Representing Mechatronic Products in Interorganizational Information Systems

Joern Kaack, Joerg Leukel and Stefan Kirn

University of Hohenheim, Information Systems II,Schwerzstrasse 35, 70599 Stuttgart, Germany {joern.kaack, joerg.leukel, kirn}@uni-hohenheim.de

Abstract. Representing mechatronic products in interorganizational information systems requires means for capturing characteristics of this type of products. This role is fulfilled by mechatronic product models which are subject of this paper. A mechatronic product consists of three components: software, electronics, and mechanics. The challenge is that diverse interrelations between their components and with other products exist; these interrelations need to be considered in respective models. The problem of current modeling approaches, however, is that they either do not provide the required richness for expressing interrelations or focus engineering tasks only by describing the product behavior. We view mechatronic product models from a logistic perspective which allows us answering two questions: What is the scope of a mechatronic product model for interorganizational information systems? Which information is required in this model (i.e., determined by the goal of logistics)? We design a mechatronic product model from this perspective. We validate our model by studying an interorganizational scenario in an automotive supply chain.

Keywords: *Logistics operations, Inter-organizational enterprise systems, Product data management systems, Product life cycle systems, Supply chain management (SCM)*

1. INTRODUCTION

Mechatronic products consist of three components: software, electronics, and mechanics. They play a major role in many industries such as medical technology, industrial automation, and automotive [1-2]. The nature of mechatronic products increases the complexity of final goods that are often composed of multiple mechatronic products as well as conventional products.

Representing mechatronic products in interorganizational information systems requires means for capturing basic characteristics of this type of products. This role is fulfilled by mechatronic product models which are subject of this paper. The challenge is that, due to the nature of mechatronic products, diverse interrelations between their components and with other products exist (in case of embedded systems); these interrelations need to be considered in respective product models.

The problem of current mechatronic product modeling approaches, however, is that they either (1) adopt conventional product models for tangible goods or (2) limit the

model's scope with regard to the product life-cycle. The former does not provide the required richness for expressing interrelations, because conventional product models rely on simple, very often hierarchical relationships only. The latter focuses engineering tasks only by describing the product behavior to design the product functionality and guarantee dependability.

We address this problem by viewing mechatronic product models from a logistics perspective. This perspective allows us answering two questions:

1. What is the degree of granularity of a mechatronic product in interorganizational information systems? It is constrained by the smallest item within a logistic process, thus an item that can be identified, ordered, and delivered.
2. Which information is required in this model? It is determined by the overall goal of logistics, thus the information must support delivering the right item in the right quantity & quality at the right place at the right place for the right cost.

In this paper, we propose a mechatronic product model from a logistics perspective. This model aims at capturing product interrelations in a better way. We validate our proposal by studying an interorganizational scenario of spare parts logistics in an automotive supply chain. We can show that the novel approach reduces representational mismatches and supports logistic decisions.

The remainder of our paper is structured as follows. In section 2, we discuss existing work. Section 3 analyzes concepts of mechatronic product modeling and relates them to the logistics perspective. In section 4, we specify our mechatronic product model based on these considerations. Section 5 shows the application of this model in an interorganizational scenario. Finally, section 6 draws conclusions and points out avenues of future research.

2. RELATED WORK

The related work can be grouped into three major areas: mechatronic engineering, product data management, and product ontology engineering.

Mechatronic engineering provides models dedicated to mechatronic products. These models describe how such a product works and which structural and functional dependencies between its components exist [3]. Therefore, these models are designed from a *dependability perspective*: They aim at allowing for dependability simulation which describes the product behavior in terms of states, events, and possible faults [4, 1]. Representation means are state graph formalism [2], for instance, whereas quantitative aspects can be expressed by colored Petri nets [5], for instance.

Product data management (PDM) provides models aimed at capturing information which describes a product in order to support product-related activities within an organization. Its main goal is supporting the management of product complexity regarding number of products, parts, versions, and interrelations [6]. PDM focuses on engineering and manufacturing. Respective models describe products from a *product structure perspective*, thus in terms of interrelated parts and properties of these parts [7]. Representation means often rely on hierarchical structures formed by 'part-of' relationships which can be complemented by vertical relationships of limited expressiveness [8]. For instance, relationships can be classified into flow of energy or

information [3]. More recently, PDM's coverage is extending to other life-cycle phases such as sales and after-sales. The semantic expressiveness of interrelations, however, does not increase.

