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

A REQUIREMENT-DRIVEN PRODUCT DEVELOPMENT PROCESS

Kjell Andersson

Abstract

Product development is often described as an iterative process of finding solutions that match specific requirements. The many dimensions of this process include time, organization, product-specific elements such as the level of abstraction and detail, and analysis to verify the product's properties.

Many types of software tools are used to generate and visualize the concept shape. These include CAD (computer-aided design) tools; tools to simulate and verify product properties, such as FE (finite element analysis) and MBS (multibody systems); and tools for handling product data such as PDM (product data management). This paper focus on the effective use of simulation software such as FE and MBS tools to support the process of verifying that a product meets the formulated requirements. The simulation software can be used for such things as selecting alternative solutions or as a final check or optimization of a solution concept. Its can be used even more effectively if it is supported by a framework for handling the information created during the verification process.

This paper presents a proposal for an information framework that can support traceability and reuse of partial results created during the verification of a specific required attribute. This framework also facilitates study of the effects of changes in the specification on product properties. The framework is illustrated in a modeling and simulation scenario for a lifting unit on a wheel loader produced by Volvo CE. This scenario focus on modeling and simulation activities and how these can be supported in a question-and-answer driven process that investigates the behavior of the lifting unit.

Keywords: Requirements, traceability, modeling, simulation

1. Introduction

A challenging task, rapidly growing in importance in product development today, is handling the large amount of data created during the design process. This task is important, for the data captures a great deal of information that, if reused, may give a company a competitive advantage. For example, the information and knowledge could be used in the redesign of an existing product for a new customer. It could also be used when comparing and evaluating different product concepts, where some of the modeling objects may be worth reusing.

The ability to reuse simulation models is also a crucial issue when it comes to increasing the use of simulation tools in industry, which will contribute to a better understanding of products' behavior and will, in turn, contribute to the design of better products. However, if simulations and other models are to be reused, it must be easy to find these models when we need them. This implies that we must structure and store these models together with their metadata describing the modeling context and the intent of the model.

In this paper the focus is on describing the different objects that represent a verification loop from an attribute in the requirements specification to a decision basis affecting the next gate in the development process. The objects presented here are all implemented as XML objects and the links between them are represented as hyperlinks.

An important property of a verification process is traceability, that is, the ability to trace what objects were created and used during a verification loop. Another important property is that the designer or the team can search for objects created earlier. For a search to be efficient and to incorporate some level of logic, it is important to specify the context in which objects were formulated.

2. Background

The task of tracing dependencies and the relations created during the verification of a product requirement is important for a requirement management system. There is relatively good support for this task in the field of systems engineering, which has developed specialized tools for requirements management [1].

According to Ramesh [2], the traceability requirement can be divided into pre- and post-traceability. The former refers to maintaining the links to the initial customer, and the latter to maintaining the links to the subsequent design solutions through functional modeling and allocation. Blanchard and Frabrycky [3] advocate a modified variant of QFD [4] as a tool for breaking down customer requirements to obtain system and subsystem requirements and eventually component requirements.

According to Sutinen [5], three basic techniques can be used to manage and maintain traceability. These are

- traceability tables
- traceability lists
- automated traceability links.

Ramesh [2] also discusses a framework for model traceability and an intelligent decision support system that uses this framework to support model development and maintenance activities in software development.

The set of models presented in this paper relate to post-traceablity. They make it possible to trace models produced during the verification process initiated in response to a stated requirement specification that eventually produced a decision basis. In order to achieve this traceability, the product model has to be extended to incorporate a number of new types of objects, such as problem statements and model specifications. Furthermore, to enable sufficient follow-up of the formulated requirements, the product model must also support traceability on the object-to-object level, or at least at the object-to-document level.

3. Design process model

To describe a requirement-driven product development process, we need a suitable design process model. In this context, requirement-driven means that we are interested in following and describing the process of going from the requirement specification, which is the starting point for a particular design phase, through its implications and relations to subsequent design activities. We also want to be able to trace the results of the synthesis and analysis activities, which are based on the requirement specification and documented in a decision basis. A decision basis in this context is a document on the basis of which management decide whether a project is worth continuing or whether it should be canceled.

