INTERNATIONAL DESIGN CONFERENCE - DESIGN 2002
Dubrovnik, May 14 - 17, 2002.

AN INTERFACE SPECIFICATION FOR PRINCIPLE SOLUTIONS SUPPORTING THE CROSS-DOMAIN DESIGN OF MECHATRONIC SYSTEMS

Stefan Möhringer and Jürgen Gausemeier

Keywords: modelling of principal solutions, conceptual design, interface specification, mechatronic systems, collaborative design

1. Introduction

The conceptual design plays an important role for mechatronic systems. Starting from the requirements, the overall function has to be derived and divided into subfunctions until every subfunction can be fulfilled by a working principle or a solution element. For this process engineers from different domains such as mechanics, electronics and software engineering need a common base for communication and cooperation. A modelling language has been developed to specify principle solutions in a cross-domain way [Kallmeyer 1998]; simular approaches are e.g. [Schön/Meerkamm 1999], [Lippolt 2000]. Starting from this model of a principle solution the components are usually worked out decentralised within the engineering departments. With the increasing design progress the interactions between components become more and more important. They must be considered at any time to ensure the system integration and a total product optimum. Furthermore these interactions are dynamic: they can change due to concept modifications, new customer requirements or proceeding component specifications. The experience in industrial projects shows that the handling of these interactions is not supported sufficiently. Common concept features and dependencies between components cannot be specified in an appropriate way. Changes are often not communicated early enough between the relevant departments. Inconsistencies and time and cost-consuming iterations are the result. Therefore engineers from the involved domains need a common method to specify the interactions between components. The cross-domain modelling language for functions and principle solutions will be presented. A method to specify interactions with the help of interface is introduced. This interface specification is integrated in an overall proceeding and supported by a software tool.

2. Cross-Domain Modelling of Functions and Principle Solutions

The semi-formal specification of functions and principle solutions enables the specialists to communicate, to generate common solution ideas and to find consensus about the principle solution.

2.1 Modelling of Functions

In order to take advantage of the functional structure functions and principle solutions should be modelled on a semi-formal level and integrated into a common model [Gausemeier/Möhringer 2001]. Therefore basic constructs for the modelling of functions are used. A method allows the modelling of polyhierarchical relations between functions and working principles/solution elements.

Basic constructs for the modelling of functions: Basic constructs are the known symbols for the representation of functions, cf. [Roth 2000] (figure 1). The emphasis lies on the functional decomposition, because in that two important effects are seen for the designing of mechatronic

products: the reduction of the complexity as well as the preparation of the partitioning, i.e. the distribution into working principles/solution elements of the involved domains and their assignment to functions. In order to increase the clarity of large hierarchical functional structures subfunctions can be aggregated to subsystems (see figure 2).

Polyhierarchical relations between functions and working principles/solution elements: The assignment of working principles/solution elements to functions can not always be realized as a 1:1 relationship; it is rather cross-linked in a polyhierarchical way [Roth 2000]. This concerns for example solution elements with a carrying function as the housing which fulfills several subfunctions (attach, carry, seal etc.). The modelling of these polyhierarchical relationships is of special importance in order to use the principles of functional integration and separation systematically and to recognize their effects at an early stage.

