

## **Combining Engineering Design Methodology and Case-Based Reasoning**

Mehmet H. Göker, M.S.E., M.Sc.

Prof.Dr.-Ing. H.

Birkhofer

Technische Hochschule Darmstadt

Maschinenelemente und Konstruktionslehre

Magdalenenstr.4

D-64289 Darmstadt - Germany

Telefax : (+49) 6151 163355

[goker@mars.muk.maschinenbau.th-darmstadt.de](mailto:goker@mars.muk.maschinenbau.th-darmstadt.de)

[birkhofer@venus.muk.maschinenbau.th-darmstadt.de](mailto:birkhofer@venus.muk.maschinenbau.th-darmstadt.de)

# Combining Engineering Design Methodology and Case-Based Reasoning

## Abstract:

Engineering Design Methodology (EDM) is a formal approach to design. It provides methods and structure and can be used as a design grammar. By combining EDM and Case-Based Reasoning (CBR), systems that support engineers performing conceptual design tasks, can be implemented. In this article we describe the preliminary system architecture of a system that supports the design engineer both in terms of methodology and experience.

## Keywords:

Case-Based Reasoning, Engineering Design Methodology, Systematic Design, Conceptual Design, Mechanical Engineering, Experience.

## 1. Introduction

One of the premises of Case-Based Reasoning (CBR) is that human beings do not reason from first principles while they are performing routine tasks, but rely extensively on their memory of past experiences [RiSch89, Ham89, Ko90]. When human beings talk, or perform simple calculations, they do this without utilising the underlying methods and structure (grammar) of the domain. These are used only, when we are faced with an unknown situation where we do not already know the solution, i.e. when we are novices in the domain [KoJo89, Mü91].

Engineers behave in a similar manner while designing. If the problem at hand is a familiar one, they solve it directly. Only novices, or experts faced with a problem they do not have experience with, use the methods of the domain extensively.

The underlying methods and structure form the grammar of a domain. A grammar provides us with means to analyse old, and to generate new cases (experiences) in the domain. For example English grammar provides us with tools to both analyse an old, and to generate a new English sentence. The "grammar" of arithmetic provides us with tools to analyse a computation and check if we calculated correctly, and to perform "harder" calculations.

We propose that Engineering Design Methodology (EDM) [PaBe93, VDI2221, VDI2222] can be used as a grammar for engineering design, and that by combining experience in the form of cases and the grammar provided by EDM, systems that assist human engineers in performing design tasks can be realised.

## 2. Engineering Design Methodology (EDM)

EDM is a formal approach to design. Its aim is to provide a theory to the engineer with which he can systematically generate and evaluate solutions to a design problem in an objective and effective manner. EDM tries to make design teach- and learnable, and to ensure that engineers arrive at optimal solutions not by chance, but by methods.

In EDM, technical objects are defined to be technical products or the preliminary stages of these. Technical objects are described and identified by means of their properties in the form of attribute-value pairs [Bi80]. Similar objects have common attributes but differ in the values of these, whereas different objects differ in their attributes and thus have different properties.

EDM provides methods and structure for design and divides the design process into following phases:

- Task clarification
- Conceptual design
- Embodiment design
- Detail design.

Figure 1 shows the phases and the result of each design phase as described in PaBe84, PaBe93, VDI2221 and VDI2222. At the end of every phase a decision has to be made whether the next phase can be taken or if previous phases have to be repeated.

### Task clarification

The design process starts out with the clarification of the task. During this phase information about the requirements to be met are collected. The result is a detailed requirements (specification) list. The required properties of the technical object are described in attribute-value/range pairs. The requirements list is basically a description of the ideal object.

### Conceptual design

From the description of the ideal object, functions that the object has to realise are derived, and a function structure is generated. Then, solutions for each of these functions are generated or found (see Fig. 2). By combining these solutions a vast amount of concept variants is generated. In order to decide which one(s) of these concept variants should be developed further an evaluation has to be made. For this purpose the variants that do not satisfy the criteria of the requirements document are eliminated first. The remaining ones are then evaluated against technical and economic criteria derived from the requirements definition document i.e. the ideal object [VDI2225, PaBe93, BiGö92]. The best variant is proposed as a solution concept to the problem.

