7th Workshop on Product Structuring – Product Platform Development
Chalmers University of Technology, Göteborg, March 24-25, 2004

FUNCTION-MEANS BASED CONFLICT ANALYSIS IN CONCEPTUAL DESIGN OF SYSTEM PRODUCTS

Hans Johannesson

Abstract

Product development is a process of uncertainties and mistakes, resulting in problems and rework. A key issue, when it comes to avoiding unnecessary rework, is the ability to evaluate intermediate steps during the development process in order to make sure that design decisions taken are feasible and will not cause downstream problems leading to backwards iteration loops during the process. Of outmost importance is the result from the concept design phase, as the selection of non-robust, non-feasible design concepts most certainly will result in a lot of downstream problems. This work therefore aims to contribute to the development of new models and procedures for analysis of potential conflicts in early system design concepts, within the framework of an integrated product life cycle model.

Analyses referred to in this work are analyses on system concepts during the system configuration design stage, when the information about the different items constituting the system to be is very limited. Target specification information such as over-all required functional system behavior and over-all system constraints are known, but descriptions of solutions fulfilling these are not.

The relation structure in a function-means tree model is used as a base for a conflict analysis procedure. Sub-solution interactions, modeled by the relation structure in the functions-means tree, have been used to identify functional couplings, in the sense of axiomatic design. The idea has then been to use the degree and character of these couplings to identify and characterize conflicts within the product system.

The results of the work so far show that relation structure based analysis is a feasible way to identify potential system conflicts in early conceptual and configuration design. As the system models grow, they however soon become hard to handle without computer support.

Keywords: Conflict analysis, function-means tree, product model, relation structure, functional requirements, constraints, interactions

1 Introduction

A well-established need for companies that develop, manufacture, and sell complex products is to be able to define and manage information about the product throughout all stages of the product life cycle. Focusing on the product development phase, there is a need to define and manage product information from early stages of development where information is quite incomplete and most often also inconsistent, and through the development process until the product information is complete and consistent. A complete and consistent product description is mandatory to be able to support design and manufacturing operations, sales and marketing, as well as service and maintenance.

The supporting information systems have to be firm in order to guide the process from its incomplete and inconsistent early state towards the complete and consistent state when the product is released for manufacturing. Yet the same supporting information systems also have to be very flexible to support the different situations and needs that inevitably occur, especially during early phases in the development of complex products.

The proposed conflict analysis procedure should be seen in a configuration and concept design context, when the information about the product to be developed is very limited. Even though this is the case, important design decisions with severe downstream consequences have to be made at this early stage. It is therefore of outmost importance to make use of the information available in the best possible way. The selection of non-robust, non-feasible design concepts most certainly will result in a lot of downstream problems. This work therefore aims to contribute to the development of new models and procedures for analysis of potential conflicts and their consequences in early system design concepts, within the framework of an integrated product life cycle model.

2 Background

In earlier work, [1, 8 and13] the author, with co-authors, has put forward generic product and product platform models. These models comprise an integrated requirements and concept part, based on function-means and axiomatic design approaches, which captures the conceptual part of a products or product platforms design rationale and design history. In this work the relation structure in such models is used, together with previous results from research on functional coupling characterization and analysis [4, 6, 7 and 14], as a base to propose a conflict analysis procedure applicable on system product concepts.

2.1 Requirement and concept model

The model concept we have introduced is a *system structure* composed of *a hierarchical function-means tree* in order to capture the design intent, rationale and history of the platform based products, and a *configurable component structure* that has the capability to define the generation of a *representation of ′physical′ hardware and software part structures*.

In order to support both early phase development, with incomplete product descriptions, and the need for detailed part representations for manufacturing purposes, different representations of the product items are used. These representations are objects in an object-relation model. Product requirements defining the functionality desired and constraining factors are modeled as *functional requirements* (FR:s) and *constraints* (C:s) respectively. A design solution fulfilling a functional requirement is modeled as a *generic design solution* (DS). A DS is the generic carrier of the wanted function or the generic means of realizing it. Note that the mapping FR → DS is 1 → 1, i.e. each FR is solved by one, and only one, chosen DS. Before the final choice is made there may be alternative DSs to consider. To implement generic design solutions into configurable realizable product items, *configurable components* (CCs) are used. A CC is a parametric representation of a family of realizable product items (hardware or software) that can be configured into an instance variant, given the current instantiation parameters. Instantiation means that the CC configures and defines an instance representation referring to ´real physical articles´.

To more fully describe and explain the product representations, documents, attribute lists and models can be linked to the DS, FR and C objects. ´Olsson matrices' can also be linked to these objects, as aids for sorting descriptive information into different life cycle phases and aspect domains. The ´Olsson matrix´, which was proposed by the late Freddy Olsson, former professor at Lund University, in his doctoral thesis from 1976, is a matrix based checklist with product life cycle phases as rows and product related aspects as matrix columns. The modeled objects and their linked items are shown in figure 1.

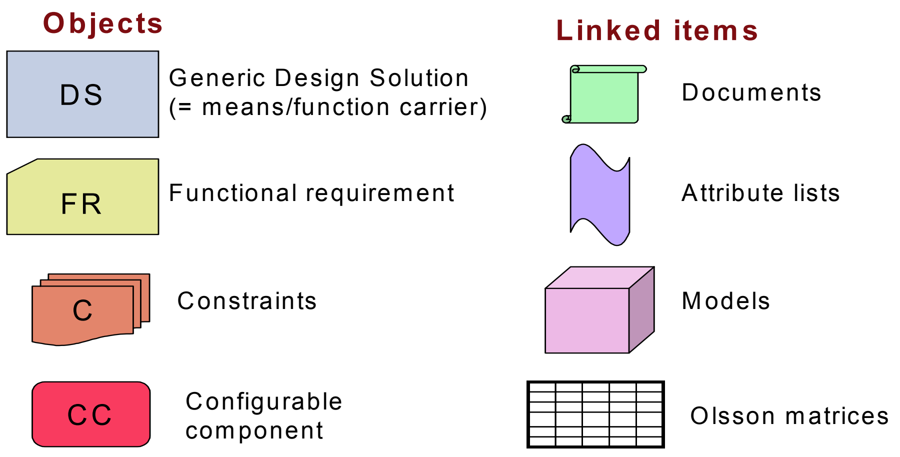

Figure 1. Modeled objects and linked items

The relations (1/1=one to one, 1/n=one to many, n/1=many to one) between the different objects and the linked items are as follows:

- DS to FR (1/n): *rf = requires_function*
- FR to target DS (1/1): *isb = is_solved _by*
- FR to non-target DS (1/n): *iib = is_influenced_by*
- DS to C (1/n): *icb = is_constrained_by*
- C to DS (1/n): *ipmb = is_partly_met_by*
- DS to DS (1/n): *iw = interacts_with*
- CC to DS (1/n): *iaio = is_an_implementation_of*
- CC to Parts (1/n): *iprb = is_physically_realised_by*
- CC to lower level CC (1/n): *icu = is_composed_using*
- Objects to attribute lists (1/n): *ha = have_attributes*
- Objects to documents (1/n): *hd = have_documents*
- Objects to models (1/n): *hm = have_models*
- Objects to matrices (n/1): *hom = have_Olsson _matrix*

