

## Ontologizing B2B Message Specifications: Experiences from Adopting the PLIB Ontology for Commercial Product Data

Joerg Leukel<sup>1</sup>, Martin Hepp<sup>2</sup>, Volker Schmitz<sup>3</sup>, Christoph Tribowski<sup>4</sup>

<sup>1</sup>University of Hohenheim, Stuttgart, Germany

<sup>2</sup>University of Innsbruck, Innsbruck, Austria

<sup>3</sup>University of Duisburg-Essen, Essen, Germany

<sup>4</sup>Humboldt University, Berlin, Germany

joerg.leukel@uni-hohenheim.de, mhepp@computer.org,  
volker.schmitz@icb.uni-due.de, tribowsc@wiwi.hu-berlin.de

### Abstract

*Data about products and services is of paramount importance in most inter-organizational business processes. For business-to-business (B2B) scenarios, a great number of XML-based message specifications are available, which cover various processes and types of transactions. These specifications support the respective data exchange tasks by providing a common representation for products and services in the form of syntactical conventions with some light-weight formal semantics. However, an analysis of the underlying models shows that technical product data and commercial product data are being represented in a fundamentally different manner. In particular show the commercial models both a higher syntactical complexity and more semantic heterogeneity. In this paper, we propose to align the representation of technical and commercial views on product data in message specifications. Our approach is based on the PLIB ontology, which was originally described in ISO 13584. We have evaluated our proposal by applying it to an industrial message specification for electronic catalogs, and can show that the novel approach reduces representational mismatches in B2B processes and simplifies systems integration in such scenarios.*

### 1. Introduction

In B2B scenarios, integrating processes, data models, and content is at the core of many implementation projects as well as fields of research [1]. In the past years, ontologies as the backbone of the Semantic Web have attracted a lot of interest with regard to solving such integration problems, because

they may help systems in mediating between representational and conceptual mismatches automatically or semi-automatically.

Current B2B integration technology, however, is still based on message-oriented protocols and XML-based message specifications [2]. Even though these specifications can be considered light-weight ontologies, because they reflect some degree of consensus over the elements and their relations in a domain, the amount of formal semantics is still too limited for empowering machines to mediate mechanically between different specifications. This observation is particularly true for the product data domain, and thus for the exchange of product data in various business applications. Much effort and resources have been dedicated to the development of specifications (sometimes incorrectly referred to as “standards”). A recent survey funded by the European Union [3], for instance, has identified more than 25 competing specifications for electronic catalogs, of 16 are XML-based. The focus on XML respectively XSDL as the formal language causes several problems with regard to semantic interoperability [4] [5]. When analyzing these specifications, it becomes evident that technical product data on one hand and commercial product data on the other hand are being represented in a fundamentally different manner.

#### 1.1 Observations

We can observe the following major problems that contribute to an unsatisfying status quo:

**(1) Complexity:** B2B message specifications tend to become very complex in terms of the number of data elements and inter-relationships. The reason is that the extensibility of XML allows defining sophisticated

data models simply by building hierarchical structures. Creating and processing, and converting such structured XML documents requires a high amount of resources. This is most evident when it comes to establishing and maintaining mappings between two competing specifications.

(2) Unclear semantics: In general, each specification defines its own data model and terminology for a similar set of requirements originating from the same domain. Due to limited semantic expressiveness of XSDL, all specifications provide an often extensive documentation describing the meaning of elements and XML attributes in natural language. The aforementioned survey reported on many poor examples of documentations with ambiguous descriptions, missing information, and inconsistencies with the formal specification [3].

(3) Competing specifications: The existing number of specifications shows that neither specification can be regarded as a dominating de-facto standard. Again, the extensibility makes XML a universal tool; hence various organizations, consortia, and companies have developed respective specifications. There are two major consequences: first, competition between specifications limits potential network effects of standardization; second, the high diversity of current specifications has not yet been reduced by reusing reference models, relevant ISO standards, nor other best-of-breed solutions for specific purposes.

(4) Lack of a common product model: In any popular message specification for product data in B2B e-commerce, there exists a clear distinction between technical and commercial product data, with very concise technical product models and a great variety of highly complex commercial product models. This observation is quite surprising, since one could assume that both areas show similar characteristics, which would form the basis for an integrated view.