Product ontology engineering provides methods for constructing product ontologies which are consensual formal definitions of the concepts and interrelations within a product domain [9]. This field has been driven by the Semantic Web and yields models that capture the semantics of products using ontology languages (e.g., [9-10]). The *ontology perspective* requires defining product concepts unambiguously and distinguishing one concept from other concepts. Therefore, product ontologies and respective models rely mainly on taxonomic relationships. Another limitation is that product ontology engineering does not fully exploit product complexity, since it regards products very often as final products being subject of interorganizational systems such as e-commerce systems and thus limits complexity to simple product features that can be expressed by property-value pairs and standardized property definitions [11].

3. PRODUCT MODELING CONCEPTS

In this section, we introduce and analyze basic concepts of product mechatronic for mechatronic product and relate them to the logistics perspective.

3.1 Product Structure: Bill-of-Material

A basic concept of product modeling is representing its structure, thus the parts that constitute the product, as a tree called bill-of-material (BOM). A BOM describes the product structure by multiple layers of detail. BOMs can be designed from various perspectives such as engineering or manufacturing. In manufacturing, the BOM describes of which assembly groups and atomic parts the final product consists. Here, the term 'part' refers to any material which is handled within the respective enterprise and thus subject of intraorganizational information systems, i.e., ERP systems.

With regard to mechatronic products, the BOM concept allows for distinguishing specific layers within the product structure. Obviously, mechatronic products form one layer above the mechatronic component layer. Since mechatronic products are often part of more complex final products, layers above the mechatronic layer group parts into modules. Figure 1 shows an example BOM by distinguishing four layers: final product, module, mechatronic product, and component. Such as BOM does not cover interrelations between elements of the same layer; hence it contains, for instance, no information about dependencies between the software and electronic component of a mechatronic product.

From a logistics perspective, the BOM concept can be employed for other purposes than manufacturing. For instance, a BOM for sales or distribution logistics contains only parts that are relevant for sales and after-sales, e.g., optional and spare parts. Therefore, 'part' refers to items that can be ordered by referring to an order number, transported, and delivered. Regarding mechatronic products, the logistics perspective constrains the scope of the product model as evident by the BOM concept: The

smallest item within a logistic process must be part of such a BOM. The consequence is that the degree of granularity is lower than those of engineering-oriented models. The latter models are more detailed (i.e., BOMs listing elements of a circuit board).

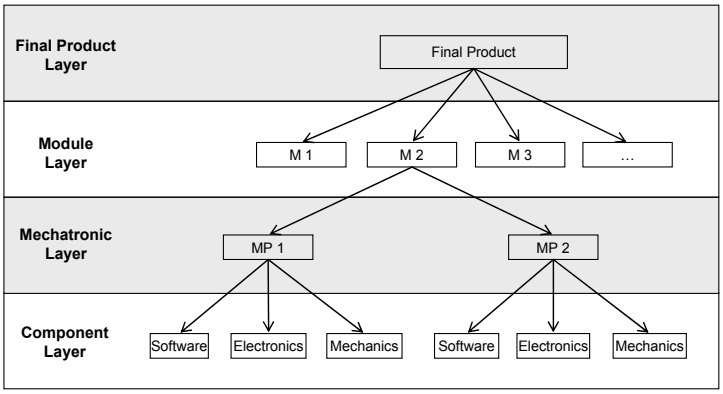

Figure 1. Example BOM for Mechatronic Products (M: module MP: mechatronic product)

3.2 Product Dynamics: Versioning

A basic concept for representing product dynamics, thus changes in product specification, is versioning. It increases product complexity and requires representational means for maintaining version numbers and describing the sequence of versions over time.

