INTERNATIONAL DESIGN CONFERENCE - DESIGN 2004
Dubrovnik, May 17 - 20, 2004.

DESIGN 2004

INFORMATION MODEL FOR THE MECHATRONIC PRODUCT FOCUSING THE FUNCTIONAL ABSTRACTION

T. Zimmerman, K. Hallin and J. Malmqvist

Keywords: information, function, mechatronic, PDM

1. Introduction

Mechatronic products with their diversity of cooperating technologies set new demands on effective and consistent information management. Modern, but traditionally mechanical products, such as cars, medical equipment and so forth, tend to contain an increasing amount of electronics and software, thus becoming more and more mechatronic. Product data management (PDM) systems are often found in the core of information system architectures constituting a data repository for product data. Information models form patterns for design of databases in this type of systems. It is thus important to define robust information models to enable a high quality management of product data.
In early phases of the development process, when the design is immature, it is often unclear whether a specific technical solution shall be implemented in hardware or software. A presumption of this work is that the concept of product function could constitute the uniting notion between hardware and software throughout the development process, thus providing a tracelink between artefacts originating from different engineering disciplines.
This work has adopted the design modelling research approach [Duffy and Andreasen 1995], see Figure 1. The reality in scope is the mechatronic product. This reality is described by a phenomenon model and by an information model. A product example is instantiated in EXPRESS-I for the purpose of information model validation. The work concludes that a link between hardware and software artefacts could be maintained by a relatively simple model comprising the functional abstraction. A model based on the semantics of transformational-based behaviour is shown to be feasible.
The point of departure for this research work is a mixture of methodologies representing a cross-section of design research applicable to the development of mechatronic products. Further, the Theory of Technical Systems [Hubka and Eder 1988], the Chromosome model [Andreasen 1992] and the existing state of the PDM discipline are taken into account. The main contributions are a phenomenon model and an information model that together captures the semantic needs of functional representation applicable for implementation in PDM systems.
The sections in this work are arranged as follows. Section 2 discusses related work. Section 3 gives a theoretical background. Section 4 scopes a comparison of methodologies and a look into the PDM discipline. Section 5 delivers a phenomenon model, a list of related information requirements and an information model. A validation is presented in Section 6. Section 7 sums up the paper in a discussion with concluding remarks.

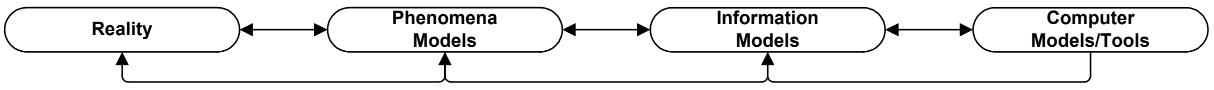


Figure 1. The adopted research approach [Duffy and Andreasen 1995]

2. Related work

A proposal for integration of hardware and software development processes in the context of mechatronic products have been made by Svensson and Crnkovic [Svensson and Crnkovic 2002]. Their work can however be extended regarding information modelling for early phases, such as conceptualisation. Korpela [Korpela 2002] proposes an object-oriented approach, with influence from Theory of Technical Systems, which suggests an UML-based (Unified Modeling Language) product model for early development phases, but does not explore the functional domain thoroughly. Two ongoing and interrelated projects that touch upon the scope of this work are UML for Systems Engineering (SE) and the ISO community's development of AP233, Systems engineering data representation [ISO 2001]. The purpose of the first project is to extend UML to cover all aspects of modelling in the Systems Engineering discipline, thus providing a modelling language for the discipline as a whole, whereas the latter project aims at providing a homogeneous information model for the exchange of product data between computer tools. Both projects are still far away from being completed but will provide interesting contributions in the future.

Another interesting approach is that of D.H Brown Associates [Collier 1999], who proposes a common product model for hardware and software, focusing information management for the development of product and subsystem variants. His work has a very broad scope but lack in details concerning the functional representation.

Earlier work by the authors [Hallin et al 2003] has compared existing phenomenon models from the hardware and software domains and concludes that parts of ISO 10303 product model [ISO 1994] can be used to implement the Chromosome model. This work takes the authors´ earlier approach further in exploring how the functional domain can be deployed for representation of the mechatronic product, emphasizing the structuring of metadata in PDM-like systems. The intent is not, as in the case of the standardization projects mentioned above, to deliver an all-embracing modelling language or a complete information model for mechatronic product data exchange. It is rather a contribution that shows how information consistency can be improved by introducing the functional abstraction in product data management systems.

3. Supporting theories and foundation

3.1 Theory of Technical Systems

Even though the Theory of Technical Systems (TTS) [Hubka and Eder 1988] primarily addresses machine systems, it still provides a basic foundation for how modern design theories and practices approaches mechatronic product development. Hubka and Eder look upon the technical system as something that in interaction with humans and other environmental elements contributes to the transformation process, whereas the technical system is the artefact to be developed. The entirety constitutes the transformation system.

