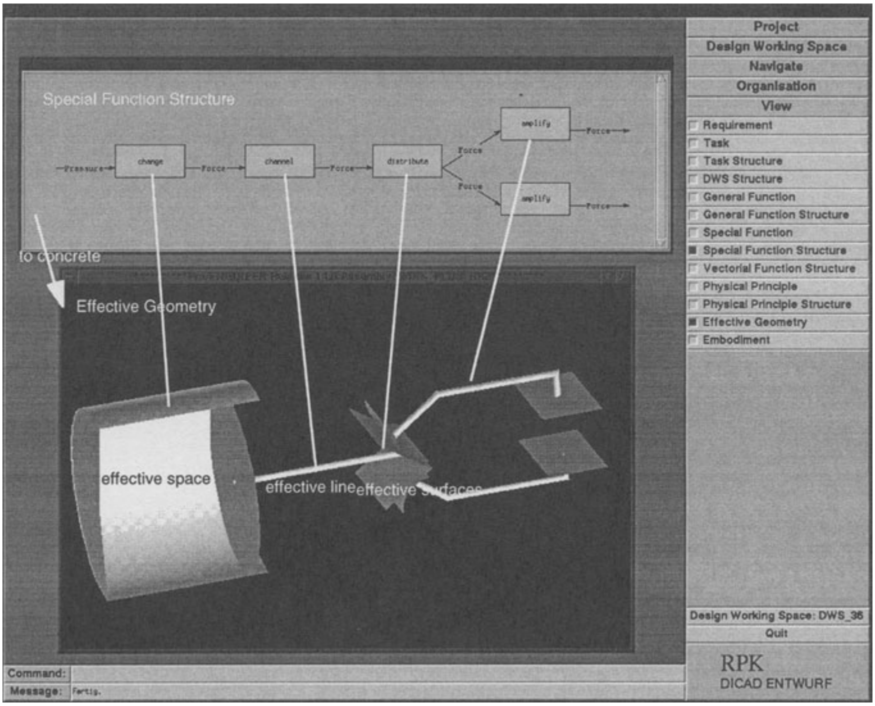


Figure 5 Elementary solution steps in the design process

CONCRETION-ABSTRACTION

The transformation of a solution state SS_i into a following, more concrete one SS_{i+1} is called *"concretion"*. Here we understand the instantiation of the product model with information belonging to a more concrete modeling layer (see Figure 6). In this way new solution properties will be added to the solution state SS_i. The example shows the transformation *concretion*, which maps the set of product functions of the robot gripper onto a set of effective elements like effective-spaces, -surfaces and -lines.

The product function which describes the transformation of pressure into force (physical effect: $F_1 = pA$) is mapped onto a effective space (pneumatic cylinder). The transmittance of force (physical effect: $F_2 = F_1$) is realized by an effective-line (bar), which is connected with two effective-surfaces (wedge) to split the force F_2 into two forces F_3 changing its directions (physical effect: $F_3 = (1/2 \tan(a)) F_2^{-1}$) at the same time. F_4, which is the output of the main function of the robot gripper is obtained by the lever-effect (Figure 3 and 6).



☐ Requirement
☐ Task
☐ Task Structure
☐ DWS Structure
☐ General Function
☐ General Function Structure
☐ Special Function
■ Special Function Structure
☐ Vectorial Function Structure
☐ Physical Principle
☐ Physical Principle Structure
■ Effective Geometry
☐ Embodiment

Design Working Space: DWS_36

Command:
Message: Fertig.

Figure 6 Example for the solution direction "concretion"

The opposition of concretion is *abstraction*. Abtraction is the transformation of the solution state SS_i into a more abstract solution state SS_{i+1} with respect to the intended design solution. Abstraction serves for recognition of the essential product properties. In consequence a more abstract solution state SS_{i+1} is one step further away from the intended design solution (see Figure 7). This step is used if one starts from a well known design to reach a new one to get new ideas or to get until then not known solutions. In this context Figure 7 shows the abstraction of the shape of a robot gripper's jaw onto its effective structure.