Consider as an example a functional requirement FR = ´Carry load´, with a linked attribute list with the attributes ´static load = X kN´ and ´dynamic load = Y kN´, a linked document containing requirement related product planning background information and another that contains building construction safety regulations. This FR ´is solved by´ the generic design solution DS = ´Console´ which has a link to a CAD file with a model of the console geometry and another link to a document containing motives for the taken design decisions related to this chosen DS. The DS is also ´constrained by´ the constraining requirement C = ´use approved standard profiles´, and this C is further connected to a standard profile catalogue via the concerned life cycle cell in a linked ´Olsson matrix´. The final implementation of the DS, i.e. the description of the console family and the rules for creating console family members, will be found in the linked object CC = ´Configurable console´. This CC contains a parameterized solid CAD model, rules for input parameter variation, an implemented method for retrieval of standard profile data from the standard profile catalogue as well as for generating the output information needed to manufacture a console instance.

Using this set of objects with linked information it is possible to describe both WHAT to design (FRs and Cs with attributes and linked documents containing regulations, standards and other referenced background information), and HOW (DSs with attributes, linked describing models and reference documents containing motives, decisions, and other relevant information) a design solution has come to be what it is. I.e. major parts of the design history can be handled in a structured manner with this approach. By combining this description with the CC concept it will then also be possible to generate the information needed to produce product instances.

The creation capturing part of the proposed product platform model is based on the enhanced function-mean tree [1, 2 and 13] with its linked items. The function-means tree is a graphical hierarchical representation showing the primary functions of a machine system supported by a hierarchy of subordinate functions, which are the requirements on the chosen means, i.e. design solutions (DSs, organs or "function carriers") fulfilling the primary functions (see figure 2).

The function-means modeling procedure is performed top-down. Starting from the top DS object, the overall FRs and Cs with linked items are identified and modeled. Then the design solutions to these FRs are derived, considering the portion of the overall Cs that are relevant for each derived DS. The new DSs and their Cs are now modeled on the next level, and the modeling proceeds from each new DS in the same manner as above.

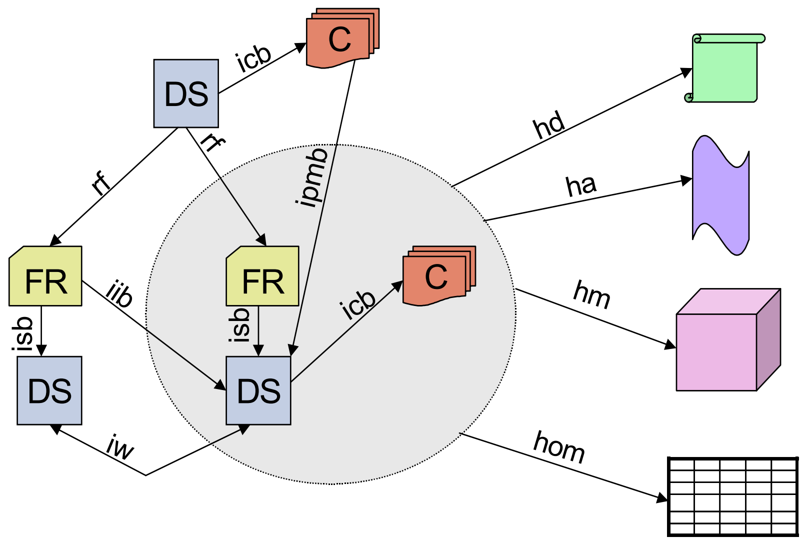

Figure 2: Function-means tree with linked items

An entity-relationship model (ER model) for the function-means tree in figure 2 is shown in figure 3.

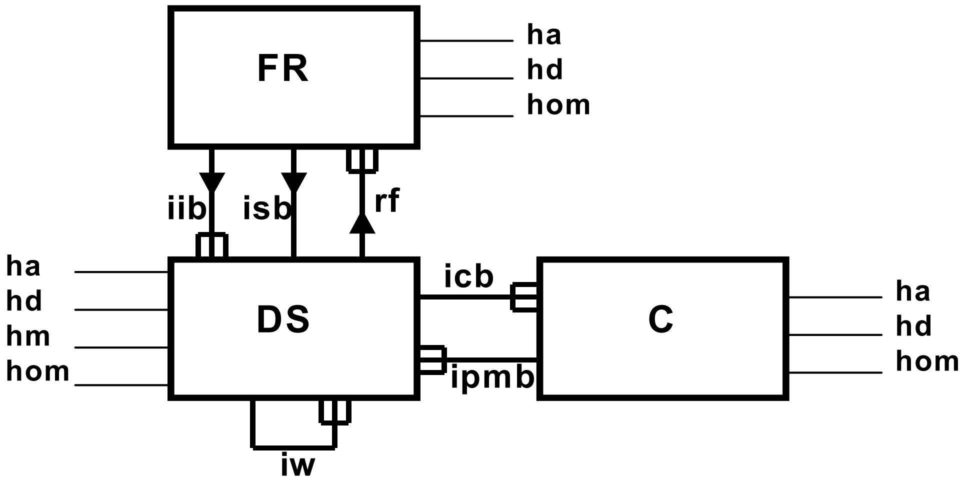

Figure 3. ER representation of function-means tree with linked items

The complete product platform model is established by linking the bottom level DSs in the function-means model to the bottom level CCs in the CC structure. The actual linking is equivalent to an identification and grouping of DSs, and their mapping, using the *′is an implementation of″* (***iaio***) relationship to a newly created (or existing and appropriate) CC. The main requirement on a bottom level CC is that it shall implement an identified constellation of bottom level DSs [2 and 8].

2.2 Model based system concept analysis

Analyses referred to in this work are analyses on system concepts during the system configuration design stage. At this early stage the information about the different items constituting the system to be designed is very limited. Target specification information, such as over-all required functional system behavior and over-all system constraints, are known in terms of over-all FRs and Cs, but descriptions of solutions fulfilling these, are a priori not. The design solutions, i.e. the different DSs in the DS structure, are evolving during the functional decomposition while creating the function-means tree, and information about the different design solutions are gained and added as attributes, linked documents or models. Depending of what models that are created and linked to a DS, solution characteristics can be analyzed and compared to required properties, but that kind of analyses is not the issue here. Instead analyses of the system concept characteristics, resulting from the chosen design solutions interactions, are focused.

Sub-solution interactions, i.e. interactions between chosen DSs, have in previous works by the author and co-authors been modeled by the relation structure in the functions-means tree [4, 6, 7, 13 and 14]. The basic idea in these works has been to identify functional couplings, in the sense of axiomatic design [12], from the system relation structure, and then use the degree and character of functional couplings to judge the ′goodness′, in terms of internal conflicts and necessary requirement fulfillment trade-offs, of the proposed system concept.