In TTS, the function is described as a property of the technical system. Its purpose is to convert an input measure to an output measure. The technical system is defined by a set of functions with certain relations. Two types of relations can be identified, causal and logical relations. Functions are linked through a flow of material and/or energy and/or information. The function structure gives means to understand the operational states of a technical system. A function may be characterized by its level of abstraction. An abstract function often has a broader variety of possible solutions as means for realizing it; inversely a more concretely defined function is more limited in its solution space. Hubka and Eder distinguish between five types of purposes for functions. The transformation function fulfils the purpose of the technical system. Then there are auxiliary functions, driving, propelling or energy delivering functions, regulating and controlling functions, and connecting and supporting functions.

Basically, there are two ways to draw up a function structure. The first emanates from the aims that are related to the technical process, concretized through “black boxes”, stipulating a certain output for a given input. The second represents a bottom-up approach, starting with a solution space consisting of organs and/or components, resulting in a “black box” function structure as well.

3.2 The process of product development

According to *Engineering Design* [Pahl and Beitz 1996], an increasingly important sphere of application of the systems approach is the function-oriented synthesis. In product development this often means describing the overall functions in some schematic presentation or model, emphasising the existence of subfunctions and their hierarchical ordering. Pahl and Beitz propose a comprehensive process for product development where the establishment of product function is an integrated component. A process model as such, is not easily situated in terms of the declared research approach, see Figure 1. It does not describe reality nor the product as a phenomenon. However it gives valuable inputs to the information domain, as it defines important inputs and outputs of the subprocesses prescribing the engineering work, and is in that respect, yet most suitable mapped to the phenomenon domain. Hence, the survey and comparison between the methodologies in Section 4 are utterly important in order to understand how they affect the information management supporting them.

3.3 Product models and the chromosome

Product models are defined differently depending on source and community. According to STEP (STandard for the Exchange of Product model data) [ISO 1994], a product model makes up the conceptual description of the product, set out in a computer interpretable language, thus capturing the product semantics. A more abstract and fundamental description of a product model is presented through the chromosome model [Andreasen 1992]. It scopes model elements and their interrelation (allocation) that directly correspond to the results of design activities, thus separating between process, function, organ and part elements, see Figure 2. The latter model better correspond to what in this paper is referred to as a product model, thus describing the most important phenomena in product development, structured in a comprehensive way. In this context a product model is more of a mental creation, than something that is to be directly implemented into a computer tool. Product models like the chromosome model are in this paper regarded as phenomena models, see Figure 1.

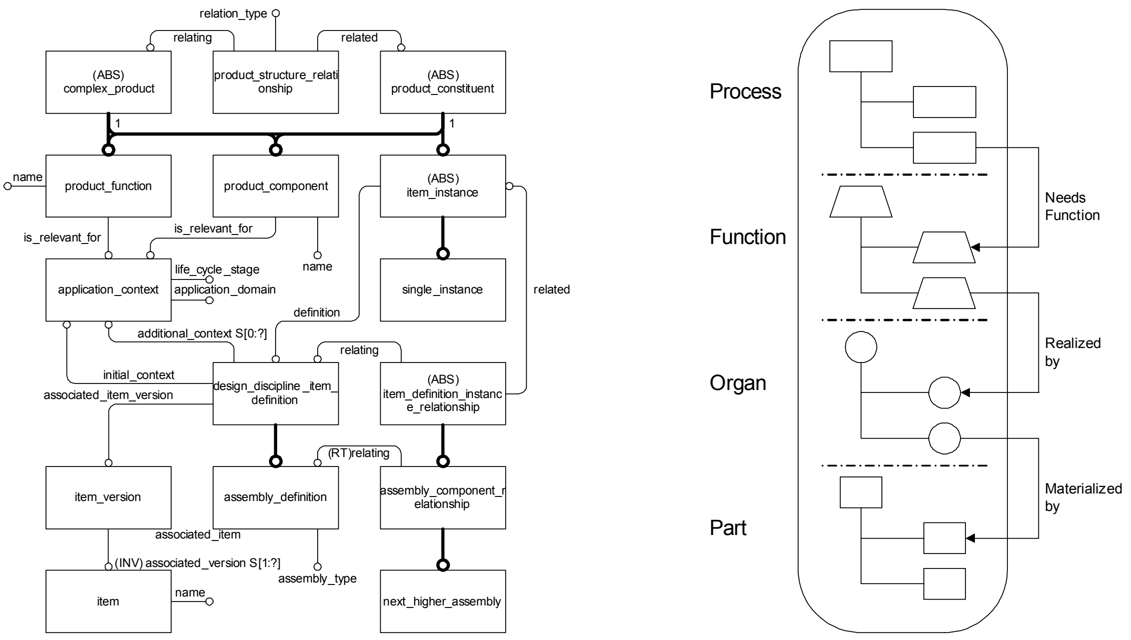


Figure 2. Subset of STEP [ISO 1994] and the chromosome model [Andreasen 1992]

3.4 Information models

A phenomenon model, like the chromosome declares the concepts and the overall relations in scope of the activity regarded. What an information model does in implementing it, is to be more precise by deploying a total approach to the modelled area, tuning into details such as prescribing auxiliary entities for the management of documents, versions, variants, status, et cetera.
Johnson and Bailey [Johnson and Bailey 2003] separate information models from computer models. Both kinds are similar in the way that they focus on concepts that information systems manipulate and hold. They are both entity-centric, include entity-to-entity relationships and allow the declaration of attributes. What differs is the level of abstraction. An information model has the engineering practitioner as the intended reader. The computer model is more detailed and holds many implementation enabling entities and modelling constructs and is thereby intended to support an implementation engineer.