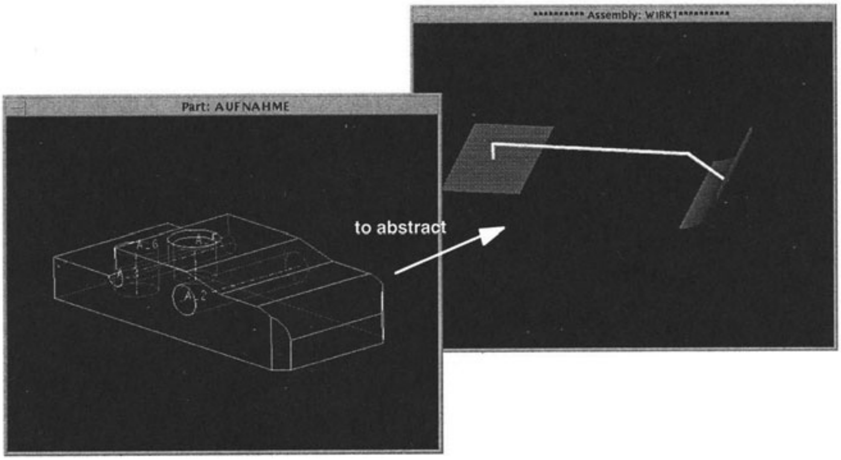


Figure 7 Example for the solution direction "abstraction"

Starting from the shape of the robot grippers's jaw, the essential features of the design are extracted. These are the two effective surfaces S_1 and S_2 where S_1 is responsible for the transmittance of a force onto the lever, which can also be extracted from the shape. S_2 is the effective surface which can be recognized as being responsible for applying the gripping force onto the handled part. The function of the whole in the middle of the jaw's body acts as a bearing which also finds it's counterpart in the effective structure. By that the shape description of the robot gripper's jaw is abstracted to it's effective structure. The effective structure can serve as the basis for a variation of the shape or it can be abstracted by itself in order to obtain another effective structure with other design properties.

DETAILING-COMBINATION

Adding more information to a design object ***within the same modeling layer*** is called ***detailing.*** When detailing a solution state SS_i to a following SS_{i+1}, the modeled design information remains on the *same level of abstraction* as it was in state SS_i. The solution direction detailing is used in order to solve a design problem by dividing it into sub-problems. The rough effective structure of Figure 3 could be detailed by example by adding two bearings to the levers or by adding a sealing to the piston inside the pneumatic cylinder. With all the information contained in the detailed form it is easier to concrete (maybe after performing other detailing steps) the effective structure to the robot gripper's shape model.

The opposition of detailing is *to combine.* It transforms a solution state SS_i into a more *general one* SS_{i+1}. This transformation step also remains on the same abstraction level. One example for combination is to summarize different sub-functions into an overall function or to omit different design objects in an effective geometry sketch in order to find the basis for a better variant of the intended product. A design on a certain level of abstraction is combined until the information which is essential for the next design step (abstraction or variation) is left.

VARIATION-LIMITING

Variation means to find a corresponding solution, in state SS_i of the intended design, other eventually better alternatives (solution state SS_{i+1}). Variation steps back to the preceding solution state SS_{i-1} and concretes this solution state to SS_{i+1}. SS_{i+1} contains design properties different from the one's of SS_i. Variation keeps the intended design's degree of concretion unchanged. What changes, is the spectrum of possible solutions on the same level of abstraction where the design is in state SS_i. As an example for the here described solution direction, the variation of a physical effect corresponding to a certain sub-function, can be mentioned. Variation of the function "to generate energy" into two variants. One generates force by applying pressure p ($F_1 = f^1(p)$), the other variant obtains force by a spring ($F_1' = f^{1'}(D)$), where D is the elasicity constant of a spring.

4 Modeling in design working spaces

In the product model there is still a lack of mechanisms which helps to structure the designed product in order to reduce the complexity of subtasks. An assembly is a structured set of parts only on the geometrical level. With respect to this problem and also focusing on the administration problem of subtasks and structuring knowledge of the design process we developed the concept of "design working spaces" which is described in this paragraph.

In this context the *main purpose of design-working-spaces is to structure design process knowledge* which is represented by design process patterns. Design process patterns support the designer by proposing to him a special design direction (design state transition), like abstraction or decomposition. Whether it should be necessary to concrete or to decompose a special design problem (task) depends on the actual design state. The *actual design state is fixed by the design-working-space.*

A *design-working-space* is an *Euclidean space* (on geometric level, Figure 8) which is available for the designer *to solve* his *design task.* The design-working-space is defined by an envelope (geometric system boundary) and its constraints (in-/outputs). The fundamental idea of modeling design-working-spaces comes from system theory and therefore design-working-spaces are not limited to the geometric. The *main purpose* of *design-working-spaces* in this context of modeling design processes is *to fix a special design state.* If a special design state has been fixed it is possible to derive new design states getting stepwise to the intended solution.