With regard to mechatronic products, dynamics is an inherent property. The reason is that its software component can be changed easily by updating the software without changing other components. These changes can even be executed by customers during the usage of the product. With regard to mechatronic products, one has to consider that each component has its own version history and contributes to the version of the product. The number of versions over time is quite different with greater dynamics in the software component (see figure 2). There also exist dependencies between software and other components, since a software update may require a change in electronics. Thus, compatibility is a major issue in mechatronic products. This aspect is even more important for final products consisting of multiple mechatronic products which control the final product's behavior. This additional information can not be expressed in version graphs though.

From a logistics perspective, versioning is important for meeting the overall goal of logistics, thus delivering the 'right item'. The attribute 'right' is mainly related to technical, functional, or qualitative characteristics of the respective and other items. Since significant changes of characteristics lead to a new version and all items can be subject of versioning, the mechatronic product model must at least include version numbers and, in addition, provide means for expressing relationships between

versions, i.e., sequence of versions, down- and upward compatibility. The latter information determines whether an item is right or not depending on other items.

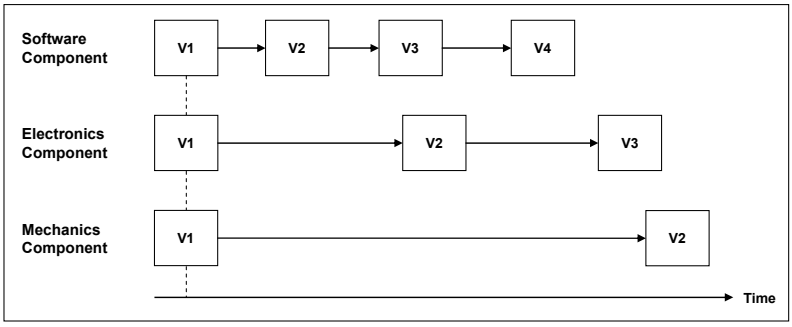

Figure 2. Example Version Graph for Mechatronic Products

Product dynamics can be captured on a wider scale by the product life-cycle which describes the 'life' of a product over time. This life-cycle consists of subsequent, though overlapping phases: product planning, engineering, manufacturing, distribution/sales, usage/service, and recycling. Mechatronic life-cycles are very different in terms of phase length, number of versions, and overlapping. For instance, in the automotive domain, long life-cycles are typical for the final product (e.g., 3 years development time; 7 years production; 15 years usage and service), whereas both software and electronics components have shorter life cycles (e.g., electronics: 4 years; software: 1 year).

4. MECHATRONIC PRODUCT MODEL

In this section, we define a mechatronic product model which addresses interrelations.

4.1 Product Interrelations

We define a product interrelation as a typed relation between at least two parts of a product. In contrast to a bill-of-material, which is structured hierarchically, product interrelations allow for describing relations apart from the hierarchy. In particular, these relations can be defined from diverse perspectives and for various reasons. Thus these relations are not confined to only one type, which is often that of 'part-of'; hence interrelations build a network of relations.

In the following, we define the semantics of our model. We focus on interrelations and abstract from the position of related parts within the product hierarchy, thus these relations can connect any part in such a hierarchy. In particular, we allow interrelations in and across all layers. For this reason, we introduce 'element' to

denote any type of module, product, part, or component that can be subject of logistics (thus it equals 'item').

4.1.1 Interrelation Type

An interrelation type defines the reason or cause why a respective interrelation between two elements A and B exists:

'Function': There is a functional interrelation between A and B, i.e., A can only provide/guarantee its functionality if B exists. Any modification of B such as version update or replacement may influence the functionality of A.

'Compatibility': This version-related interrelation between A and B describes which version is required respectively not permitted. This type refers to upward and downward compatibility. The latter says that version Y is an improvement of version X, thus Y performs at least equal to X and more; whereas X is only able to perform little of release Y.

'Time': This time-related interrelation between A and B describes that a particular version of B becomes outdated depending on its date. This type allows for replacing obsolete parts B due to changes of A.

'Context': The interrelation results from the final product's context, e.g. region, culture, or law restrictions. It allows for customizing products due to different contexts and guaranteeing that context-specific requirements are being met.