As a base for identification and characterization of functional couplings, Pimmler and Eppingers [11] classification and characterization of parts or sub-systems interactions have been used. Their way to describe and quantify interactions have been adopted and further completed by Johannesson [4 and 6] for functional coupling identification. According to

Johannesson, a functional coupling is a result of the interaction between two ′incompatible′ product items, or between two items where at least one ′constitutively influences′ the other. Functional couplings then lead to functional requirement fulfillment trade-offs in order to remove the incompatibility or the harmful constitutive influence.

The need for functional requirement fulfillment trade-offs is really what these kinds of analyses aim to identify. To be able to do this is very important, as trade-offs most often mean that stated requirements can not be completely fulfilled, when conflicting design solutions are balanced in order to make interacting solutions compatible, or to avoid harmful constitutive influence.

If possible, functional couplings and requirement fulfillment trade-offs should be avoided. If this is not possible they should be identified and modeled, and then be dealt with in the best possible way. The function-means tree model has been shown to be an efficient aid for this purpose. Analysis methods with similar purpose are Design Structure Matrix (DSM) based methods [3 and 9]. In the literature DSM methods have been used for analyzing conflicts related to information availability and sub-system dependencies in both product and process development.

In the present work the frame of reference for internal product conflicts, functional couplings and requirement trade-offs has been broaden, and the procedure for f-m-based functional coupling analysis has been refined.

3 Internal product conflicts

Functional couplings that lead to necessary requirement fulfillment trade-offs do not occur in a system product, unless there are harmful interactions between internal design solutions that cause requirement fulfillment conflicts. Basic key issues to discuss when forming a base for functional coupling analysis are therefore *product requirements* and *internal product interactions.*

3.1 Product requirements

The term “Product requirement” is here defined in the same way as the term “Criterion” was defined by Schachinger and Johannesson [13]. As mentioned in section 2.1, a distinction is made between functional requirements (FRs) and constraints (Cs). An FR characterizes a functional need of a design solution whereas a C bounds its solution space. A design solution is created to fulfill the FR and the effect of the C is to sort out the solution alternatives that are within the allowed solution space. Thus a functional requirement can be said to be a *solution driver* and a constraint a *solution restrainer*.

Product requirements can also, according to Pahl and Beitz [10], be classified as *′demanded′* or *′whished′*. Both FRs and Cs can be classified this way. A demanded requirement must always be completely fulfilled by a feasible design solution, whereas a design solution may be feasible while fulfilling only parts of a whished requirement. Different whished requirements are more or less important to fulfill. The relative importance of a whished requirement is therefore usually stated with a weight factor. Demanded and highly weighted whished requirements are important to fulfill, and design solution interactions that hinder or restrict the possibilities to do that are the conflict causes of interest here. This is always true for demanded or highly weighted whished FRs. For corresponding Cs, further considerations must be made.

Constraints in a system product are different in the ways they are affecting, shared between and handled when satisfied, by different design solutions. There are

1. **´over-all´ constraints**, like quality, reliability, safety and alike, that should be met by all design solutions in a system product. At least on higher hierarchical product levels they are valid equally for all part system solutions. Conflicts caused by this kind of constraints can, but must not, arise in situations where a choice between alternatives is affected by the constraints. If they do, they hinder or restrict a design solutions functional requirement fulfillment. There are also situations, on lower hierarchical levels, when these kinds of constraints are met by transformation into functional requirements that are solved by dedicated design solutions. They can then be involved in conflicts after having been transformed to functional requirements.
2. **´shared´ constraints**, like cost, weight, available total space and alike, that should be shared in different negotiated portions between different design solutions.
3. **´directed´ constraints**, like standardization or manufacturing restrictions valid for a certain derived and chosen design solution. These kinds of constraints can but must not necessarily lead to conflicts with other design solution. If they do they hinder or restrict a design solutions functional requirement fulfillment.

In the following only constraints causing conflicts are discussed.

3.2 Internal product interactions

As mentioned in section 2.2, Pimmler and Eppingers [11] classification and characterization of parts or sub-systems interactions have been adopted earlier and further completed for functional coupling identification by the author. This previous proposition will in the following be somewhat modified.

First of all – the earlier distinction between ´incompatible´ and ´constitutively influencing´ design solutions is no longer regarded to be necessary. The reason is that the resulting effect, of two design solutions not ´fitting together´, and having to be changed in order to ´fit´, in both cases per definition leads to a negotiation with product requirement trade-offs. Constitutive influence is therefore also regarded as a kind of design solution incompatibility in the following. The earlier statement defining ´functional coupling´ can therefore now be restated as follows:

A ´requirement fulfillment coupling´ (RFC) in a system product is a result of the interaction between two ´incompatible´ internal design solutions. To achieve necessary ´compatibility´, design solution changes and product requirement fulfillment trade-offs are needed.

Note the change of expression from ´functional coupling´ to ´requirement fulfillment coupling´ (RFC) in the statement. This is due to the recognition that fulfillment of constraining requirements (Cs) may also be affected by trade-offs or compromises as a result of internal design solution incompatibility.

Interactions of interest between items in ´multi-technology´system products are (following Pimmler and Eppinger):

- Spatial interactions
- Energy transfer
- Signal or information transfer
- Material transfer

´Spatial´ should here be given a wide interpretation and cover both interactions in the geometrical space and in the information technological space. Incompatibilities to consider when analyzing RFCs in ´multi-technology´ system products are:

- **Resource availability** – the available resource (for example space) is not sufficient for meeting the total resource demand from the interacting solutions, and this make them incompatible
- **System technologies** – the proposed different system technologies (including system physics, material, standards, transfer protocols, etc.) for the interacting solutions are not compatible
- **Constitutive influence** – one solution is exposed to a load from an interacting solution, which make the solutions incompatible
- **Time dependencies** – the functions of the interacting solutions are timely incompatible (i.e.can not be executed at the same time)

Interactions between design solutions that are incompatible in any of these respects will thus cause RFCs leading to product requirement trade-offs when redesigning the interacting design solutions to make them compatible.

3.3 Conflict modeling

In a function-means tree model the potential conflicts, characterized as RFCs, are seen [7 and 14] as different kinds of closed relation loops (see figure 4). The first one is a loop starting at a constraint (C) that two or more design solutions have to meet together. The loop goes from the C and follows an ***ipmb***-relation to the first design solution. Thereafter it continues following ***iw***-relations to the rest of the design solutions in the interaction chain. From the last design solution in the chain the loop is closed back to the C by another ***ipmb***-relation (C_a – DS_1 – DS_2 – C_a in figure 4).