Design-working-spaces are *defined* and will be builded up by the following rules:

- A design-working-space consists of a set of elements and of a set of relationships between the elements.
- Elements of a design-working-space are informations of the design stages (Figure 1), like requirements, product functions or physical principles. Relationships between the elements are general or special magnitudes like energy, information, matter or force, torque etc.
- Every design-working-space can be subdivided in independent sub-spaces. If elements of different sub-spaces will be grouped together then this sub-spaces are called overlapping design-working-spaces.
- Every design-working-space and every sub-space is defined by a system boundary. The system boundary is specified by its envelope, making available the maximum of geometric space for designing, and its effective geometry. The effective geometry defines the point at which the physical event (phenomena) takes place.
- A system boundary of a design-working-space has one or more in-/outputs.

- If a design-working-space has no in-/outputs then we talk about a closes design-working-space, on the other hand about an open design-working-space.

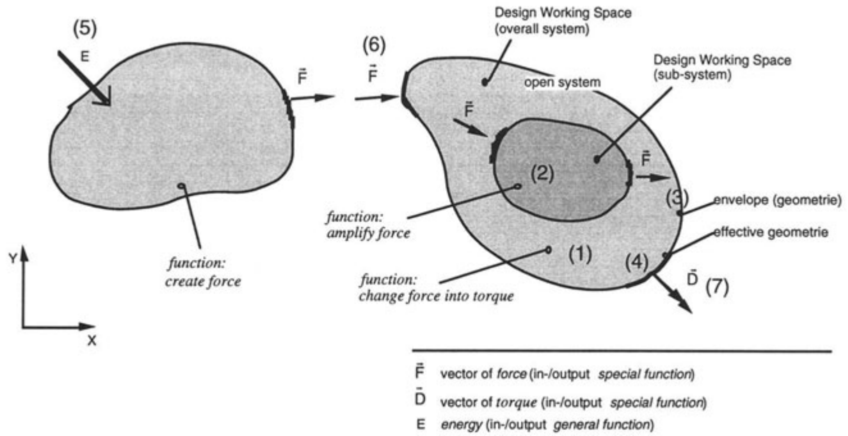
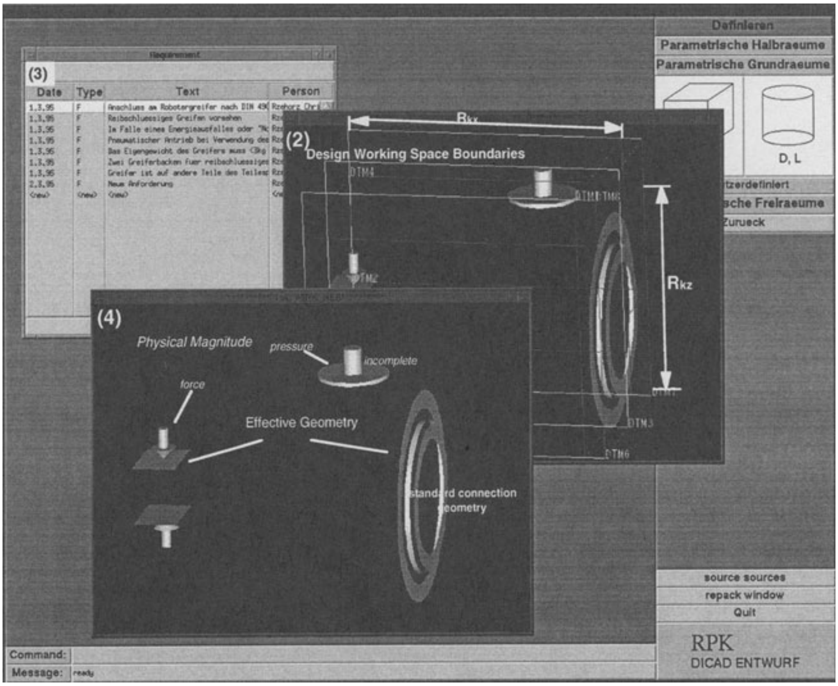


Figure 8 Basic concept of design-working-spaces

In Figure 8 there are three design-working-spaces which have to fulfil a special product function, like *change force into torque* (1) or *amplify force by force* (2). The system boundary of a design-working-space is clearly defined by its *maximum envelope* (3) and *effective geometry* (4) and by the *physical in-/outputs* like the physical magnitudes force F and torque D; the envelope and the effective geometry is represented by free form surfaces. The envelope describes the maximum space inside which a special problem has to be solved. The effective geometry is described by *effective spaces* and *effective surfaces* which transmit for example forces. The relationship between the design-working-spaces is established exemplary by the general magnitude *energy* (5) and the special magnitudes *force* (6) and *torque (7)*.



