Semi-Automatic Semantic Lifting of XML to a Target Ontology

Christian Knauer, David Urbansky
TU Dresden
01062 Dresden
christian.knauer, david.urbansky@tu-dresden.de

Johannes Meinecke
SAP Research Center
Dresden
01069 Dresden
johannes.meinecke@sap.com

Daniel Schuster, Philipp Katz, Alexander Schill
TU Dresden
01062 Dresden
daniel.schuster, philipp.katz,alexander.schill@tu-dresden.de

ABSTRACT
This paper shows an approach for a semi-automatic transformation of existing XML data or schema to a given target ontology. We developed methods to automatically map complex types from XML to OWL concepts, XML attributes to OWL properties and relations. Due to a lack of semantics in XML, the transformation cannot be completely automated and we integrate the automatic tools in an editor where the domain expert can modify the transformations. The tool has been evaluated on real industry data from the domain of service engineering within the Aletheia [11] research project.

General Terms
Ontologies

Keywords
XML, OWL, semantics

1. INTRODUCTION
The extensible markup language (XML) has become the de facto standard as a B2B exchange format [2]. However, XML does not capture any formally described semantics. Such a lack of semantics prohibits the integration of data from different sources. Nowadays many companies pursue the goal of combining data from different heterogeneous sources such as databases, document management systems, and legacy applications automatically.

Usually, data from diverse sources differ from each other in their syntactical description. For an automatic integration, the semantics of the data is indispensable. Using ontologies with their formal conceptual model solves the described issue and enables software to understand, share, and reason about the data. Hence, it is necessary to lift pure syntactical described data up to a semantic level.

The main contribution of this paper is the extension of known xml lifting and matching approaches by an automatic matching algorithm. We show that the automatic matching suggestions works reliable and helps the human expert to transform his documents faster than with manual matching only.

In the next section we review and compare related work on lifting XML files to more semantic formats. The subsequent sections will then explain our goal of an semi-automatic lifting approach and our concept. In the last sections we give an insight in the implementation and evaluate our approach on two independent scenarios.

2. RELATED WORK
Several strategies for lifting have been proposed in the literature. In [6] XML documents are translated into RDF graphs and XML Schemas are lifted to OWL ontologies. However, both translation approaches are independent and the generated instances may not respect the OWL model created from the XML Schema.

These issues have been solved in [1]. Bohring proposes a strategy of how OWL ontologies may be generated automatically out of existing XML data. For that purpose, he established suitable mappings between the different data model elements of XML and OWL. In his paper, he presents a framework, which converts XML Schema to an OWL model. Based on this step a suitable stylesheet for converting XML data at the instance level is created. Nevertheless the described approach in [1] does not consider the already existing target ontology. Similar to the work of [6], a new ontology is created.

Reif et al. [5] present in their paper an approach considering an already existing ontology. They introduce a technique to extend existing Web engineering techniques to develop semantically tagged web applications. Thereby, the mapping between XML and the ontology have to be created manually on the design level. Based on the mapping an automatic XML lifting is executed.

In [10] a framework has been developed, which resembles the described one in [1]. However, as distinguished from [1], the mapping proceeds by using an already existing OWL ontol-

	manual mapping	automatic mapping	tool supported	automatic transformation	generic mapping	existing ontology	entire process
[FZT04]		X		X	X		
[BA05]		X		X	X		
TUD	X			X		X	X
[RJG04]	X			X	X	X	X
[ADMR05]	X	X	X		X	X	
[RRC06]	X		X	X	X	X	X
Goal	X	X	X	X	X	X	X

Figure 1: Overview of related work.

ogy. Furthermore, [10] also aimed at providing tool support for such a user controlled mapping. Thus, a mapping tool is presented in [10], which enables a manual, user controlled mapping between XML schema and the OWL target ontology. Afterwards, it enables the instance data transformation.

Figure 1 shows an overview of the presented related work in this field. We compared the research efforts along seven dimensions. The Figure shows that [10] already meets all but one of our requirements. We therefore build upon the work of [10] and add automatic mapping algorithms. Rather than a manual mapping, we focus on a semi-automatic mapping, based on the framework, which was developed in [10]. Especially, we tried to extract semantic information out of XML documents and their XML schema in an automatic manner in conjunction with reviews by the user.

