INTERNATIONAL DESIGN CONFERENCE - DESIGN 2004
Dubrovnik, May 18 - 20, 2004.

INTEGRATED MODELLING OF CONCEPT SYNTHESIS AND EARLY CALCULATIONS USING EXCHANGEABLE PRINCIPLE SOLUTION ELEMENTS

S. Wilhelms

Keywords: conceptual design, product modelling, concept synthesis

1. Introduction

Design methodology, e.g. [Pahl et al 2003], provides process models (with the major steps task clarification, conceptual design, embodiment design and detail design) as well as structures for describing the designed artefact (horizontal function structure indicating functions and connecting flows between them, vertical causal structure visualised e.g. in function/means trees or working structures).
Due to the nature of conceptual design, early design phases are usually qualitative, e.g. which functions are necessary, how they are decomposed into sub-functions, by which solution principles a specific function can be solved or which one of them is chosen in a specific concept. Quantitative dimensioning is meaningless until the qualitative parameters have been determined ([Koller 1998], ch. 8.4). This implies too often that calculations are not carried out until early embodiment design, where usually only one or a few principle solutions remain.
On the other hand, it is recommended to consider several alternative principle solutions ([Pahl et al 2003], ch. 2.2.4). Therefore, a co-evolution of concepts and early calculations is desirable. When the qualitative determination of the principle solutions and early calculations indicating their properties during concept synthesis are carried out concurrently, a larger number of concepts can be treated quantitatively, and the reached final concepts can thus be verified to a larger extent.
Adequate support is desired in order not to increase the necessary time effort for these additional calculations. Advanced information technology offers potential support by applying constraint networks, e.g. [Kleiner et al 2003]. Feature-based information models for conceptual design have been presented in literature, e.g. [Golob et al 2002]. A support for integrated embodiment design and calculations has previously proven suitable, see [Lindemann et al 2000], and in this article, a similar support for integrating qualitative conceptual design and early calculations is elaborated on. In this case, functions, solution principles (means) and concepts with their respective parameters are modelled, cf. the information model proposed in [Wilhelms 2003].
This article focuses on how principle solution elements should be modelled in order to facilitate easy exchangeability of their quantitative description. During conceptual design, there is a need of being able to rapidly change solution elements in order to explore the solution space and reach the best qualitative principle solutions. Such a change could for example occur when noticing that a motor will have an insufficient torque for a specific task, and therefore needs to be replaced by a motor with gearbox. As the fulfilled function “provide torque” is the same, the solution elements “motor” and “gear motor” should be modelled in a modular way so that they can be exchanged in a single operation, without the calculation model inside these elements needing any additional manual corrections.

2. Method

Starting from the basic idea of the information model for concept synthesis from [Wilhelms 2003], a strategy and methods for easily exchangeable principle solution elements have been developed by theoretical reasoning.
To verify the suitability, the methods have been implemented in a software prototype, which permits the change of principle solution elements by a single operation in a function/means tree view. The constraint network is automatically updated to reflect the new state. The suitability has been studied by modelling an example from a design task from industry previously carried out by students.

3. Results

The approach presented here uses requirements, functions, means, parameters and constraints to describe principle solutions. The strategy for modelling concept synthesis is to apply both a function/means tree (cf. [Hansen 1995]) and a parallel constraint network in an integrated way (see 3.10), in this way storing the developed concepts, permitting early calculations on them and allowing an easy exchangeability of elements (see 3.2). As there appears to be no general and systematic method for breaking down the specifications of a composite system into specifications for the sub-systems [Hansen 1995], the breakdown is instead made on functions, to which requirements then are assigned.

3.1 Modelling strategy for concept synthesis using constraint network

The concept synthesis uses functions and means, visualised in a function/means tree, cf. Figure 1. Functions (shown as trapezoids) express desired behaviour. Means (shown as rectangles) are used to describe the principles by which the desired behaviour can be acquired. Concepts are constituted by a list of chosen means and a list of chosen parameter values.
Early calculations are described using requirements, parameters and constraints. Requirements (shown as circle with exclamation mark) are used to state the limits of parameters. Parameters (shown as circles) quantify the solutions. Constraints are used to connect parameters and provide a way of explicitly modelling the relations and maintaining consistency by propagating changes.