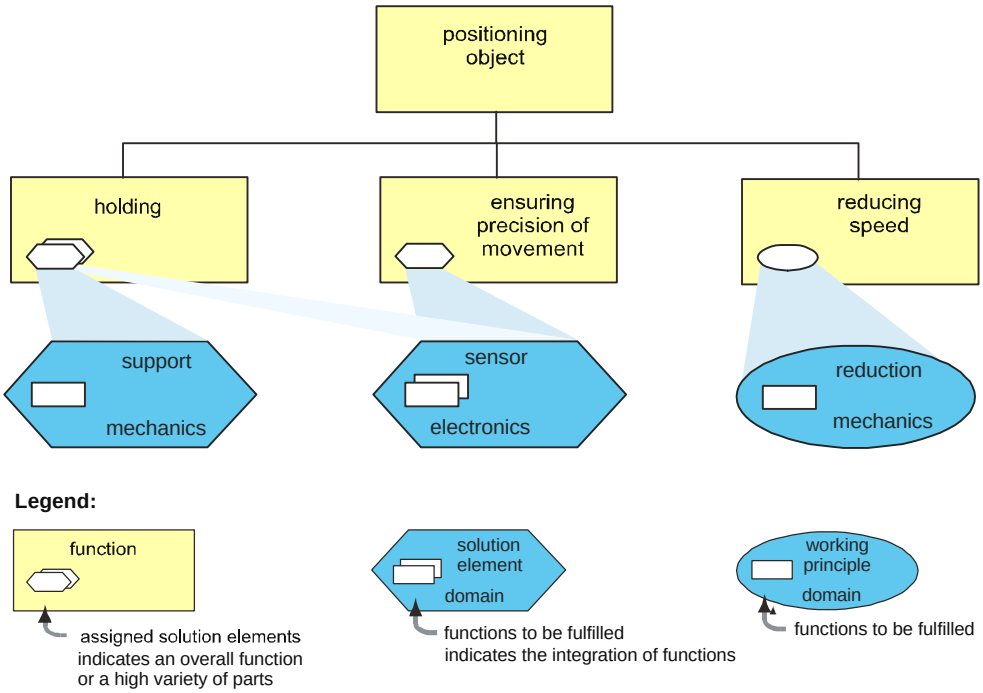

Figure 1. Modelling of polyhierarchical relations between functions and working principles/solution elements

Does e.g. a solution element fulfill several wanted functions (functional integration), the unwanted, so-called parasitic functions increase mostly as well which come from this solution element [Kallenbach et al. 1997]. If these reciprocal actions are made visible, the developers can choose the functional integration in a way that preferably few parasitic functions arise. The relationships between functions and working-principles are modelled in figure 1: The reference of a function to a or several working principles/solution elements occurs by the reduced corresponding construct. In this example the function "holding" is fulfilled by the solution elements "support" and "sensor". In both views – functional view and view of working principles/solution elements – the relationship between functions and fulfilling elements remains always visible. This encourages the designer to apply the principles of functional integration and separation and to check the emerging effects permanently.

2.2 Modelling of Principle Solutions

The principle solution is the result of conceptual design: It is considered as the coarse, but fundamental determination of the physical and logical mode of action of the future product. In the following a graphical method for the integrative and cross-domain modelling of principle solutions is presented [Kallmeyer 1998], [Gausemeier et al. 2001] (figure 2).