3. SEMI-AUTOMATIC MAPPING

Reaching the goal to create a concept for lifting purely syntactically described data up to a semantic level in a semi-automatic way, we conducted an XML structure analysis.

Due to the fact that XML enables only structuring data without formal semantics, several different XML documents may be created for one use case with the same meaning. Because the semantics of an XML document is implicit and is not based on a semantic data model, a new approach for recognizing the informal semantics within XML and mapping rules to OWL are necessary. Within our investigation, comparing to [1], we focused on the three constructs of an OWL ontology: *Class*, *Datatype Property*, and *Object Property*. Hence, we can employ the following XML constructs as possible mapping candidates for the three OWL constructs.

XML elements, which contain other elements or have at least one attribute, are candidates for an OWL Class. However, simple XML elements, i.e., elements without child elements, and XML attributes are imaginable candidates for an OWL Class mapping as well. Concerning Datatype Properties,

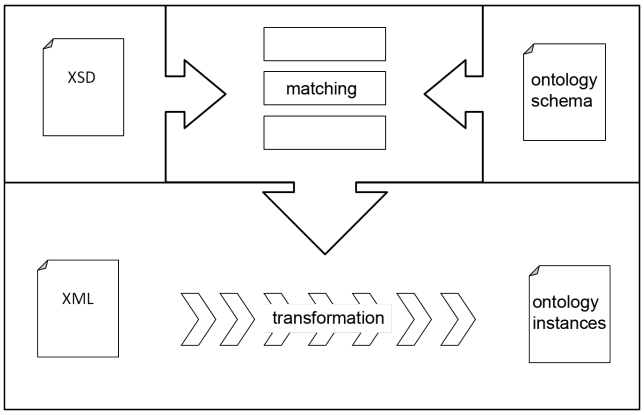

Figure 2: Conceptual approach of XML to OWL mapping.

we can identify simple XML elements and XML attributes as mapping candidates. XML elements with child elements might represent OWL Datatype Properties as well. However, in this case the XML document structures the scenario in more detail than the ontology does. Hence, it needs external knowledge to decide, if the XML element with its nested elements reflects the considered OWL Datatype Property. Already at this stage in our preliminary consideration our assumption for a semi-automatic approach with human interaction was confirmed. Considering the last OWL construct, we notice for our industry scenario two kinds of relations between concepts within an XML document. On the one hand, relations in the XML document between OWL Class candidates are constituted by nesting elements. On the other hand, relations between XML elements could exist by using reference identifiers. Hence, our matching approach has to consider this issue as well.

4. THE SEMI-AUTOMATIC MAPPING APPROACH

Similar to [10], we distinguish between schema mapping and instance transformation as shown in Figure 2. For lifting a special kind of XML document, we therefore firstly create a mapping between an XML schema and an OWL schema. Based on this mapping, every XML document with its data, which belongs to the XML Schema, can automatically be transformed to an OWL document according to the target ontology.

For building up such a mapping we designed a semi-automatic matching process. In the first step both schemas have to be parsed. Afterwards, the matching process tries to find correspondence between the XML and OWL artifacts. For this purpose, according to the XML structure analysis, the matching process is divided into three algorithms: Class Matching, Datatype Property Matching, and Object Property Matching.

Figure 3 shows the sequential execution of these algorithms. All of them operate according to the same principle. Firstly, the candidate retrieval identifies possible candidates for a mapping by a heuristic search, based on the assumptions of the XML structure analysis. By using the set of pos-