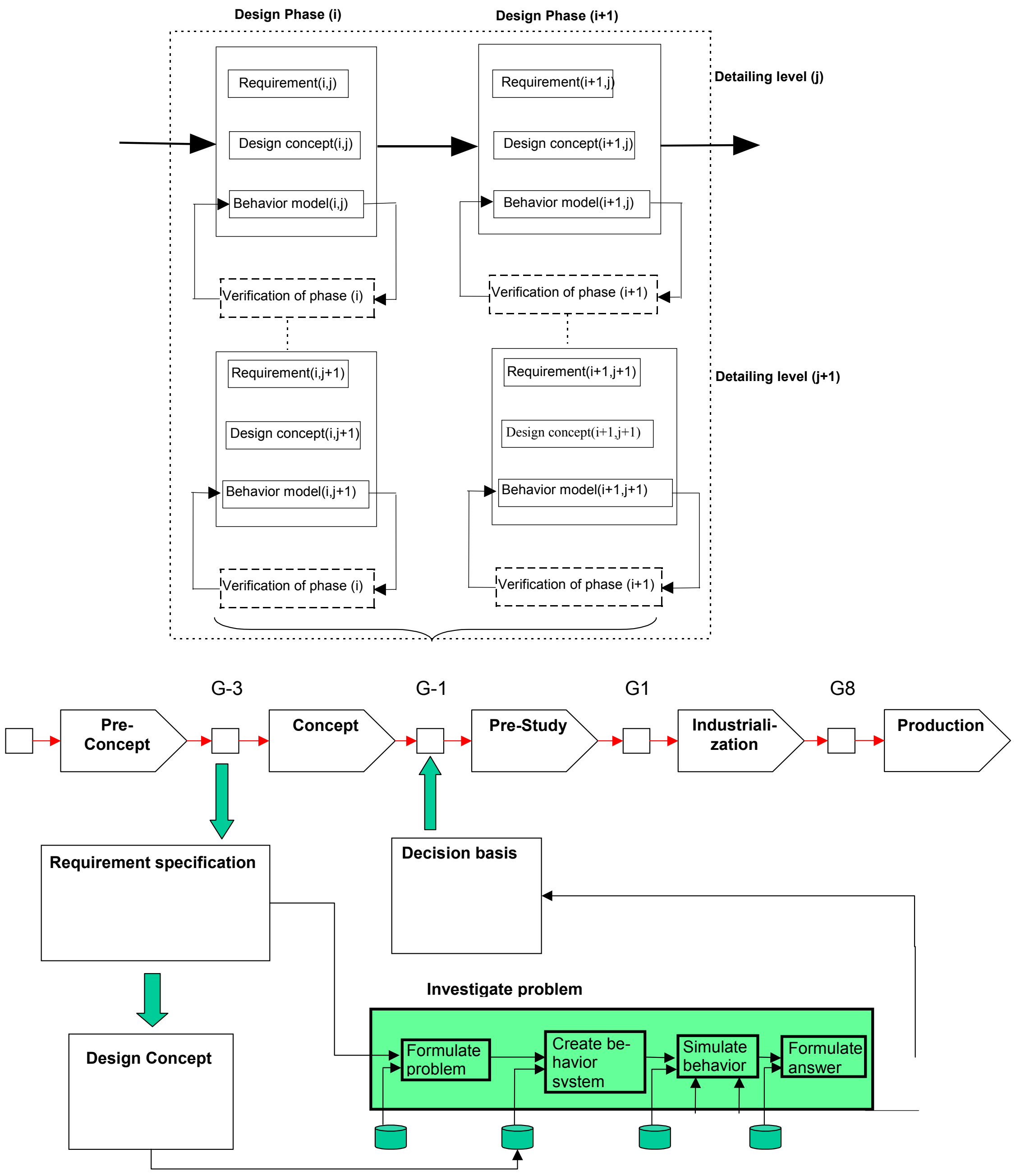


Figure 1. Relation between a generic design process model and a stage-gate model [6].

On the basis of the work by Malmqvist [1] and Andersson [7], a design process model that is capable of describing problem statements and model specifications as separate objects has been formulated by the author [6]. This design process model is based on relating the generic design process model to a more specific activity-oriented stage-gate model, as shown in figure 1.

3.1 Activities in the design process

The activities that are of interest are those that have a key impact on the decision made by the management at each gate. Figure 1 is a schematic representation of two main types of activities: generation of the design concept and analysis of this concept with respect to the requirements imposed by the specifications. We will assume that we have a well-defined concept and will thus concentrate on the activities involved in the investigation of the properties and behavior of this concept.

This investigation can be seen as a verification loop for each requirement attribute, as illustrated in figure 2. This verification process is initiated by the requirement specification and results in a decision basis for project management.