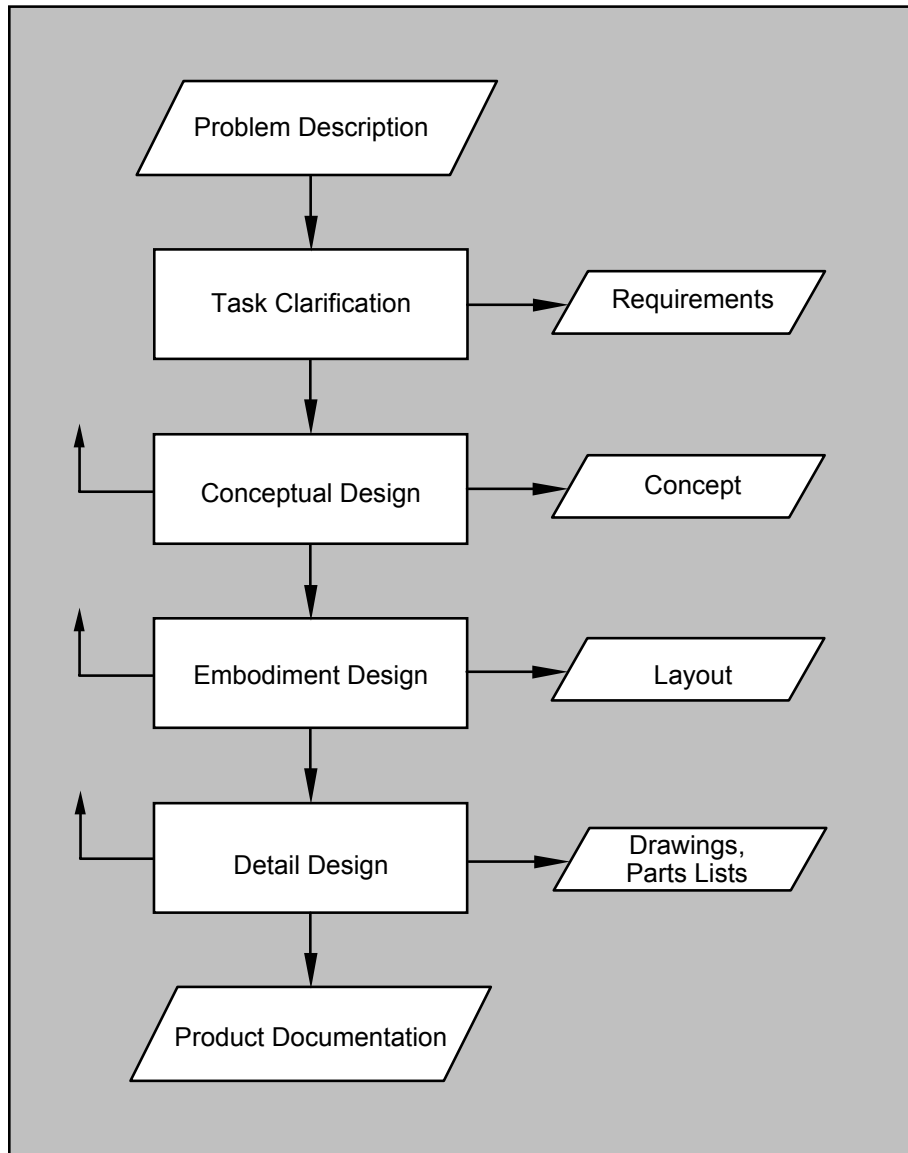


Figure 1: Design Phases according to EDM

### **Embodiment design**

In the embodiment design phase the engineer refines the concept and determines the layout and the geometry of the technical object in accordance with the description given in the requirements document. The arrangement, form, dimensions and surface properties of all parts of the technical object are optimised, checked against errors, and fixed.

### **Detail design**

In this phase the design is finalised and all detail drawings, parts lists, and production documents are generated.

For each of these phases, especially for the conceptual design phase, a vast amount methods to generate new, and to modify, evaluate and refine available solutions exist. Solution catalogues on various levels of abstraction are also available [Ro82, PaBe93].

Although EDM-research has been continuing for many years now, its acceptance among design engineers is not as broad as it should be [Mü91]. This is actually not quite surprising if we consider the proposed similarity between a grammar of a domain and EDM. Like a grammar, engineers tend to apply EDM consciously especially when they are hard-pressed for solutions and know no way out.