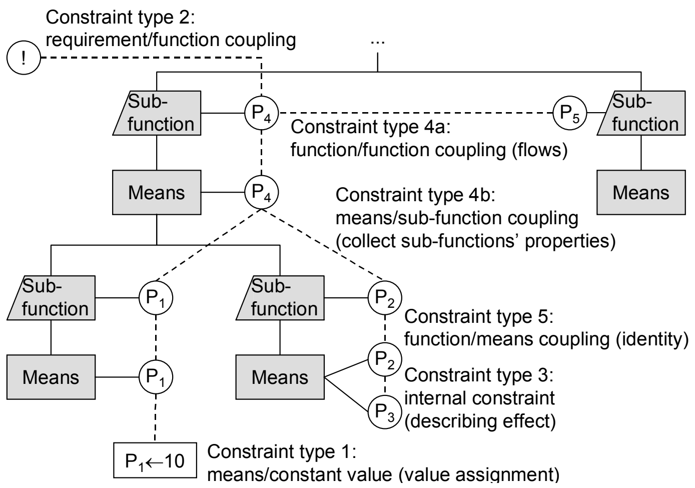

Constraint type 2:
requirement/function coupling
!
...
Sub-function
P4
P5
Constraint type 4a:
function/function coupling (flows)
Means
Constraint type 4b:
means/sub-function coupling
(collect sub-functions' properties)
P1
P2
Constraint type 5:
function/means coupling (identity)
P3
Constraint type 3:
internal constraint
(describing effect)
P1←10
Constraint type 1:
means/constant value (value assignment)

Figure 1. Constraint types

In general, constraints can occur in several forms: as value assignments (a numeric value is assigned to a parameter), as production rules (if condition A is the case, then execute procedure B) or as algebraic constraints (P_1, P_2, … relate according to a formula; equation or inequality), cf. [Lin et al 2002]. Here, constraints (shown as dashed lines in Figure 1) are used for three purposes:

- **Value assignment** (records a decision regarding a parameter value, e.g. the assignment of a concrete value to the transmission ratio of a gear pair, see type 1 below)

- **Requirement–function assignment** (declares a function responsible for fulfilling a goal-setting requirement, e.g. a function “provide torque” is set as being responsible to fulfil a requirement stating the necessary torque, see type 2 below)
- **Relations between parameters** (models various relations between functions and means, see detailed examples for types 3-5 below)

For the mentioned purposes, five constraint types operating on different objects are used:

- Type 1: **Value assignment** to parameters in means (values are never assigned to parameters of functions, cf. 3.1.1), e.g. a concrete value assignment “$i \leftarrow 2$” being made on the internal parameter “gear transmission ratio i” of a means “gear pair”
- Type 2: **Requirement–function coupling** (requirements are never coupled to means in order to ensure easy exchangeability, cf. 3.1.1), e.g. a constraint “$M \geq 10$ Nm” being set on the output flow parameter “torque M” of a function “provide torque”
- Type 3: **Internal constraints** between parameters of the same means (applied to describe physical effects, e.g. the working principle $M_2=i \cdot M_1$ or $i=n_1/n_2$ for the gear pair, but even more complex simulation models are imaginable)
- Type 4a: **External relations** between parameters of two different functions (to describe the function structure, e.g. connecting the input flow “torque M_1” of a gear pair fulfilling the function “multiply torque” to output flow “torque M” of “motor” providing “create torque”)
- Type 4b: **External relations** between a parameter of a means and a parameter of one of its sub-functions (used to collect parameters from sub-functions, cf. 3.1.2, e.g. to calculate a total mass m_{tot} from all contributing sub-functions m_1, m_2 etc.)
- Type 5: **Identity constraints** between a parameter of a function and the corresponding parameter of one of its means, namely the one that is currently selected into a concept (cf. 3.2.1)

The constraints are processed in an incremental constraint network, to reduce recalculation efforts and change as little of the design as possible. As far as possible, all constraints are satisfied in this way, but irresolvable states (conflicts) can occur.
While choosing or modelling means is the method applied to define the design degrees of freedom, value assignments are the central method to determine design degrees of freedom. Choosing a transmission ratio for a given output torque will e.g. result in a determined input torque. If no value is assigned to the transmission ratio, a degree of freedom in design is prevailing, as either the input torque or the transmission ratio can be modified to meet the prescribed output torque.