4. Overall prerequisites

This section compares four design methodologies in order to clarify a theoretical basis for the functional domain. Further some overall conditions of the PDM discipline are outlined.

4.1 Comparing four methodologies

The four compared methodologies are *Engineering Design* [Pahl and Beitz 1996], *System Engineering and Analysis* [Blanchard and Fabrycky 1998], *Product Design* [Otto and Wood 2001] and *A methodology for development of mechatronic systems* [Shakeri 1998]. In comparing these methodologies special focus will be put on outlining how and why the functional abstraction is deployed..
Engineering Design is a systematic methodology for design of traditional mechanical machines, which spans from the establishment of customer needs to component detailing. *Systems Engineering and Analysis* is a contribution with focus on the system-level design and is less detailed on the component level. *Product Design* is in many senses similar to *Engineering Design* but the scope is extended to cover re-design aspects, platform development and design-for-X. *A methodology for development of mechatronic systems* is as the name implies a design methodology especially developed for mechatronic products. It is a system-level object-oriented approach that takes influence from the engineering design research area.
From a process perspective the four methodologies have a somewhat similar understanding of the development process, see Figure 3. Initially there exists a customer need, subsequently a conceptual and/or a preliminary system design is developed that establish the major building blocks of the system and finally the system is detailed into a producible unit. Some difference exists in what parts of the development process that are emphasized.

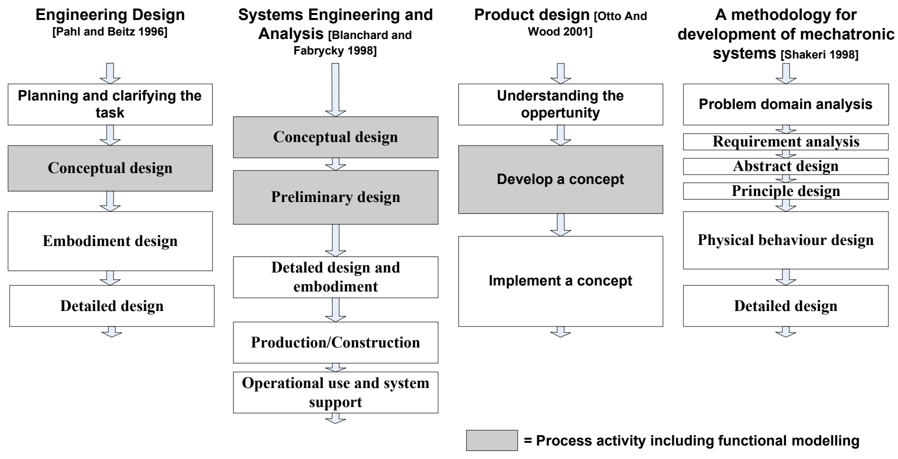

Engineering Design [Pahl and Beitz 1996]
Planning and clarifying the task
Conceptual design
Embodiment design
Detailed design
Systems Engineering and Analysis [Blanchard and Fabrycky 1998]
Conceptual design
Preliminary design
Detaled design and embodiment
Production/Construction
Operational use and system support
Product design [Otto And Wood 2001]
Understanding the oppertunity
Develop a concept
Implement a concept
A methodology for development of mechatronic systems [Shakeri 1998]
Problem domain analysis
Requirement analysis
Abstract design
Principle design
Physical behaviour design
Detailed design
▨ = Process activity including functional modelling

Figure 3. Comparing the process aspect of the four methodologies

All methodologies adopt a TTS-like approach where system behaviour is expressed by functions, inputs and outputs. The functional abstraction bridges the gap between the customer need and the final solution, the form, and constitutes a neutral solution-independent representation of how the design shall operate. In Table 1, the methodologies are outlined in terms of what concepts that are covered and what approach that is adopted to represent function. Shakeri differs by introducing a completely object-oriented approach that is solution centric and has a system design focus. He adopts state-based behavioural concepts clarifying the change in internal state over time. This is not covered by the other authors. All methodologies agree on the need for a transformational description of how the functions of the system interact in realising its total behaviour. Functions are either captured by functional flow block diagrams (FFBD) [USAF 1968] or black-box charts illustrating how inputs in terms of material, energy or information, is transformed from a initial state to a wanted state in conformity with the TTS, thus capturing the transformational aspects of behaviour. Further it is recognised how the functional abstraction is a key enabler for mapping requirements/needs to physical/technical solutions. Pahl and Beitz state that "*Functional modelling provides a natural forum for abstracting a design task"*. Blanchard and Fabrycky too, stresses that functions are utterly important as a solutions independent abstraction. To sum up, we state that the scope and complexity of the actions taken in the establishment of product function, in many aspects, is very similar amongst the compared processes. The information content and its interrelations show considerable similarities in terms of used abstractions.

