A CONCEPTUAL FOUNDATION FOR VERSATILE E-COMMERCE PLATFORMS

Prof. Dr. Ulrich Frank
Institut fuer Wirtschaftsinformatik, Universitaet Koblenz-Landau
Rheinau 1, 56075 Koblenz, Visitor Address; Rammsweg 1, Tel:
E-Mail: ulrich.frank@uni-koblenz.de

ABSTRACT

Electronic commerce demands appropriate reorganization of business processes made in conjunction with the introduction of powerful information systems. There are numerous already-proven concepts from business management and information systems that can be partly relied on. There is, however, a series of specific demands that must be accounted for in the design of specialized systems. The following article is directed at object-oriented modeling of products that are offered on e-commerce websites. Unlike traditional inventory management, products traded on e-commerce websites are often not fully recognized at the time when the system is constructed. Instead there is the need to register new product types during the run time of a system. In addition, the representation of product variants and the construction of individual product configurations should be possible. Motivated by a series of additional demands to be leveled at systems such as this, two obvious modeling approaches will be described first. A powerful approach will then be mapped out, which is based in part on the use of meta concepts. This approach originated within the design of a reference model for e-commerce platforms on the Internet.

1. Motivation

The initiation and processing of business transactions over the Internet, or, in short, “e-commerce” requires, next to suitable marketing concepts, the reorganization of business processes and, if applicable, strategic alliances between various firms. In addition, great significance should be attached to powerful information and communication systems. In order to cope with the functions linked to this, one can fall back on a large range of proven concepts from business management theory, information systems research or computer science. However, at the same time, e-commerce demonstrates a series of additional, distinctive features that bring with them attractive research problems. From an information systems perspective, the following aspects are of particular importance:

Increasing automation of business processes: even when e-commerce seems to focus at first sight on the generation of business over the Internet, (and to which some current offers actually limit themselves), economic reasons suggest an integration and extensive automation of all involved processes, such as procurement, logistics and financial transactions. The technical support for processes of this kind requires systems that guarantee a high degree of reliability and integrity. Amongst other things, concepts must be drawn up that allow close integration of the different systems operated by the various companies involved in a business process.

Need to cover various business models: the pre-mature nature of e-commerce implies that the decision for a generic business model, like B2B or B2C may have to be revised over time, for instance by offering goods both to businesses and to consumers or to act as a broker rather than as a retailer. In addition to that it may be necessary to change particular aspects of a business model. This may be the case for new forms of pricing. Within traditional business, prices are, as a rule, determined by the supplier. Only in some cases can they be individually adapted through negotiations with an appropriately authorized employee. Trade over the Internet offers a range of pricing mechanisms (diverse forms of auction, time discounts, bundling of demands etc) that have been known about for a long time, but that obtain a new quality by allowing for – from the perspective of the supplier – automated price negotiations. In order to stay competitive, it may be necessary to allow for various pricing mechanisms.

Contingency of the products offered: the diversity of products offered over e-commerce websites on the Internet is sometimes considerable; in some cases – for example in auction platforms – it is almost unlimited. In contrast to department stores or mail order companies that also often offer a wide array of products, the range does not have to be planned in the medium term. Instead, it is possible to make diverse, almost totally unpredictable changes to the product range on a daily basis; today, leather seats; tomorrow, helicopters. Since all of these products are managed by information systems and numerous operations are to be performed by them, they must be described *meaningfully* with regard to corresponding use cases**.** Therefore, it is all about the seeming paradox of sufficiently describing objects of which one has, up to that point, no knowledge.

The dynamics of e-commerce markets demands one to be highly adaptive. This implies, together with the high degree of automation required for corresponding business processes, a considerable challenge for

information systems research. To increase the productivity of constructing and maintaining information systems, it is essential to have access to reusable, adaptable artifacts, such as classes or components. Furthermore, there is need for communication interfaces (exchange formats) that are semantically rich (in order to foster automation) and versatile (to cover a wide range of communication needs).

For both objectives concepts of products are essential. Firstly, they are required to specify software artefacts that are used to build or maintain information systems. Secondly, they are needed to specify interfaces for exchanging electronic documents like orders or invoices. Concepts of products that can be found in database schemata of today's ERP systems are not flexible enough because they were designed for traditional business models. On the other hand, EDI protocols like UN-EDIFACT do not allow for comprehensive descriptions of arbitrary products because they are based on the assumption that the firms involved in a business transaction have a common understanding of the referenced products. Therefore the challenge for information systems research consists, amongst other things, of finding abstractions of products that offer a high degree of flexibility and integrity to e-commerce platforms.

2. Requirements for the Modeling of Products

In specialist shops, qualified sales assistants are at the customer's disposal to find the product in question required and, if necessary, to provide the customer with additional information. The function of the sales assistant in Internet trading is replaced by suitable product classifications and descriptions. Corresponding catalogues, sometimes called "Internet Electronic Product Catalogs (IEPC), have been discussed for some time as being fundamental components of trading sites. The demands made in connection with catalogues of this kind understandably focus above all on the presentation of products to the customer. Therefore, according to Timm and Rosewitz IEPCs are aimed at putting "multimedia product representations" [TiRo98, p. 118] at the customer's disposal. A similar understanding of product catalogues is found in the work of Keller and Genesereth [KeGe97].

