INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN
ICED 03 STOCKHOLM, AUGUST 19-21, 2003

EFFICIENT SUPPORT FOR THE PRODUCT DEVELOPMENT PROCESS – COMPUTER-BASED METHOD-TOOLS FOR THE CONCEPTUAL PHASE

Walter, S., Sauer, T., Weiss, S., Birkhofer, H.

Abstract

This paper presents an approach to support the application of product development methods with a software-tool. The system is intended to support both the designer in practice and the student at the university.

In order to keep the system simple a special software-tool should not be generated for each method. Rather, the methods have to be described and structured in such a way that elementary fragments can be separated. To achieve an overall value it must be possible to create more than just one or a few method-tools from these elementary fragments.

Keywords: computer-aided learning, knowledge-based engineering, knowledge-based systems

1 Introduction

In times of fast changing global markets, business companies need high dynamics and flexibility in developing new products.

In actual and future product development, especially in highly competitive markets, there are two basic problems: charges increase progressively in the developmental period while the time-to-market simultaneously decreases. Products which yesterday seemed to have potential in becoming market leaders may be antiquated today. That’s why business companies sorely need efficient method application and software-tools that allow products to be developed more quickly.

2 Actual application of product development methods

Although the positive influences of method-application in product development projects are observed, there are a few points of criticism. Partially complex and less intuitive proceedings in methods may strain the patience of the users or maybe even frustrate them.

Furthermore, there is nearly no chance of reconstructing which course a project has taken when no methods have been applied. These may be some causes for the slight range of method application. It could also be due to too little experience or bad experiences with an incorrect method-application or the application of an inapt method.

Often, the benefits of documentation are not evident to the user. Only when the results of a method application are used in the following steps does the user recognize the benefits. The documentation that accompanies the method application makes sense to the user only when it is used later. Documentation without multiple usage is exactly what it seems to the user to be – a waste of time.

There are some further reasons for the slight use of methods:

In contrast to students, designers in industry have a certain level of experience and their own specific problem-solving behaviors. Foreign and different behaviors are not always welcome to them. [2]

In addition, many more causes of problems in method application are to be found in [7]: Changed circumstances in product development, lack of cognition concerning aims, limits and possibilities of methods and consequential errors, inadequate customized methods concerning specific circumstances or even the absence of schooling.

Due to that and the fact that an objective cost-benefit analysis is hardly possible, the designer develops a doubtful attitude towards method applications. [7]

3 Approach for improvement

The point is to support the application of product development methods.

Based upon the experiences with ‚thekey to innovation' [4], the department of product development and machine elements (pmd) at the Darmstadt University of Technology has endeavored to generate a system which is both an application-system for daily use in industry projects and a teaching- and learning-system for product development education at universities: this system is known as *pinngate*.

3.1 Components of pinngate

The complete pinngate system will contain the following components:

- Knowledge-base which provides concrete solutions at different levels of product development. The solutions are extensively described in order to enable easy use.
- Teaching- and learning-system based on modularized product development knowledge to fulfill specific user needs. The contents of product development knowledge are modularized in different granularity and integrated into a knowledge network. [1]
- Product development method-tool(s) which will be ready to use in different situations. These specific situations could be a designer, who wants to solve an actual design task, or a product development method used in education, that shall be explained by demonstrating the method-tool.
- Navigator which supports the user in finding his way within the system.