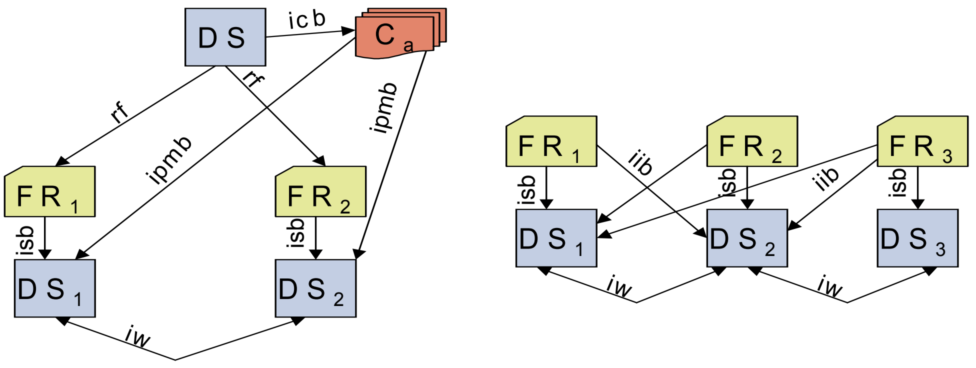

Figure 4. Conflict indicating relation loops

If the previous loop that started from a constraint is denoted ´C-driven´, the next, which starts with a functional requirement, is denoted ´FR-driven´. It starts with an FR and follows an ***isb***-relation or an ***iib***-relation to the first design solution in a chain of interacting design solutions.

It continues, following an ***iw***-relation, to the next design solution and goes back to the FR or follows an ***iw***-relation to the next design solutions in the chain and returns finally from the last design solution following an ***isb***- or an ***iib***-relation back to the FR (FR_1 – DS_1 – DS_2 – FR_1 or FR_2 – DS_1 – DS_2 – FR_2 or FR_3 – DS_1 – DS_2 – DS_3 - FR_3 or FR_3 – DS_2 – DS_3 - FR_3 in figure 4).

The conflict indicating relation loops in the function-means tree can also be given matrix representations. In figure 5 the conflict indicating relation loops from figure 4 are shown using C-DS-, DS-DS- and FR-DS- matrices. A C-DS-matrix describes the relations between the constraints (Cs) of a ′parent′DS and its DS ′children′. A DS-DS-matrix, which actually is a DSM-matrix, describes the internal relations between DSs with the same DP ′parent′. An FR-DS-matrix shows the relations between DSs with the same DP ′parent′ and their governing functional requirements, i.e. an FR-DS-matrix is the same as an axiomatic design matrix.

C \ DS	DS_1	DS_2
C_a	ibmb$_{a1}$	ipmb$_{a2}$
C_b	0	0
C_c	0	0

DS \ DS	DS_1	DS_2
DS_1	▨	iw
DS_2	iw	▨

Relation loop C_a – DS_1 – DS_2 - C_a

FR \ DS	DS_1	DS_2	DS_3
FR_1	isb$_1$	iib$_{12}$	0
FR_2	iib$_{21}$	isb$_2$	0
FR_3	iib$_{31}$	iib$_{32}$	isb$_3$

DS \ DS	DS_1	DS_2	DS_3
DS_1	▨	iw	0
DS_2	iw	▨	iw
DS_3	0	iw	▨

Relation loops: (FR_1 – DS_1 – DS_2 – FR_1), (FR_2 – DS_1 – DS_2 – FR_2), (FR_3 – DS_1 – DS_2 – DS_3 - FR_3), (FR_3 – DS_2 – DS_3 - FR_3)

Figure 5. Matrix representations of C- and FR-driven relation loops

As there are two different kinds of relations involved in each relation loop, two matrices are needed to identify each loop. To see a C-driven loop a C-DS-matrix, showing ipmb-relations and a DS-DS-matrix showing iw-relations, are needed. FR-driven loops containing ***isb***-, ***iib***- and ***iw***-relations requires FR-DS- and DS-DS-matrices to be seen.

A conflict indicated of a relation loop like the C_a – DS_1 – DS_2 – C_a –loop can for example be a geometric tolerance chain where the dimensions of the two solutions are constrained by the over-all tolerance C_a. The FR_1 – DS_1 – DS_2 – FR_1- or FR_2 – DS_1 – DS_2 – FR_2-loops are examples indicating ordinary functional requirement fulfillment conflicts, where interacting design solutions, DS_2 or DS_1, hinder or restrict the fulfillment of the functional requirements FR_1 or FR_2 respectively. We call this an *′ordinary′* FR-driven relation loop.

The FR_3 – DS_1 – DS_2 – DS_3 - FR_3-loop (or the FR_3 – DS_2 – DS_3 - FR_3-loop) can be an example of a conflict where the FR is a transformed over-all constraint which fulfillment is heavily dependent on properties of the interacting design solutions. Such loops are called *′C-transformed′* FR-driven relation loop. That kind of conflicts can for example arise when a geometrical constraint is transformed to a functional position requirement for an assembly, and that requirement is to be solved using geometrical features located on the design solutions, which are parts of the assembly. Note however that the origin of an FR does not affect the following analysis of the relation loop.

4 Design procedure – conflict analysis scenarios

Based on the previous discussion about design solution interactions and requirement fulfillment couplings (RFCs) as causes for conflicts, a design procedure with different conflict analysis scenarios can be identified. The procedure comprises the following common procedural steps when conceptually developing an arbitrary design solution (DS) somewhere in an f-m hierarchy:

1. Identify, and restate if necessary, the Cs that apply on the DS.
2. Formulate the functional sub-requirements (FR_is) for the DS.
3. Seek sub-solution alternatives and chose sub-solutions (DS_is) that fulfill the FR_is and meet the Cs (using morphological and selection matrices).
4. Transfer over-all DS constraints (over-all Cs) from the parent DS to its DS_i children. Indicate with ***ipmb***-relations.
5. Split the shared DS constraints (shared Cs) between the DS_i children. Indicate with ***ipmb***-relations.
6. Add new design solution dependent constraints (directed Cs) to affected DS_i children.
7. Identify and describe interactions between DS_i children and indicate the interactions with **iw**-relations and potentially influenced FRs with ***iib***-relations.
8. Analyze the FR-driven ***isb*** – ***iw*** – ***iib***-relation loops and C-driven ***ipmb*** – ***iw*** – ***ipmb***-relation loops considering incompatibilities in order to identify requirement fulfillment couplings (RFCs).

To be able to fully describe the interactions and the requirement fulfillment couplings on the present hierarchical f-m structure level, it may be necessary to continue the decomposition of some (sometimes all) of the DS_is to the next lower level, following the above procedure in eight steps. The additional knowledge then gained is then brought back up to the higher level to explain and describe what was initially unclear. It might even be necessary to continue the decomposition even further to explore uncertainties at the initial level.