The interrelation types abstract from current types that describe only physical relations between elements such as 'part-of', 'substance-of', and 'member-of' [12]. These types have been derived from the overall goal of logistics of delivering the 'right item' and reflect different aspect of what is 'right'.

4.1.2 Cardinality

The cardinality defines how many elements are related with other elements due to a defined interrelation. Here we employ common cardinality means: one-to-one (1:1), one-to-many (1: N), many-to-one (N:1), and many-to-many (N: M). In addition, we allow determining fix numbers for N such as 1:4.

The cardinality means can also be used for meeting the logistics goal of delivering items in the right quantity; for instance, by defining the minimum number of related items.

4.1.3 Direction

The direction defines whether an interrelation has to be interpreted unidirectional (one-way) or bidirectional (two-way). The former distinguishes dependent elements (B) and elements which impact others (A). In case of a bidirectional interrelation, both sides are subject of a two-sided impact. It is possible to substitute a bidirectional interrelation with two single bidirectional interrelations.

4.1.4 Effect

The effect defines the consequences of an interrelation, thus the impact on the depending elements. We employ means of Boolean algebra:

- Inclusion (→): The dependent elements are required; otherwise the product has an invalid state which may harm its behavior.
- Exclusion (¬): The dependent elements are not permitted, thus A and B may not exist at the same time in the same context.
- Disjunction (∨): The dependent elements are connected by the logical OR. At least one element must be present to fulfill the criteria of dependability.
- Conjunction (∧): The dependent elements are connected by the logical AND. It means that all dependent elements must be present.

4.2 Conceptual Modeling

Next, we transform the semantics into a conceptual UML model. We employ class diagrams which provide associations as a basic formalism for interrelations.

4.2.1 UML Associations

Associations describe relationships between classes. Adopting this formalism for product interrelations requires mapping the concepts defined in section 4.1 to respective UML modeling primitives as follows:

- Each element belonging to an interrelationship is represented by a class.
- Each interrelation is represented by an association.
- The cause of an interrelation defines the name of the respective association.
- The cardinality of an interrelation defines the multiplicity of the respective association.
- The direction of an interrelation defines the direction of an association.

It is, however, not possible to include the effect of an interrelation in the respective association. Depending on the number of interrelated elements (here: classes), one has to distinguish binary and ternary associations. Figure 3 shows two respective examples. A severe weakness of this modeling approach is that UML associations limit the number of interrelated classes (element types) to three.

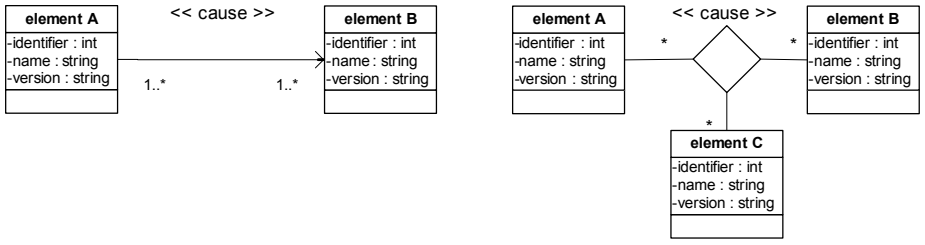

Figure 3. UML Class Diagram with Binary and Ternary Associations Respectively

4.2.2 UML Association Classes

To compensate the stated limitations of associations, we replace the association with an association class which enables including additional semantics by means of attributes. The resulting model is shown in figure 4.

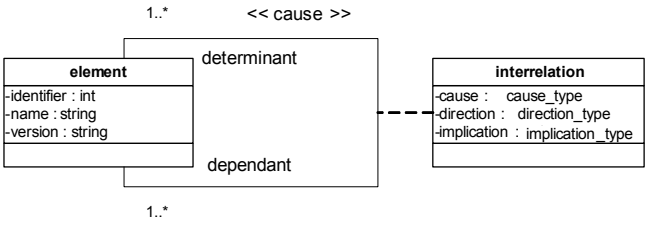

Figure 4. UML Class Diagram with Association Class

In particular, we make the following design decisions:

- Elements belonging to an interrelation are represented by a single class. Different elements can be distinguished by its identifier and version.
- Each interrelation is represented by an association class which is connected via two associations with the respective element class. The first association has the role name 'determinant' and denotes elements which impact others. The second association characterizes elements ('dependant') which are dependent from others, thus from a determinant.
- The cause, direction, and effect of an interrelation are represented by respective attributes of the association class.
- The cardinality of an interrelation is represented by the multiplicity of the associations.