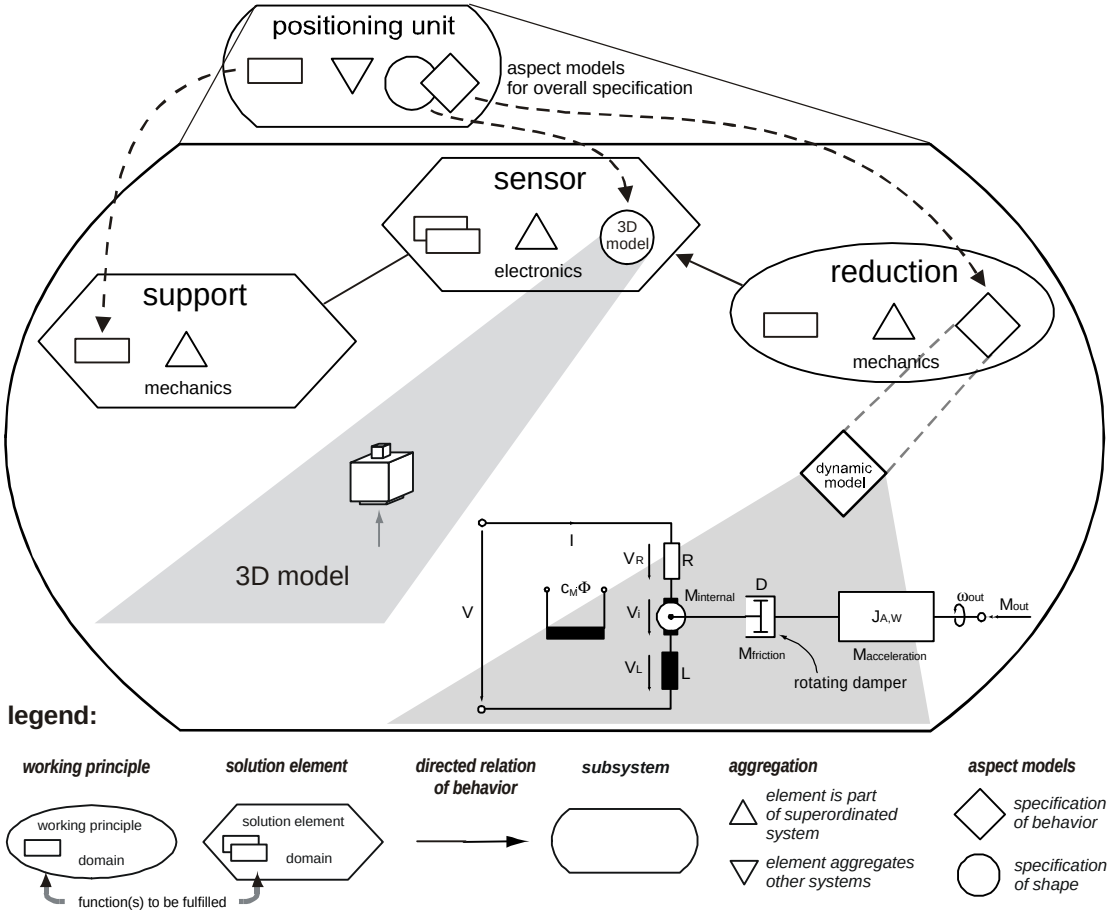

Figure 2. Example for the modelling of principle solutions of mechatronic systems – hierarchical network of working principles, solution elements and their relations

System elements are working principles and solution elements which are graphically represented and characterized be the name and the corresponding engineering domain. Working principles are represented by ovals whereas solution elements are represented by hexagons. Describing the relations between system elements the principle solution can be modelled as an entire system. Relations of behavior show the cooperation between elements. Relations for boundary conditions describe non-functional dependencies as e.g. the determination of the hardware for working principles of the domain software.

System elements can further be aggregated because of logical or spatial aspects. A logical aggregation indicates function units. For this purpose sub-systems are modelled. The spatial hierarchical modelling may be deployed if the spatial design governs the product design. Within mechatronic product design the spatial design is mostly less important than the functional aspects. Aggregated elements are grouped together in a sub-system (positioning unit in figure 2). The grouped elements are indicated by an arrow pointing upwards. Additionally the name of the sub-system is noted. The sub-system contains an arrow pointing downwards to indicate further elements on a more detailed level.

Behavior and shape of the system elements are specified by the proper models which are known and established within the different engineering domains. In figure 2 for example the shape of the position sensor is specified by a 3D-CAD-model, the behavior of the working principle reduction by a dynamic model. These aspect models are part of the overall aspect models specifying the total behavior and shape of the superordinated system (positioning unit in figure 2). It is not aim of this graphical method to introduce an “esperanto” for the specification of behavior and shape. Therefore established domain-specific system description techniques such as VHDL, Petri Nets or UML are used and integrated. This ensures a smooth transfer to the following domain-specific design steps. Interactions between components can be analysed e.g. with the matrix of interaction [Gausemeier et al. 2001] and modelled

with the general flow variables *material*, *energy* and *information* and block representation [Pahl/Beitz 1996] or bondgraphs and statecharts [Schön/Meerkamm 1999].

3. Interface Specification

After having analysed and modelled the interactions of the principle solution on a cross-domain level it is necessary to continue detail work in each engineering domain. According to modular design the interactions should be localized within modules and the interactions between modules should be minimised [Bertram et al. 2000], [Smith/Duffy 2001]. In order to meet the overall system requirements the identified interactions between modules can be specified by interface. The interface specification consists in a standardised description of input and output features with corresponding values of each module. By connecting corresponding input and output features of different modules, the interactions can be specified. The interface is represented by a table with columns for features, target values, modification values (change request) and actual values, each for input and output of one component. The interface specification will be used on the basis of the modelled principle solution which comprises the modules and their basic interactions (figure 3).