Date	Type	Text	Person
1.3.95	F	Anschluss an Robotergreifer nach DIN 49	Rzehorz Chr
1.3.95	F	Reibschluessiges Greifen vorsehen	Rz
1.3.95	F	Im Falle eines Energieausfalles oder "N	Rz
1.3.95	F	Pneumatischer Antrieb bei Verwendung des	Rz
1.3.95	F	Das Eigengewicht des Greifers muss <3kg	Rz
1.3.95	F	Zwei Greiferbacken fuer reibschluessiges	Rz
1.3.95	F	Greifer ist auf andere Teile des Teiles	Rz
2.3.95	F	Neue Anforderung	Rz
<new>	<new>	<new>	<ne

Figure 9 Design-working-space derived from the requirement list

Figure 9 shows a design-working-space which is directly derived from the requirement list (3). This requirement list contains information of product requirements which have been aquired from the customer and formalized. The design-working-space for the robot gripper is defined by its connection geometry (4), which connects the robot gripper with the robot arm, by its envelope (2), defining the maximum mechanical designing space, by its overall function "to grip" and by its effective geometry (4) to handle the products. The effective geometry is connected with physical magnitudes (4). The physical magnitudes are either geometrically defined, like the gripping force F, or geometrically undefined, like the energy supply, which will be geometrically defined later.

Design-working-spaces will be defined by half-spaces, by prismatic spaces or by free form surfaces. The design-working-spaces of the example are basic prismatic spaces.

For the distributed problem solving there are needed operations for the decomposition of design-working-spaces, to solve sub-tasks of sub-spaces in parallel and to relate special problem solver to a sub-space. A problem solver can be a human beeing (designer) or a machine.

In Figure 10 there is shown the design-working-space exemplarily decomposed into two sub-design-working-spaces. The physical constraints on the system boundary of the new decomposed sub-design-working-spaces will be inherited from the super-design-working-space, in this way generating a complex network of relationships.

The two sub-design-working-spaces $R_{k1.2}$ and $R_{k1.1}$ are related with each other geometrically by the length of its edges and physically by its physical magnitudes $[(L_{1.2}<L_1 - L_{1.1}), \dots, (L_{1.1}<L_1), \dots ; F_1]$ (for abbreviations please see also Figure 13).

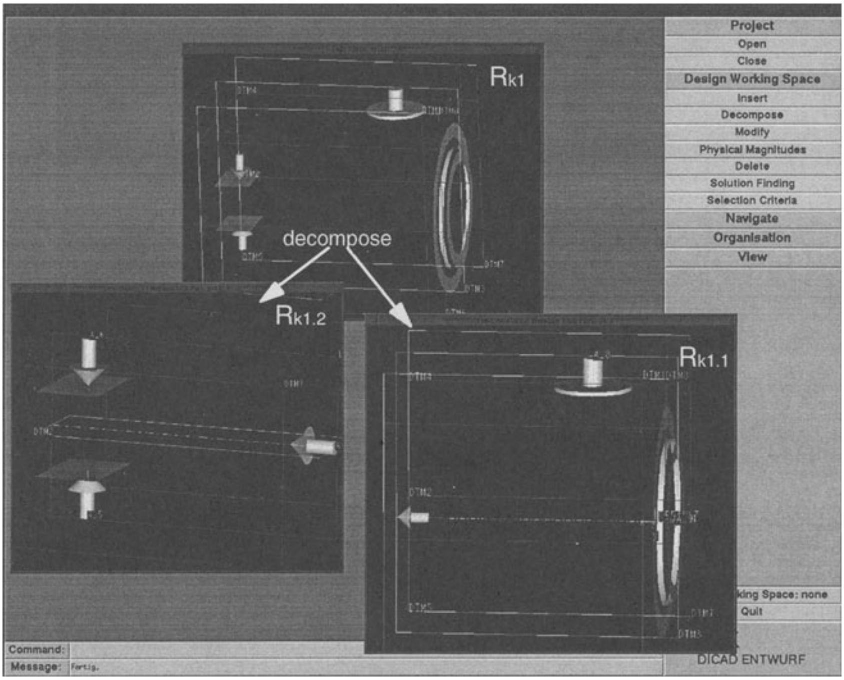


Figure 10 Decomposition of a design-working-space in two sub-spaces

For the decomposition of prismatic design-working-spaces there is generally value the following equation:

$$\sum_{d=1}^{3}\sum_{i=1}^{n} R_{k_{id}} \leq R_{k_d};$$

d: dimension of the spaces {x, y, z}
n: number of sub-spaces

The geometrical relationship means that the sum of lengths of the sub-space's edges has to be less than or equal to the maximum of the length of the edges of the super-space R_{k1}. The relationship $L_{1.2}<L_1-L_{1.1}$ is a directed one meaning that the length L of the sub-space $R_{k1.2}$ is dependent of the length L of the sub-space $R_{k1.1}$. The *directed dependency* of that relationship results directly from the sequence of problem solving which is determined by *general design rules* resp. process control strategies.