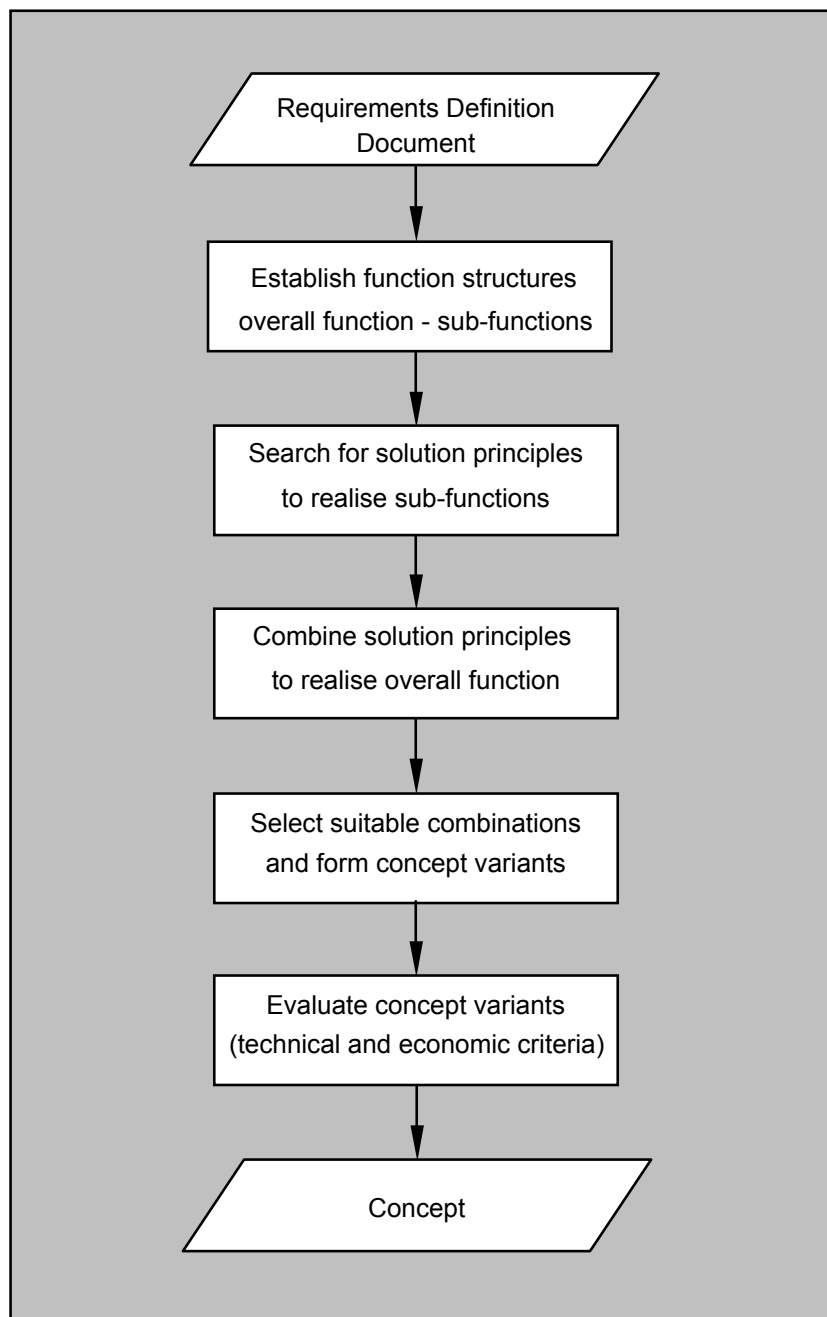


Figure 2 : Phases of conceptual design



### 3. Experience

Experience is the knowledge or skill a person has gained by doing and seeing things [Ox93]. Human beings store their experiences in the form of abstract mental concepts<sup>1</sup> which form the basis of intelligent behaviour and of communication [Vgo62, Wro91]. They enable us to understand things easier, to see the relations between objects and give us a means to store and access our knowledge in an efficient way.

Mental concepts collect a set of individual objects into a group with certain common properties [Wro91]. During this collection process only the relevant properties are taken into account whereas the irrelevant properties are ignored (abstraction).

Cognitive psychology gives following models for the structure of mental concepts in human beings [Wro91]:

- Classical model
- Probabilistic model
- Exemplar model
- Hybrid model
- Microtheory model

In the classical model a mental concept is defined by a set of necessary and, together, sufficient features. Mental concepts are organised hierarchically and all features are of structural nature. In the probabilistic model the mental concept is still defined by a set of features, but these need not be defining but only typical. Each feature has a measure of typicality attached to it and the classification of an object is done on a probabilistic basis. Exemplar models do not have a set of defining features but instead a set of defining exemplars. An object is classified according to its similarity to the exemplars of the mental concept. Hybrid models mix the defining-feature based and exemplar based models to form a system where both defining features and a set of exemplars exist. Microtheory models, on the other hand, use the rules governing the classification of the object as the core of the mental concept model.

The experience of the engineer plays a central role throughout the design process. Transferring the problem at hand into tasks to which solutions can be found is a process that is based on the ability of the engineer to relate the problem to his prior experiences and his domain knowledge (see also [Ru85, EhRu87]).

We believe that in mechanical engineering, designers store a description of what a technical object usually does and its basic properties, along with some examples and counterexamples as concepts. They also use their world knowledge [Mu88] to understand the nature of the object and the requirements of the context first. This is in support of the hybrid and the microtheory models.

---

<sup>1</sup>In order to distinguish concepts in the sense of cognitive science from (technical) concepts in the EDM sense the term "mental concept" is introduced here.



The knowledge, engineers store in mental concepts are either objects or methods, each of which can be either global or local (see table 1).

	Objects	Methods
Global	Object Classes	Heuristics, Strategies
Local	Solutions, Concrete Objects	Tactics, Procedures

Table 1: Types of knowledge stored in mental concepts

In the transportation domain we can give the following example for the fields in table 1:

Concrete Object : "Porsche 911 Carrera-4, 1993"  
 Object Class : "Cars"  
 Procedure : "Driving to my home from my office"  
 Strategy : "Getting from point A to point B on land in the shortest time"

Table 1 has a recursive nature in terms of abstraction hierarchy. A global object or method will have local objects and methods which in turn are global for other, more concrete, objects and methods.

If we consider the example given above and take "means of transportation" as the object class, then "cars" can be considered a local object. The strategy for "getting from point A to B in the shortest time" is more global than "Getting from point A to point B on land in the shortest time", which becomes a local method in this abstraction level.