Table 1. Modelling techniques and modelled concepts of the methodologies

	Pahl and Beitz	**Blanchard and Fabrycky**	**Otto and Wood**	**Shakeri**
Scope	Need(task)→ product documentation	Need→ system design	Need→ prototype	Need→ system design
Level	System/component	System/subsystem	System/component	System/component
Abstractions	Requirement Function Modules System layout Drawings Part lists Assembly instructions	Requirement Function Design parameters Modules System design	Customer need Function Modules Platforms Variant Concept	Domain modelling Requirement Class/system layout Signal/message Process State
Solution independent	Yes (Function)	Yes (Function)	Yes (Function)	No (class diagrams)
Behaviour	Transformational	Transformational	Transformational	Transformational state-based
Language	Black box	FFBD	FFBD	UML, MSC (Message Sequence Chart), SDL (Specification and Description Language)

4.2 Managing information in the PDM discipline

In addition to the prerequisites for management of product function that could be extracted from the previous section, there is a need for consideration of the reality that the PDM discipline of today delivers. As have been pointed out [Dahlqvist et al 2001], there is a gap in cultural as well as computer support aspects between the engineering disciplines that develops hardware and software artefacts. A notable strength of PDM systems is their ability to comprehend and relate multiple structures of information, such as functional, product, manufacturing and maintenance breakdowns of the product. For an increased efficiency in mechatronic product development Dahlqvist et al suggest compatible product models and compatible version management for systems that manages hardware and software product data. They do not endeavour an identical development processes for hardware and software. Neither do they imagine a complete product model for both engineering domains. However, common checkpoints are recommended throughout the development process. For a synchronized information management, enabling features like common version management, they suggest a universal abstraction

level for what is later to be implemented as hardware and/or software. They do not stress a foundation that integrates hardware and software on a technical level. Instead, they emphasize the importance of homogeneous ways of expressing metadata. The requirement and the conceptual phase are indicated as being very similar for both hardware and software.
A relevant reference-work [Hamer and Lepoeter 1996], points out important phenomena in scope of information management in the PDM discipline. Five dimensions that concerns abstraction and integrity of product data are comprised – version, views, hierarchy, status and variants.
Engineering work is normally recorded in partial steps, where every step brings about an update in the describing information content. The version dimension addresses this state and handles the ongoing modification process during products development. Of course, this concern also the objects and models used for describing product function. The view dimension is intermediated because the description of a modern (mechatronic) product could be hard to comprehend, due to its complexity. There is a need for an ability to highlight only one or just a few aspects of the product at the time. A view could comprise, in a computer support tool, hardware or software design or why not, only the transformational aspects of a product with respect to the function domain. An effective management of the hierarchy dimension is the main weapon to combat complexity during product development. Hardware designs are commonly created as hierarchies of assemblies, subassemblies, components and parts. Software design is developed through a division into applications, packages, components and separate procedures. A complex (mechatronic) product's description in a division of functions, spanning from abstract to concrete, bring the question of hierarchy to the fore. The status dimension, usually implemented through workflows, has to do with the maturity of the product description, as it progresses from a rather unfinished to an obsolete state. The variants dimension is interesting if more than one variant, or platform, of the (mechatronic) product is to be developed. Then this dimension deals with the commonalities and differences between variants.
In attaining a uniting notion for hardware and software, special interest is set on the view and hierarchy dimension of the PDM discipline. A functional view of the product enables a common sphere for comprehension, discussion and decision-making amongst engineers of different disciplines, essential for a coordinated development process. A hierarchy dimension adds positive effects in terms of comprehension.

5. An information model for mechatronic function

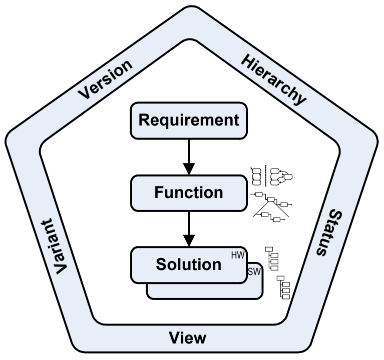
This section discusses important phenomena in the functional context and delivers a phenomenon model with correlating information requirements. Emanating from this, an information model is proposed.