With regard to the realization of powerful information systems, the restriction to presentations of product descriptions is not adequate. Therefore, we will not use the term product catalogue from now on. We will instead apply the term *conceptual product modeling*. A conceptual product model, such as a data or an object model, can be used to guide the implementation of corresponding information systems. It is also a blueprint for the specification of communication interfaces. Before we consider current approaches to model products, we will have a more detailed look at the requirements a corresponding system should fulfill.

In principle, it should be possible to represent any products (or more exactly: types of products).

The registration of new product types should not require a change of the program code or of the database schema, because in view of the general availability of the system and the frequent appearance of new product types this would not be acceptable.

In addition to this there are requirements that correspond to a great extent to those at which the product catalogues already mentioned aim:

The objects stored on the basis of conceptual product models should support the customer when searching for suitable products.

With this, the description of products from the point of view of the customers should have a sufficient content and be fairly set out for all their various needs in a detailed and clear form. This requires that both a customer searching a particular type of furniture and another one searching for an excavator should be able to specify relevant features. Common search engines are usually not satisfactory. They operate on full text representations that do not include product descriptions as semantic structures. For example, imagine to search for a dining-table with a glass top and mahogany table-legs. While it would not be possible to express exactly what you are looking for, a search engine would retrieve pages that contain both glass and mahogany tables but maybe none that satisfies the relevant search criteria.

Besides that, there are a series of requirements that are connected to the automated use and care of product data:

The product descriptions should allow for the collection, over a period of time, of relevant marketing knowledge about the purchasing behavior of customers. This requires the analysis of buying preferences with respect to specific features of products.

The system should extensively shut out the registration of meaningless product descriptions. That implies to define products types that restrict the possible descriptions of corresponding instances.

The same is valid for the registration of customer orders:

It should be possible to represent *product variants* as such, since in this way not only can the analysis of purchasing habits mentioned above be supported, but also redundancy of data registration and use can be avoided: Features that have been described for a product type already do not have to be described again when it comes to specify corresponding variants.

The customer should be in the position to specify individual *configurations*, e.g. by assigning equipments to a car. To avoid orders that cannot be satisfied, it is essential to allow for correct configurations only.

Different forms of pricing and price assigning (for product types and single products) should be possible.

The product descriptions should be versatile enough to satisfy diverse communication interfaces (e.g. to customers, suppliers, banks, and logistical partners).

Within specialized communication protocols for e-commerce (see 5) as well as in state of the art ERP systems, one can find various approaches to describe products. Most of them are based on one of the two prototypical approaches we will consider in the following section.

3. Current Approaches to the Modeling of Products

The modeling of products is a central component of many business management applications, for instance of goods trading systems, production planning and taxing systems or of ERP system in general. The focus of our analysis is limited to approaches of this type that are found in trade with a diverse product range.

3.1 "Flat" Product Concepts

In order to be able to represent as many product types as possible with just one concept, it is necessary to restrict the concept to features that all product types have in common. The differentiation between various product types follows solely through specific initializations of the generic features. In the following examples we use classes to define concept. The notation used corresponds to UML. The class specification in Figure 1 is an example of a product concept of this kind.

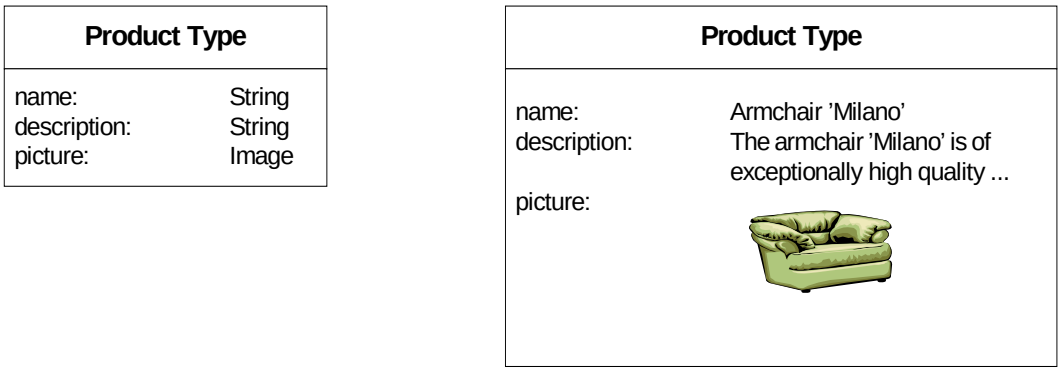

Figure 1: Generic product class and example

As the indication of this generic class shows, this instance deals not with the representations of single real-world objects, but much more with product types. Therefore it does not allow for representing of concrete product instances, for example a particular washing machine. However, often a mapping of product instances to objects is not appropriate anyway because the expenditure linked to the necessary identification of the real-world objects is usually far too large – think for example about the bottles in the stock-room of a drinks company. For trading sites on the Internet representations of product types will generally be sufficient. If the characteristics of concrete product instances play a role, for example for second-hand vehicles, there is need for additional classes, which allow the representation of related instances (a detailed representation of different alternatives can be found in [Fran99]).