3.1.1 Placement of constraints

The division into internal (type 3) and external (type 4a/b) constraints facilitates establishing constraints that will continue to operate correctly even if the means selection is changed. Instead of placing constraints directly between means’ parameters, external constraints are always set on parameters belonging to functions (i.e. solution-independent, desired functionality), and thus work regardless of which means is chosen. External constraints of type 4a are never directly set on means (i.e. solution-dependent, achieved functionality) in order to ensure exchangeability. Only type 4b is allowed to connect to a means, but will only connect to its own sub-functions. Internal constraints, however, belong to the means and get relevant once the means is chosen for a concept.
The presence of a large number of constraints at lower hierarchy levels would substantially complicate the deselection of a means and selection of another one. The strategy to achieve easy exchangeability therefore is to avoid relations in deep hierarchy levels. Relations are placed at the highest possible level which is semantically correct (an external relation to the machine consuming the torque in the example in Figure 2 would e.g. be placed at parameter “M” of the function “provide torque”, and not at “M_2” of the function “multiply torque”, which is only present in some concepts).
Besides obtaining models with exchangeable elements, the described environment can also constitute a step towards easier supporting collaborative work, when elements are autonomously designed by different individuals (e.g. means maintained by different experts or multidisciplinary work on different means belonging to special engineering disciplines).

3.1.2 Properties from sub-functions

In order to be able to model constraints on the highest possible level, as described in 3.1.1, parameters are duplicated in functions and means, and lower-level parameters are passed upwards to higher hierarchy levels. In the simple example shown in Figure 2, a function "provide torque" can be solved by either a large motor or a smaller motor in combination with a torque multiplier. In the case where the latter is chosen (selected items shown in grey), the provided output torque "M" for the means "geared motor" on the higher level is collected from the parameter "M_2" of function "multiply torque" on the level below. The actual choice is not automated, but made by the user, assisted by the collected properties and the alternatives' degree of requirement fulfilment. In this way, the automatically set constraint between "provide torque" and "geared motor" (cf. 3.2.1) at the higher level is operational regardless of which means is chosen and regardless of how the sub-tree of those means is composed.

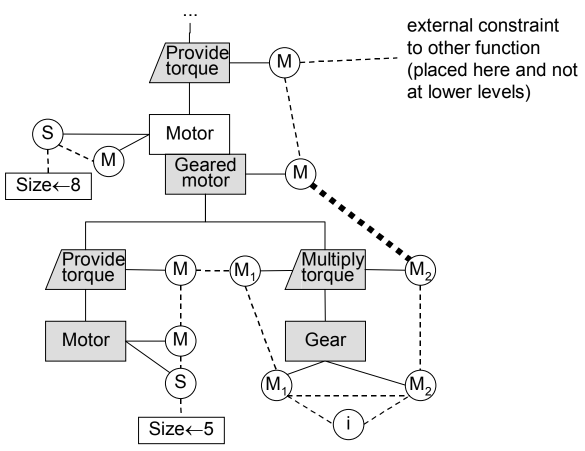


Figure 2. Example of a property (M) being collected from a sub-function's property (M_2)

3.1.3 Explicit value assignments

Value assignments are used to record degrees of freedom in design that have been determined. Examples of value assignments are shown in Figure 2, where two such describe the size of the motor, indicating that the stand-alone motor will have to be larger (e.g. "size 8") than a gear motor (e.g. "size 5"). As it can be seen, value assignments are part of a concept and not of function or means, as the same motor description (e.g. M-n or size-cost characteristics) is used for both size variants.

3.2 Easy exchangeability

In order to ensure the easy exchangeability of principle solution elements, it is necessary to provide methods to perform automatic setting and removal of the couplings between functions and means.

3.2.1 Automatic constraints between function and selected means

Identity constraints (constraints of type 5) are essential for connecting a function and a means that is selected in a certain concept. Upon selection of a means into a concept, these constraints are automatically set, and removed again when deselecting the means. In this way, the general means description is connected to the task-specific function structure. Figure 3 shows an example, where the initially selected "alternative means 1" (left) is deselected and replaced by a selection of "alternative means 3" (right) and the identity constraint (shown as thick line) automatically is adapted.

It is not trivial to identify the right means parameter, to which the function parameter should be automatically constrained. For this purpose, a naming convention, using e.g. the general set of quantities in [Roth 1994], ch. 5.5.1, or the one-time manual assignment of the corresponding parameter pair upon adding the means is feasible. In the prototype implementation, the parameter name, taxonomy and flow type (e.g. "M", torque, output flow) are used to identify the right parameter pair.