It becomes evident that current product data exchange suffers from limitations of XML and XSDL modeling as well as from the artificial divide between technical and commercial product data. We aim at solving the described problems by ontologizing respective specifications.

## 1.2 Related work

Related work to ours can be found in (1) ontology engineering, (2) product data management, and (3) B2B e-commerce research in general.

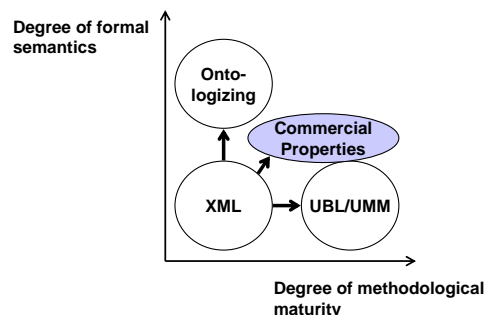
Early with the emergence of XML as a universal language for data exchange on the Web, general criticisms on XML and XML schema languages has been expressed with regard to its modeling paradigms,

unclear semantics of concepts such as the hierarchy relation, and new integration problems due to its extensibility [4] [5].

There exists a wealth of research and initiatives on how to overcome limitations and potential dangers of XML by improving the quality of the process leading to an XML message specification. For instance, the Universal Business Language (UBL) is being based on so called “core components”; these are modeling guidelines to reduce heterogeneity in naming, nesting, and data typing. It is complemented by the UN/CEFACT Modeling Methodology (UMM), which adopts concepts from object-orientation and business process modeling [6].

The Semantic Web postulates to describe information resources semantically, thus to enable machine-readable semantics, unambiguous interpretation of formally defined concepts, and reasoning on distributed knowledge bases. Since ontologies form the backbone of this vision, “ontologizing” is the process of describing something semantically with the aforementioned intentions. For instance, the work by Foxvog and Bussler [7] aimed at ontologizing the ANSI X12 EDI format, which consists of more than 1,400 data elements. They employed four different ontology languages. Ontologizing, or in a more general wording, lifting the semantic level of B2B message specifications eases, in general, content integration [8]. For product data integration, Omelayenko and Fensel developed a two-layered integration approach by separating syntax layer, data model layer, and ontology layer [9]. Hepp proposed a methodology for deriving OWL ontologies from product classification schemes, such as eCl@ss [10].

Reviewing related work, we can identify two distinctive paths of research: (1) improving model quality and (2) ontologizing specifications. The respective two dimensions (degree of methodological maturity, degree of formal semantics) span the matrix as shown in figure 1.



**Figure 1: Approaches to advanced B2B message specifications**

We position our approach on the right and half up. The first step of our way of ontologizing is to make the semantics explicit, thus reconstructing the semantics hidden in XML tree hierarchies and related documentations. In that sense, we aim at providing a methodology for restructuring B2B message specifications. We still reside, however, in the XML world; hence we will not reach a higher degree of formal semantics, e.g., first-order logic. Since our approach addresses the described integration problems in the product data domain, we call the results “commercial properties”, which resolve the dividing line between technical product data (therefore properties) and commercial product data.

### 1.3 Our contribution

The contribution of our paper is as follows: (1) We motivate and present an approach to ontologizing B2B message specifications for the domain of product data, (2) make this approach operational by developing a methodology for XML schemas, and (3) show the proof of concept by applying it to an industrial message specification. Hence we can (4) clearly demonstrate that the approach results in both a significant reduction of the complexity of message specifications and a useful augmentation in formal semantics as provided by the PLIB ontology.

The remainder of our paper is structured as follows: In section 2, we give a survey of technical and commercial product data with regard to formal representations and available standards. In section 3, we propose to base all product data on the PLIB ontology. Eventually, we define in section 4 a methodology for ontologizing respectively restructuring XML schemas that address commercial product data. In section 5, we report on experiences with applying our approach to a message specification for electronic product catalogs. In section 6, we draw conclusions and point to future research issues.

## 2. Technical vs. commercial product data

In this section, we give a summary of the field of product data and its usage in B2B scenarios. We distinguish technical and commercial product data, both being essential for most business applications.

### 2.1 Technical product data

Definition: Technical product data is defined as all data that describes technical, thus functional and/or physical characteristics of a product. Generally,

technical product data is static and not subject of personal, mental or organizational intensions of the viewer. In other words, this data is not determined by its context. For instance, it concerns dimensions, material, design, shape, and other features (e.g., physical, electrical, and biological).