Obviously requirement 1 is fulfilled in so far as a description for all products is conceivable that corresponds to this concept. Requirement 2 is fulfilled unreservedly, since the creation of a new product class merely requires the instantiation of an object. Requirements 3 and 4 can also be fulfilled using this approach. It remains though to be recorded that both the search for features and the information about product characteristics are not supported by corresponding, explicitly made concepts in the class description, but simply through character strings. The collection of knowledge about commercial decision-making processes is certainly not excluded by a semantically flat product concept. At the same time, it must be taken into account that a product concept of this kind does not support the differentiated analysis of consumer behavior, since the product types that are offered cannot be distinguished conceptually. Requirement 6 marks a persistent weakness of this concept, since in conceptual terms almost all nonsensical product descriptions are possible. Specifying product variants is not feasible on a conceptual level since it is not possible to show which characteristics distinguish one variant from another (requirement 7). For similar reasons it is not possible to represent product configurations (requirement 8).

In view of trade with products, it may well be noticeable that important characteristics such as buying and selling prices or stock are not contained within the class description. Since the keeping of stock may possibly lie completely outside of the company, namely with a logistical partner, the management of stock should also take place there. It would certainly be of consideration as to whether this externally managed stock would be accessed through operations that are offered by the class. The same goes for prices. Even when pricing

mechanisms are used that lead to differing selling prices for products of a certain type over the course of time, the selling price, or to be precise the lowest selling price, must be determined somewhere. That can come about with help from the attributes of the class. It is however also conceivable to refer suppliers to contracts with general specifications for pricing mechanisms, in which terms and conditions are specified. Similar considerations go for the two approaches not introduced as yet. Prices and stock present therefore no fundamental criteria for our comparison. In this respect the class description shown in Figure 1 is indifferent in comparison with the consideration of different price mechanisms (requirement 9). Communication with external partners demands context specific information about products, like for example weights and measures or also the respective warning classifications. As long as this information is standardized - that is, that it exists in the same form for all products - it could be reproduced by means of suitable attributes. If, however, certain information is specific to single product types, the approach is no longer persuasive, since it would in this case lead to conceptual redundancy and therefore to confusion for those which are in charge of recording new product types.

To summarize this approach is an example to show that concepts with relatively small semantic content (semantic being understood as content of information) can be used for a wide spectrum of cases. They imply, however, severe disadvantages. First, they limit the chance for powerful retrieval, because features that may be relevant for searching a particular product type are not part of the conceptual description. Second, they provide only partial support for the automation of business transactions. For instance, it is not possible to specify product configurations. Third, they jeopardize the integrity of a system by permitting diverse interpretations (with respect to automatic interpretation, the string used to describe a product type can mean anything).

CLASS HIERARCHIES

If it is necessary to express differences between product types on a conceptual level, additional classes are required. In order to take advantage of commonalities of various product types, it seems appropriate to associate these classes through generalization/specialization relationships. In Figure 2 an example for a corresponding generalization hierarchy is given.

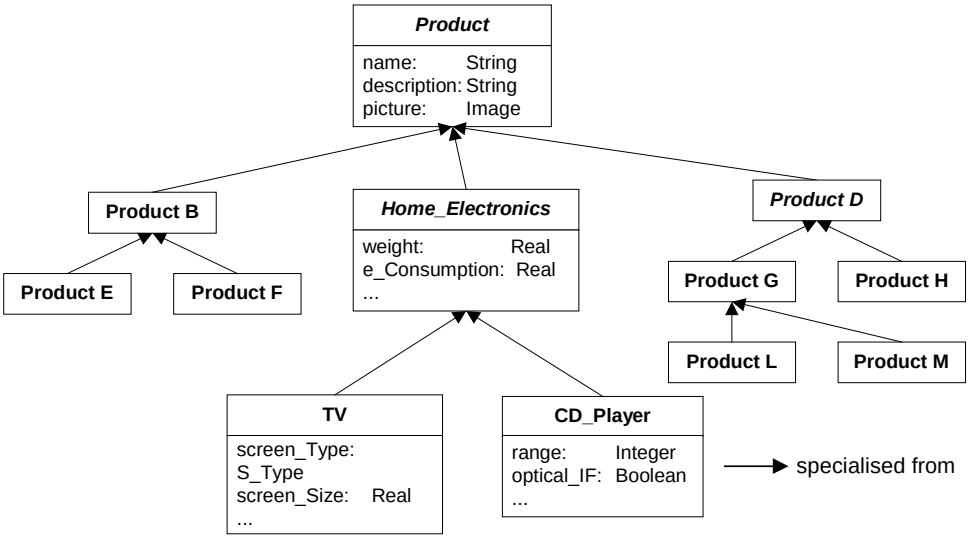

Figure 2: Example of a generalization hierarchy

In comparison with the flat product concept sketched out above, a class and not its instances serve here to represent product types. This requires the introduction of many more specialist classes. Notwithstanding this, it may be that there is only one instance of a particular class, since mapping of single real world objects to one instance is too costly (see 3.1). In principle the quality of a generalization hierarchy of this kind is dependent on whether the similarities that are the basis for generalization relationships are fundamentally stable over time for the respective application domains. This can only be tested in a particular case. In the following section we will limit ourselves to looking at the general assessment of this approach in view of the sketched-out requirements.