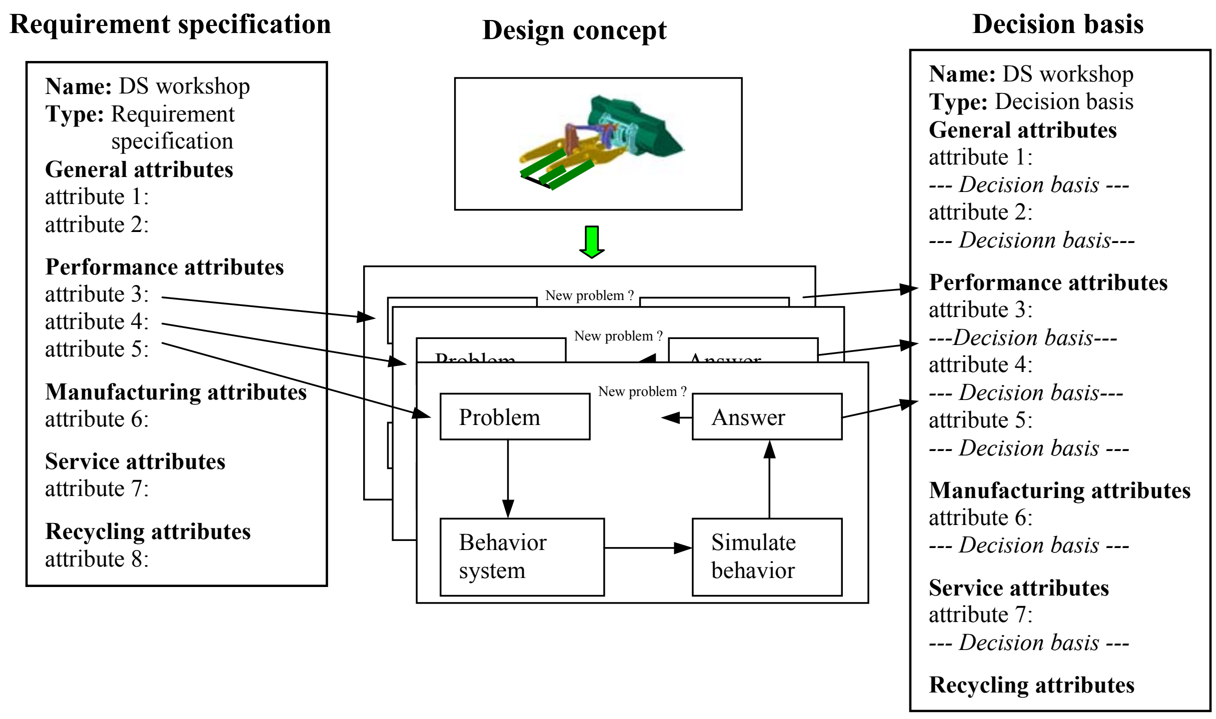


Figure 2. A loop for verification of requirement attributes

The results of the activities involved in verification of the demands and wishes formulated in the requirement specification are documented in a decision basis. This decision basis, the requirement specification, and the design concept constitute the main parts of the design model (figure 3), which is a subset of the total product model.

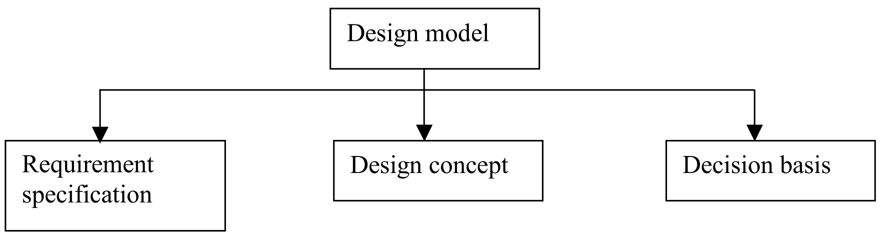


Figure 3. Main components of a design model

3.2 Requirement specification

The main purpose of a requirement specification is to set the target values that the product must meet. It should have a clear and simple structure that makes it easy to find the target values specified for a specific attribute. The requirement specification is often well-defined in industry today, using either an internal or a more general external standard, such as MIL-STD-490A [8].

In order to support better traceability of attributes in the requirement specification, these should be represented as separate objects that can contain both meta-information and values such as text, integer, or real values, or references to other types of objects such as graphs and figures (see figure 8).

4 The problem investigation loop

The base for the verification in figure 1 is the activity named "investigate problem". This is the core activity in the problem investigation loop, which aims to verify that the current product concept meets the specified product properties. The main activities in this loop are illustrated in figure 4, which is followed by a short description of each activity. The VISP tool in figure 4 is prototype software being developed in the VISP research project to support these activities.

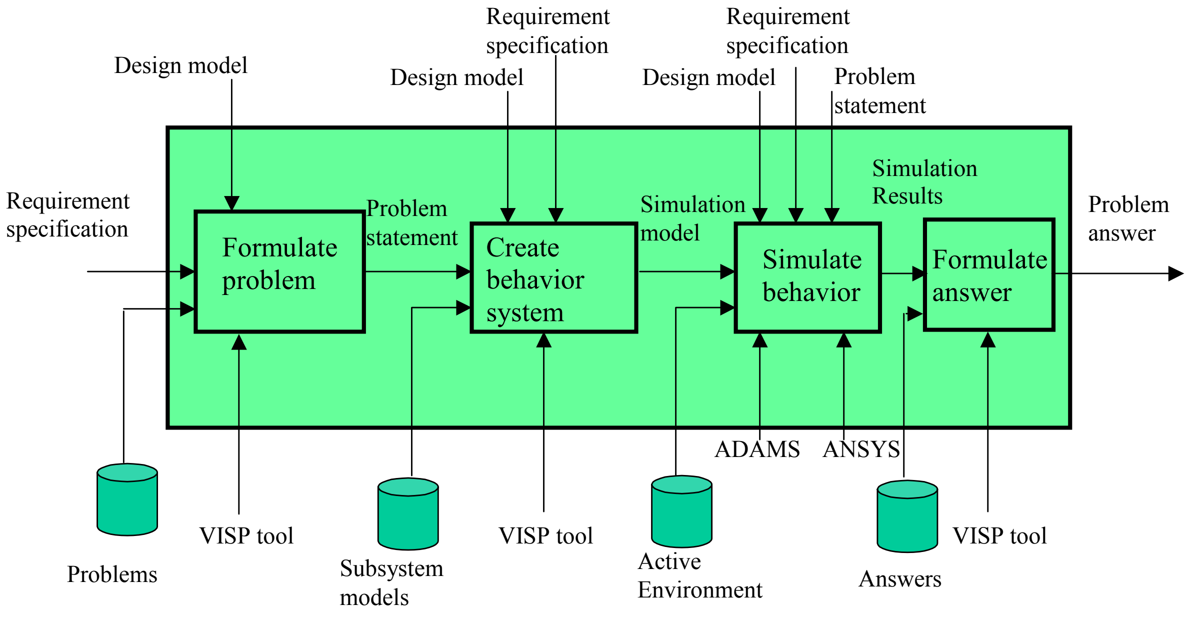


Figure 4. The activity "investigate problem"

Formulate problem

Formulating the problem is the first in a sequence of activities that focus on exploring the properties of the design concept. More precisely, this activity requires the designer to define the problem using some sort of web-based interface where the problem is formulated as a question. This information is then stored as an instance of the "problem" object.