Methods only make sense if combined with objects. From table 1 we see that four different combinations (table 2) are possible :

	Concrete Objects	Object Classes
Tactics, Procedures	Real world tasks	Understanding, Explanation
Heuristics, Strategies	Policies, Standards	Philosophy, Theory, Grammar

Table 2: Methods and object combinations

Considering the above given example once more we can see that in this context a real world task is "Driving to my home from my office in a Porsche 911 Carrera-4, 1993". The strategy given above applied to the concrete object yields "Getting from point A to point B in a Porsche 911 Carrera-4, 1993 in the shortest time", which is an object-specific strategy (Marketing strategies also fall into this group). The procedure applied to the object class yields "Driving to my home from my office in a car" i.e. an explanation or description, whereas using the object class in the strategy yields "Getting from point A to point B by car in the shortest time" which is a theoretical planning problem.

We believe that the process of combining local and global objects and methods both in one context and cross-contextual is one of the most important aspects of human creativity and has to be incorporated in systems that try to model it.

## 4. The Case-Based Conceptual Design System

### 4.1 Case-Based Reasoning

A Case-Based Reasoner solves new problems by adapting solutions to previously solved similar problems to the new requirements [Ham89, RiSch89, Ko90]. It is built upon the premise that human beings reason mostly from experience and not from first principles. A Case-Based Reasoning (CBR) System :

- finds and retrieves solutions from its Case-Base that have met the same or similar requirements,
- adapts the retrieved solutions to meet the current requirements,
- evaluates and if necessary repairs the adapted solution,
- suggests the solution it found to the user,
- and learns new cases from the solutions it generates.

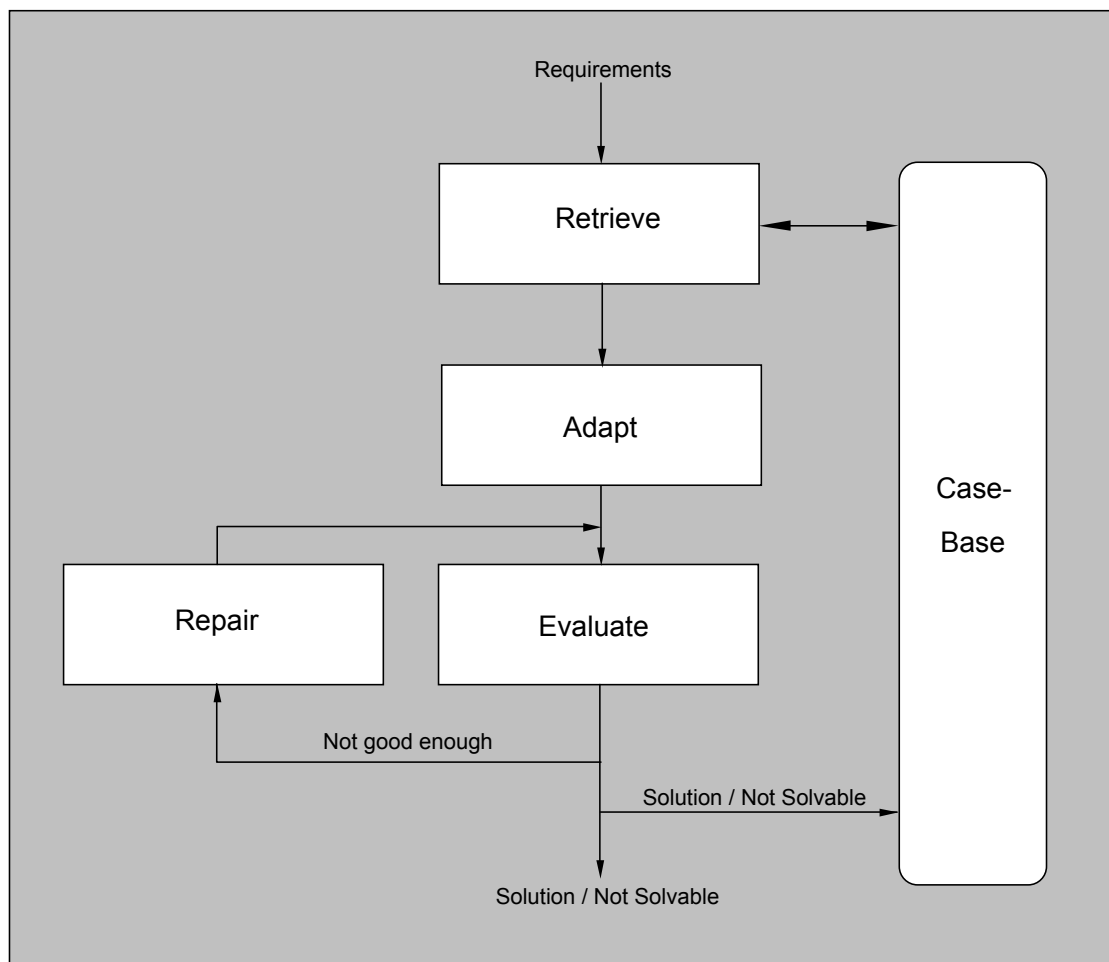


Figure 3: Case-Based Reasoning Flowchart

For simplicity the reasoning mechanism of the CBR system (retrieval, adaptation, evaluation and repair) is separated from the Case-Base in Figure 4 and is called the Reasoning Engine. The system is also complemented with a thesaurus which is used for indexing and retrieval purposes and the Control-Knowledge Base which contains global methods for the domain.