Assuming that all uncertainties have been sorted out, there are now two different analysis and trade-off scenarios to consider:

- **Scenario 1:** Functional requirement trade-offs due to requirement fulfillment couplings
- **Scenario 2:** Constraint trade-offs due to requirement fulfillment couplings

In scenario 1 the FR-driven relation loops with ***isb***- and ***iib***-relations are focused. Each loop here indicates a potential RFC. The first priority in this scenario is to try to change or redesign each influencing DS_i so that the harmful interactions, i.e. the RFCs indicated by the ***iw***- and ***iib***-relations, can be removed or at least diminished. If they can be removed, no functional requirement trade-off is needed, as initially influenced FR_is are not influenced any longer. If they can not be removed, trade-offs between the influenced FR_is and influencing DS_is with their governing FR_is must be carried out.

Removal of ***iib***-relations corresponds to changing non-zero to zero matrix elements in the FR-DS-matrix (or the axiomatic design matrix). When the changes or redesigns of the influencing DS_is are finished, the ***iib***-relation structure in the f-m model and the FR-DS-matrix are also changed. The new structure, or the new matrix, is now rearranged to an uncoupled (diagonal axiomatic design matrix) or semi-coupled (triangular axiomatic design matrix) form if this is possible. If not, a form as close to triangular as possible is desired.

If the final choice of a DS concept has not been made, the redesigned DS alternatives should be compared, by using their rearranged f-m structure or FR-DS matrix representations. The alternatives to choose first are the ones that fulfill all demands and have diagonal or triangular matrices. Besides fulfilling the demands these also fulfill the axiomatic design independence axiom, and are ′good designs′ in an axiomatic sense. If such alternatives can not be found, the ones with FR-DS-matrices as close as possible to diagonal or triangular form should be chosen. Among DS alternatives with identical matrices that fulfill the demands, the one that best fulfills the whished product requirements should be chosen.

Based on the rearranged f-m-structure or matrix of the finally chosen DS concept, an optimal DS_i detail design task plan can be developed [5]. Given the chosen redesigned DS_i concepts, following this procedure will maximize the functional requirement fulfillment of DS, and following the developed task design plan will minimize the number of required design iterations due to RFCs in the DS detail design process.

In scenario 2 the C-driven relation loops with ***ipmb***- and ***iw***-relations are of interest. Also in this case redesign of the interacting design solutions in order to remove the harmful interactions causing the RFCs is the first action to take. If this is possible, a constraint C_x can be transferred to (over-all C), split between (shared C) or imposed on (directed C) the involved DS_is (as C_{xi}s) without trade-offs and compromises between the solutions, as the solutions are no longer interacting. Each DS_i can be developed to fulfill its FR_i and meet its Cx_i without consideration of other DS_is.

If the interaction can not be removed, the nature of the C_x determines the actions that follow. If the C_x is an over-all C that shall be transferred to the interacting DS_is, or a directed C that is to be imposed on some of the DS_is, the involved interacting solutions may have to (but must not) be incompatible in order to meet the transferred or imposed constraint. The DS_is involved in the identified RFCs in step 8 above, have already been recognized to be incompatible, and trade-offs, i.e. adaptations of these involved DS_is with compromises between the fulfillment of their FR_is, are needed.

If the C_x is a shared constraint that causes an identified RFC, a negotiated split of the constraint C_x between the DS_is must be done. The consequences of such a split are adaptations of the involved DS_is with compromises between the fulfillments of their FR_is. This is typical in resource availability incompatibility cases, where the DS_is together require more resources to fulfill their FR_is than the total resource available.

If, in this scenario, the final choice of DS concept has not been made, the number of problematic C-driven relation loops, i.e. RFCs, in the f-m structures or FR-DS matrices of the DS alternatives should be compared. The alternatives to choose first are the ones that fulfill all demands and have the least number of RFCs. Among the remaining DS alternatives, with identical f-m structures or matrices that fulfill the demands, the one that best fulfills the whished product requirements should be chosen.

5 Concept analysis example

The proposed conceptual design and analysis method is in the following described by going through the different steps in designing and analyzing a hydraulic cylinder platform concept, i.e. creating a description of a generic hydraulic cylinder family concept. The hydraulic cylinder example has been chosen of pedagogical reasons and for reasons of complexity. Such a product is complex enough to capture the main design and analysis issues, but is still not too complex to allow the reader to grasp the different items contained and their internal relations.

The point of departure of the configuration, concept design and analysis procedure is that a hydraulic cylinder family or platform is to be developed, considering the overall functional requirements and constraints associated with this particular platform. For the whole cylinder family there are two generic functional requirements:

- FR_1 = Move load
- FR_2 = Contain pressurized fluid

Associated with these requirements are additional information like load, pressure, stroke and velocity ranges and other performance parameters that are not specified further. In this example only three of the constraints on the cylinder family and their implications will be discussed. These three constraints are:

- C_a = X different cylinder sizes are allowed in the cylinder family
- C_b = There is a maximum weight Wx for each cylinder size
- C_c = Use approved standard components where possible

Also for the constraints, there is additional associated information not accounted for here, as it is not contributing to the understanding of focused issues. The design solutions chosen to fulfill the FRs while meeting the Cs are:

- DS_1 = Piston rod system
- DS_2 = Cylinder tube system

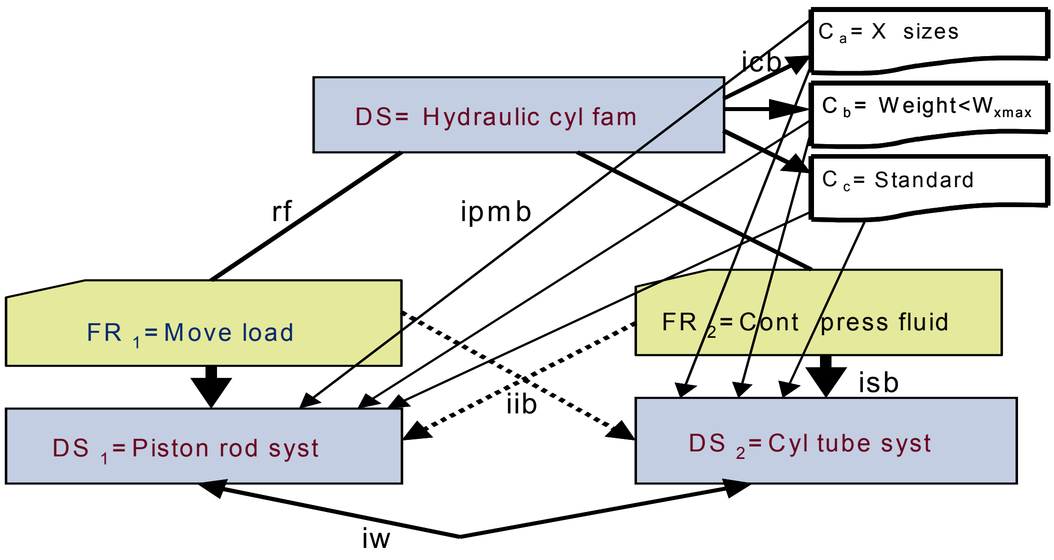

Figure 6. Hydraulic cylinder family f-m model

In figure 6 the first level of the function-means modeled cylinder family is shown. The model reflects the stage of modeling when the design procedure ´8 point list´ has been worked through for the top level DS. For this DS there are three relation loops indicating potential RFCs:

- The FR-driven loop $FR_1 – DS_1 – DS_2 – FR_1$
- The FR-driven loop $FR_2 – DS_2 – DS_1 – FR_2$
- The C-driven loop C_b - $DS_1 – DS_2 – C_b$

Note that over-all constraints, C_a and C_c, are assumed to be met by DS_1 and DS_2 without causing any RFCs. The shared constraint C_b, that is to be split between DS_1 and DS_2, is on the other hand involved in a potential RFC.