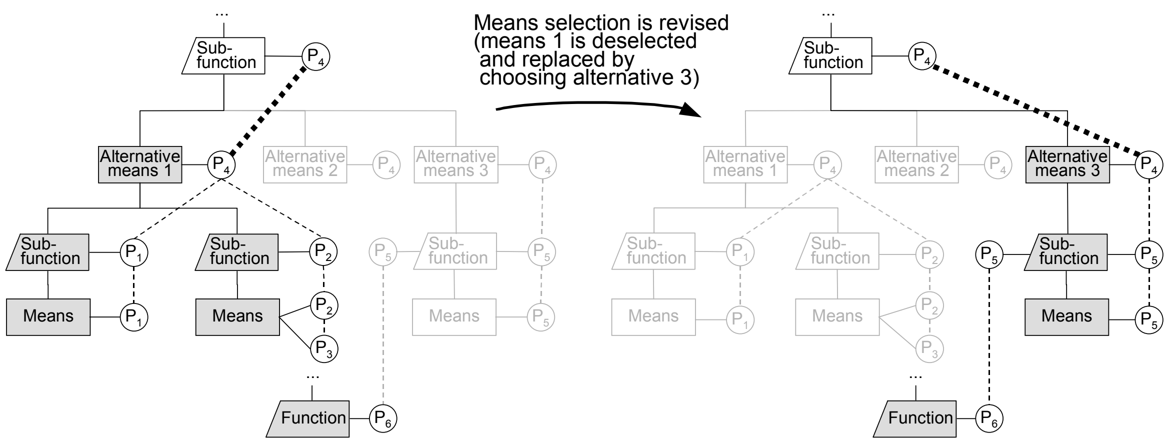
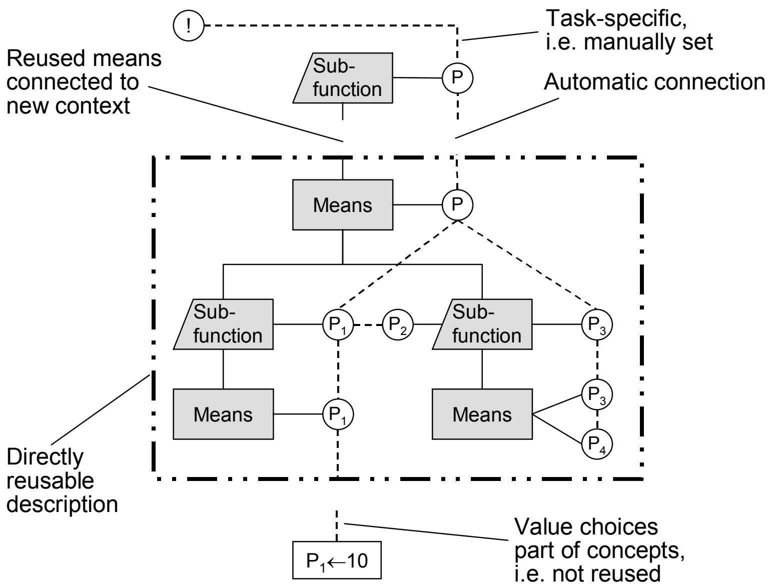


Figure 3. Automatic replacement of identity constraint connecting functions and means

3.2.2 Reuse of entire segments

As widely agreed upon in design methodology, reuse is initiated on the modelled functions. In this way, means (effects with the necessary parameters to describe them) to solve these functions can be found ([Pahl et al 2003], [Koller 1998]). Reuse of entire segments is supported as shown in Figure 4, where the contents inside the thick border are the directly reusable parts of the description.



Task-specific, i.e. manually set
Reused means connected to new context
Sub-function
Automatic connection
Means
Sub-function
Sub-function
Means
Means
Directly reusable description
Value choices part of concepts, i.e. not reused
P1←10

Figure 4. Reuse of entire segment constituting a principle solution element

Not directly reusable are the function on which the search is initiated and the related requirements, which are specific to each individual task. Even the value choices are specific to the concepts and will have to be adapted. Sub-functions and their means including constraints on the contrary are reusable.

3.3 Example

As an example, a rock drill was modelled with a prototype of a software tool implementing the described models and using the presented strategy and methods. The rock drill, taken from an industrial project carried out earlier by students, needs among other functionality the function to transfer the reactive longitudinal forces to the surrounding. In Figure 5, the part of the function/means tree showing the support of the drilling reactive force against the surface of the hole is shown. Means denote entities used to solve a function. In order to better describe the essence of a concept, means can besides function carriers (e.g. rotating rolls) even be abstract principles (e.g. friction), cf. [Andreasen 1980].