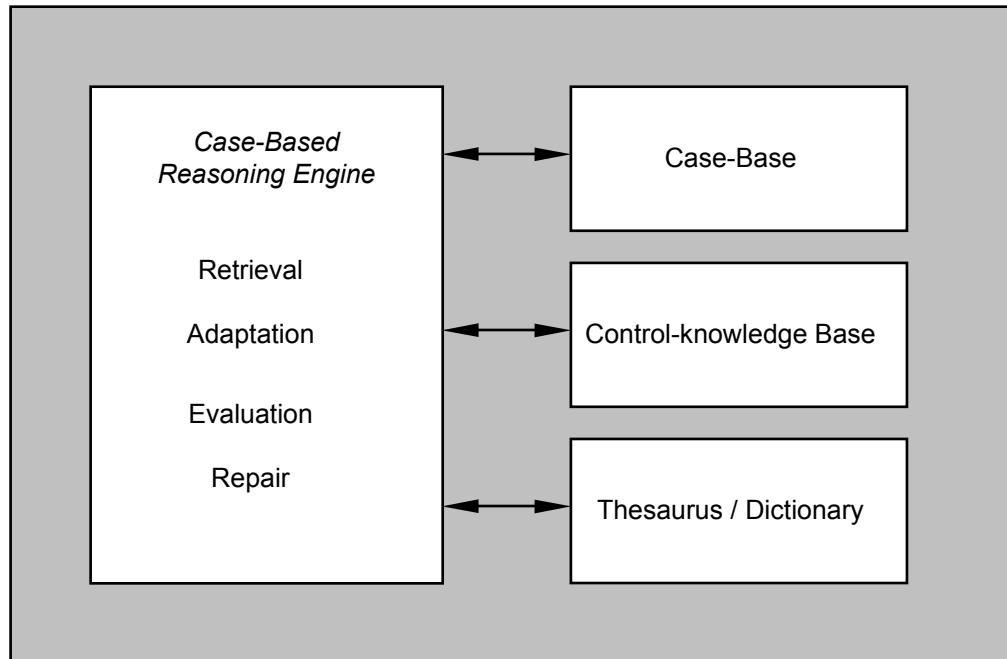


Figure 4: CBR-Module

## 4.2 EDM and CBR

As described in 2, EDM provides methods (both local and global) and structure for the design process and can be used as a grammar.

Local EDM methods are applicable only in a specific design phase and/or to specific objects, whereas global methods are applicable throughout the design process and form the control knowledge. Heuristics like "a technical object has to be simple, definite and safe" fall into this group.

By treating local methods, the object-class and the solutions experience of a design engineer as cases, the global methods as control knowledge and using the representation structure provided by EDM, we can implement a Case-Based system which supports the design engineer both methodologically and with experience.

Figure 5 shows the basic architecture of such a Case-Based Conceptual Design System which supports the design engineer during the task clarification and conceptual design phases in the above mentioned way.

## 4.3 The Case-Base

### 4.3.1 Case Types

The cases in the Case-Base of the system can be of three different types :

- Class-Cases
- Solution-Cases
- Local-Method-Cases

Class-Cases contain problem independent knowledge. They store knowledge that is not directly associated with a specific problem but that may be considered general engineering knowledge (physical effects, material properties, environmental specifications,...). They provide solutions to a class of problems.

Solution cases map specific requirements to functions, specific functions to physical effects, specific physical effects to effect carriers, specific effect carriers to geometry and specific geometry to production processes. They contain solutions for the task at the given phase of design. The resulting solution does not necessarily have to be on the following level of concretisation but can also skip one or several phases of the design process and thus concretisation levels. A solution which has been put together by joining partial solutions in various levels of concretisation will be as abstract as the most abstract of the partial solutions used.