Information systems: Technical product data resides mainly in IS for product data management (PDM), engineering management (EM), and more recently, product life-cycle management (PLM). Very often, some basic technical product data is also subject of enterprise resource planning systems (ERP), thus this data needs to be integrated with PDM systems.

Standards: For technical product data, the STEP family of standards (ISO 10303) is of essential importance as it provides a comprehensive set of models for representing product data [11]. In terms of formalization, STEP relies on EXPRESS (ISO 10303-11), a modeling language derived from an extended ER model including additional expressiveness (i.e., constraints, data types). For the course of our paper, we look only at the complementary ISO 1584 standard. Other parts of STEP address bills-of-material and geometric model data, being of interest primarily for engineering and production planning, thus the early phases of the product life-cycle. For B2B e-commerce, this data is less relevant. ISO 1584 contains a conceptual model for product data that can be captured by properties and respective values [12].

Semantics: ISO 1584 (parts library, PLIB) aims at grounding product data on unambiguous, semantically well-defined, globally unique properties being accepted by specific branches of industry [11]. This aim is being targeted by (1) a set of strict attributes as well as semantic relationships for defining each property in a common manner, (2) global unique identifiers (GUID) for each defined property, and (3) a standardization process for reaching domain consensus. Once defined, PLIB-compliant properties should be made accessible by so called dictionaries. A dictionary contains definitions of product categories and respective properties.

### 2.2 Commercial product data

Definition: Commercial product data is defined as all data that describes commercial non-functional characteristics of a product. Generally, this data is subject to intensions of the company that manufactures, supplies or purchases the product (i.e., business goals, product strategies). It can be subdivided in data for identification, pricing, ordering, logistics, etc. In contrast to technical product data, it is

on the instance level often dynamic (i.e., prices) and context-dependent (i.e., customer-specific).

**Information systems:** Commercial product data resides in ERP systems, which provide comprehensive models and process definitions for materials management, with an emphasis on the late stages of the product life-cycle (i.e., distribution).

**Standards:** A great number of XML message specifications are at hand, though few can be regarded as de-facto standards, if any. ISO standards are still missing. Very often, ERP systems and conventional EDI formats served as an inspiration for developing these specifications. In our previous works [13] [14], we have analyzed the models for commercial product data. The results pointed to a great diversity both in covered requirements and resulting model structures. In general, XML documents get deep and complex due to extensive usage of nesting, intermediate container elements, and often proprietary data typing.

**Semantics:** Due to the usage of XML and XSDL, the semantics of commercial product data is interwoven with the document syntax, thus it is mainly expressed by element names and nesting and remains highly informal. It is thus inaccessible to machines. A lot of semantics is contained in the supplementing documentations, which are neither complete nor consistent and thus again not machine-readable.

### 3. Adopting the PLIB ontology

In this section, we propose to base all product data on a single conceptual model, namely on the PLIB ontology. PLIB ontology is a recently emerging term for ISO 13584, reflecting that it can serve as a domain ontology [15]. The business-wise motivation for our proposal is that product data is being created, used, maintained, and exchanged in various business processes that are all part of product data management. Therefore, integrating product data, specifically bridging technical and commercial product data on the conceptual or semantic level can be expected to result in a significant reduction or even elimination of integration costs that are caused in today's IS.

#### 3.1 Semantic foundations

ISO 13584 consists of several parts. For our purpose, mainly part 42 describing the conceptual model is of interest. ISO 13584 also contains instances of this model, thus property definitions for specific branches of industry (part 501: measuring instruments; part 511: mechanical systems and components for general use). In that sense, PLIB fulfills basic

requirements on an ontology, though it is not described in any of the popular ontology languages for the Semantic Web.

We have already seen that PLIB requires expressing all technical product data as property-value pairs with the properties being defined according to the PLIB model. In order to apply this model to commercial product data, we have to ask whether the model is able to represent commercial product data as well.