Process control strategies will be described in the next paragraph, so that for the explanation of the relationships and constraints between the design-working-spaces it should be enough to know that there is a given control strategies: "design along the flow of force, begin at the energy supply (resp. system input)".

Every solution depends on the solutions already established defining a kind of ranking; in doing so the first solution has always the highest priority. The determined sequence of problem solving leads to the fact that the execution of operations on the $R_{k1.2}$ depends upon the state of $R_{k1.1}$ resp. the relationships and constraints between $R_{k1.2}$ and $R_{k1.1}$. For example, if operations will be executed on $R_{k1.2}$ changing the physical input constraint F_1 it will come into conflict with $R_{k1.1}$[2] because $R_{k1.1}$ has a higher priority than $R_{k1.2}$.

As mentioned above in this context the main purpose of design-working-spaces is to structure the designed product in order to *reduce the complexity of subtasks*, to *structure design process knowledge* and to fix special *design states* so that new design states can be derived getting stepwise to the intended solution. To derive new design states it is necessary to represent this knowledge in an appropriate model which will be done in the design process model described in the next paragraph.

5 Design Process Model

Depending on his experience and skill a designer choses an appropriate path to perform the necessary design steps depending on the actual state of the design. Today's CAD systems do not offer any support in finding the next design steps which should be performed in order to reach the design goal. For this problem we present a concept for the realisation of a knowledge based design-process-control system for CAD. This process control is a component of an intelligent design system which pre-selects the most promising of the next possible design steps. It also aids the designer in selecting the next design step which should be performed in order to solve the design problem in the (hopefully) most efficient way. In this context for a better understanding of the concept presented in the following a definition of the term "design step" is given:

A design step Ds is the combination of a non-empty set of modeling commands C_j which transforms a design object O which is in the solution state SS^O_i into the following solution state SS^O_{i+1}.

$$Ds := \{C_n: S_j\ (C_j\ (SS^O_i) = SS^O_{i+1}\}$$

According to this definition, a design step is the action which solves a given sub-problem of a design task according to the requirements specifying the task.

The design process control of an intelligent CAD system serves to support the designer to navigate through the design process on an optimal route towards the intended design. This means to find an optimal sequence of design steps which lead from the requirements specification to the description of the found solution.

This process control must be based on the description of the designer's work. Figure 11 shows the conceptual architecture of the design process model consisting of the three components: dynamic design model, process control strategies and process patterns. In the following we describe these three components.

---

[2] F_1 is physical output of $R_{k1.1}$

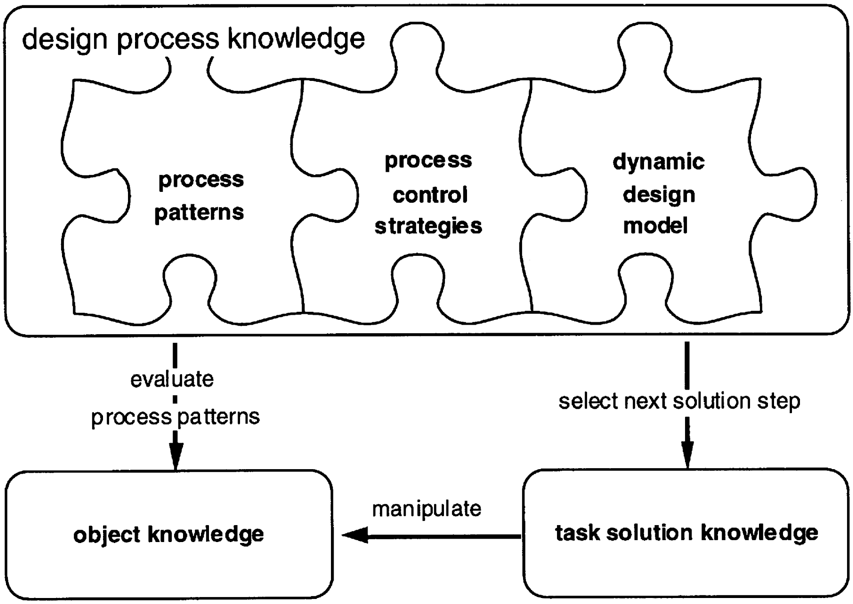


Figure 11 Overview of the process model architecture

DYNAMIC DESIGN MODEL

The *dynamic design model* describes the *dynamic behaviour of the design objects*. The *dynamic behaviour* is described by the *states* which a design object can adopt together with *state transitions* between these states as well as *actions* causing those transitions and the *conditions* under which they are performed (Grabowski, Lossack, Weis 1995). *Design objects* are *objects* of the *design stages*, like General Function, Special Function etc., and will be grouped in design-working-spaces. There are five different states a design object can adopt. These states are:

1) defined,
2) detailed,
3) varied,
4) combined and
5) evaluated

A design object can not be in any other state but the five above. Figure 12 shows as an example the dynamic model of the object General Function.

As the General Function specification layer is the most abstract layer in the concretion hierarchy (for a more detailed description see Grabowski, Lossack, Weis 1995), the design starts with the description of the General Function of the intended product. When the product requirements are specified the object General Function (GF) is in its initial state. By initiation the object GF changes its state from the *start state* to the state *defined*. The two *transitions* starting at the state *defined*, symbolize that there are two possibilities for the next transitions. One is the transition to the state *varied*, the other is the transition to the state *detailed*. The transition to the state *varied* is marked with the *condition* [GF.detailed ≠ NULL]. This means, this transition is only then performed if the instantiation of the object General Function has already been detailed before.

In consequence, if the General Function has not been detailed yet, the transition to the state *detailed* is accomplished. The new state *detailed* is then reached by the action *detail.* After detailing, the results of the action must be evaluated (how the evaluation is performed will be described in the following). By this the state *evaluated* is reached. The evaluation calculates a result (evaluation.result) which describes the quality of the found solution which was gained by the design step currently performed. The result of the evaluation is then compared with a given limit. If the result is better than the limit, the final state the object General Function is reached. This means that the design for the General Function is accepted. Here a message is sent in order to cause the sending of the message *concrete* to the object Special Function (SF). This means, the General Function object is concreted to the corresponding special function.

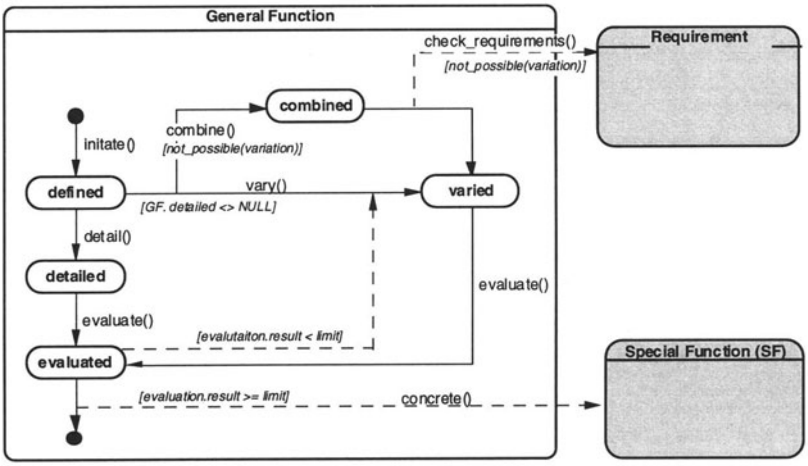


Figure 12 Dynamic model of the object "General Function"

If the result is worse than the limit, a message is sent to the transition *vary*. This message causes the execution of the variation of the General Function object. In our model, the transition *vary* is modelled as an alternative transition. This means, if the intended variation can be performed, the state *varied* will be reached. Then the properties of the object will be evaluated in analogy to the object in the state detailed. If the variation cannot be executed (symbolized by the condition [not_possible(variation]), the alternative transition *combine* is started. This alternative causes the combination of the object so that the object will reach the state *combined.*

Following this methodology, the dynamic behaviour of the design objects describing a product in all design stages have been modelled. In our process model, the *dynamic design model* is responsible for low level tactics dealing with *finding the next command* which should be applied to a selected design object. The dynamic design model cannot be used for chosing the next design object if another object is finished. This is the problem the second layer.

PROCESS CONTROL STRATEGIES

The process control strategies are heuristics describing a path which lead to the solution of a design task. Because of their task-independencies, process control strategies form the basis for a process control of an intelligent CAD system. This means they describe independently from the specific design task a path through the design leading to the intended task solution. The knowledge which is necessary for this task is based on the general design methodology. As examples for such a design strategy "design along the flow of force" or "design along the main functions" can be mentioned. The problem solving starts at:

1) the system input,
2) the system output or
3) the system in- and output at the same time.