Create behavior system

The second activity is initiated by the formulated question and starts with a check whether if everything needed to solve the problem and answer the question is known. If not, an additional problem must be formulated. This additional problem needs to be solved first so that all necessary information is available before tackling the initial problem. Next, a model specification is drawn up in which we state what type of information we need to get from the simulation; for example, whether we need to know the dynamic or static properties of the product concept.

We then proceed to configure a system model that can be used to explore the properties of the product concept. This configuration must be in accordance with the model specification and uses pre-made subsystem models stored in a modeling database (see [9], [10]).

Simulate behavior

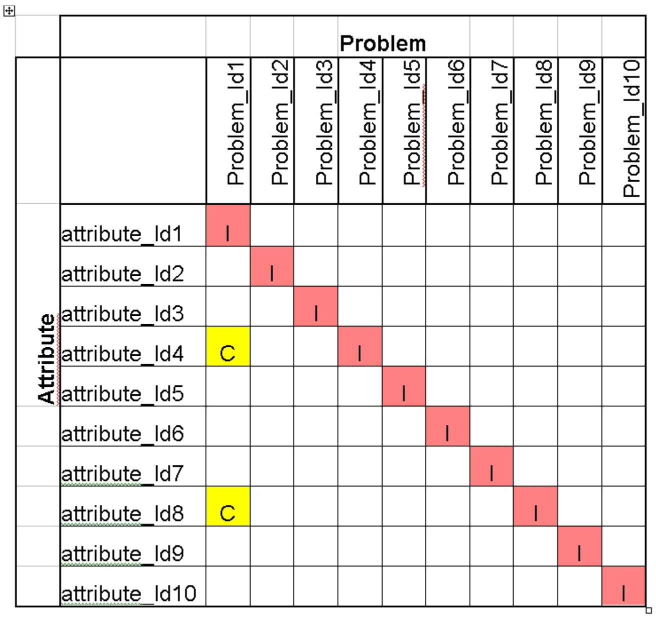
In this third activity, the configured model is taken as input and additional constraints are put on it based on the active environment in which it will operate. One or more analyses are then performed. For these analyses, the simulation model is imported into the selected analysis tool (e.g. ADAMS) where the additional conditions are defined.

Formulate answer

Last but not least, the generated simulation and the calculated results are evaluated. This evaluation is a crucial task that focuses on establishing whether the simulated behavior resembles the corresponding physical behavior within acceptable limits. The result of this activity is an answer to the original problem formulated in a document with graphs and drawings, which is represented as a "problem answer" object.

4.1 The problem investigation matrix

The verification loop is initiated by an attribute in the requirement specification. However, each loop may yield results that can contribute to the verification of other attributes. Thus before starting a new loop by formulating a new problem, it is important to know whether any earlier results can contribute to this verification. But there has been a lack of any simple method or tool for visualizing these relations. To address this lack, I have developed a problem investigation (PI) matrix (see figure 5). In this matrix the requirements attributes are listed on the left-hand side and the problem formulations are listed at the top. The relation between a problem formulation and an attribute is marked in the matrix. A letter "I" means that this is the attribute initiated this problem formulation, and a letter "C" means that that the answer to this problem formulation contributes to the verification of this attribute.



		Problem									
		Problem_Id1	Problem_Id2	Problem_Id3	Problem_Id4	Problem_Id5	Problem_Id6	Problem_Id7	Problem_Id8	Problem_Id9	Problem_Id10
Attribute	attribute_Id1	I									
	attribute_Id2		I								
	attribute_Id3			I							
	attribute_Id4	C			I						
	attribute_Id5					I					
	attribute_Id6						I				
	attribute_Id7							I			
	attribute_Id8	C							I		
	attribute_Id9									I	
	attribute_Id10										I

Figure 5. The Problem Investigation (PI) matrix

This method provides a visual representation of which analyses have been done and whether they may contribute to the verification of attributes other than the one that was the focus of the original problem. It can also support a search for verification of earlier product variants by illustrating dependencies and links to other objects in the matrix. Information that is of interest both when looking at earlier products and during the development of a new product can be shown, thus enabling objects to be traced from the problem object, or answer documents to be identified as the result of the formulated problem.

4.2 Tracing dependencies

The proposal that the PI matrix should be used as a tool for visualizing relations between attributes and problems automatically leads to the question of how to trace these relationships. Sutinen [5] discusses three basic techniques that can be used to manage and maintain traceability information. These are

- traceability tables
- traceability lists
- automated traceability links.

The PI matrix falls into the first of these categories. To be able to be used with this technique, the objects have to be described with sufficient meta-information. It is also desirable that the problem fields contain links to the described objects. A request, which falls into the third category above, to show a network graph of objects related to a problem object in the PI matrix will in general result in a graph similar to that shown in figure 6. The dashed lines on the lower right in the figure indicate that if some of the information needed to solve the initial problem is missing, a new problem must first be formulated and solved. In the lower left part of figure 6, the dashed lines indicates that solving one problem may lead to new questions and problems that have to be addressed before an answer to the initial problem can be formulated. The dashed lines further illustrate the iterative nature of this evaluation and verification process.