Since the number of product classes used in a generalization hierarchy of this kind is non-determined, as many product types as one likes can be represented (requirement 1). The introduction of a new product type certainly demands the expansion of the database schema – or of program code respectively (requirement 2 is not fulfilled). Requirements 3 and 4 are better fulfilled than by flat product concepts, since the conceptual peculiarities of product types can be represented as such. For the same reason the prerequisites for the analysis of customer behavior are more favorable (requirement 5). Since it is possible to describe the characteristics of a

product type in a complex and, if necessary, restrictive manner, the range of nonsensical initializations can clearly be limited (requirement 6). The representation of product variants (requirement 7) is not provided for at first in this approach. It is certainly badly suited to corresponding expansions, too. In the simplest case a variant is characterized in that single features show certain states (a certain color for instance). If one wished to cover up this case with an additional class that is associated with a product class, one would have to adopt all the attributes of the product class whose forms can vary – even if possibly one only needs one attribute in a particular case. If a variant shows additional features (like for instance a sunroof in a car) that other features require (e.g. a certain furnishings package) or exclude (e.g. air-condition), the representation of a product type by *one* class is no longer adequate because it would not allow to properly describe valid sets of features (if they were modeled as attributes). This argument is even more applicable to configurations (requirement 8), since a variant can be interpreted as a particular configuration. With regard to the pricing mechanisms (requirement 9) this approach is to be assessed in a similar manner to the former – except for the circumstance that special pricing mechanisms could be assigned to product types. Since this approach allows a semantically rich description of product types, the chances of satisfying external communication interfaces (such as standardized message types) is clearly better than in the first approach.

The following table shows the comparative assessment in a simplified form:

Table 1: Comparative Assessment of current Approaches

Requirement	"Flat Concept"	Specialization
1	+	+
2	+	-
3	~	~
4	~	~
5	o	+
6	-	+
7	-	-
8	-	-
9	~	~
10	-	+

+ requirement fulfilled - requirement not fulfilled o requirement partially fulfilled ~ indifferent

4. A Meta-Model oriented Approach

Both approaches discussed up to this point demonstrate considerable weaknesses in that they do not allow for the description of variants and individual product configurations in a satisfactory way. In addition to this, the possibly weighty disadvantage for the otherwise powerful approach (specialization hierarchy), remains; namely that the introduction of new product classes demands an intervention into program code and the database schema. Since the permanent smooth operation of an e-commerce platform is an essential requirement, avoiding frequent changes to the program code or schema is obligatory. Neither approach is therefore satisfactory.

The search for an alternative approach aims above all at overcoming the conflict between the range of the concepts used (if possible all conceivable product types should be covered) and their semantics. In view of the circumstance that the products to be modeled are not well known in advance, solving this conflict seems hardly possible. On closer inspection this restriction relatives itself: even when the concrete forms of certain product types cannot be made known to us in advance, we can develop perfectly substantial ideas as to which product types are possible in principle – or, to put it another way, which *languages* one needs in order to describe a product class in the sense of the requirements already named.

4.1 Central Concepts

The problem shown above is not new. In certain ways – if on a clearly higher level of abstraction - it is fundamentally valid for the construction of models within software engineering. Also there one does not know the peculiarities of the possible domains that may ever be modeled at first. However, if one has an idea of how domains can be structured in general, one can at least specify a task- or domain specific modeling language. In this way a purposeful description of facts unknown at first can be - at least indirectly - supported. The specification of description concepts requires an appropriate language that is specified for example through a meta model. A meta model serves to describe the concepts used on the object level. For instance, a meta model of an object modeling language would define concepts such as 'Class', 'Attribute' or 'Operation'. Besides this, the problem demonstrates a clear similarity in the description of aggregate compositions, as it plays a role in production planning and control with recourse to models for lists of parts. Hence, the approach sketched out in the following section demonstrates parallels both to meta modeling and to the modeling of aggregates. Instead of describing specific product types, our approach defines suitable concepts to describe product types. In other

words: It is based on a generic (we could also say: 'ontological') notion of 'Product Type'. On the one hand it promotes a much higher level of abstraction and flexibility than approaches that specify product types directly, giving the user the freedom to define any product type he may need. On the other hand, it guides the user with the definition of new product types by offering specialized concepts and by enforcing rules to apply them.

Figure 3 shows the central classes of a corresponding object model.