Because the design strategies are an important part of our design process model we explain in the following the first strategy mentioned above. The design example is the robot gripper.

Applying the strategy "design along the flow of force, begin at system input" leads to begin the design at the input of pressure given in the requirements specification. First the function "change pressure into force" then the next function "channel force" have to be modelled (see also Figure 6). The modeling of these two functions leads to the division of the first design-working-space into two sub-spaces (see Figure 13).

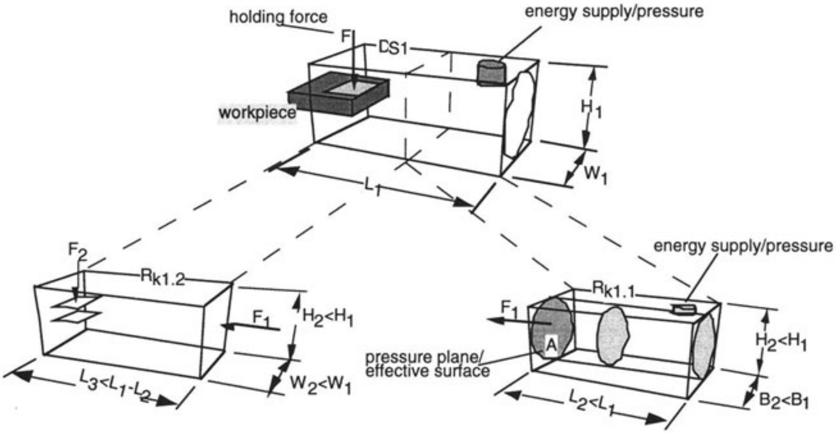


Figure 13 Example for the design strategy: "design along the flow of force, begin at the system input"

It is easy to see that these general, task independent strategies can show any path leading to the solution of the design task. But this path usually will not be optimal. In this context the term optimal means to design a product which meets optimally the specified requirements applying the least possible design steps. Finding such an optimal path it is necessary to have knowledge about optimal design processes depending on the respective design task. For that reason we developed a concept for the acquisition and retrieval of process patterns.

PROCESS PATTERNS

Process patterns describe the proceeding of a designer when solving a certain sub-task of a given design task. The structuring and decomposition of the design task into sub-tasks is performed by the help of design-working-spaces. In addition to this by design-working-spaces all product describing informations will be assigned to the respective sub-task. Therefore design-working-spaces can be applied to store the current solution state of a design. This means if we fix the solution state of a design by design-working-spaces, we have to protocol the modeling commands which the designer performs. The result can then be stored as a so called process pattern. Figure 14 gives an idea of the information model of process patterns.

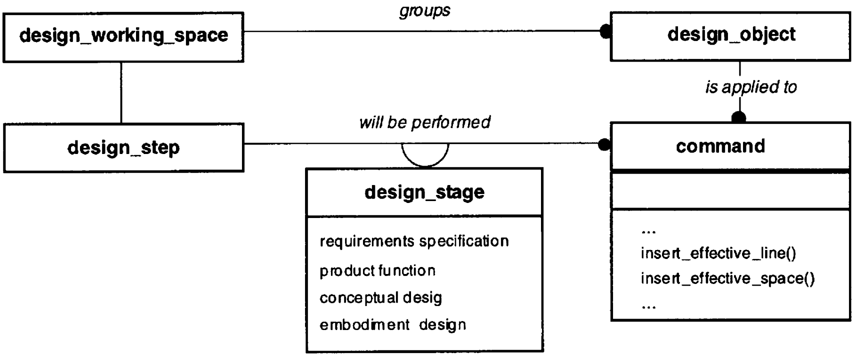


Figure 14 Information model of the process pattern

A design step is directly connected to a design-working-space. The design step is performed by multiple commands which are each applied on a design object. The working out of a command depends on the respective design phase. The design phase can be one of the following: Requirements specification, functional modeling, conceptual design or embodiment design.

This information model builds the basis for the process patterns. Based on the concept described above the process patterns can be acquired by the design system from the designer's work. By a search for similar design-working-space, the process patterns can be retrieved and applied to the new design.

6. Conclusions and future work

We have modeled and verified the dynamic model in a small application on the DIICAD product model. Modeling design solutions in design working spaces and saving these design solutions as solution patterns in the product model is possible. With design-working-spaces we find similar solution patterns for a given problem. This is realized by using a case-based-reasoning approach which is implemented on KEE and ACIS. At the moment we are able to do this for the requirements modeling (Kläger 1993) and the functional (Huber 1994) design stages in a top down approach and for solution patterns (Suhm 1993) described in the mentioned Phd thesises of Kläger (1993), Suhm (1993) and Huber (1994).