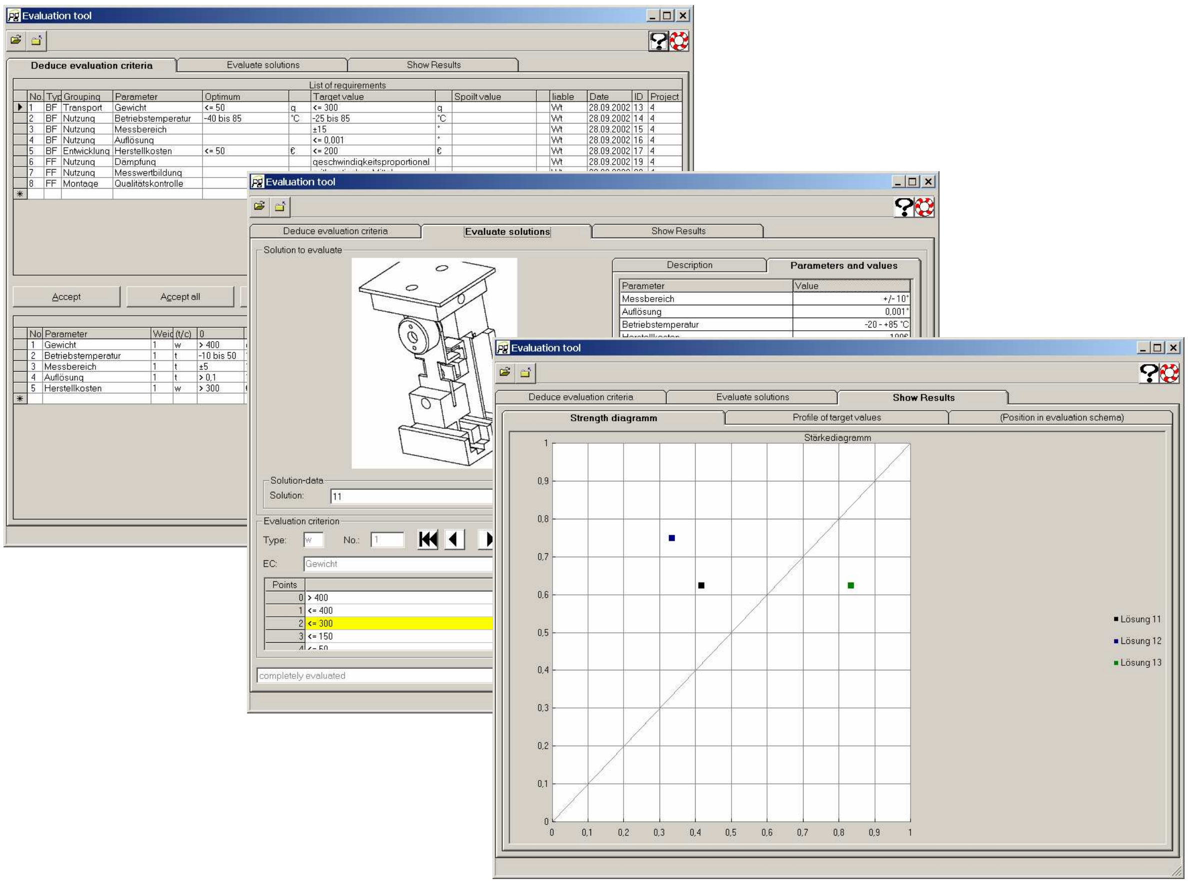
The individual components team up over interfaces. The system shares the redundance-free stored information and data. Method-tools, providing background knowledge to solve design tasks and a knowledge-base to work with are given to the users.

The persistent usage of already existing data (knowledge-base), edited data and newly generated data reduces the documentation effort mentioned above. The data that is necessary for the method application shall be provided, prepared and arranged by the system. Newly generated data will be saved and provided for following steps.

3.2 Software-tool for evaluation

As a first step, a software-tool with which one can evaluate solutions for given functions was created. This prototypical tool for the evaluation method is structured in a process-related way. The software-tool reproduces the designers' proceeding in the following steps:

- first, the user obtains the (existing) list of requirements from which he can deduce evaluation criteria and edit and/or complete them
- then, the user can evaluate (existing) possible solutions in respect to the assessed evaluation criteria
- finally, the user can display the results of the evaluation in different ways (e. g. strength diagram)



List of requirements

No.	Typ	Grouping	Parameter	Optimum		Target value		Spoilt value		liable	Date	ID	Project
1	BF	Transport	Gewicht	<= 50	g	<= 300	g			Wt	28.09.2002	13	4
2	BF	Nutzung	Betriebstemperatur	-40 bis 85	°C	-25 bis 85	°C			Wt	28.09.2002	14	4
3	BF	Nutzung	Messbereich			±15	°			Wt	28.09.2002	15	4
4	BF	Nutzung	Auflösung			<= 0,001	°			Wt	28.09.2002	16	4
5	BF	Entwicklung	Herstellkosten	<= 50	€	<= 200	€			Wt	28.09.2002	17	4
6	FF	Nutzung	Dämpfung			geschwindigkeitsproportional				Wt	28.09.2002	19	4
7	FF	Nutzung	Messwertbildung										
8	FF	Montage	Qualitätskontrolle										