There are three different types of properties in PLIB: non-dependent properties, dependent properties, and conditions. For instance, the color of a button is not dependent on other properties, whereas the revolutions per minute of an electrical engine depend on the input voltage. The latter is the condition under which the dependent property is being defined. Additionally to the 'depends\_on' relation, it is possible to give a formula defining the value of the respective property by referring to other properties (e.g.,  $\text{area}=\text{width}*\text{length}$ ). This model has been developed for capturing physical characteristics and inter-relations, and looks rather simple; however, it can easily be adopted for commercial product data. The price of a product, for instance, may depend on a set of multiple factors, such as time and order quantity [13]. If we regard all these factors as PLIB-compliant properties and establish the respective 'depends\_on' relationships, then we can express the same information in another way, which is fundamentally different from existing, nested XML document structures. It has to be noted that the resulting XML instances are nearly flat, not deeply nested. The reason is that PLIB allows only one single relation. Its interpretation is as follows: if a property is defined as dependent from one or many other properties, then the instantiation of this property requires to instantiate the conditions also. In the pricing example, the price amount has always to be augmented with instances, thus concrete time and order quantity intervals.

#### 3.2 Organizational foundations

Since the PLIB ontology is only the semantic backbone of ISO 13584, we investigate additionally its organizational foundations. This approach is motivated by the idea of unambiguous, semantically well-defined, globally unique properties itself. This idea requires establishing procedures for finding consensus on these properties as well as infrastructures being capable of fulfilling these needs. We have to repeat the concept of PLIB dictionaries here, which are collections of PLIB-compliant properties (and product categories). The tern dictionary often comprises not only the items, but also an underlying information

system giving access to the items as well as automating the standardization process.

One could object that raising the semantic level in our approach does only move the standardization, and therefore the integration problem to another level (organizational vs. syntactical level). However, there are three arguments for the contrary. First, any standardization work takes place on the organizational level, with interactions between interested parties. Here we just streamline this work by adhering to a unified, globally accepted conceptual model, thus we prevent the creation of highly customized, proprietary XML document structures from the beginning. Second, there exist several reference implementations of PLIB-compliant IS that manage standardization processes and grant access to the dictionary items (e.g., [17]). Third, the number of potential commercial properties is by nature limited and much smaller than those of technical properties. Whereas new technical properties result from innovation in products and services, the commercial domain is quite stable in the number of properties and static in its structure (while highly dynamic on the instance level!). A good example are ERP systems that incorporate comprehensive models for the respective areas (e.g., for pricing, ordering, and logistics). Specifically, ERP models for materials management are at the core business functions and do not change over years. Therefore, it is feasible to expect that standardization of commercial properties should be possible on a quite abstract level in terms of industry coverage.

## 4. Methodology

In this section, we define a methodology for ontologizing respectively restructuring XML schemas that address commercial product data. It shows the conversion of XML schemas of current B2B message specifications into those that adopt the PLIB ontology. Candidates for properties are all elements and XML attributes in the respective commercial product models.

### 4.1 Elements

Elements can be directly converted into non-dependent properties, if the element is not part of a sequence with a multiplicity greater than 1. For instance, the following schema defines such an element for the EAN product identifier (often part of the basic product model, therefore not deeply nested):

```
<xsd:element name="EAN">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:maxLength value="14"/>
      <xsd:minLength value="1"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

```
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
```

We extract the following PLIB-relevant information: property name equals element name; the property is mandatory and univalent; data type is being mapped to 'string\_type' with the restriction mapped to PLIB's 'value format' attribute (cf. section 4.4). The essential 'definition' attribute can not be filled automatically, because this information is not part of the schema (i.e., human-language description of the meaning). The compulsory version information can be derived from the schema, e.g., date of the XSD file. If the element has cardinality greater than 1, we allow multiple values for the respective property as well.

### 4.2 Elements with attributes

In XML, attributes can be attached to elements in order to provide additional information on the element and to specify the element content similarly to a type, e.g., the product ID element has an attribute that says whether the ID is that of the supplier, buyer, or a third party:

```
<xsd:element name="Supplier_PID">
  <xsd:complexType>
    <xsd:simpleContent>
      <xsd:extension base="xsd:string">
        <xsd:attribute name="type" use="required">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:minLength value="1"/>
              <xsd:maxLength value="50"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:attribute>
      </xsd:extension>
    </xsd:simpleContent>
  </xsd:complexType>
</xsd:element>
```

Such attributes require defining a dependent property for the element, and a condition for the attribute, thus the element value can only be interpreted correctly in conjunction with the attribute value. All other PLIB-relevant information on the attribute-derived property can be extracted respectively needs to be added manually in the same way as for elements.

### 4.3 Sequence of elements

The previously described rules concern only a very small part of actual XML schemas, since they do not cover nesting as it is realized by building sequences of elements that again can incorporate sequences.

At first sight, a sequence of cardinality 1 could be interpreted as a logical group of its included elements. Therefore, the sequence would be used to group related information only, but all included elements are

independently from each other; hence the respective properties are of that kind, too. However, often semantic relationships exist between these elements, though they can not be extracted on a formal basis. The reason is that this information is not part of the schema, but contained in the supplementing documentation or coded in the element names. For instance, let `OrderInfo` be a sequence of three elements: `OrderUnit` (the unit of measurement for ordering, e.g., box), `ContentUnit` (the unit of measurement of the product itself, e.g., bottle) and `ProductID` (ID of the base product delivered in the bottle). In this case, the latter element must not be confused with a regular product identifier, since its interpretation needs to consider the context of `ContentUnit`. Therefore, `ProductID` serves as a condition for the dependent property `ContentUnit`. All these relationships need to be added manually to the PLIB-compliant property definitions by carefully searching for respective descriptions or names.

Sequences with cardinality greater than 1 require that all but one of its sub-elements are converted into dependent properties with making the left out one the condition for all other properties. Otherwise it would be impossible to relate property values to each other. The following example demonstrates this rule.

```
<xsd:element name="Transport" maxOccurs="unbounded">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="Location"/>
      <xsd:element ref="TransportRemark"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

For each product, multiple transportation information can be specified in the `Transport` element, which includes the location of delivery and a remark. The interpretation is that for each location only one remark can be specified, thus `Location` can be regarded as unique in the context of each `Transport` element. Therefore, we define a multivalent property for `TransportRemark`, which is dependent on the respective (multivalent) `Location` property. The same applies to sequences where two or more elements are unique; hence the respective properties represent the condition for all other sub-elements. It has to be stressed that choosing the right conditional properties can not be automated, since it depends on the semantics of the sequence, which is not formally described in current message specifications.

#### 4.4 Nested sequences

A nested sequence consists not only of elements, but includes at least one other sequence, thus another

sequence of elements. In this case, converting the latter sequence must consider whether some or all of its elements are dependent on elements respectively derived properties of the former sequence.

#### 4.5 Identification and naming rules

Each PLIB property requires a GUID ('code' attribute) and name ('preferred name' attribute). This information can be extracted from the XML schema as follows: In PLIB, the GUID is defined as a 3-tuple: organization, dictionary item, and version. We choose an ID for the organization that developed the schema, define a range of consecutively numbered IDs for the dictionary items (properties), and add a version ('1.0' for the initial ontologizing process).

XML element names can be mapped to the property name directly, if element names are unique in the schema. When parsing the XML schema, the same element may appear more than once (being used in multiple contexts). Then we do not define a new property, but set the cardinality of the already existing property to multiple (if it is 1). However, this procedure depends on the actual schema. If element names are not unique, we have to use the full path instead of the name only.

XML attribute names are often generic and seldom unique (e.g., 'type', 'version', 'code'). Therefore, we build the property name by adding the respective element name (e.g., 'Supplier\_PID/type').

#### 4.6 Data typing and domains

The PLIB ontology contains a comprehensive system of data types for restricting property values. This system includes in total 23 data types that are arranged in a type hierarchy. Since this hierarchy is differently from the XSDL type system, we defined a mapping to enable automated conversion.

In addition to predefined types of the XSD namespace, current B2B message specifications make extensive usage of restricting standard types and defining customized types, especially enumerative types (i.e., for country codes, currencies, languages, price types, product categories, etc.). Therefore, we transformed those domain definitions into PLIB-compliant ones. It has to be noted that the expressiveness of the PLIB type model is lower, thus not all type-related information can be mapped to PLIB. This otherwise loosed information could be included in the generic PLIB 'remark' and 'note' attributes, though these attributes have no formal semantics.

## 5. Ontologizing BMEcat 2005

In this section, we report on experiences with applying our approach to the message specification BMEcat 2005 for electronic catalogs. We choose BMEcat [16], because it is highly relevant in Europe and at the same time a quite comprehensive specification, e.g., in terms of the number of elements and the overall structural complexity. The process of ontologizing is as follows: First, we define commercial properties by extracting relevant information from the current BMEcat schema. Second, we restructure this schema by removing all commercial product models. In addition, we implemented a converter that transforms existing BMEcat instance documents into those that contain only property-value pairs.