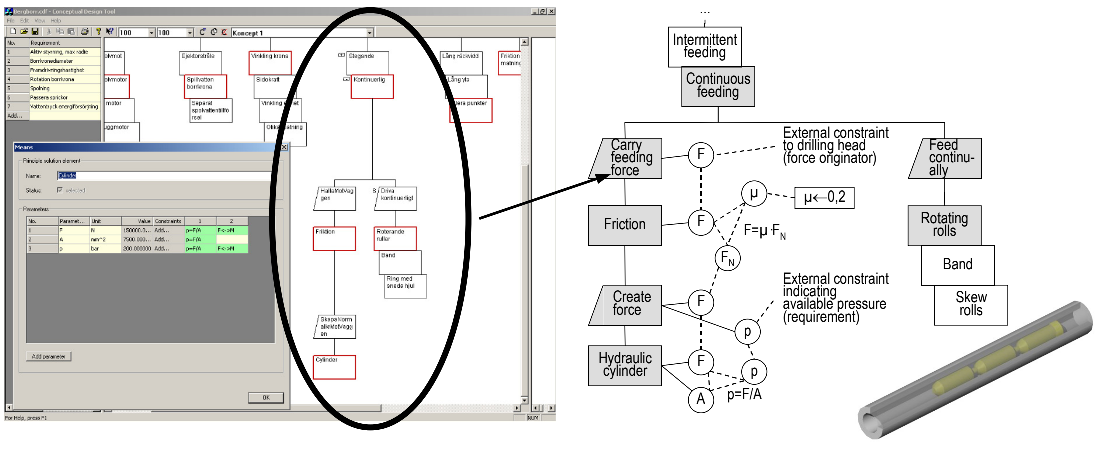


Figure 5. F/M tree view in the prototype (left) and corresponding network segment (right)

4. Conclusions

In this article, the use of constraint networks for supporting both concept synthesis and early calculations is suggested. The presented approach facilitates bidirectional, continuous integration of concept synthesis and calculations. Bidirectional refers to the propagation of changes (i.e. that changes regarding the requirements or means selection are propagated to the calculations, and that changed values in calculations are propagated to functions and requirements). By supporting the determination of quantitative properties already in the early phases, concepts can easier be validated. It is, however, worth noting that it is not trivial to determine the properties in early phases, but already the work with a subset of all properties, namely the ones that are conceptually important, should yield advantages in conceptual design. Easy exchangeability of principle solution elements facilitates exploring the solution space and finding the best quantitative solution.

References

Andreasen, M. M., "Syntesemetoder på systemgrundlag", diss., Lunds tekniska högskola, 1980.

Golob, B.; Jezernik, A.; Hren, G., "A feature based approach for conceptual design", Proc. Design 2002, Vol. 1. Zagreb : Faculty of mechanical engineering and naval architecture/The design society, 2002, pp 483-488.

Hansen, C. T., "An approach to simultaneous synthesis and optimisation of composite mechanical systems", Journal of engineering design, Vol. 6, No. 3, 1995, pp 249-266.

Kleiner, S.; Anderl, R.; Gräb, R., "A collaborative design system for product data integration", Journal of engineering design, Vol. 14, No. 4, 2003, pp 421-428.

Koller, R., "Konstruktionslehre für den Maschinenbau", Berlin : Springer, 1998.

Lin, L.; Chen, L.-C.: "Constraints modelling in product design", Journal of engineering design, Vol. 13, No. 3, 2002, pp 205-214.

Lindemann, U.; Amft, M., "Rechnergestützte Integration von Gestaltung und Berechnung im Konstruktionsprozeß", Konstruktion, Vol. 52, No. 10, 2000, pp 78-82.

Pahl, G.; Beitz, W.; Feldhusen, J.; Grote, K. H., "Konstruktionslehre", Berlin : Springer, 2003.

Roth, K., "Konstruieren mit Konstruktionskatalogen", Vol. 1: Konstruktionslehre, Berlin : Springer, 1994.

Wilhelms, S., "A conceptual design support system using principle solution elements", Proceedings ICED 2003, Stockholm, Glasgow : The design society, 2003, pp 295-296.

Sören Wilhelms
Linköpings universitet, Department of Mechanical Engineering, Division of Machine Design
581 83 Linköping, Sweden
Telephone: +46 13 28 27 30, Telefax: +46 13 28 56 70
E-mail: sorwi@ikp.liu.se