The two FR-driven loops indicate that the fulfillment of FR_1 is influenced by DS_2, and that the fulfillment of FR_2 is influenced of DP_1. In the first case the function ´Move load´, is influenced by the counteracting friction force from the ´Cylinder tube system´ on the ´Piston rod system´. In the second case the function `Contain pressurized fluid´ is influenced by leakage caused by the moving ´Piston rod system´. Both cases are coupled, as fluid leakage and friction forces are effects appearing in the piston seal and piston rod seal contacts. Seal contacts with low friction, and a better fulfillment of the functional requirement ´Move load´, result in higher leakage flows, i.e. a diminished fulfillment of the functional requirement “ Contain pressurized fluid´, and vice versa. When designing the seals a trade-off between leakage and friction, i.e. between ability to move load and ability to contain pressurized fluid, is needed. The piston seal is part of the piston rod system and the piston rod seal of the cylinder tube system. Thus the actual trade-off will be done when designing these solutions on lower levels in the f-m hierarchy.

The C-driven loop indicates that the shared weight constraint C_b has to be met by the two interacting design solutions together. This means that the total weight W_{xmax}, which must not be exceeded, has to be split between the two solutions. The portions have to be negotiated aiming at optimizing the summed functionality of DS_1 and DS_2 compared to FR_1 and FR_2.

The continued decomposition of DS_1 is shown in figure 7. The model reflects the stage of modeling when the design procedure ´8 point list´ has been worked through for design solution DS_1.

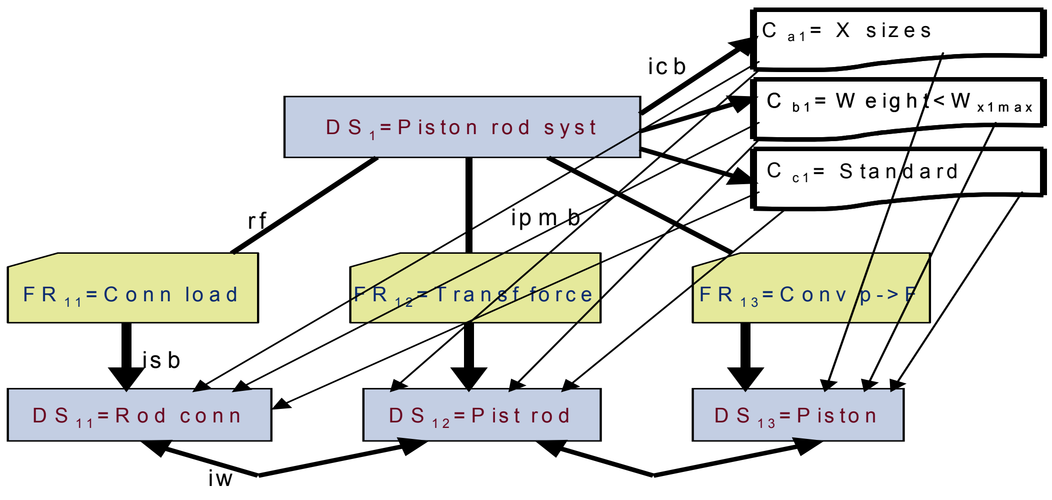

Figure 7. Piston rod system family f-m model

For DS_1 there are no identified FR-driven loops causing RFCs. FR_{11}, FR_{12} and FR_{13} are assumed all to be fulfilled without influences from the identified interactions. As for DS on the previous level, the only constraint that is involved in a C-driven loop causing an RFC, is the shared weight constraint C_{b1}.

The C-driven loop, $C_{b1} – DS_{11} – DS_{12} – DS_{13} – C_{b1}$, indicates that the shared weight constraint C_{b1} has to be met by the three interacting design solutions together, i.e. the total weight W_{x1max}, which must not be exceeded, has to be split between the three solutions. The portions have to be negotiated aiming at optimizing the summed functionality of DS_{11}, DS_{13} and DS_{13} compared to FR_{11}, FR_{12} and FR_{13} respectively.

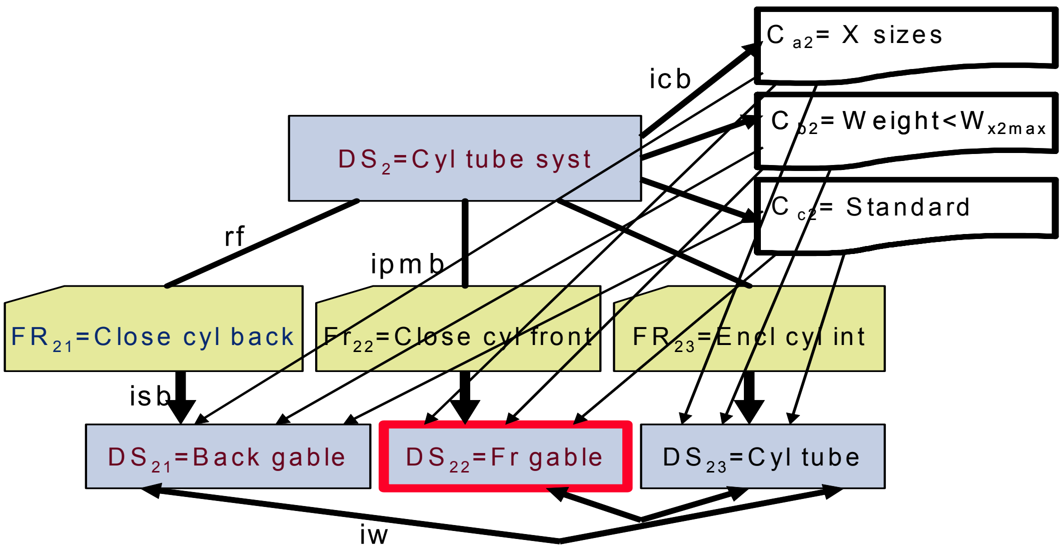

Figure 8. Cylinder tube system family f-m model

The continued decomposition of DS_2 is shown in figure 8. The model reflects the stage of modeling when the design procedure ´8 point list´ has been worked through for design solution DS_2.

Also for DS_2 there are no identified FR-driven loops causing RFCs. FR_{21}, FR_{22} and FR_{23} are assumed all to be fulfilled without influences from the identified interactions. As for DS_1, the only constraint that is involved in a C-driven loop causing an RFC, is the shared weight constraint C_{b2}.