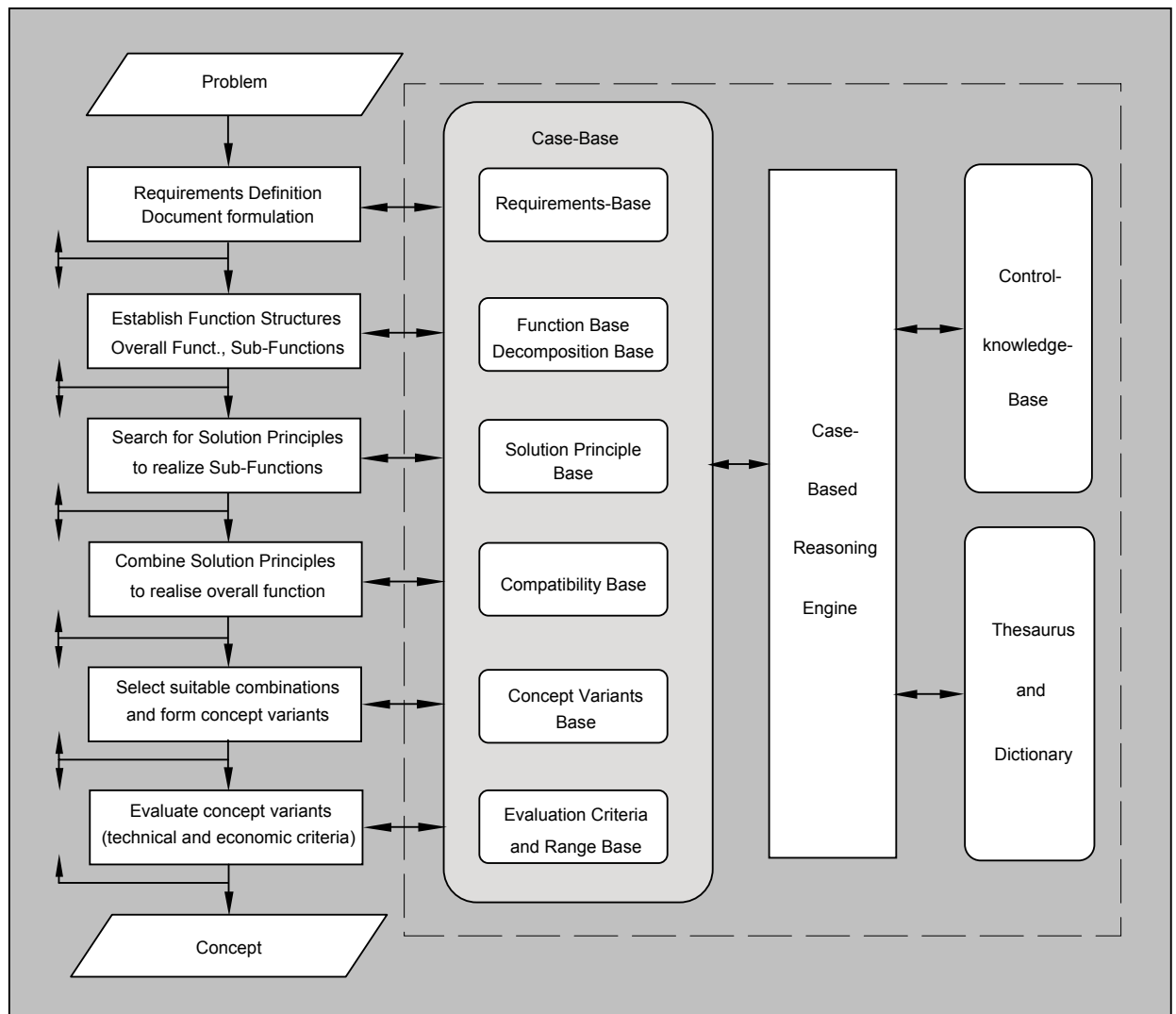


Figure 5: System Structure of the Case Based Conceptual Design System

There is an obvious overlapping between the Class- and Solution-Cases. Class-Cases, providing solutions to a class of problems have to contain solutions to specific problems too. Local-Method-Cases contain tactics, procedures and rules as well as their applicability and boundary conditions. They store information like procedures to generate new variants, for evaluation and selection etc.

### 4.3.2 The Content of the Case-Base

Figure 5 shows that the Case-Base contains several Sub-Case-Bases which relate to separate phases of the design process. Although this separation is artificial and the number of cases which skip several phases will increase due to the learning aspect of this system, it helps clarifying the structure and content of the Case-Base with respect to the design phases. Table 3 lists possible contents for each case type in an ideal case-base for each design phase.

An important aspect of the structure shown in table 3 is that the local methods of EDM are found in the Case-Base whereas the global methods are incorporated in the system architecture as control knowledge.

Type	Class-Case	Solution-Case	Method-Case
Requirements	<ul style="list-style-type: none"> <li>• Implicit requirements (environment, safety, user friendliness etc.)</li> <li>• Knowledge regarding neighbouring systems</li> </ul>	<ul style="list-style-type: none"> <li>• Specific Requirements Definition documents for existing, technical objects</li> <li>• Mappings to following phases</li> </ul>	<ul style="list-style-type: none"> <li>• Checklists for Requirements Specification</li> <li>• Neighbouring Systems analysis</li> </ul>
Function, Function Structures (Decompositions)	<ul style="list-style-type: none"> <li>• Engineering education</li> <li>• Physics, Mechanics etc.</li> <li>• Conversion methods between energy, matter and signal</li> </ul>	<ul style="list-style-type: none"> <li>• Function structures of existing devices</li> <li>• Mappings to following phases</li> </ul>	<ul style="list-style-type: none"> <li>• Function sequencing techniques</li> <li>• Decomposition tactics</li> <li>• Planning techniques</li> </ul>
Solution Principles	<ul style="list-style-type: none"> <li>• Physical effects to realise functions</li> <li>• Principal solutions</li> <li>• Effect carriers</li> </ul>	<ul style="list-style-type: none"> <li>• Specific technical objects, mechanisms or devices</li> <li>• Mappings to following phases</li> </ul>	<ul style="list-style-type: none"> <li>• Variation techniques</li> <li>• Checklists for property determination to generate new principal solutions</li> </ul>
Compatibilities	<ul style="list-style-type: none"> <li>• Scientific Education</li> <li>• Domain Specifics</li> </ul>	<ul style="list-style-type: none"> <li>• Incompatible Solutions list</li> <li>• Patents</li> </ul>	<ul style="list-style-type: none"> <li>• Algorithm for Compatibility checks</li> </ul>
Concept Variants	<ul style="list-style-type: none"> <li>• Abstract solutions</li> <li>• Analogies from other domains</li> </ul>	<ul style="list-style-type: none"> <li>• Specific products, mechanisms, devices</li> </ul>	<ul style="list-style-type: none"> <li>• Concretisation techniques and tactics</li> <li>• Morphological matrix</li> </ul>
Evaluation Criteria and Value Ranges	<ul style="list-style-type: none"> <li>• Engineering education</li> <li>• Dimensions and ranges of properties</li> </ul>	<ul style="list-style-type: none"> <li>• Evaluation criteria and associated value/grade mappings</li> <li>• Existing evaluations</li> </ul>	<ul style="list-style-type: none"> <li>• Criteria determination checklists</li> <li>• Evaluation algorithms</li> </ul>