5.1 Phenomenon model and information requirements

According to the design methodologies discussed in Section 4, there are at least three different representations that are vital to use for successful design. First of all, the need that is addressed must be fully understood and defined. Secondly the behaviour of the system must be formalised in terms of its internal transformations by means of functions, including flow of inputs and outputs. And finally the selected technical solution must be defined. As pointed out in Section 4, it is important to take certain phenomena into consideration to achieve consistent management of information, all with respect to the version-, view-, hierarchy-, status- and variant-dimension. Dahlqvist et al recognises that a common abstraction is needed to map different engineering domains to each other. Pahl and Beitz, Blanchard and Fabrycky, and, Otto and Wood recognise the importance of abstracting the design problem to a solution independent level by means of the functional abstraction. In the context of information management we propose that the functional domain shall be the bridge between representations of different technology and thus that transformational behaviour shall be introduced in PDM systems.
A use of state-based behavioural modelling, as in Shakeri's methodology, see Section 4.1, is of course important for describing product behaviour. However, from a mechatronic information management perspective, where it is crucial to capture the tracelinks between solutions of various technologies, the

transformational-based approach may be preferred. As Oliver et al [Oliver et al 1997] concludes, the transformational-based approach's founding artefacts (function) can easily be mapped onto multiple structures of solutions. That feature is needed in the management of product data in order to couple technical solutions from different engineering domains. The energy, material and information, as parts of constituting body of the transformational representation, can in this context act as initiators for establishing the functional relation between what is to be structures of both hardware and software design artefacts.

In Figure 4, a phenomenon model for design information is presented. In the centre, the trace from requirements to function and further to solutions can be found. Solutions may be built from different technologies and a function is the natural link between them in accordance with the presumptions made in the commencement of this work. A pentagon is drawn around the vital representations, illustrating impact of the five dimensions of PDM. The phenomenon model is interpreted in terms of a set of information requirements, see Figure 4, in agreement with the line of argument in this and previous sections.



#	Requirement	Comment
1	The concept of function shall be modeled as an object with the capability of classification	Function may be specialized into ´main´, ´auxiliary´, et cetera
2	Function inputs and outputs shall be modeled with the capability of classification	Inputs and outputs may be specialized into ´energy´, ´material´ and ´information´
3	Allocation of exterior objects to and from a function object/model shall be possible	For instance, requirement objects may be allocated to functions. Functions may be allocated to organs, components or parts, et cetera
4	Integrity shall be maintained	The mechatronic design (including functions) shall be under version and workflow (status) control
5	A comprehensible design shall be provided by modelling abstractions	The abstractions of hierarchy, variant and view shall be modeled

Figure 4. Phenomenon model and information requirements

5.2 The information model

In this section some basic modelling constructs for capturing the interrelations of mechatronic product functions are proposed. The constructs, see Figure 5, stand in correlation to the requirements and the phenomenon model presented in the previous section. No proposals for the function's integrity and no variant or view management constructs are modelled, see Requirement #4 and #5. This is done for reasons of delimitation.

The product function concept is modelled by the *product_function* entity in order to meet information requirement #1. To enable classification of functions the *function* entity has been typed with the attribute *type*, which can hold applicable values such as 'main' or 'auxiliary'. As the vocabulary and number of function types may vary in different applications this is captured in the model by an attribute and not by inheritance which is a more static construct. Requirement #2 of capturing the flow of operands (I/O) between different product functions is modelled with the *operand* entity. Operands may, according to Hubka and Eder's classification scheme, be of three different categories, namely energy, material or information. As these three types have different attributes and as they represent a fixed set that will not vary over time they are modelled by an inheritance construct. The *operand* constitutes input and output of the transforming functions and thus has relation(s) to one or many functions through the *product_function_operand_relationship* entity. Note that this implies that operands can be produced and consumed by one or several functions. Functions are in the context of information management often ordered in hierarchies and thus information requirement #5 set demands on such capabilities. In the proposed information model, the aspect of decomposition is solved by the *product_structure_relationship* entity. Functions are by means of this entity ordered in structures that however do not always appear in a laddered formation. The *relation_type* attribute controls the purpose of the relationship. The most applicable *product_function_relationship* type, in the functional context is 'decomposition'.