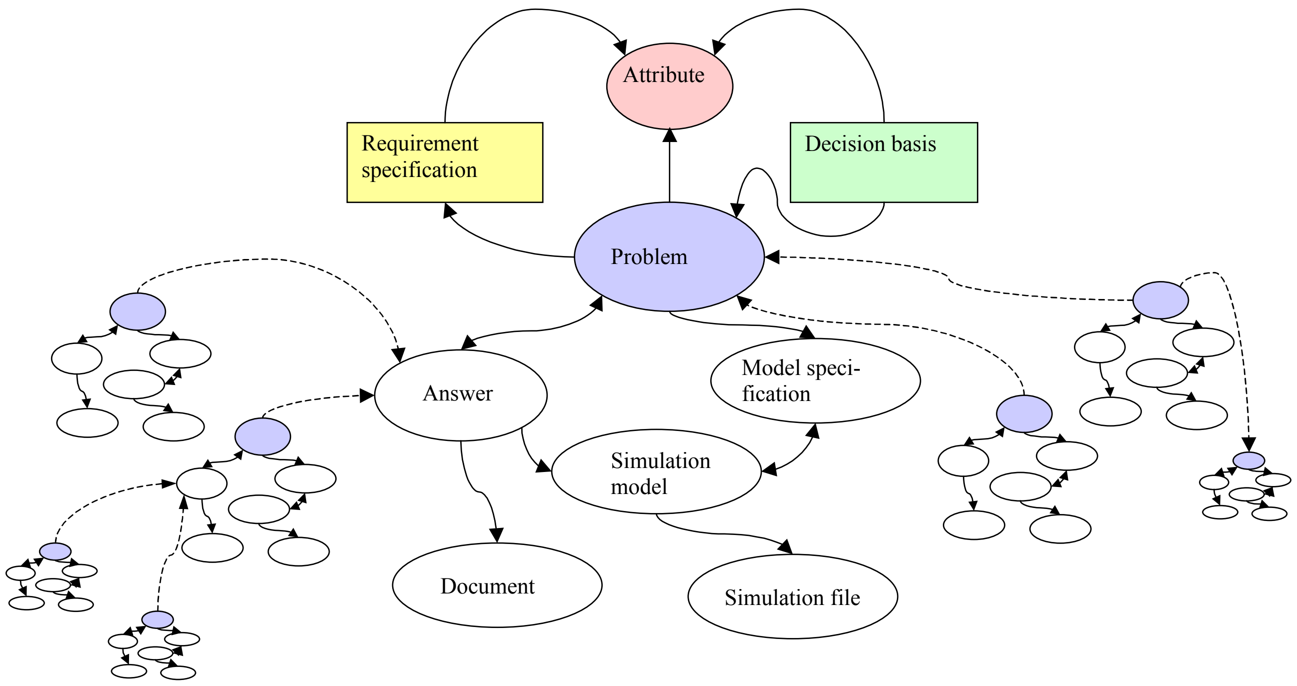
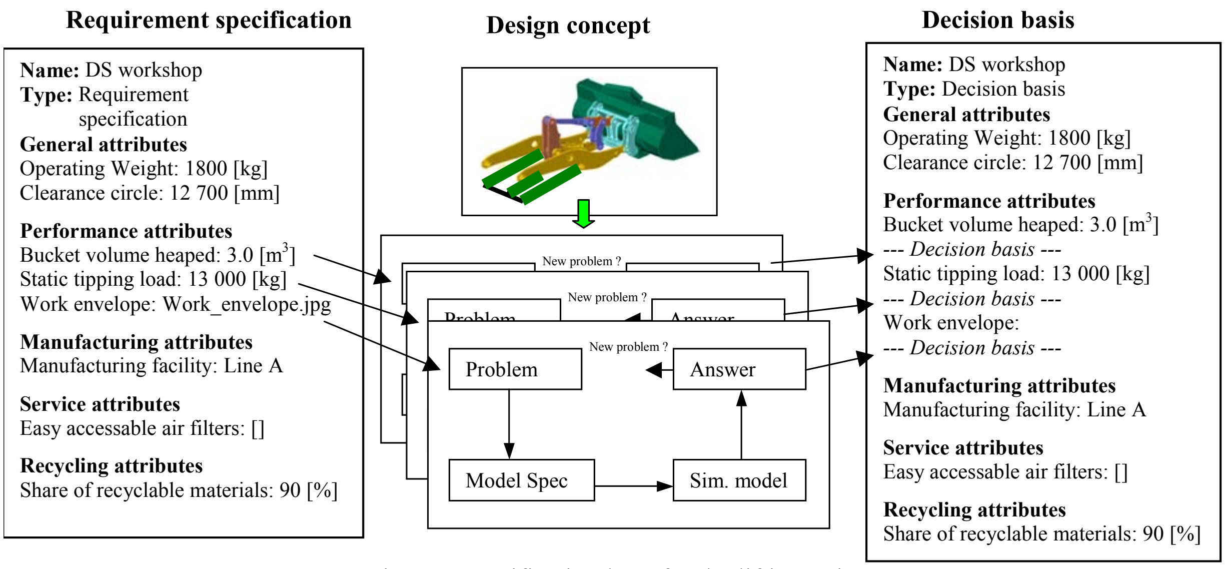


Figure 6. A network graph illustrating traceability links

5. Example: A wheel loader from Volvo CE

The framework presented here will now be used in a modeling and simulation scenario for a lifting unit in a wheel loader from Volvo CE. This scenario focuses on the modeling and simulation activities and on how these can be supported in an iterative question-and-answer driven process of investigation of the behavior of the lifting unit.