5. SCENARIO: AUTOMOTIVE SUPPLY CHAIN

In this section, we show the application of the proposed model in an interorganizational scenario in order to give evidence of its validity. We use the scenario technique as a descriptive design evaluation method. The scenario considers a multi-tier automotive supply chain. It includes suppliers of software, electronics and mechanics, a car manufacturer, and garages as part of the service infrastructure.

Maintenance, repair and vehicle recalls are common events in the service and usage phase of product instances. A major part of these events are caused by electronics and software failures in mechatronic products. In these cases, garages need exact and rich information regarding the components to be replaced and other components which may be affected directly or indirectly because of side effects.

We study the following recall: The recall concerns an electronic stability control (ESC) system which needs to be replaced for a predefined set of cars; this new version 3.0 works only properly if the brake assist system's (BAS) version is 7.0 or higher. This interrelation requires that every ESC replacement based on the recall has to check the BAS version in the respective vehicle and, if necessary, has to upgrade this one to version 7.0 or higher. Figure 5 shows the respective object model (an instantiation of the model of figure 4).

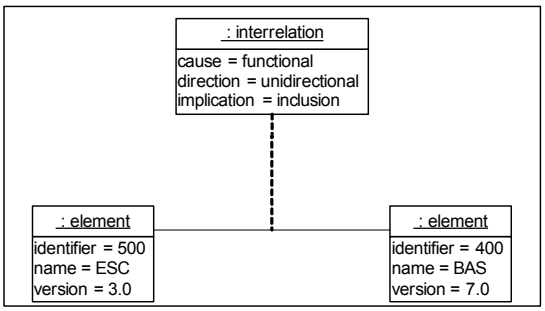

Figure 5. Vehicle Recall

Independently from the vehicle recall, the BAS underwent modifications by the respective supplier during a periodical revision process. Due to this process, the version 6.3 – which had been introduced earlier – provides a substantial new feature which required changing two components of the tire control system (TCS); the microcontroller (software) needs to be updated and a wire which connects both BAS and TCS needs to be replaced by a shielded one. The model in figure 6 shows this interrelation.

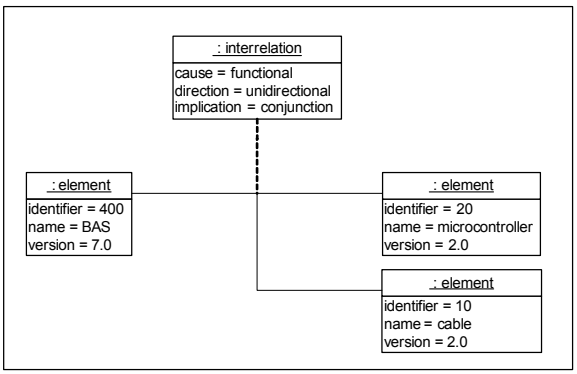

Figure 6. BAS Revision

At this point, it becomes evident that the recall-triggered modification of the BAS may require two additional modifications of the TCS, depending on the current configuration of each individual vehicle. This dependency between recall and revision can be made explicit by linking the two respective models. Due to sharing the same conceptual model, a transitive dependability between ESC and TCS can be derived as shown in figure 7.