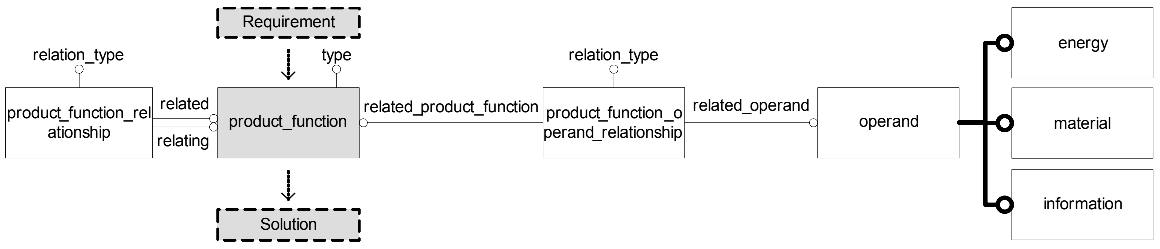


Figure 5. Information model construct

Information requirement #3 calls for the ability to trace from requirements to function and from function to solution. This requirement is not explicitly met in the information model presented in Figure 5, but it is indicated that requirements shall be related to the realizing *function* and that the *function* shall be related to the materialising solution. To capture the relation between requirements, functions and solutions, the constructs of Figure 5 have in Figure 6 been united with parts of the main structure engine of (STEP) Application protocol 214 [ISO 2003]. This part of ISO 10303 has been recognised to formalise the information content of the chromosome model and further to be applicable in information management for mechatronic products [Hallin et al 2003]. The protocol has a sophisticated solution for managing conceptual product structures in early design phases. The entities *product_function* and *product_component* are specializations of the *complex_product* and *product_constituent* entities where *complex_product* holds one or many *product_constituent* through a *product_structure_relationship*. A *product_component* is defined as an element in a conceptual product structure. The possible relationships between *product_function* and *product_component* reveal more of the nature of the two entities. A *product_function* may be considered as the functionality to be fulfilled by a *product_component*, a *product_component* may be considered as a means to realize a *product_function* thus capturing the relation between functions and solutions. In the united model the entity *product_function_relationship* has been replaced with *product_structure_relationship* of the application protocol. The relation between functions and requirements is captured by an association of the *design_constraint* entity to *product_function* in the role of a *complex_product*.

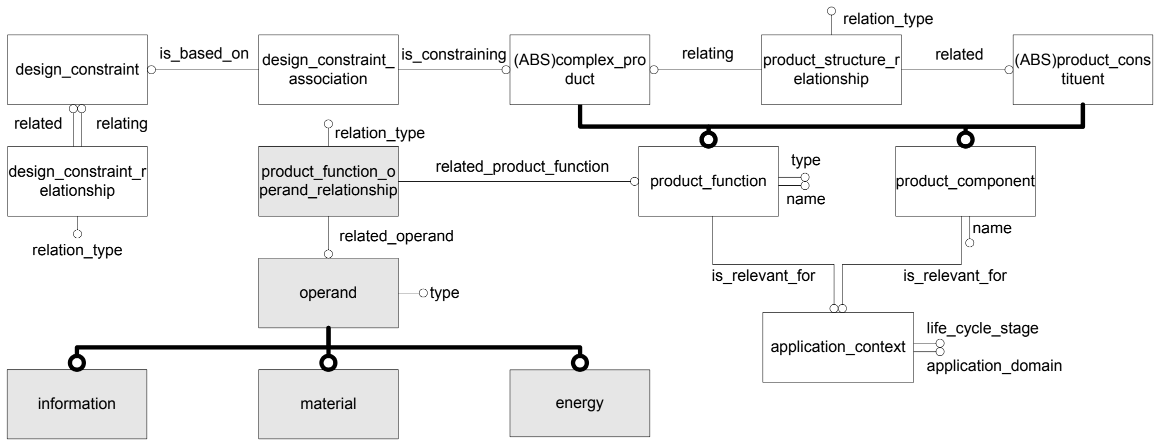


Figure 6. Information model

6. Case study

The information model has been validated by instantiation with product data describing a parking heater system, which can be found in many new cars. The system is analysed in terms of its function and architecture (structure). The primary function of the heater is to guarantee a satisfying temperature inside the car when the engine is not running, typically used to make sure that the car is warm when the driver enters. The heater consists of the following main parts: A fuel pump for diesel or petrol supply, a control unit, a circulation pump for heated media circulation, a burner for burning of fuel, a heat exchanger and an ignition system.

The information model has been instantiated with product data associated with the heaters prime functions (*initiate heater*, *heat media* and *circulate heat)* in order to show how the requirements presented in section 5.2 are fulfilled. Figure 7 presents the instantiated information model and illustrates how the parking heater's *control unit* interacts with the *heater* by means of a operand (information) flow between the components corresponding product functions. It is further illustrated how *design_constraints,* i.e. product requirements, connect to *product_functions* and further on to *product_components*. The hierarchical aspect of *product components* is captured in the case of the *control unit*. Further the figure illustrates how product functions are interconnected by means of operands. The instantiated model is written in the EXPRESS-I [ISO 1997] language whilst the emphasised subset is illustrated by simple boxes and lines in order to make it more comprehendible.
The instantiation diagram shows that functions and operands can be captured in an information model. Further it is illustrated how functions can be decomposed in to subfunctions and how traceability from requirements to functions and solutions can be achieved. The primary contribution to the existing ISO information model is the addition of operands carrying energy, material and information that constitutes a horizontal causality binding between functions. This addition has increased the information model's scope concerning behaviour, enabling an improved understanding of how different solutions interact in implementing a product function. This contribution is of special interest when there is a lack of spatial relations between technical solutions, as in multitechnology assemblies.