To start with, we assume that we have a situation as illustrated in figure 7. We will now go through the steps leading from the requirement specification to the decision basis for verifying the work envelope for the wheel loader. In figure 7, this demand is specified by the attribute “Work envelope”.



Requirement specification

Name: DS workshop
Type: Requirement specification
General attributes
Operating Weight: 1800 [kg]
Clearance circle: 12 700 [mm]

Performance attributes
Bucket volume heaped: 3.0 [m^3]
Static tipping load: 13 000 [kg]
Work envelope: Work_envelope.jpg

Manufacturing attributes
Manufacturing facility: Line A

Service attributes
Easy accessable air filters: []

Recycling attributes
Share of recyclable materials: 90 [%]

Decision basis

Name: DS workshop
Type: Decision basis
General attributes
Operating Weight: 1800 [kg]
Clearance circle: 12 700 [mm]

Performance attributes
Bucket volume heaped: 3.0 [m^3]
--- *Decision basis* ---
Static tipping load: 13 000 [kg]
--- *Decision basis* ---
Work envelope:
--- *Decision basis* ---

Manufacturing attributes
Manufacturing facility: Line A

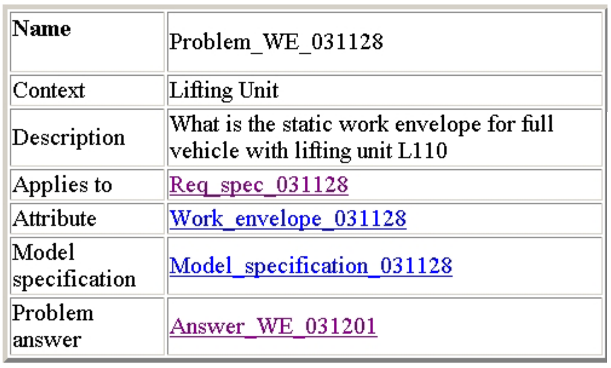
Service attributes
Easy accessable air filters: []

Recycling attributes
Share of recyclable materials: 90 [%]

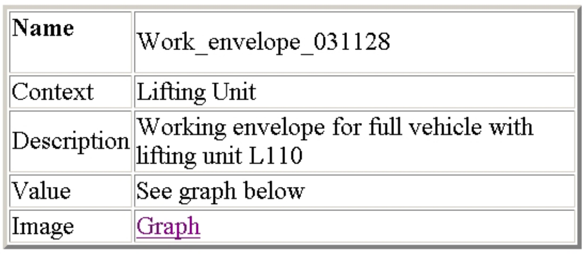
Figure 7. Verification loop for the lifting unit

Formulate problem

In this first verification step, we have to define the problem in terms of a question that indicates what attribute it applies to and in which requirement specification. For the “work envelope” attribute, this question is formulated as follows: “What is the static work envelope for a full vehicle with lifting unit L110?” Note that this attribute is represented as a separate object as shown in figure 8. We also have to define the context for this problem, which in this case is the lifting unit. This information is stored in a problem object, to which we will also add references to the model specification created to solve this problem as well as a reference to the answer to this question (see figure 8).



Name	Problem_WE_031128
Context	Lifting Unit
Description	What is the static work envelope for full vehicle with lifting unit L110
Applies to	Req_spec_031128
Attribute	Work_envelope_031128
Model specification	Model_specification_031128
Problem answer	Answer_WE_031201



Name	Work_envelope_031128
Context	Lifting Unit
Description	Working envelope for full vehicle with lifting unit L110
Value	See graph below
Image	Graph

Figure 8. A problem description object (left) and a description of the requirement attribute “work envelope” (right)

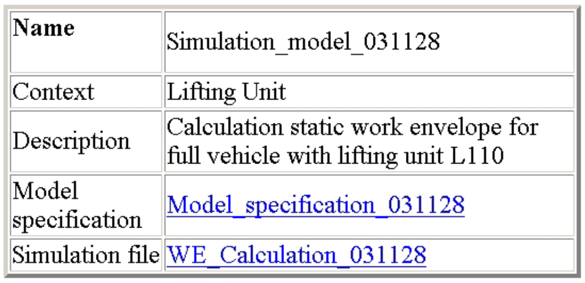
Create behavior system

Once we have created a problem description, the next step is to define a model specification that defines what simulation submodels we use and how these should be connected. We also have to define what simulation tool should be used to simulate the system behavior. In this example, we decided that a rigid ADAMS model is sufficient to give a rough estimate in this early design phase.

Then we configured the system model by selecting pre-made ADAMS subsystem models from a modeling database. In these subsystem models we separated the connection interfaces to other subsystem models (see Andersson [9], [10]). This enables us to explicitly define the connections between the subsystems.