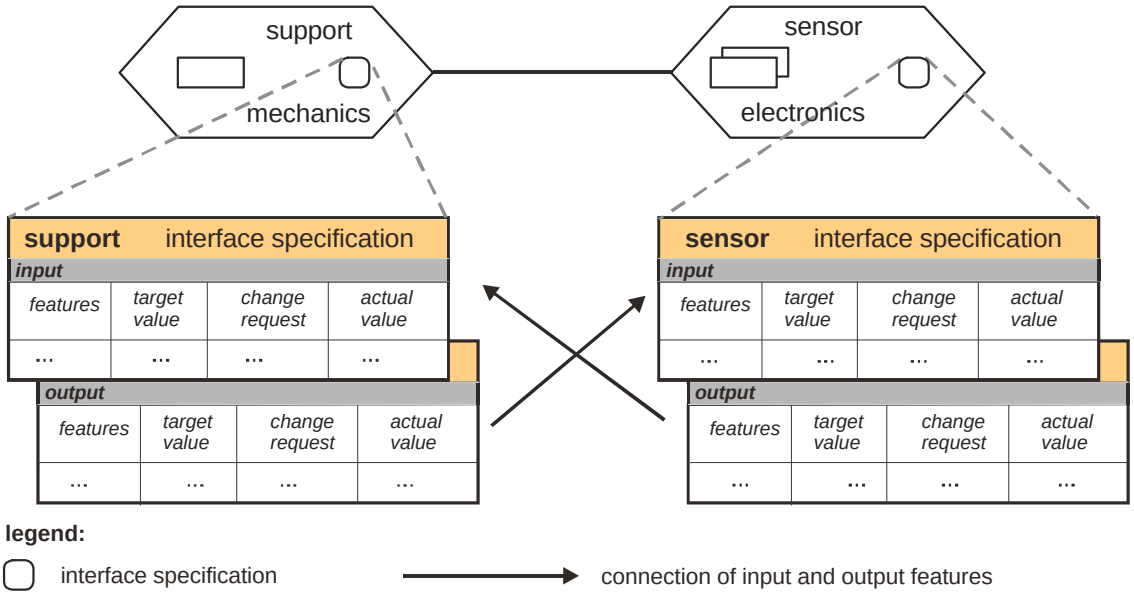

Figure 3. Interface specification linked to each working principle/solution element

The designer will specify first the features and the target values (e.g. from the requirements list) of the modules he is responsible for. The principle solution model (including appropriate aspect models) indicates him the interactions with other modules. In cooperation with the involved domains the designer will then specify in detail the interface by connecting input and output features. In the range of the target values he can independently design his modules and define the actual values. As soon as he needs to differ from the predefined values the column for change request becomes important. He can enter the new value and this value will automatically appear in the corresponding interface tables as a request. If the colleague accepts the requested value it will be transformed to the new actual value. An example explains how the communication between designers is supported (figure 4). According to the cross-domain concept the target value for the frequency of the step motor is 800 Hz. During the domain-specific design the responsible designer needs to change the value of frequency. He will enter a new target value into the column “change request – step motor”. The request will automatically forwarded to the interface specification table of the related control unit and appears in the column “change request – interface”. The responsible designer of the control unit can now investigate whether the new target value is acceptable within his design restrictions. If he accepts he will confirm the request entering “o.k.” in the column “release – control unit”. The release will as well automatically forwarded to the interface specification of the step motor (column “release – interface”).

The change management process has been successfully finished, the new target value will be changed to 900 Hz. The tool-based interface specification has been introduced within an industrial project.