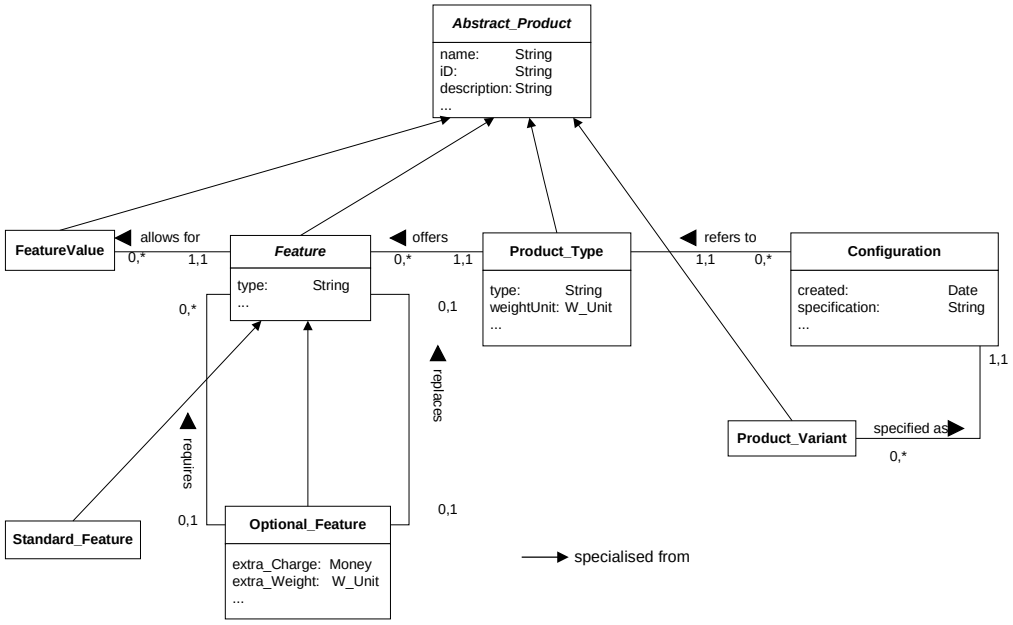

Figure 3: Meta modelling-based approach for the description of configurable product classes

This model is valid for all conceivable product types. To introduce a new product type, one would instantiate an object of Product_Type first. Then one would assign the necessary number of features – either instances of Standard_Features or Optional_Feature. Features may be available in various form (color, size, etc.). The range of available feature forms can be specified by assigning corresponding instances of Feature_Value. Associations between features (replaces, requires) allow one to express configuration constraints. All the requirements formulated in section 2 are fulfilled using this approach, whereby as in the other approaches pricing mechanisms are abstracted. However, one must consider that only a part of all conceivable configuration constraints can be expressed (through the association between Optional_Feature and Feature). For instance, it would not be possible to specify that a sun roof is available only for black cars, because that would require one to refer to the state of a Feature_Value, not just to the existence of another feature. However, a limitation of this kind is not a serious drawback in view of the intended scope of application. It would not be economic anyway if a trading site reproduces the detailed configuration possibilities of a complex product (a car, for example) in the same detail as the respective manufacturer. Therefore, for complex models the model is suited to describe subsets of all possible configurations. All individual configurations are managed as instances of Configuration, whereby the attribute specification serves to describe an individual configuration in a suitable language (for example as an instance of a corresponding XML document type). Figure 4 clarifies this fact with an example. It represents standard and optional features of a particular sofa type and thereby expresses the set of possible configurations. In addition to that it shows an excerpt of a particular configuration that is represented as an XML string, which would be stored with an instance of the class Configuration.

Even when this approach supports a high degree of flexibility and concurrently a high level of integration and all in all, fulfils to a great extent the demands shown, it also has drawbacks. These can be traced back above all to the fact that this approach introduces an abstraction that may possibly not be expected of the system user, and – as already mentioned – cannot be directly supported by current software tools. If, for example, all types of vehicle with a motor having a power output of more than 100 KW are searched for, then a corresponding inquiry, either in SQL or an object-orientated programming language would be trivial, if there existed a corresponding class such as 'Car'. With our approach the specification of the inquiry is less intuitive, because the inquiry is directed at first to meta concepts – like Product_Type or Standard_Feature. Only after this do the searches for instances of these meta concepts follow. To make matters more difficult, an important function of the database schema, or more precisely, of system-wide class specifications, is missing: the management of unified indicators (class names, attributes and operations). In order to prevent a rapid growth of indicators, we work on a dictionary and rules for data registration (see 6).