No	Parameter	Weig	(t/c)	0
1	Gewicht	1	w	> 400
2	Betriebstemperatur	1	t	-10 bis 50
3	Messbereich	1	t	±5
4	Auflösung	1	t	> 0,1
5	Herstellkosten	1	w	> 300

Parameter	Value
Messbereich	+/- 10°
Auflösung	0,001°
Betriebstemperatur	-20 - +85 °C

Solution: 11

Type: w No.: 1

EC: Gewicht

Points	
0	> 400
1	<= 400
2	<= 300
3	<= 150
4	<= 50

completely evaluated

Figure 1 Screenshots of the evaluation-prototype

The problem is that this prototypical software (Figure 1), in the latest version, acts as one independent software-tool. It may be supplied with external rendered data (e. g. list of requirements) and is able to handle and analyze these data and to show the results.

But in the end it should not be the intention to create an extra software-tool for each product development method, which categorically distinguishes how each method is handled. The objective should be instead to create a flexible software-tool that is able to realize different method-applications with similar handling.

4 Approach for the transfer of methods into a flexible software-tool

The aim is to implement product development methods into software-tools and to do that, above all, with as little expense as possible. To reach that aim the methods should be structured or even standardly described. This would mean that no product development method would have to be transferred to an extra software-tool. The specific software-tools could be put together from only a few "standard-elements" and could be flexibly configured.

4.1 The structure of the software-tool

This approach attempts to build up the product development methods from simple lists, respectively as composites of lists and their conjunctions. In other words, the lists are a kind of 'common denominator'. By combining, orientating and setting conjunctions between lists, different methods can be represented.

In the following examples, the performance shall be illustrated at different levels of complexity.

4.2 Simple list – Listing

The simplest case is a plain listing as in the given example: a list of functions (Figure 2). The functions which are realized with a subject are listed here.

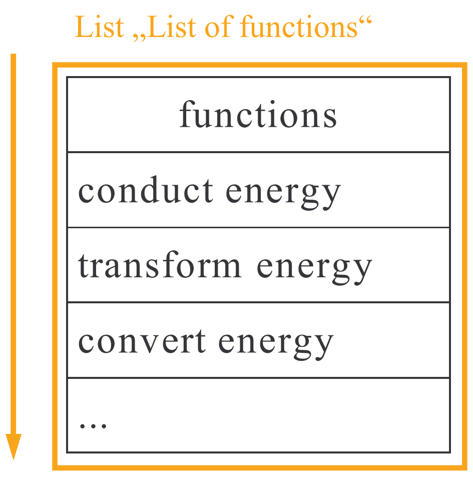


Figure 2 List of functions as simple example for a listing

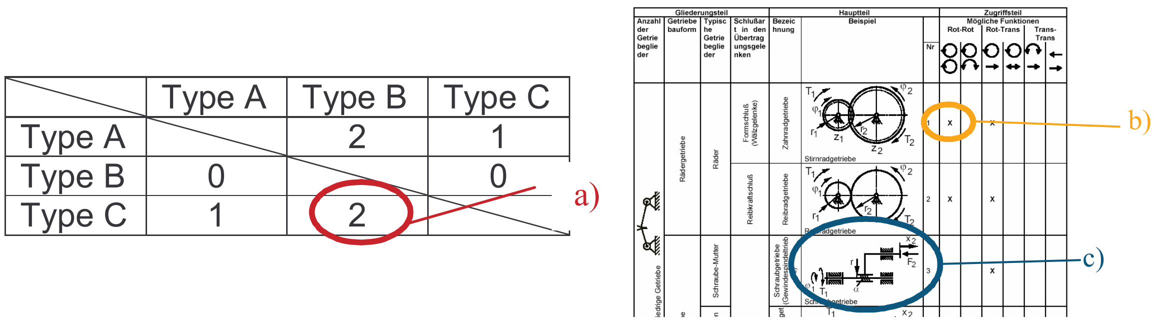
In this example no conjunctions are necessary since there is only one list. Examples with more lists will follow later on.