The C-driven loop $C_{b2} – DS_{21} – DS_{23} – DS_{22} – C_{b2}$ indicates that the shared weight constraint C_{b2} has to be met by the three interacting design solutions together. This means that the total weight W_{x2max}, which must not be exceeded, has to be split between the three solutions. The portions have to be negotiated aiming at optimizing the summed functionality of DS_{21}, DS_{22} and DS_{23} compared to FR_{21}, FR_{22} and FR_{23} respectively.

The continued decomposition shall be performed along the same lines as seen above. To illustrate some additional issues of interest only the ´Front gable´ will be further decomposed. This is shown in figure 9. The model reflects the stage of modeling when the design procedure ´8 point list´ has been worked through for design solution DS_{22}.

Within the front gable system there are two identified FR-driven loops indicating RFCs. The gable rod hole and the fluid connection are both interacting with the gable body as they are both integrated parts of the body. The potential RFCs have to do with the gable body dimensions being big enough to incorporate a functional requirement fulfilling rod hole and a functional requirement fulfilling fluid connection. This might lead to necessary functional requirement fulfillment trade-offs concerning FR_{222} and FR_{223} at the detail design stage. The reason may be that the space provided within the gable body will be restricted due to for example weight constraints. Note that FR_{221} is not negotiable as the cylinder front must be covered if the cylinder shall work at all.

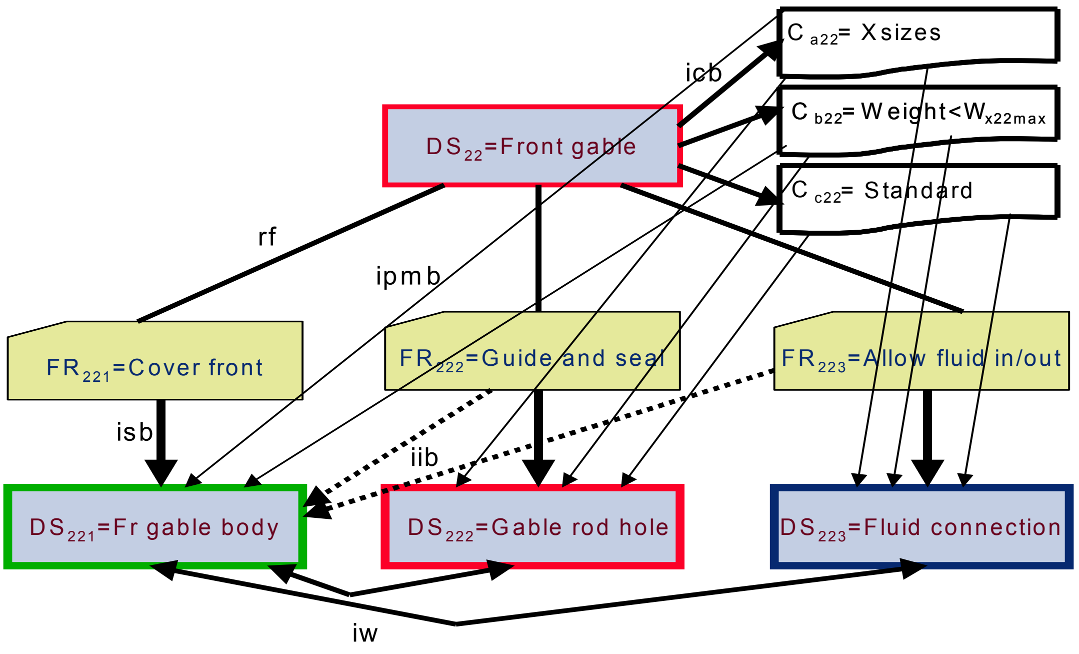

Figure 9. Front gable family f-m model

There is one C-driven loop, $C_{b22} - DS_{222} - DS_{221} - DS_{223} - C_{b22}$, within the front gable system indicating that the shared weight constraint C_{b22} has to be met by the three interacting design solutions together. This means that the total weight W_{x22max}, which must not be exceeded, has to be split between the three solutions. The portions have to be negotiated aiming at optimizing the summed functionality of DS_{221}, DS_{222} and DS_{223} compared to FR_{221}, FR_{222} and FR_{223} respectively. As indicated above, the trade-off is in this case involved in the FR-driven loop caused trade-offs, as the gable body weight might restrict the gable body dimensions.

The final decomposition step shown here is the decomposition of the gable rod hole DS_{222}. This is shown in figure 10. The model reflects the stage of modeling when the design procedure ´8 point list´ has been worked through for design solution DS_{222}. The gable body is not further decomposed. The fluid connection is, but that is not shown here as this decomposition not results in any new aspects that have not already been discussed. The gable rod hole is decomposed into the ´Rod hole´ and three standard components. The ´Rod hole´ is interesting in itself being a design solution with functionality, but without a physical body of its own. Instead it will be realized as a cavity in the gable body.

Within the gable rod hole system there are three identified FR-driven loops indicating RFCs. The three standard components ´Bearing´, ´Rod seal´ and ´Scraper´ are all interacting with the

´Rod hole´ geometry. The shape, dimensions, tolerances and surface roughness of the rod hole geometry are crucial for the functional performances of these three components. The potential RFCs have to do with the rod hole dimensions being sufficient to incorporate a functional requirement fulfilling bearing, a functional requirement fulfilling rod seal and a functional requirement fulfilling scraper ring. This might lead to necessary functional requirement fulfillment trade-offs concerning FR_{2221}, FR_{2222}, FR_{2223} and FR_{2224} at the detail design stage. The reason may be that the dimensions of the rod hole geometry will be restricted by for example weight constraints. Note that part of the actual interaction between DS_1, the ´Piston rod system´, and DS_2, the ´Cylinder tube system´, takes place in the interfaces between the piston rod surface and the bearing, the rod seal and the scraper ring. The additional information about this interaction gained at this level can be transferred back to the higher level to give a more detailed description of the interaction between DS_1 and DS_2. The actual trade-off between the requirement fulfilments of DS_1 and DS_2 is done at this level when the seals and the bearing are designed in detail.