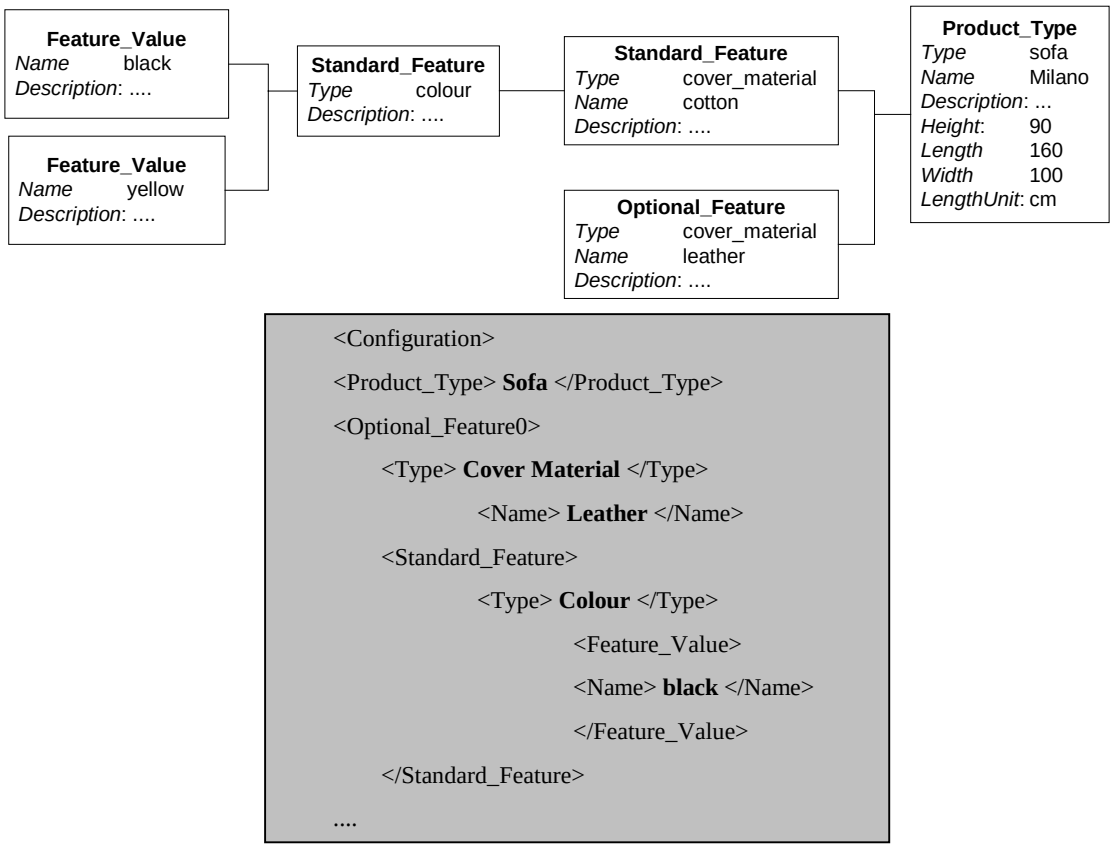

Figure 4: Exemplary instancing of the model with corresponding specifications of a particular configuration using XML

It certainly cannot be expected of customers to use the abstractions deployed in the meta model. But this really poses no severe problem. In the end, the objects that together represent the description of a product type can be mapped to a dynamically generated user-interface. For instance, with the example in fig. 4, a corresponding user-interface would include fields labeled as 'Cover Material:', 'Color' etc. However, to those that register new product types, the structure of the model cannot be completely concealed. While the recursive relationships between Optional_Feature and Feature can be left out of account, the registration of product features will not be possible in all cases by simply accumulating features. Those that register new, more complex product types must bring with them a considerable abstractions ability. They must decide which features are appropriate for the decomposition of a product on each different level. In the example given in Figure 4, it has yet to be decided as to whether differentiation should be made first on the basis of color then material, or the other way round.

ADDITIONAL CONCEPTS

The model shown in Figure 3 is not sufficient to represent all product-related data. That goes for, amongst others, listed prices, stocks and packaging. In order to be able to take different forms of pricing into account, it is not sensible to relate selling prices directly to the product classes. Instead it is recommended that prices, or more precisely the pricing mechanisms (as in auctions) be described in associated classes. The management of stocks also recommends further abstractions, since various forms are conceivable: the management of a single physical stock-room, recourse to an external stock-room managed by a logistical partner or contracts in which the suppliers guarantee full delivery within a certain timeframe. Through this the contracts fulfill the function of a virtual stock-room.

In contrast to traditional trading, delivery to the customer is an obligatory part of the order processing. Therefore data that is needed for the completion of corresponding logistics processes has great significance. This includes weight and measures of the packaging, for example. Figure 5 shows an extension of the model concerning the description of packaging units. It corresponds to a great extent to the well-known "composite pattern". The cardinality 1,1 that is assigned to the class Package in the association with Aggregated_Package is intended. The packages contained in a collection packet are not represented as an object each, since the identification of packages in the real world would be too costly (see above). Therefore only the type of

packaging will be referenced, the respective number of which will be managed through the attribute no_of_items.