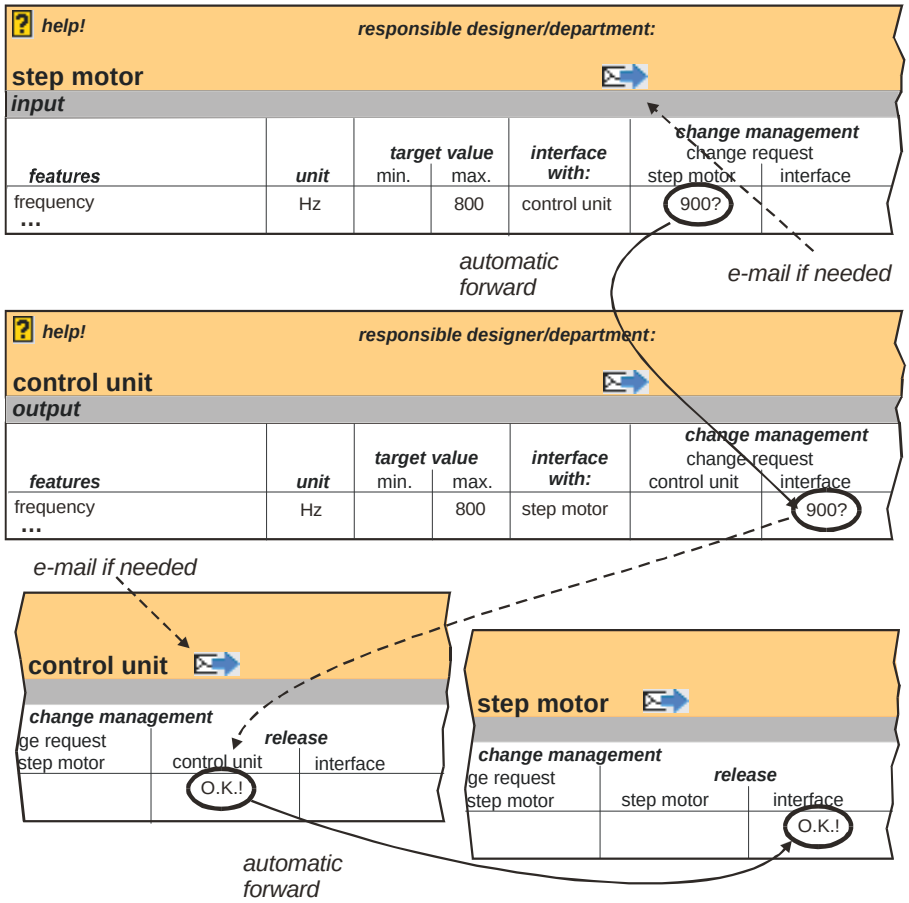

Figure 4. Proceeding of the interface specification exemplified by the change management

4. Software Tool Supporting the Cross-Domain Design

For the cross-domain design a tool was developed on the basis of iGraphx-Professional (Software tool for the modelling of business diagrams, Micrographx, http://www.micrografx.com/igrafx/professional) and tested in industrial projects. It enables the convenient definition and visualisation of the graphical constructs as well as the dynamic alternation between hierarchies and views (functions, principle solutions, domain-specific aspect models, interface specification). In the following the basic functionality will be explained. Figure 5 shows the desk top with the basic function of the tool. The needed constructs are deposited in the symbol library as graphical symbols. After placing the symbol on the graphical surface the input editor in which the necessary dates are specified opens itself. Besides name and domain of the constructs the relationships of aggregation can be defined and the aspect models (behavior model and shape model) and the interface specification can be referenced by arbitrary files. By clicking the wanted interface specification the necessary tool is started automatically (e.g. MS Excel supporting the interface tables) and opens the file. The designer can manage the specifications and communicate with his design colleagues from other departments.

5. Conclusions

A cross-domain interface specification integrated in the semi-formal modelling of functions and principle solutions has been presented. It helps that engineers can work independently based on well-defined interfaces between related modules. Modifications and their effects become transparent; a “negotiation function” ensures that intended modifications of one module leading to an inconsistency with another module can only be done in close co-ordination with the concerned department.