Simulate behavior

Next, the configured simulation model was loaded into the simulation software, in this case ADAMS, where additional constraints needed to be added. In this example, we had to define how to control the movement by defining the restrictions on the maximum and minimum lengths of the cylinders. The outer boundary of the working envelope could then be obtained by using an external force dragging the bucket to its maximum reach positions. Figure 9 illustrates the simulation model object for this example as well as the ADAMS model for this calculation. This object contains references to the model specification as well as to the ADAMS file that contains the results data.



Name	Simulation_model_031128
Context	Lifting Unit
Description	Calculation static work envelope for full vehicle with lifting unit L110
Model specification	Model_specification_031128
Simulation file	WE_Calculation_031128

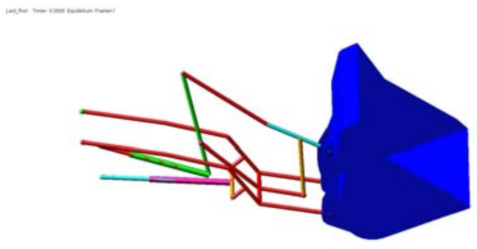


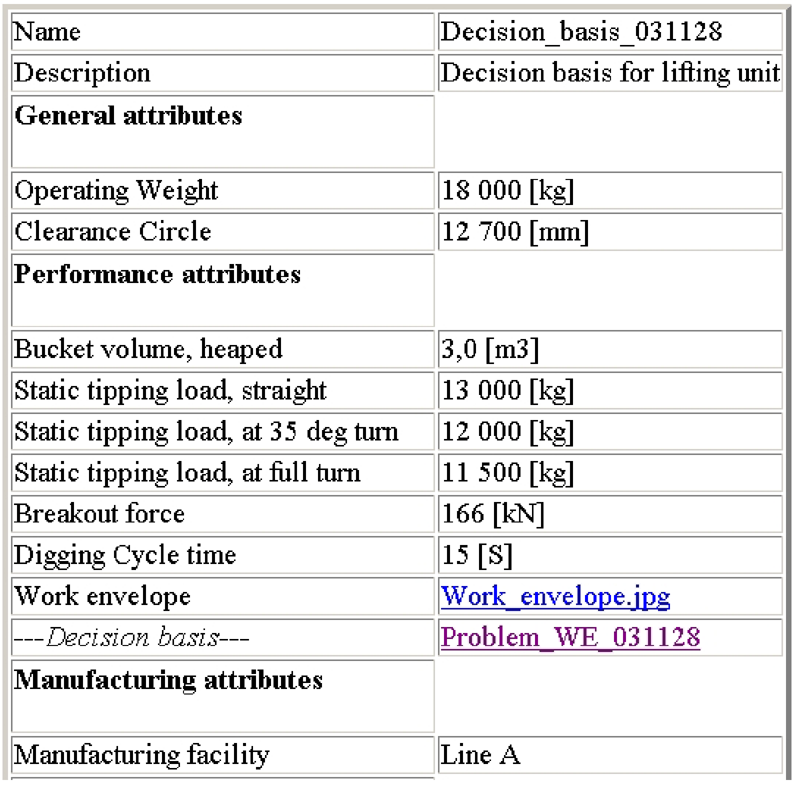
Figure 9. A description of the simulation model (left) and the corresponding ADAMS model (right)

The final activity after performing the behavior simulations was to interpret the results of the simulations and to formulate an answer to the question posed in the problem definition.

This answer may be formulated in a text-based document containing tables or graphs supplied by the analysis tool. The conclusions to be drawn from the analysis should be clearly stated.

Decision basis

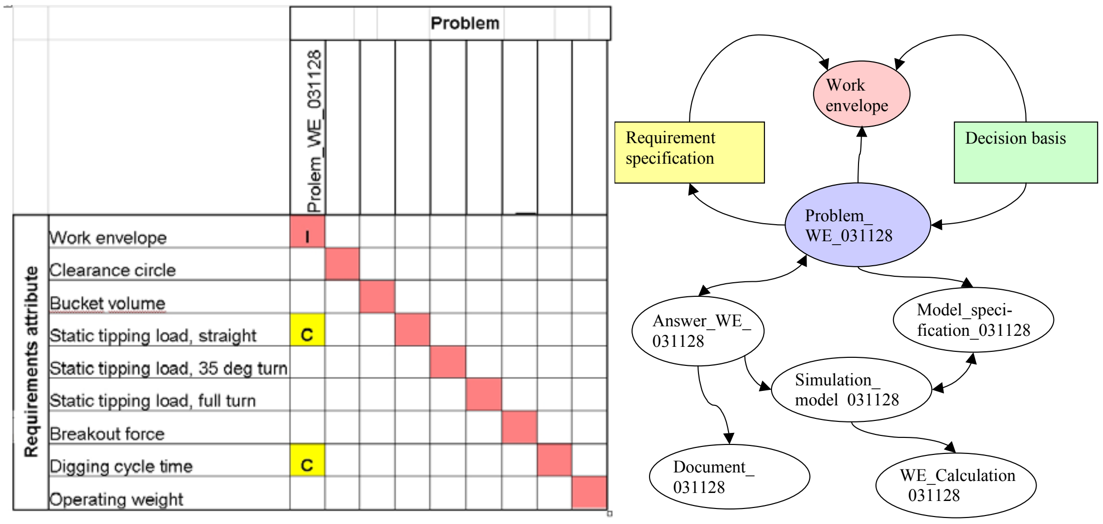
Investigation of the product properties of the actual design concept is an iterative process, with a problem definition object being created for each attribute in the requirement specification that has to be verified during a specific development phase. This iterative process results in a decision basis that is successively filled in (see figure 10). This iterative process is illustrated as a loop in figure 7, where only part of the requirement specification and the decision basis is shown.