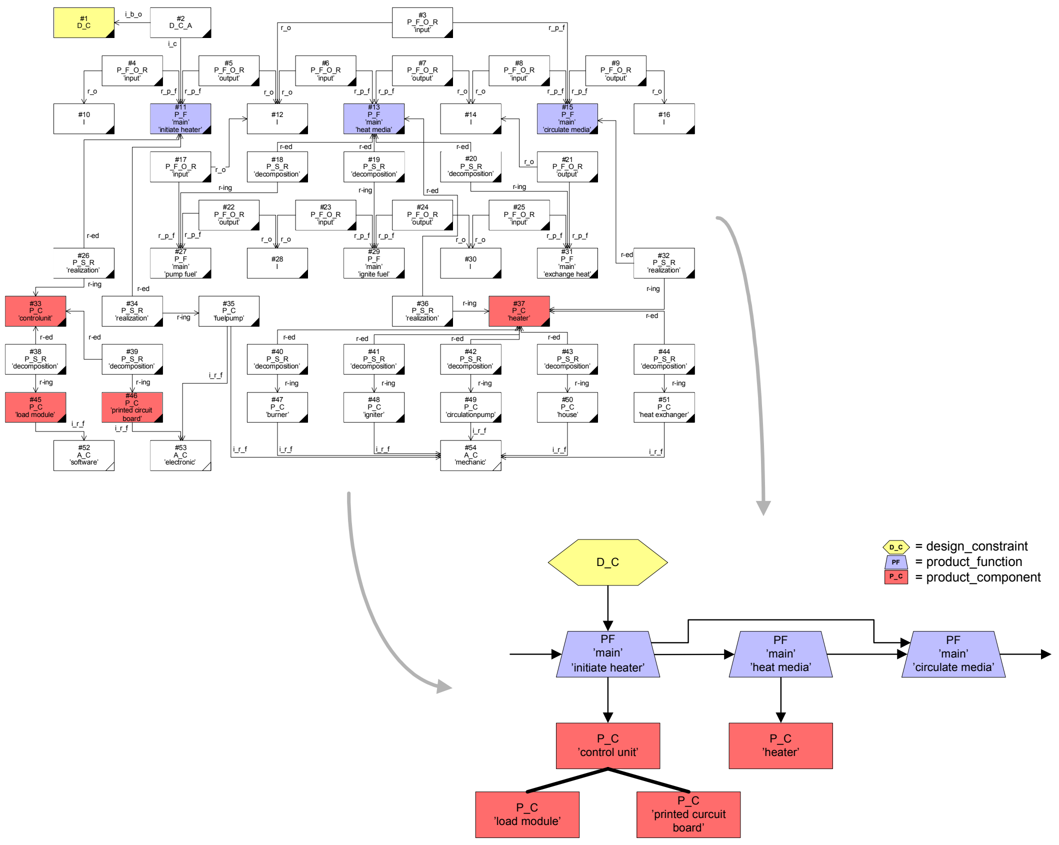


Figure 7. Instance diagram in EXPRESS-I (to the left) and its correlating simplification

7. Discussion and conclusions

In order to achieve consistent information management for the mechatronic product it is important to understand what type of information that is generated throughout its development process. The nature of mechatronic products is heterogeneous with a diversity of cooperating technical solutions, often of different technology. The presumption of this work has been that the functional abstraction can constitute a bridge between such technical solutions, enabling a more solid representation of the mechatronic product in product data management systems.

The basics of functional representation are delivered by Hubka and Eder in the Theory of Technical Systems. Their framework is described and implemented in the four design methodologies that have been object of survey and comparison in Section 4.1. In general, they all describe an analogous approach towards the establishment of product function, advocating transformational-based behaviour as the most suitable abstraction in conceptual product development phases. From an information management perspective, aiming at PDM-like implementations, a straightforward approach for the handling of product functions has been showed to be appropriate. The proposed information modelling constructs, see Section 5, have shown to sufficiently establish the tracelink between design artefacts. Modelling function is in many respects an abstract task. Thus the hierarchy dimension is important as it enables engineers to grasp, focus and thereby comprehend their own portion of work.

In the continuing design work, an ability of linking previously produced information with present information is obtainable. As design descriptions get more and more detailed and, as the development process progress, the necessity for a variety of information abstractions falls natural. This work focuses the functional abstraction, conveying also a higher (requirements) and a lower (solutions) abstraction level.

The phenomenon model and information requirements presented in Sections 5.1– 5.2 represent a summary of what has to be caught by an information model with respect to mechatronic products. They do not drive the development of an extensive information model but stress the importance of capturing the functional abstraction in a model for mechatronic products. The proposed constructs are incorporated into the STEP model, entailing that all the capabilities set by the information requirements are fulfilled. To validate an information model with instance diagrams is a first important step, but does not constitute an absolute verification for the accuracy of the information model. However, the validation case has indicated that the model satisfactorily fulfils the requirements.