4.3 Conjunctions between the lists

It shall be possible to establish different conjunctions to associate the list entries. Figures, predications (logical or mathematical) and even totally free input will be possible.

- Figures: A figure as a conjunction between two list entries could occur, for example, in a comparison. The correlation between compared subjects could be described with the figures 0 (less important), 1 (equal importance) or 2 (more important). (Figure 3a)
- Predications: Mathematical predications, such as "greater than", "less than", "equal to" or "is subset of", "is set union" are imaginable, as well as

- Logical predications: For example, in an evaluation where a “cross” signalizes the conjunction between the evaluation criterion and the reached value, or as shown in the example below in a design catalogue. (Figure 3b)
- Free input: Even free input such as text or pictures/sketches shall be able to be inserted (Figure 3c).



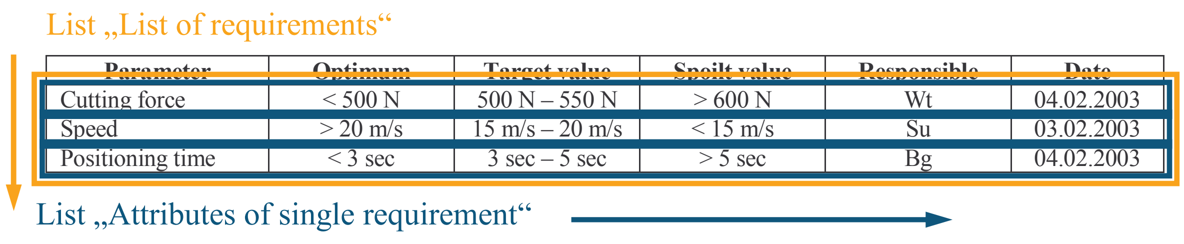
	Type A	Type B	Type C
Type A		2	1
Type B	0		0
Type C	1	2	

Figure 3 Examples of conjunctions

4.4 Simple composites of lists – Listing of lists

A composite from two lists could be, for example, a list of requirements (Figure 4). A single requirement can be understood as a simple list which contains the listed attributes (e. g. criterion, optimum, target value, spoilt value, responsibility, date) of the requirement. The order of the requirements may be arranged by the user in order to structure the list of requirements thematically. But there is no mandatory overvaluation because of the sorting.

The conjunctions between the lists may be characterized as “textual addition”. The statements of the single requirements are accumulated. A further connection between the listed requirements is not inevitable.



List „List of requirements“
List „Attributes of single requirement“

Parameter	Optimum	Target value	Spoilt value	Responsible	Date
Cutting force	< 500 N	500 N – 550 N	> 600 N	Wt	04.02.2003
Speed	> 20 m/s	15 m/s – 20 m/s	< 15 m/s	Su	03.02.2003
Positioning time	< 3 sec	3 sec – 5 sec	> 5 sec	Bg	04.02.2003

Figure 4 List of requirements as a combination of different lists

In this arrangement, only the header column of the evolving table is fixed by the titles (cutting force, speed and positioning time) of the arranged list (requirements). The number of (filled) columns depends on the user. In this example, a requirement could perhaps have only a target value and no optimal or spoilt value.

In a morphological box (in this case: partial solutions for partial functions) no row has to be filled completely. The number of solutions for a function may vary from one function to another. (Figure 5)