Table 3: Content of the ideal Case-Base

Actually the table has a third dimension which is the context of the given design situation. The applicability of the methods and the properties which are relevant for the Solution- and Class-Cases are very much context dependant.

### 4.3.3 The Structure of the Case-Base

Roth describes in [Ro82] details regarding the storage and usage of engineering knowledge in what he calls design catalogues. He distinguishes between object, solution and operation catalogues which have the same content as the class, solution and method cases respectively. He describes methods for generating these catalogues, for their application as well as several samples and a dictionary for attributes that can be used for indexing.

Design catalogues (table 4) contain three different areas :

- partitioning area
- main area
- access area

Partitioning Area		Main Area			Access Area								Supplement
Type of Surface-contact	Type of Force transmission	Equation	Name	Sample		Maximal transmittable Torque	Torque transmission depends on	Axial Forces are supported	Over-load reaction	Concentricity	Shaft axially movable	Shaft angularly mountable	Comments
1	2	1	2	3	Nr	1	2	3	4	5	6	7	8
Normal  (shape bound)	direct	$M_t = (d_m/2) * A_{\tau tot} * \tau_{max}$ $M_t = (d_m/2) * A_p * P_{max}$	Profile-Shaft		1	large	Form factor	no	break	yes	possible	in steps	
	indirect	$M_t$ : transferable Torque $d_m$ : average applicable diameter	Form-elements		2	small		possible					
Tangential  (friction bound)	indirect	$M_t = M_f = F_r * d_m / 2 = F_n * \mu * d_m / 2$ $\alpha$ :angle $M_f$ :Frictional	Clamp element		3	small to large	Temperature,  Rotational Forces,  Axial Forces	yes	slide	yes	only if $F_A > F_r$	continuous	Easy to produce and assemble
	direct	Torque $F_r$ : Friction Force $d_{st}$ : pin diameter	Stress element		4	medium		possible		possible			
Tangential and normal	indirect		Pre-stresses connection		5	small		possible	break	no	no	possible	

Table 4: Sample Design Catalogue

The partitioning area contains attributes that partition the solution space in a way that the completeness of the catalogue and the uniqueness of the elements is ensured. The main

area contains the main description and/or sketch of the element. The access area contains the indices that are used to access the elements in the design catalogue. These indices do not necessarily have to be the same attributes as the ones in the partitioning area but should be oriented towards the applicability of the elements.

This organisation seems very appropriate for the creation and the usage of the Case-Base too. The indices for partitioning ensure the uniqueness of the cases whereas the access indices ensure the applicability oriented retrieval.

Another very useful aspect of this organisation is that design catalogues of the type described in [Ro82] exist in a large number and can easily be transferred to the Case-Base. A computer-aided method to generate and use design catalogues has also been described in [De90].

#### **4.4 The Thesaurus**

The thesaurus and dictionary provide a standard for describing and accessing cases. In order to provide stable communication with the system, indices (terms) have to be standardised. This does not necessarily have to be a global standard, but can be user specific. The indices used in the Case-Base have to be from the dictionary or have to be added (see also [BaZe92]).

#### **4.5 The Control-Knowledge-Base**

As described in 4.2 the global methods of EDM that are applicable throughout the design process form the control knowledge of the system. These are methods that deal with the global consistency and properties of the technical object as well as methods for abstraction/concretization, generalisation/specialisation and aggregation/decomposition (see also [Su93]).

#### **4.6 The Case-Based Reasoning Engine**

The tasks of retrieval, adaptation, evaluation and repair are based on EDM methodology. Retrieval is basically done by template matching. It is assumed that the more properties and attributes the two objects have in common the more they are similar. Currently the effect of attributes causing dissimilarity is not being taken into account. Adaptation and repair are both being done interactively with the user. We define the adaptation process to be the adjustment of the values of properties or the addition of missing properties to the retrieved object, whereas repair is the removal of incompatibilities and unwanted properties from the adapted object. The adapted object has both the properties of the retrieved object and the properties of the ideal object described in the requirements definition document. In the repair step some of the properties that come from the original, retrieved object are being removed. The result the system delivers may have more properties than were required by the user, but this does

not necessarily have to be a negative effect. For the evaluation process the techniques of EDM are being applied directly [VDI2225, PaBe93, BiGö92].