Modeling the design process in the described way is a promising approach. We have modeled the product life cycle and verified the approach in a small prototype. In the SFB346 (this is a special research area which is set up at the University of Karlsruhe by the German Research Community) a language has been developed to describe dynamic models. We consider it an important point that in the future basic research has to be done in developing a methodology to build dynamic models. Our next step will be to implement our concepts; the process model as a whole (dynamic design model, process control strategies and process patterns).

References

Benz T.: 1990 *Funktionsmodellierung als Basis zur Lösungsfindung in CAD-Systemen (Functional Modeling as Basis for the Solution Finding in CAD Systems)* Universität Karlsruhe (TH).

Birkhofer, H.: 1980 Analyse und Synthese der Funktion technischer Produkte (Analysis and Synthesis of the Function of Technical Products), Dissertation TU Braunschweig.

Grabowski, H, Lossack, R.-S., Weis, C. 1995 *Supporting the design process by an integrated knowledge based design system,* contribution to the IFIP WG 5.2 workshop on Formal Design Methods for CAD, held in Mexico City, 1995.

Grabowski, H; Rude, S.; Lossack, R.-S.: 1994 *Konstruktionsarbeitsräume als Benutzungsschnittstelle für die Modellierung von und mit Features (Design-working-spaces as user interface for modeling features),* Workshop "Feature Technology in Design and Manufacturing", 29.-30. August 1994, Universität des Saarlandes

Grabowski, H; Rude, S.; Lossack, R.-S.: 1995 *A contribution to research directions for artificial intelligence in design,* in: Fourth Workshop on Research Direction for Artical intelligence in Design; John S. Gero (ed.), Fay Sudweeks; Key Centre of Design Computing

Huber R.: 1994 *Wissensbasierte Funktionsmodellierung als Grundlage zur Gestaltsfindung in Konstruktionssystemen /(Knowledge based functional Modeling as Basis for the Shape Computing in Design Systems),* Aachen: Shaker, 1994 (Reihe Konstruktionstechnik), Zugl.: Karlsruhe, Univ., Diss., , Institut für Rechneranwendung in Planung und Konstruktion (RPK).

Ken Wallace, Nigel Ball, Ming-Xi Tang: *AI in mechanical engineering design,* John S. Gero, F.Sudweeks (eds.), Fourth Workshop on Research Directions for Artificial Intelligence in Design, University of Twente, Enschede, The Netherlands, 6. January 1995.

Kläger R.: 1993 *Modellierung von Produktanforderungen als Basis für Problemlösungsprozesse in intelligenten Konstruktionssystemen (Modeling of Product Requirements as Basis for Problem Solving Processes in intelligent Design Systems)* Aachen: Shaker, (Reihe Konstruktiostechnik), Zugl.: Karlsruhe, Univ., Diss., 1993, Institut für Rechneranwendung in Planung und Konstruktion (RPK).

Koller, R.: 1985 Konstruktionslehre für den Maschinenbau (Design Theory for the Mechanical Engineering), Springer-Verlag.

Krumhauer P.: 1974 Rechnerunterstützung für die Konzeptphase der Konstruktion (Computer Support for the Conceptual Phase of Design), Dissertation TU Berlin.

Lossack R.-S.: 1995 *Neue Konzepte in der Verteilten Produktenwicklung: Modellierung von Konstruktionsarbeitsräumen (New concepts in distributed product development: modeling of design-working-spaces),* contribution to the SFB346 workshop on Innovative Product Development, held at the University of Karlsruhe, Germany, 1995.

Pahl, G.; Beitz, W.: 1994 *Engineering Design,* Springer-Verlag.

Roth, K.-H.: 1994 *Konstruieren mit Konstruktionskatatogen,* Springer-Verlag.

Rutz A.: 1985 *Konstruieren als gesdanklicher Prozeß (Design as Intellectual Process)* KM Lehrstuhl für Konstruktion im Maschinenbau TU München, Dissertation.

Suhm A.: 1993 *Produktmodellierung in wissensbasierten Konstruktionssystemen auf der Basis von Lösungsmustern (Product Modelling in Knowledge Based Design Systems on the Basis of Solution Patterns)* Aachen: Shaker, (Reihe Konstruktionstechnik), Zugl.: Karlsruhe, Univ., Diss., 1993, Institut für Rechneranwendung in Planung und Konstruktion (RPK).