Name	Decision_basis_031128
Description	Decision basis for lifting unit
General attributes	
Operating Weight	18 000 [kg]
Clearance Circle	12 700 [mm]
Performance attributes	
Bucket volume, heaped	3,0 [m3]
Static tipping load, straight	13 000 [kg]
Static tipping load, at 35 deg turn	12 000 [kg]
Static tipping load, at full turn	11 500 [kg]
Breakout force	166 [kN]
Digging Cycle time	15 [S]
Work envelope	Work_envelope.jpg
---Decision basis---	Problem_WE_031128
Manufacturing attributes	
Manufacturing facility	Line A

Figure 10. Decision basis for the lifting unit

Once we had completed one verification loop for the attribute "work envelope" in figure 7, we could use the PI matrix to visualize the current status of the total verification process in this design phase. The appearance of the PI matrix after this first loop is shown in figure 11, together with a graph containing objects and the relation created during this verification loop. This graph can be generated using traceable information starting from the problem object.



		Problem								
		Prolem_WE_031128								
Requirements attribute	Work envelope	I								
	Clearance circle		🟥							
	Bucket volume			🟥						
	Static tipping load, straight	C			🟥					
	Static tipping load, 35 deg turn					🟥				
	Static tipping load, full turn						🟥			
	Breakout force							🟥		
	Digging cycle time	C							🟥	
	Operating weight									🟥

Figure 11. PI matrix (left) and a network graph (right) of created objects for the wheel loader

6. Summary and Conclusions

Product development is a very complex task that involves many people of different professions working together over a long period of time towards the same goal or temporary goal. A common way of dealing with complexity is to simplify and concentrate on what are considered to be the most important properties. Of course, the decision about what is most important is always influenced by one's own preferences or responsibilities. In this paper I have chosen to focus on the aspect of product development that deals with the verification of product requirements.

This paper presents a proposal for an information framework that can support a requirement-driven product development process. This framework is complementary to the work of Sutinen [5] and Malmqvist [1]. It focuses on enabling traceability for decisions based on simulation activities, where their work focuses on traceability for models in early design phases and addresses such issues as requirements, functions, and design concepts.

The framework is based on a design process model that can treat problem statements, model specifications, simulation models, and problem answers as separate objects. This design process model is based on the work of Malmqvist [1] and Andersson [7]. It enables a fine granularity level of information with object-to-object traceability between the attributes in the requirements specification and the estimated product properties. It also enables partial results obtained during the verification of a specific requirements attribute to be traced and reused, as well as enabling study of the effects that changes in the requirements specification have on product properties.

The presented framework is illustrated by a modeling and simulation scenario involving a lifting unit in a wheel loader from Volvo CE. This scenario focuses on modeling and simulation activities and on how these can be supported in an iterative question-and-answer driven process of investigation of the behavior of the lifting unit.

Much further research is still needed to address questions such as the following:

- How will the model deal with the relations between requirements on the top level and on subsystem levels?
- Often only a subset of the total requirements are used to guide the work during the early design phases. How will these be incorporated in the model?
- How can the PI matrix be further developed to deal with a situation where a number of problems are initiated for a requirement attribute?
- During development, there are constant changes in a product's shape and detailing level and alternatives exist. How can the model be extended to accommodate this?

Acknowledgments

The work presented here was performed in a VISP research project and was financially supported by the Swedish Agency for Innovation Systems (VINNOVA).

References

[1] Malmqvist J., "Implementing requirements management: A task for specialized software tools or PDM systems?," Proceedings NordDesign 2000, DTU, Copenhagen 2000.

[2] Ramesh B., "Representing and reasoning with traceability in model life cycle management," Annals of Operations Research, Vol. 75, pp. 123–145, 1997.

[3] Blanchard B. S., Fabrycky W. J., "Systems Engineering and Analysis", Prentice Hall, New Jersey, USA.

[4] "Quality Function Deployment," American Supplier Institute, 1989.

[5] Sutinen K., Almefelt L., Malmqvist J., "Implementation of requirements traceability in systems engineering tools," Proceedings of Product Models 2000, Linköping 2000.

[6] Andersson, K., "Simulation in product design: An iterative question–answer driven process," Proceedings of Design 2004, Dubrovnik, Croatia, May 2004.

[7] Andersson K., "Modeling and simulation as a control means for product development," 6th International Conference on Management of Technology, MOT97, Gothenburg, June 1997.

[8] MIL-STD-490A, "Military Standard: Specification Practices," Department of Defense, Washington D.C: 1985.

[9] Andersson, K., "A design process model for behavior simulations of complex products," ASME, DETC '99, Las Vegas September 1999.

[10] Andersson, K., "Simulation of a tunnel drilling sequence to determine loads on a rock drilling equipment," ADAMS User Conference, Rome 2000.

Kjell Andersson, Associate professor
KTH, Machine Design
SE- 100 44 Stockholm, Sweden
Tel: +46 8 790 6374
Fax: +46 8 20 2287
e-mail: kan@md.kth.se