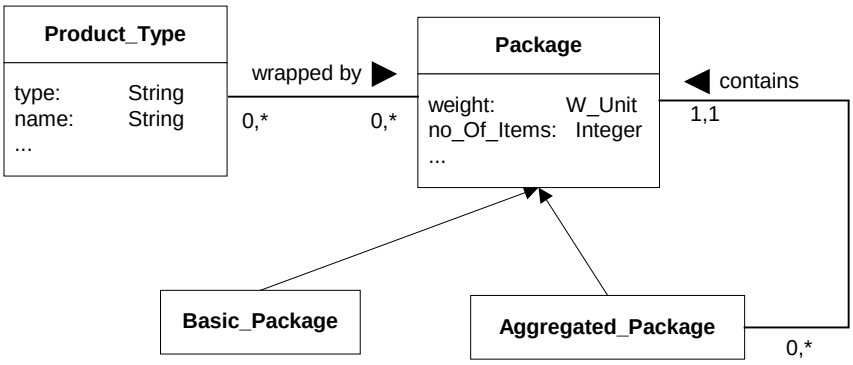

Figure 5: Expansion of the model for packaging

If it is necessary to manage single instances of the modeled product type, the classes Product_Type, Configuration and Product_Variant have to be associated with a respective class whose instances take over this function.

5. Related Work

With respect to their central significance for electronic trade, it is hardly surprising that product descriptions are the subject or a series of specialist research activities. Some of the companies running trading sites are currently pursuing ambitious approaches, too. This concerns above all three domains not entirely free of overlaps: the categorization of products, the registration of product catalogues and the exchange formats based on them, as well as the development of ontologies. The formation of hierarchically-ordered product categories is not directed at a description of product types by means of conceptual features. Instead, categories are defined extensionally through listing the ordered products. Dictionaries of this kind that should above all support the search for products are therefore comparable to the keywords in a library. An example for a categorical system of this kind is the index managed by the UN [UNSP99] that is partly shown in fig. 6. It corresponds to the common use of the term "Meta" in the Internet that indexes of this type are also described as meta-data [MiFe99]: "Meta-data are defined lists of keywords describing features of available information." ([StSc00a], p. 705) It does not need to be emphasized that this is a clearly different understanding of the term from that which is the basis for the approach proposed here.

-family-[5211] Accommodation furniture
  -class-[521115] Furniture
    -commodity-[52111501] Sofas
    -commodity-[52111504] Entertainment centers
      ....
[49] Musical Instruments, Recreational Equipment, Supplies and Accessories
[51] Drugs and Pharmaceutical Products
[52] Furniture, Furnishings, Domestic Appliances and Consumer Electronic Products
[53] Apparel, Luggage, and Personal Toiletry Products

Figure 6: Extract from the UN's product categories

Product catalogues (the IEPCs mentioned already) aim to standardize product descriptions in order to be able to deploy standard tools to support the exchange of product related data. In this respect they are comparable with the approach introduced here, as they define a structure for the description of products. There is a range of companies and consortia that try to establish corresponding structures as standards. Amongst others in this series are the "Common Business Library" (xCBL) [Comm99] or cXML [Arib99]. They both use XML or extensions of XML to define document types, like 'Order Request', 'Order Response' etc. The proposed document type definitions feature an impressive number of specific product details. xCBL, for instance, includes various 'Hazard Packing Codes' that can be assigned to a product ([Comm00], p. 279). While these are examples of generic product features, neither xCBL nor cXML allow adding individual product features. Instead, both approaches are similar to UN-EDIFACT in the sense that they use product references only, assuming the participating parties share a common understanding of these product IDs. Fig. 7 shows an example instance of

the document type ‘Contract’ within cXML. Using item IDs only is certainly not sufficient to provide for powerful search mechanisms.

```
<ItemSegment segmentKey=’Detroit’>
 <ContractItem>
  <ItemID>
     <SupplierPartID>12345</SupplierPartID>
  </ItemID>
  <UnitPrice><Money currency=’USD’>1.50</Money></UnitPrice>
  <Extrinsic name=’URL’> http://www.workchairs.com/CoolProducts
</Extrinsic>
 </ContractItem>
 <ContractItem>
  <ItemID>
   <SupplierPartID>12347</SupplierPartID>
  </ItemID>
  <UnitPrice><Money currency=’USD’>111.50</Money></UnitPrice>
```

Figure 7: Example instance of the document type ‘Contract’ within cXML (www.cXML.org)

In Germany, BMEcat ([HRS99], [HüSc00]) has drawn astounding attention within a short time. BMEcat specifies a few XML-document types for the exchange of product data. It also includes meta concepts (“Article_Features”, “Feature_Group”…). It does not, however, allow for describing product variants or configurations. All these approaches are in fundamental contrast to those discussed in sections 3 and 4, as they do not provide conceptual models that could be used for system development. Conceptual models are ultimately more versatile, since they guide the implementation of systems and the generation of XML document types.