#### **4.7 Learning new Cases**

The ability to learn new cases is one of the most important aspects of Case-Based reasoning systems.

In the system the proposed solutions as well as intermediate steps can be saved as new cases in the Case-Base. Due to the fact that they are problem specific these will be Solution-Cases most of the time. The result of the learning process will be, that there will be more and more solution cases that skip one or more design phases. This will yield a Solution-Case-Base with cases of various concretisation levels. We believe this is also the case in human engineers.

#### **5. Current Status**

Currently the project is in the implementation phase. At the same time research is continuing regarding the representation and use of cases in the mental processes of the human engineer. It is obvious that we will never be able to create the Case-Base shown in Table 3 in its entirety, but we are trying to realise the system with a working minimum. We are also negotiating with industry partners to serve as a test site for the system.

#### **6 Conclusion**

We propose that the methods and structure provided by Engineering Design Methodology (EDM) can be utilised as a design grammar in a Case-Based Conceptual Design System. The methods of EDM and the experience of the design engineer formulated in cases complement each other. By formulating the experience of the design engineer in Class-, Solution- and Method-Cases we are able to describe a system architecture of a system that supports the design engineer both in terms of methodology and experience.

## References

- BaZe92 T.Bardasz, I.Zeid, "Cognitive model of memory for mechanical-design problems", Computer Aided Design, Vol.24, Number 6, pp.327-342, June 1992
- Bi80 H.Birkhofer, "Analyse und Synthese der Funktionen technischer Produkte", Fortschrittsberichte VDI-Z, Reihe 1, Nr.70, 1980
- BiGö92 H.Birkhofer, M.Göker, K.H.Beelich, "Beurteilen" in "Wissensbasierte Systeme für Konstruktion und Arbeitsplanung", VDI Gesellschaft Entwicklung Konstruktion Vertrieb (VDI-EKV) and Gesellschaft für Informatik (GI) ed., VDI-Verlag, Düsseldorf 1992
- De90 T. Derhake, "Methodik für das rechnerunterstützte Erstellen und Anwenden flexibler Konstruktionskataloge", Dissertation, Technische Universität Braunschweig, 1990
- Ru85 Andreas Rutz, "Konstruieren als gedanklicher Prozeß", Technische Universität München, Dissertation, 1985
- Su93 Alexander Suhm, "Produktmodellierung in wissensbasierten Konstruktionssystemen auf der Basis von Lösungsmustern", Dissertation Universität Karlsruhe, Verlag Shaker, Aachen, 1993
- EhRu87 K.Ehrlenspiel, A.Rutz, "Konstruieren als gedanklicher Prozeß", Konstruktion 39, H.10, pp 409-414, Springer Verlag, 1987
- Ham89 Kristian J.Hammond, "Case-Based Planning - Viewing Planning as a Memory Task", Academic Press Inc, HBJ Publishers, San Diego, 1989
- Ko90 Janet L. Kolodner, "An Introduction to Case-Based Reasoning", School of Information and Computer Science, Georgia Institute of Technology, GIT-ICS-90/19, 1990
- KoJo89 Laura Kochevar, Paul Johnson, "Problem Solving is What You Do When You Don't Know What to Do", Eleventh Annual Conference of the Cognitive Society, pp 615-622, 16-19 August 1989, Ann Arbor, Michigan, Lawrence Erlbaum Associates, Hillsdale 1989
- Mu88 Murphy, G.L. "Comprehending Complex Concepts", Cognitive Science, 12, pp 529-562, 1988
- Mül91 J.Müller, "Akzeptanzbarrieren als berechnete und ernstzunehmende Notwehr kreativer Konstrukteure", Proceedings ICED 91, pp769-776

- Ox93 Oxford Advanced Learner's Dictionary of Current English, Fourth Edition, Oxford University Press, 1993
- PaBe84 G.Pahl, W.Beitz, "Engineering Design", Springer Verlag, Berlin, Heidelberg, 1984
- PaBe93 G.Pahl, W.Beitz, "Konstruktionsmethodik", 3rd. Ed., Springer Verlag, Berlin, Heidelberg, 1993
- RiSch89 Christopher Riesbeck, Roger Schank, "Inside Case-based Reasoning", Lawrence Erlbaum Associates, Publishers, Hillsdale 1989
- VDI2221 Verein Deutscher Ingenieure, VDI Guideline 2221: "Methodik zum Entwickeln und Konstruieren technischer Systeme und Produkte", Beuth Verlag, Berlin 1986
- VDI2222 Verein Deutscher Ingenieure, VDI Guideline 2222, Part 1: "Konstruktionsmethodik, Konzipieren technischer Produkte", Beuth Verlag, Berlin 1977
- VDI2225 Verein Deutscher Ingenieure, VDI Guideline 2225, "Konstruktionsmethodik - Technisch-Wirtschaftliches Bewerten", Beuth Verlag, 1977
- Vgo62 Lev S. Vygotsky, "Thought and Language", The M.I.T. Press, Cambridge, 1962
- Wro91 Stefan Wrobel, "Concept formation in Man and Machine: Fundamental Issues"; Arbeitspapiere der GMD, Gesellschaft für Mathematik in der Datenverarbeitung mbH., 1991