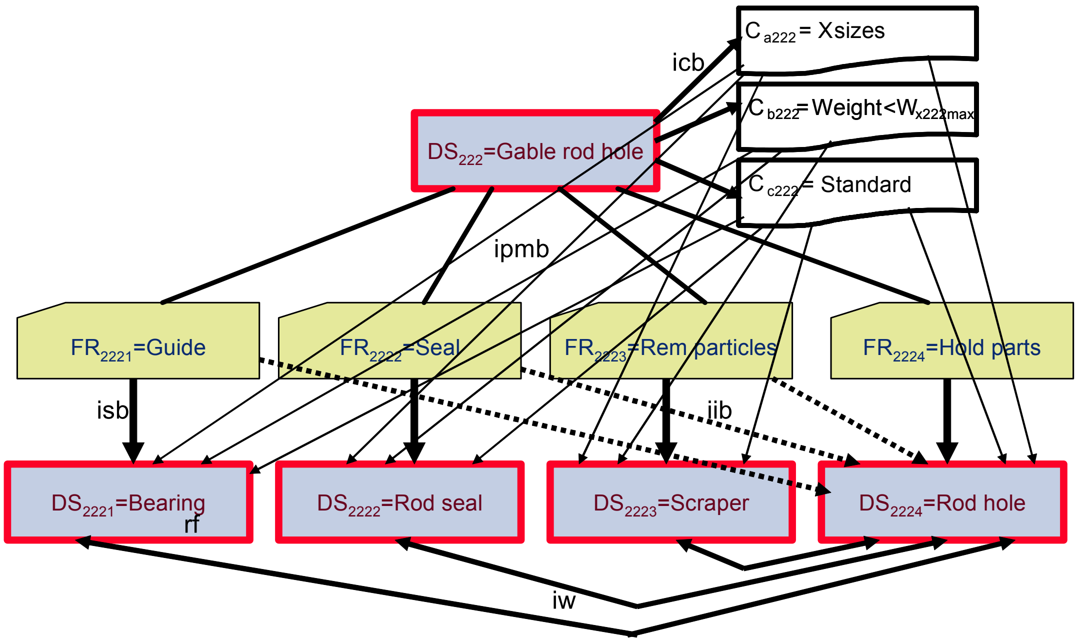

Figure 10. Gable rod hole family f-m model

Within the gable rod hole system, a potential C-driven RFC is indicated by three different C_{b222}-driven interconnected relation loops:

- $C_{b222} – DS_{2221} – DS_{2224} – DS_{2222} – C_{b222}$
- $C_{b222} – DS_{2221} – DS_{2224} – DS_{2223} – C_{b222}$
- $C_{b222} – DS_{2222} – DS2_{224} – DS_{2223} – C_{b222}$

This means that the shared constraint C_{b222} shall be split between *the indirectly interacting* design solutions DS_{2221}, DS_{2222} and DS_{2223}. These three are indirectly interacting as they are all individually interacting with the weightless design solution DS_{2224}, which has no ***ipmb***-relation to C_{b222}, as the rod hole, being a cavity, has no mass. This means that the total weight $W_{x222max}$, which must not be exceeded, has to be split between the bearing, the rod seal and the scraper ring. The weight portions have to be negotiated aiming at optimizing the summed functionality of DS_{2221}, DS_{2222} and DS_{2223} compared to FR_{2221}, FR_{2222} and FR_{2223} respectively

6 Conclusion

The proposed conflict analysis procedure should be seen in a configuration and concept design context, when the information about the product to be developed is very limited. Even though this is the case, important design decisions with severe downstream consequences have to be made at this early stage. It is therefore of outmost importance to make use of the information available in the best possible way. The enhanced function-means tree, with its relation structure, which is created as a first step during the configuration and concept design stage, provide information for useful internal system conflict analysis. The proposed analysis procedure makes it possible to already during the conceptual stage identify, and even resolve, potential conflicts. Based on the analysis results and the insights gained, plans can also be made early to handle identified unresolved conflicts during the following detail design phase. By doing this the risk for unexpected downstream problems causing backwards iteration loops and unnecessary re-work is decreased.

In a future scenario, all product and process information in companies will be handled by computer based product life cycle management (PLM) systems. Such systems will contain integrated product life cycle models of different kinds. Linked design rationale models, here exemplified by the enhanced function-means tree, product property models, for product verification, and other product life cycle models to support different product life cycle activities, should be parts of such future PLM systems. Design rationale models, based on the enhanced function-means tree, have the potential to provide support for analysis and early design decision making. Furthermore, such models with their developed relation structures will also support change management with their built-in traceability capabilities.

References

[1] Andersson, F., Nilsson, P. and Johannesson, H., "Computerbased Requirement and Concept Modelling - Information Gathering and Classification", Proceedings of ASME Design Engineering Technical Conferences, DETC2000/DTM-14561, Baltimore, Maryland, 2000.

[2] Claesson, A., Johannesson, H. and Gedell, S., "Platform Product Development: Product Model – A System Structure Composed of Configurable Components". Proceedings of ASME Design Engineering Technical Conferences, DETC2001/DTM-21714, Pittsburgh, Pennsylvania , 2001.

[3] Eppinger, S.D., Whitney, D.E., Smith, R.P. and Gebala, D.A., "A Model-based Method for Organizing Tasks in Product Development", Research in Engineering Design, 6, 1994, pp. 1-13.

[4] Johannesson, H., "On the Nature and Consequences of Functional Couplings in Axiomatic Machine Design", Proceedings of ASME Design Engineering Technical Conferences, 96-DETC/DTM-1528, Irvine, CA, 1996.

[5] Johannesson, H., "Design Project Planning Using Functional Coupling Analysis", Proceedings of NordDesign '96, HUT, Espoo, Finland, 1996.

[6] Johannesson, H., "Application of Interaction and Functional Coupling Rules in Configuration Design", Proceedings of ASME Design Engineering Technical Conferences, DETC97/DTM-3869, Sacramento, CA, 1997.

[7] Johannesson, H. and Söderberg R., "Structure and Matrix Models for Tolerance Analysis from Configuration to Detail Design", Research in Engineering Design, 12 pp. 112-125. Springer-Verlag London, 2000.

[8] Johannesson, H., Claesson, A., "Systematic Product Platform Design - A Combined Function- Means and Parametric Modelling Approach", To appear in Journal of Engineering Design,Volume 16, Number 1, February 2005.

[9] Krishnan, V., Eppinger, S.D. and Whitney, D.E., "A Model-based Framework to Overlap Product Development Activities", Journal of Management Science, 4, pp. 437-451, 1997.

[10]Pahl, G. and Beitz, W., "Engineering Design – A Systematic Approach", The Design Council, London, 1988.

[11]Pimmler, T.U. and Eppinger, S.D., "Integration Analysis of Product Decompositions", Proceedings of ASME Design Engineering Technical Conferences, DTM-94, Book No.H00914, DE-Vol.68.

[12]Suh, N.P., "The Principles of Design", Oxford University Press, New York, 1990.

[13]Schachinger, P. and Johannesson, H., "Computer Modelling of Design Specifications", Journal of Engineering Design, Vol. 11, No. 4, pp. 317-329, 2000.

[14]Söderberg, R. and Johannesson H., "Tolerance Chain Detection by Geometrical Coupling Identification", Journal of Engineering Design, Vol 10, No. 1, pp. 5-24, 1999.

Corresponding author
Professor Hans Johannesson
Department of Product and Production Development
Chalmers University of Technology
SE-412 96, Göteborg, Sweden
Tel: +46 31 772 1363
Fax: +46 31 772 1375
e-mail: hans.johannesson@me.chalmers.se