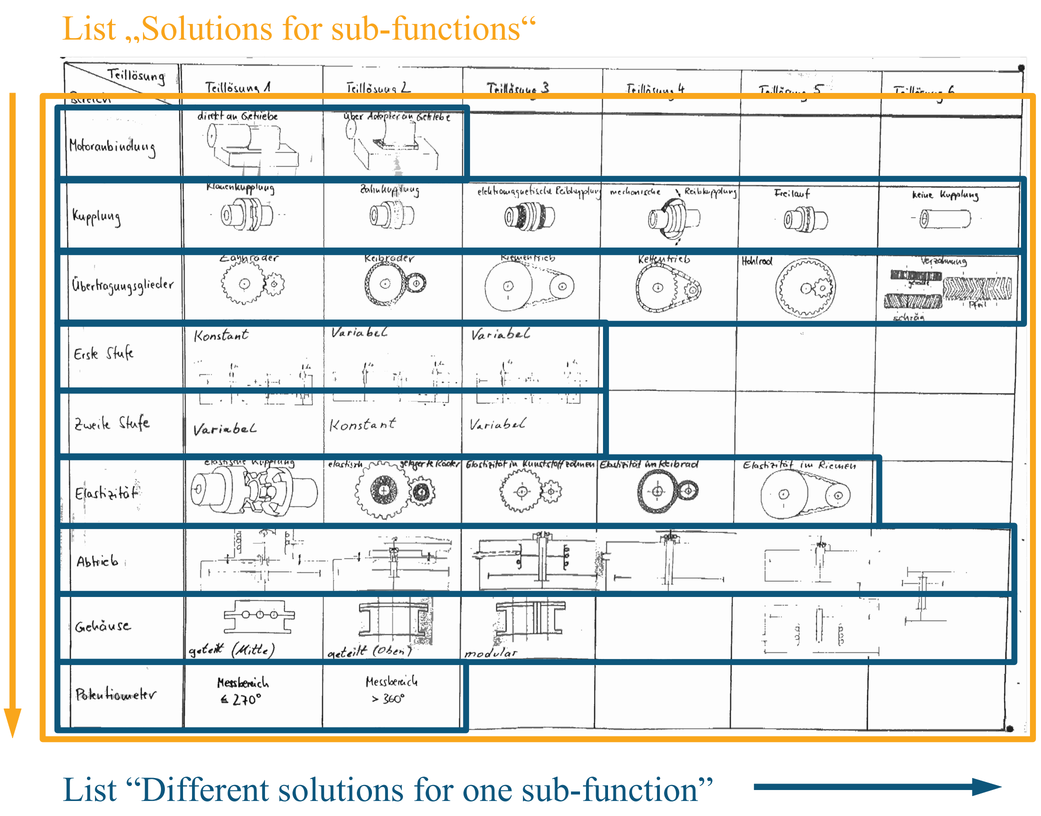
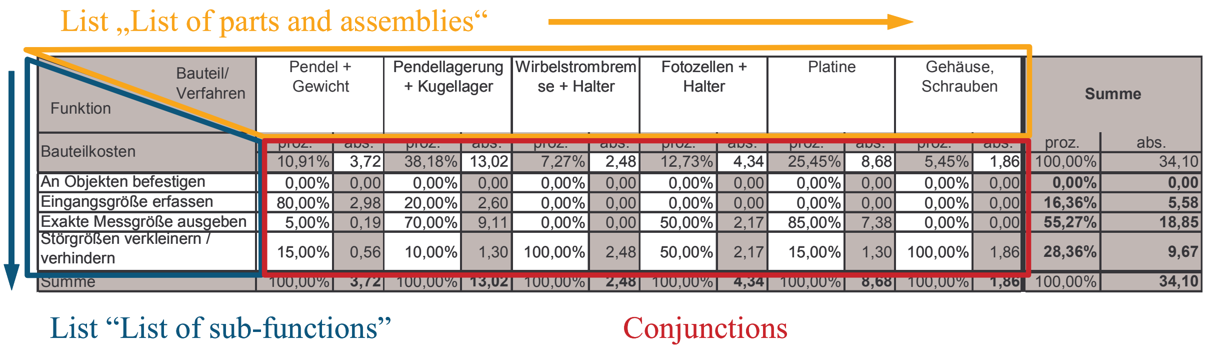


Figure 5 Example for combined lists with fixed header column and flexible header row – Morphological box

4.5 Composite of lists – Any conjunction

It is not always the case that there are fields left empty, as mentioned in 4.4. There are also composites of lists in which every field has to be filled in.

If there are two lists linked normally, a field “number of entries list 1”-times-“number of entries list 2” is stretched as in a comparison or as shown here in a cost analysis. Both the number of columns and the number of rows are fixed due to the linked lists. (Figure 6).