The term “ontology” is not used in a unified way. It seems however to predominate the idea that an ontology contains the formalized specifications of domain concepts. A product ontology would start out with a claim similar to that of the modeling approaches discussed in this article. A product ontology would define core concepts of products in order to establish a generally accepted reference language for dealing with products. An example of aproduct ontology is available on the Stanford Ontolingua Server (http://www.ksl.Stanford.EDU/software/ontolingua/). However, the concepts remain on a superficial level. There are just five features that can be used to specify a product type: Has-Model-Number, Has-Special-Discount, Has-Warranty, List-Price, Net-Weight. There is also an industrial consortium that aims at unified ontologies to guide the design of 'Internet-based electronic marketplaces' (http://www.ontology.org). It remains to be noted that until now no elaborate ontology about products seems to exist.

6. Conclusions and Future Work

The concepts for product modeling presented in this article were created during the development of a reference model for trading platforms on the Internet. The reference model exists currently in the form of an object model with approximately 100 classes as well as numerous models of corresponding business processes. Next to trade with physical products, the reference model also supports trading with financial services. In contrast to physical products, financial services are modeled on the basis of class hierarchies. That is based on the assumption that the variety of product types in financial services is clearly smaller than for physical products. The reference model allows for detailed descriptions of various business models related to e-commerce. Our work on conceptual product modeling is part of the research project ECOMOD (‘Enterprise Modeling for E-Commerce’) that is funded by the German National Science Foundation. The project aims at a method that supports the development and maintenance of high-performance and flexible infrastructures for electronic business. The main focus is on the automation of procurement processes that include the automatic (pre-) selection of suppliers and goods – based on a semantic search for appropriate products and services. With this, both business management and software-engineering concepts must be taken into consideration. Since different views of a company and its environment are to be reproduced, traditional models of software technology are not sufficient. Instead, the method to be developed is based on multi-perspective enterprise

models, which allow one to represent a company from various perspectives (e.g. strategic, organizational, information system, [Fran97]).

Our future work will concentrate on more powerful abstractions of mechanisms to support the negotiation of prices (such as various types of auctions) - and, indeed, of mechanisms to support the negotiation of terms and conditions more generally. We will also define protocols to handle external exceptions that may occur during business processes, e.g. delays or loss of information within logistical processes. In parallel to this, the reference model is being implemented in order to prove the efficiency of the underlying concepts. For this purpose, we use an object-oriented programming language (Java) and a relational database which is hidden behind an object-oriented framework.

REFERENCES

Ariba Inc.: cXML/1.1 Ariba Inc. 1999 (http://www.cmxl.org/home/, 6/16/2000)

Commerce One: Common Business Library, Version 2.0, rel. 3, (http://www.comerceOne.com/xml/, 6/16/2000)

Frank, U.: Enriching Object-Oriented Methods with Domain Specific Knowledge: Outline of a Method for Enterprise Modelling. Arbeitsberichte des Instituts für Wirtschaftsinformatik. Universität Koblenz, No. 4, 1997, (http://www.uni-koblenz/~iwi/, 6/16/2000)

Frank, U.: Applying the MEMO-OML: Guidelines and Examples. Arbeitsberichte des Institut für Wirtschaftsinformatik der Universität Koblenz-Landau. Nr.11, Universität Koblenz-Landau 1999 (http://www.uni-koblenz/~iwi/, 6/16/2000)

Hümpel, C.; Renner, T.; Schmitz, V.: BMECat Specification. Version 1.01 (http://www.bme-cat.org/, 6/16/2000)

Hümpel, C.; Schmitz, V.: BMEcat – An XML Standard for Electronic Product Data Interchange. In: Turowski, K.; Fellner, K. K. (Ed.): XML Meets Business: 1. Deutsche Tagung XML 2000, Tagungsband. 2000, pp. 1-11

Keller, A.; Geneserath, M.: Using Infomaster to Create a Housewares Virtual Catalogs. In: The International Journal of Electronic Commerce and Business Media, Vol. 7, No.4, 1997, pp. 41-44

Milstead, J.; Feldman, S.: Metadata: Cataloging by Any Other Name… In: ONLINE, January 1999, http://www.onlineinc.com/onlinemag/metadata/, 6/16/2000)

OBI Consortium: Open Buying on the Internet. Technical Specifications, Release V2.1.OBI Consortium. (http://www.openbuy.org/obi/specs/, 6/16/2000)

Stanoevska-Slabeva, K.; Schmid, B.: Internet Electronic Product Catalogs: An Approach Beyond Simple Keywords and Multimedia. In: Computer Networks, Vol. 32, No. 6, 2000, pp. 701-715

Stanoevska-Slabeva, K.; Schmid, B.: Cross-Supplier Bundling of Tourist Products with Multi-Vendor Electronic Catalogs. In: Hansen, H. (Ed.): Proceedings of the European Conference on Information Systems 2000 (CD)

Timm, U.J., Rosewitz, M.: Electronic Sales Assistance for Product Configuration. In: Proceedings of the 11th International Bled Electronic Commerce Conference: Electronic Commerce in the Information Society, 1998, pp. 118-125

United Nations Standards Products and Services Classification Code Organization: United Nations Standard Products and Services Classification. United Nations Standards Products and Services Classification Code Organization, 1999 (http://www.spsc.org/browse/, 6/16/2000)