The concluding remarks from this work are:

- Scope, complexity, line of action, and information managed in modern design theories are very similar regarding the establishment of product function in conceptual product development phases.
- A sufficiently defined link between hardware and software artefacts in mechatronic product development could be maintained by a relatively simple model for the functional abstraction, at least with the management of metadata in mind. A model based on the semantics of transformational-based behaviour has been shown to be feasible.
- The information model implements the important phenomena and requirements that could be derived from modern design methodologies.

Future work ought to encompass aspects of non-functional dependencies in the mechatronic product. There is also a need for exploiting the theory and the ideas further in computer tools, bringing the verification level beyond what can be delivered by instance diagrams.

Acknowledgement

This work was financed by Vinnova (Swedish Agency for Innovation Systems), whose support is gratefully acknowledged.

References

Andreasen, M M, “Designing on a "Designers Workbench" (DWB) “, Proceedings 9th WDK Workshop, Rigi, Switzerland, 1992.

Blanchard, B S, Fabrycky, W J, “Systems Engineering and Analysis”, third edition, Prentice Hall, Upper Saddle River, New Jersey, USA, 1998.

Dahlqvist, A P, Asklund, U, Crnkovic, I, Hedin, A, Larsson, M, Ranby, J, Svensson, D, “Product Data Management and Software Configuration Management – Similarities and Differences”, The Association of Swedish Engineering Industries, 2001.

Collier, W, “A Common Specification for Systems-Based Product Modeling – Product Definition & Commercialization”, D.H Brown Associates, Inc, Port Chester, New York, USA, 1999.

Duffy, A H B, Andreasen, M M, “Enhancing the evolution of design science”, Proceedings 10th International Conference on Engineering Design ICED 95 (WDK 23) – Vol.1, Praha/Heurista, Czechoslovakia, 1995, pp 29-35.

Hallin, K., Zimmerman, T., Svensson, D., Malmqvist, J., “Modelling information for mechatronic products”, Proceedings 14th International Conference on Engineering Design - ICED 2003, Stockholm, Sweden, 2003.

Hamer, P V D, Lepoeter, K, “Managing Design Data: The Five Dimensions of CAD Frameworks, Configuration Management, and Product Data Management”, Proceedings IEEE – Vol.84, 1996, pp. 42-56.

Hubka, V, Eder, W E, “Theory of Technical Systems”, Springer-Verlag, Berlin Heidelberg, Germany, 1988.

ISO TC 184/SC4, “ISO/TR 10303-12: Industrial automation systems and integration – Product data representation and exchange – Description methods – Part 12: The EXPRESS-I language reference manual”, ISO, Geneva, Switzerland, 1997.

ISO TC 184/SC4, “ISO/WD PAS 20542: Industrial automation systems and integration – Product data representation and exchange – Systems engineering data representation”, ISO, Geneva, Switzerland, 2001.

ISO TC 184/SC4, “ISO 10303-1: Industrial automation systems and integration – Product data representation and exchange – Part 1: Overview and fundamental principles”, ISO, Geneva, Switzerland, 1994.

ISO TC 184/SC4, “ISO 10303-214: Industrial automation systems and integration – Product data representation and exchange – Part 214: Application protocol: Core data for automotive mechanical design processes”, ISO, Geneva, Switzerland, 2003.

Johnson, J, Bailey, I, “How does AP233 support a Systems Engineering process (e.g. ANSI/EIA-632)?”, Proceedings INCOSE 2003 – 13th Annual International Symposium, Washington, DC, USA, 2003, pp. 1431-1443.

Korpela, T, “Product modelling in early phases of the design process”, Proceedings 7th International Design Conference DESIGN 2002 – Vol.1, Marjanović, D (editor), University of Zagreb and The Design Society, Dubrovnik, Croatia, 2002, pp 169-176.

Oliver, D W, Kelliher, T P, Keegan, J G, ”Engineering Complex Systems with Models and Objects”, McGraw-Hill, New York, USA, 1997.

Otto, K, Wood, K, “Product Design – Techniques in Reverse Engineering and New Product Development”, Prentice Hall, Upper Saddle River, New Jersey, USA, 2001.

Pahl, G, Beitz, W, “Engineering Design – A Systematic Approach”, second English edition, Springer-Verlag, London, UK, 1996.

Shakeri, A, “A methodology for development of mechatronic systems”, ISBN 82-4710-340-0, Norwegian University of Science and Technology, Trondheim, Norway, 1998.

Svensson, D, Crnkovic, I, “Information management for multitechnology products”, Proceedings 7th International Design Conference DESIGN 2002 – Vol.1, Marjanović, D (editor), University of Zagreb and The Design Society, Dubrovnik, Croatia, 2002, pp 551-559.

USAF, “MIL-STD-499 Functional Flow Diagrams”, DI-S-3604/S-126-1, 1968.

Trond Zimmerman, M.Sc.
Chalmers University of Technology, Department of product and production development
SE – 412 96 Göteborg, Sweden
Telephone: +46 (0) 31 772 1378, Telefax: +46 (0) 31 772 1375
E-mail: trond.zimmerman@me.chalmers.se