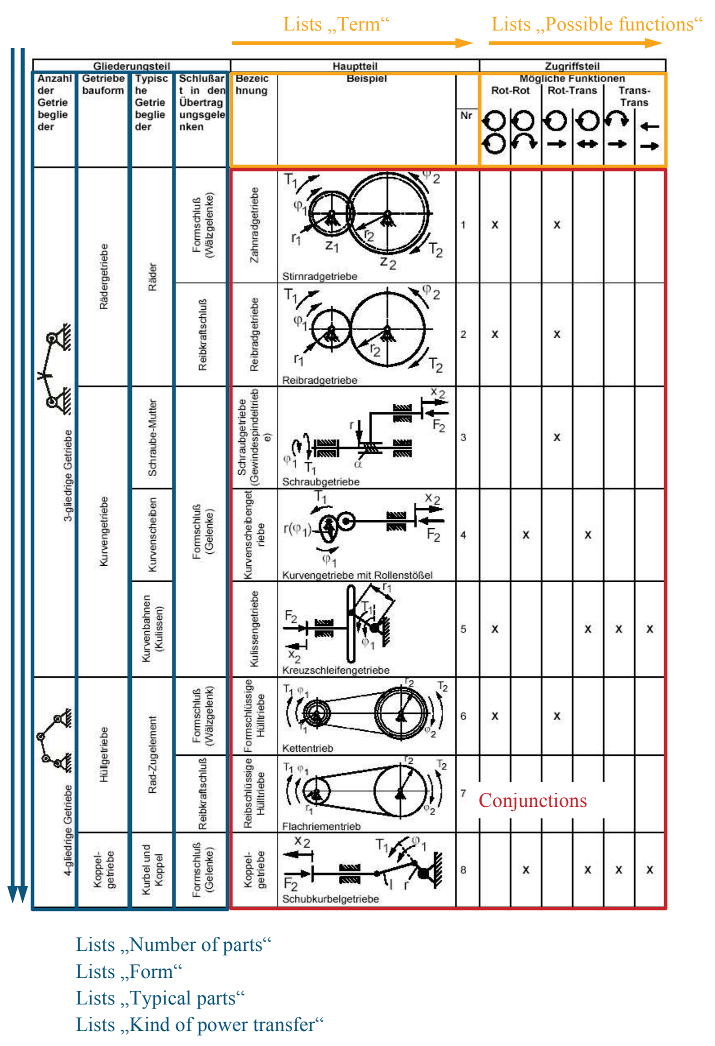

List „List of parts and assemblies“
List “List of sub-functions”
Conjunctions

Bauteil/ Verfahren / Funktion	Pendel + Gewicht		Pendellagerung + Kugellager		Wirbelstrombremse + Halter		Fotozellen + Halter		Platine		Gehäuse, Schrauben		Summe	
	proz.	abs.	proz.	abs.	proz.	abs.	proz.	abs.	proz.	abs.	proz.	abs.	proz.	abs.
Bauteilkosten	10,91%	3,72	38,18%	13,02	7,27%	2,48	12,73%	4,34	25,45%	8,68	5,45%	1,86	100,00%	34,10
An Objekten befestigen	0,00%	0,00	0,00%	0,00	0,00%	0,00	0,00%	0,00	0,00%	0,00	0,00%	0,00	**0,00%**	**0,00**
Eingangsgröße erfassen	80,00%	2,98	20,00%	2,60	0,00%	0,00	0,00%	0,00	0,00%	0,00	0,00%	0,00	**16,36%**	**5,58**
Exakte Messgröße ausgeben	5,00%	0,19	70,00%	9,11	0,00%	0,00	50,00%	2,17	85,00%	7,38	0,00%	0,00	**55,27%**	**18,85**
Störgrößen verkleinern / verhindern	15,00%	0,56	10,00%	1,30	100,00%	2,48	50,00%	2,17	15,00%	1,30	100,00%	1,86	**28,36%**	**9,67**
Summe	100,00%	**3,72**	100,00%	**13,02**	100,00%	**2,48**	100,00%	**4,34**	100,00%	**8,68**	100,00%	**1,86**	100,00%	**34,10**

Figure 6 Example of combined lists with a fixed header column and a fixed header row – Cost analysis

In this example of a cost analysis, no empty fields are allowed as a conjunction. If the intention is to state that there is no connection between two entries, at least the conjunction “0” has to be set.

As an example of a conjunction of more than two lists, a design catalogue is shown (Figure 7).