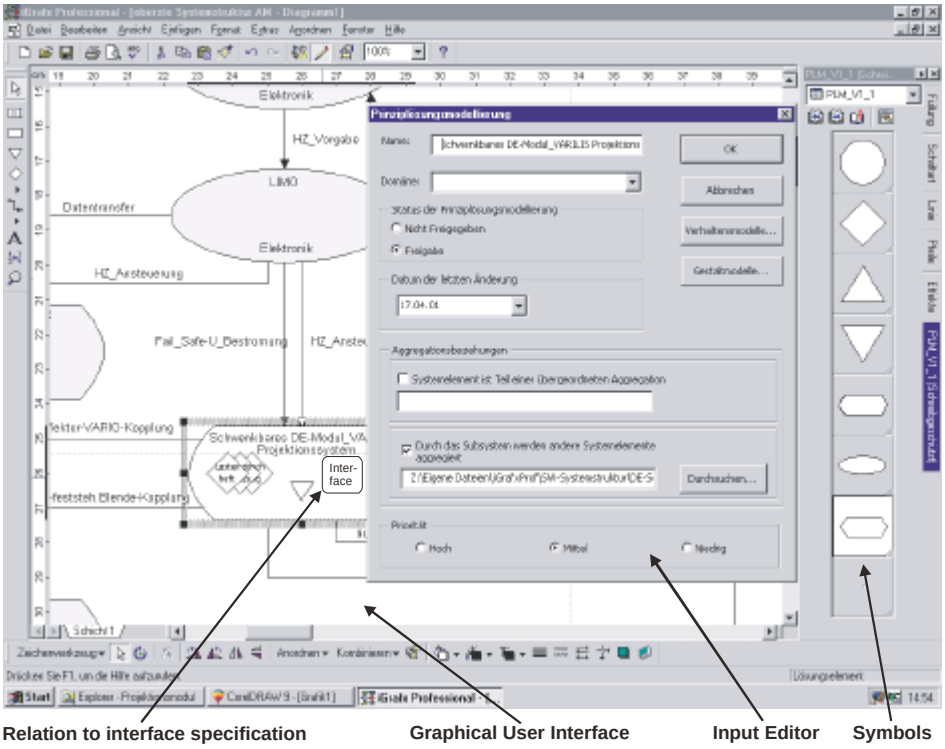

Figure 5. Overview on the basic functions of the software tool

The progress of development during conceptual design is documented frequently in non-structured form. These information can be associated in a simple way and made accessible to all developers. Domain-specific design can be continued based on the same information and software tool.

References

Bertram, T.; Petersen, J.; Flores, P.T.; Lapp, A.; Walther, M.; Schirmer, J., "Objektorientierte Ordnungsstrukturen mechatronischer Systeme", in: Konstruktion,10/2000, pp. 48 – 50.

Gausemeier, J.; Ebbesmeyer, P.; Kallmeyer, F., "Produktinnovation - Strategische Planung und Entwicklung der Produkte von morgen", Carl Hanser Verlag, 2001.

Gausemeier, J.; Möhringer, S., "Integration der Funktions- und Prinziplösungsmodellierung mechatronischer Systeme". In: Design for X, 12. Symposium, Neukirchen, 11./12. Okt. 2001, Erlangen, 2001.

Kallenbach, E.; Birli, O.; Saffert, E.; Schäffel, C., "Zur Gestaltung integrierter mechatronischer Produkte", in: Tagung Mechatronik im Maschinen- und Fahrzeugbau, Moers, 10.-12. März 1997, VDI Berichte 1315, VDI-Verlag, Düsseldorf, 1997, pp. 1 - 14.

Kallmeyer, F., "Eine Methode zur Modellierung prinzipieller Lösungen mechatronischer Systeme", Diss., Fachbereich Maschinentechnik, Universität Paderborn, HNI-Verlagschriftenreihe, Band 42, Paderborn, 1998.

Lippolt, C., "Eine domänenübergreifende Konzeptionsumgebung für die Entwicklung mechatronischer Systeme", Shaker Verlag, Aachen, 2001.

Pahl, G.; Beitz, W., "Engineering Design", Springer, London, 1996.

Roth, K., "Konstruieren mit Konstruktionskatalogen", Band 1, 3. Auflage, Springer, Berlin, 2000.

Schön, A.; Meerkamm, H., "Components for a Mechatronic Design Workbench", in: Proceedings Vol. 2, ICED 99, Technische Universität München, München, 1999, pp. 817 – 822.

Smith, J.S.; Duffy, A.H.B., "Modularity in Support of Design for Re-use", in: Design Management - Process and Information Issues, ICED 01, Professional Engineering Publ., Bury St Edmunds, London, 2001, pp. 195-202.

Dr.-Ing. Stefan Möhringer
University of Paderborn, Heinz Nixdorf Institute, Fürstenallee 11, D - 33102 Paderborn, Germany
Phone: ++49-5251-606267, Fax: ++49-5251-606268
Email: moehr@hni.uni-paderborn.de