### 5.1 Property extraction

The BMEcat schema provides a comprehensive model for commercial product data. Defining PLIB-compliant properties for this model requires applying the methodology described in section 4 on the respective part of the XSD schema. Therefore, the property extraction tool works directly on the schema. However, not all relevant schema information is formally described in the XSD file, but is contained in the supplementing documentation (e.g., integrity constraints and implementation guidance). The documentation has more than 700 pages. For instance, several elements possess a unit of measurement, though the unit is not explicitly stated in the schema (e.g., DELIVERY\_TIME element with unit 'work days'). This information has to be added to the property definition manually (PLIB's 'unit' attribute).

The property extraction results in a total number of 126 commercial properties (see table 1).

**Table 1. Results of property extraction.**

Data Area	BMEcat model	# of properties	# of depends on relations
General	PRODUCT	3	4
ID, description	DETAILS	38	14
Ordering	ORDER_DETAILS	10	12
Pricing	PRICE_DETAILS	28	31
Logistics	LOGISTIC_DETAILS	22	15
Multimedia	MIME_INFO	7	6
Relationships	REFERENCE	6	12
Contact info.	CONTACTS	2	4
Remote info.	IPP_DETAILS	6	5
	<b>Total:</b>	126	103

### 5.2 Schema restructuring

Once we have defined the set of commercial properties, all respective commercial product models

become obsolete; hence we can remove them from the schema. However, we have to check whether the BMEcat model for property-value pairs is compatible with PLIB properties. This is not the case, since it lacks the 'depends\_on' relationships, thus each property value is purely independent from others. Therefore, we extend the respective FEATURES model by adding new elements for dependent properties and conditions. The results of restructuring the BMEcat 2005 schema are presented in table 2.

**Table 2. Results of schema restructuring.**

	Original Version	Ontologized Version	Change
No. of elements	224	26	-88.4%
No. of simple XSD types	34	6	-82.4%
No. of complex XSD types	12	3	-75.0%
XSD: no. of characters	107,340	20,762	-80.7%

### 5.3 Instance document conversion

Real-world BMEcat catalogs need to be converted in correspondence with the restructured schema. We implemented a respective converter; the resulting documents contain only property-value pairs and conditions, hence the documents are rather flat than deeply nested. In our test scenario, we converted an industrial catalog of 338 products. The maximum tree depth of the relevant document part was reduced from 4 for the BMEcat 2005 document to only 2 for the new instance based on 20 actually required commercial properties (table 3). This reduction, however, does not reflect the fundamentally different document structure, because the catalog did not use the more complex data areas to the full degree (i.e., price information). The growth in number of lines results from flattening the structure, while the growth in file size can partly be attributed to long element names in the BMEcat model.

**Table 3. Results of document conversion.**

	Original Version	Ontologized Version	Change
Tree depth	4	2	-50.0%
No. of lines	17,712	59,267	+234.6%
file size [KB]	739	1,816	+145.7%
compressed file size [KB]	34.3	66.2	+93.0%

## 6. Conclusions

We aimed at solving major problems in the product data domain by ontologizing respective message exchange specifications. We evaluated our proposal by applying it to an industrial message specification for electronic catalogs. In particular, our approach addressed the following issues: We showed that adopting the PLIB ontology (1) reduces the complexity of message specifications greatly, (2) makes semantics

contained in deeply nested XML schemes explicit due to the PLIB property model, (3) helps to yield harmonized message specifications being based on a common set of commercial properties, and (4) integrates technical and commercial product data on the same conceptual model.

The PLIB ontology calls for representing all product-related information as properties and respective values. Our experiences from ontologizing the BMEcat specification indicate that this procedure is possible and results in the described improvements. However, adopting the PLIB ontology is a demanding task, since it requires making semantics explicit, which is informally described in current specifications. This task can be supported by software tools that implement the methodology and assist the expert in choosing the right properties, dependencies, and conditions. In addition, PLIB properties need to be defined in a standardized form, though not all required information is available in current specifications and its supplementing documentations. Thus, this task of lifting the semantic level requires manual intervention.