Lists „Term“
Lists „Possible functions“
Conjunctions
Lists „Number of parts“
Lists „Form“
Lists „Typical parts“
Lists „Kind of power transfer“

Gliederungsteil				Hauptteil			Zugriffsteil					
Anzahl der Getriebeglieder	Getriebebauform	Typische Getriebeglieder	Schlußart in den Übertragungsgelenken	Bezeichnung	Beispiel		Mögliche Funktionen					
							Rot-Rot		Rot-Trans		Trans-Trans	
						Nr	↻ ↻	↻ ↷	↻ →	↻ ↔	↷ →	← →
3-gliedrige Getriebe	Rädergetriebe	Räder	Formschluß (Wälzgelenke)	Zahnradgetriebe	Stirnradgetriebe	1	x		x			
			Reibkraftschluß	Reibradgetriebe	Reibradgetriebe	2	x		x			
	Kurvengetriebe	Schraube-Mutter	Formschluß (Gelenke)	Schraubgetriebe (Gewindespindeltriebe)	Schraubgetriebe	3			x			
		Kurvenscheiben		Kurvenscheibengetriebe	Kurvengetriebe mit Rollenstößel	4		x		x		
		Kurvenbahnen (Kulissen)		Kulissengetriebe	Kreuzschleifengetriebe	5	x			x	x	x
4-gliedrige Getriebe	Hüllgetriebe	Rad-Zugelement	Formschluß (Wälzgelenk)	Formschlüssige Hülltriebe	Kettentrieb	6	x		x			
			Reibkraftschluß	Reibschlüssige Hülltriebe	Flachriementrieb	7						
	Koppelgetriebe	Kurbel und Koppel	Formschluß (Gelenke)	Koppelgetriebe	Schubkurbelgetriebe	8		x		x	x	x

Figure 7 Example of combined lists with more than two lists – Design catalogue

The combined lists are the “number of parts”, “form”, typical parts” and “kind of power transfer” in the columns, and “term” and “possible function” in the rows. In the field of conjunctions the different types of conjunction are set, as mentioned above.

5 Outlook

To support the application of product development methods effectively such a software-tool has to be simple to use, easy to learn and for the most part self-explanatory. To prove this, further steps must be taken:

- First, some methods have to be realized in such a software-tool to show the feasibility.
- The software-tool should be tested in a small project (e. g. student research project or final project), later on maybe in some cooperation project with industrial partners.
- The gained insights should be continuously optimized.
- At the same time the number of realized product development methods could be expanded.

References

[1] Birkhofer, H., Berger, B., Walter, S., "Modularization of Knowledge – A New Approach in the Field of Product Innovation", Proceedings of the 7th International Design Conference DESIGN 2002, Dubrovnik 2002, Vol. 1, pp. 289 - 294.

[2] Birkhofer, H., Kloberdanz, H., Sauer, T., Berger, B., „Why methods don't work and how to get them work", Proceedings of the 3rd international seminar and workshop 'Engineering Design in Integrated Product Development', EDIProD 2002, Zielona Gòra – Lagòw 2002, pp. 29 - 36.

[3] Birkhofer, H., Kloberdanz, H., Berger, B., Sauer, T., „Cleaning up design methods – describing methods completely and standardized", Proceedings of the 7th International Design Conference DESIGN 2002, Dubrovnik 2002, Vol. 1, pp. 17 - 22.

[4] Birkhofer, H., Lindemann, U., Albers, A., Meier, M., „Product Development as a structured and interactive network of knowledge – A revolutionary approach", International Conference on Engineering Design ICED 2001, Glasgow 2001.

[5] Ehrlenspiel, K., „Integrierte Produktentwicklung", München, Hanser 1995.

[6] Pahl, G., Beitz, W., „Engineering Design", 2nd Edition, Springer Publisher, London 1996.

[7] Zanker, W., „Situative Anpassung und Neukombination von Entwicklungsmethoden", Dissertation, Lehrstuhl für Produktentwicklung, TU München, 2000.

Stephan Walter
Product development and machine elements, pmd
Darmstadt University of Technology
Magdalenenstrasse 4
D-64289 Darmstadt, Germany
phone: +49 (0) 6151 / 16 33 78
fax: +49 (0) 6151 / 16 33 55
eMail: walter@pmd.tu-darmstadt.de