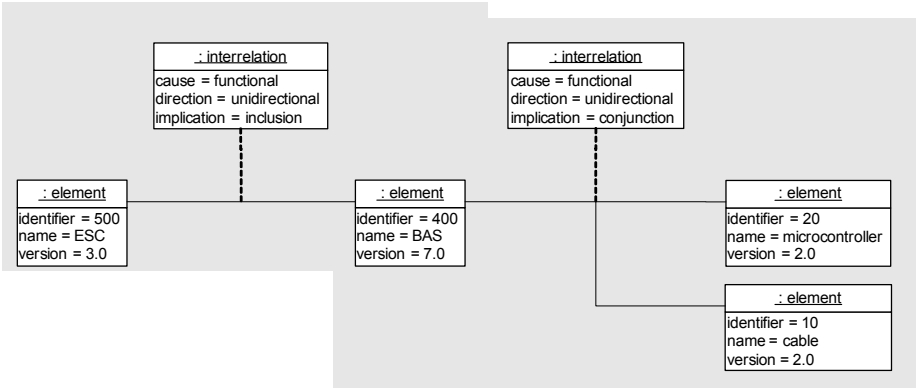

Figure 7. Vehicle Recall and BAS Revision

This additional information ensures that implementing the product recall considers all other relevant interrelations which exist in the product interrelation network.

6. CONCLUSIONS

In this paper, we have proposed a mechatronic product model aimed at capturing interrelations as required by interorganizational information systems. The designed representation formalism captures interrelations in and between mechatronic products. In particular, we analyzed conventional concepts of product modeling and employed UML class modeling. We demonstrated the application of the novel approach in a scenario of spare parts logistics and provided first evidence of validity that the model reduces representational mismatches and supports logistic decisions.

The current model has two main limitations. First, it concerns specifically interrelations beyond 'part-of' relationship types, thus it does not address other relevant aspects of product structures. Therefore, future work on potential ways of coupling the current model with other, existing models is foreseen. Second, the current modeling approach aims at providing a richer set of relationship semantics, though it does not employ dedicated formal languages for making this semantics explicit. We acknowledge a clear linkage to ontology languages based on a higher logic such as first-order logic.

REFERENCES

1. M. Broy, Automotive Software Engineering, in *25th International Conference on Software Engineering* (IEEE Computer Society: Los Alamitos, CA, 2003), pp.719-720.
2. O. Larses, *Modern Automotive Electronics from a Dependable Systems Perspective*, Technical Report (Royal Institute of Technology: Stockholm, 2003).

3. S. Burmester, H. Giese, and M. Tichy, *Model-Driven Development of Reconfigurable Mechatronic Systems with Mechatronic UML*, in LNCS 3599 (Springer: Berlin 2005), pp.47-61.
4. M. Tiechem, M. Storm, M. Andreasen, and M.K.J. Callum, Product structuring – an overview, in *Proc. of 11th International Conference on Engineering Design* (1997).
5. G. Moncelet et al., Analysing a mechatronic system with coloured Petri nets, *Intern. Journal of Software Tools for Technology Transfer*. Volume 2, Number 2, pp.160-167, (1998).
6. D.J. Chen and M. Törngren, Towards A Framework for Architecting Mechatronics Software Systems, in *Proc. of 7th IEEE International Conference on Engineering of Complex Computer Systems* (IEEE Computer Society: Los Alamitos, Cam, 2001), pp.170-179.
7. T. Männistö, H. Peltonen, T. Soininen, and R. Sulonen, Multiple abstraction levels in modelling product structures, *Data & Knowledge Engineering*. Volume 36, Number 1, pp.55-78, (2001).
8. T.K. Peng and A.J.C. Trap, A step toward STEP-compatible engineering data management: the data models of product structure and engineering changes, *Robotics and Computer-Integrated Manufacturing*. Volume 14, Number 2, pp.89-109, (1998).
9. M. Hepp, Products and Services Ontologies: A Methodology for Deriving OWL Ontologies from Industrial Categorization Standards, *International Journal on Semantic Web & Information Systems*. Volume 2, Number 1, pp.72-99, (2006).
10. T. Lee, J. Shim, H. Lee, and S.G. Lee, *A Pragmatic Approach to Model and Exploit the Semantics of Product Information*, in LNCS 4244 (Springer: Berlin, 2006), pp.242-266.
11. J. Leukel, Standardization of Product Ontologies in B2B Relationships-On the Role of ISO 13584, in *Proc. of 10th Americas Conference on Information Systems* (AIS, 2004), pp.4084-4091.
12. S. Virtanen, Reliability in product design – Specification of dependability requirements, in *Proc. of Annual Reliability and Maintainability Symposium* (1998), pp.82-88.