The main limitation of our approach is that we still reside in the XML world, thus we aimed at restructuring XML schemes and adopted an ontology that is not formally described in any of the current ontology languages for the Semantic Web. The motivation for this is twofold. First, there are still significant performance and scalability constraints in current ontology infrastructure. Repositories and reasoners for OWL-DL, for example, do still not scale well enough for handling the data volume of real-world business transactions, as has been reported in [10]. There is still competition among multiple ontology language proposals, in particular with regard to computationally less expensive subsets of OWL, e.g., OWL-DLP. Ontologizing BMEcat into OWL-DL might, at this stage, create an ontologized version that cannot be used in real-world scenarios due to the high reasoning costs of OWL-DL, while the migration to a future ontology language might render the manual work of ontologizing obsolete.

Second, we think it is crucial to prove first that aligning the two perspectives of a technical and a commercial view on product data, based on the semantics of PLIB, and yielding one harmonized conceptual model is by itself advantageous. This approach has also the benefit that our findings are of direct practical relevance. They can be used for operational systems on current technology, while preparing the ground for lifting the new conceptual model to a future version to be based on more expressive ontology formalisms.

## References

- [1] M. Stonebraker and J. M. Hellerstein, "Content Integration for E-Business", *Proc. of ACM SIGMOD 2001*, Santa Barbara, California, USA, 2001, pp. 552-560.
- [2] C. Bussler, "B2B Protocol Standards and their Role in Semantic B2B Integration Engines", *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, vol. 24, 2001, pp. 3-11.
- [3] *CEN Workshop Agreement 15045:2004 E - Multilingual catalogue strategies for eCommerce and eBusiness*, Brussels, Belgium, July 2004.
- [4] S.E. Madnick, *The Misguided Silver Bullet: What XML Will and Will NOT Do to Help Information Integration*, Paper #111, Center for eBusiness@MIT, MIT Sloan School of Management, Cambridge, Massachusetts, USA 2001.
- [5] D. Fensel, *Ontologies*. Springer, Berlin et al., 2004.
- [6] Hofreiter, C. Huemer, and Naujok, K. "UN/CEFACT's Business Collaboration Framework", *Proc. of MKW104*, Essen, Germany, 2004, pp. 29-43.
- [7] D. Foxvog and C. Bussler, "Ontologizing EDI: First Steps and Initial Experience", *Proc. of DEEC 2005*, Tokyo, Japan, 2005, pp. 49-58.
- [8] D. Fensel, D.L. McGuinness, E. Schulten, W.K. Ng, E.-P. Lim, and G. Yan, "Ontologies and Electronic Commerce", *IEEE Intelligent Systems*, vol. 16, 2001, pp. 8-14.
- [9] B. Omelayenko and D. Fensel, "A Two-Layered Integration Approach for Product Information in B2B E-commerce", *Proc. of EC-Web 2001*, Munich, Germany, 2001, pp. 226-239.
- [10] M. Hepp, "Products and Services Ontologies: A Methodology for Deriving OWL Ontologies from Industrial Categorization Standards", *Int. J. on Semantic Web & Information Systems*, vol. 2., 2006, pp. 72-99.
- [11] *ISO 10303-1:1994 Product data representation and exchange - Part 1: Overview and fundamental principles*, Geneva, Switzerland, 1994.
- [12] *ISO 13584-1:2001 Parts library - Part 1: Overview and fundamental principles*, Geneva, Switzerland, 2001.
- [13] O. Kelkar, J. Leukel, and V. Schmitz, "Price Modeling in Standards for Electronic Product Catalogs Based on XML", *Proc. of WWW2002*, Honolulu, Hawaii, USA, 2002, pp. 366-375.
- [14] V. Schmitz and J. Leukel, "Findings and Recommendations from a Pan-European Research Project: Comparative Analysis of E-Catalog Standards", *Int. J. of IT Standards and Standardization Research*, vol. 3, 2005, pp. 51-65.
- [15] J. Leukel, "Standardization of Product Ontologies in B2B Relationships - On the Role of ISO 13584", *Proc. of AMCIS 2004*, New York, USA, 2004, pp. 4084-4091.
- [16] <http://www.bmecat.org>.
- [17] N. Ondracek and S. Sander, "Concepts and benefits of the German ISO 13584-compliant online dictionary www.DINsml.net", *Proc. of CE 2003*, Madeira, Portugal, 2003, pp. 255-262.