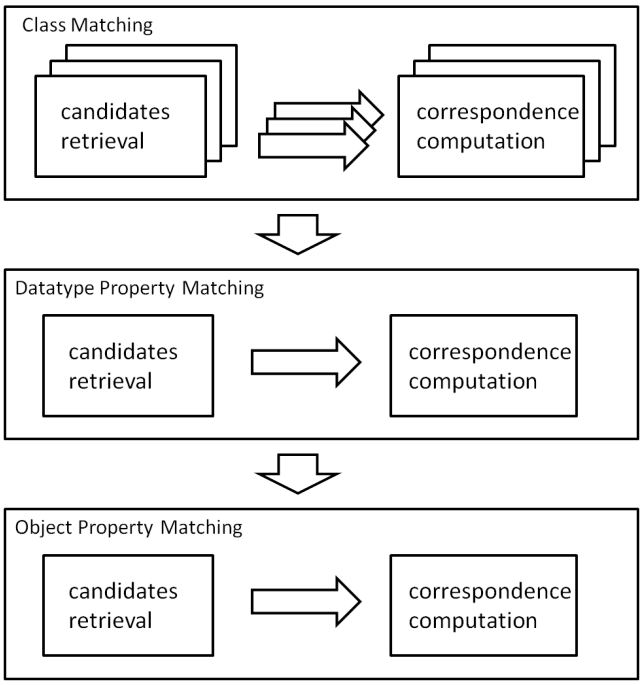

Figure 3: Three steps of semi-automatic mapping.

sible candidates the correspondences will be computed in the second step. For this, we use two term-based matching techniques (compare with [4]). Thus, our correspondence computation approach uses a string-based as well as language-based matching technique. Due to the variety of string metrics, we evaluated five preselected string metrics with a small test set.

In the final analysis the Monge Elkan [7] metric has delivered the best results for our test set. Not only using a syntactical comparison we also consider lexical-semantic relations between the XML and OWL artifacts as well. Therefore, our correspondence computation regards for example synonyms or hyponyms by using Wordnet as external source. To do this, we use the metric of Pirrï£¡ and Seco [8]. Nevertheless, for both matching techniques thresholds are required. These were acquired during evaluation.

In the following, the three matching algorithms will be described more in detail.

4.1 Class Matching

In the first step the class matching algorithm is situated in the candidate retrieval phase. According to our XML structure analysis, only XML elements with child elements are selected for the candidate set in the iteration. In the following correspondence computation step the candidate set is processed.

Thereby, the labels of the XML elements as well as the labels of their complex types are compared with the concept labels of the ontology. At this stage the already explained term-based matching techniques are called into action. Once consonances are noticed, a class mapping between an XML element and an OWL concept is created. At the end, it might be possible that some concepts of the ontology are left and are not mapped. When this case occurs, the algorithm begins a second iteration. Thereby, it regards the fact, that not only XML elements with child elements could be candidates for a class mapping.

Hence, a new candidate set is created by considering XML elements without child elements as well as XML attributes. However for limiting the candidate set, structural relations between already mapped XML elements and possible candidate mapping candidates are regarded. XML elements without child elements or XML attributes only might be candidates if a structural relation to another already mapped XML element exists. With this approach all considered mapping variants between XML and OWL concerning class mappings are taken into account.

4.2 Datatype Property Matching

In the XML structure analysis, we have established XML elements without child element and XML attributes as potential candidate for Datatype Property mappings. Hence, the candidate retrieval phase determines these XML artifacts. To do this, the already created class mappings of the class matching algorithm are involved as well. Because of these class mappings, the XML elements representing OWL concepts are known. Beginning from these particular XML elements, the corresponding Datatype Property candidate in the XML subtree is searched. With this approach the candidate retrieval considers structural relations within the XML document and forms the set for the following correspondence computation.

The second stage is proceeding as usual. With the help of the term-based matching techniques, string-based, as well lexical semantic-based comparisons taking place. All in all, the Datatype Property Matching algorithm ignores XML elements with child elements as candidates. In the case of mapping an XML element with child elements to an OWL Datatype Property, the XML document contains more scenario information than the ontology. Hence, the algorithm has to decide, if the complex XML element really represents the same semantics as the OWL Datatype Property. For such a decision the user has to be involved.

4.3 Object Property Matching

The Object Property Matching concept bases on the XML structure analysis as well. Similar to the Datatype Property Matching, the input for creating object property mapping is the class mapping set. By using the ontology information about the range and the domain of an object property in conjunction with class mapping set, the algorithm is able to identify probable candidates. After a successful correspondence computation, mappings are created. However, ontology relations between two concepts might exist more than once. As these relations differ from each other by their name and accordingly by their semantics, the algorithm is not able to resolve the semantics only by a label. Thus, a user interaction is necessary in such a case.

5. IMPLEMENTATION

We decided to integrate our proof of concept in the JXML2OWL Mapper from [10]. Figure 4 shows the screenshot of the extended tool.

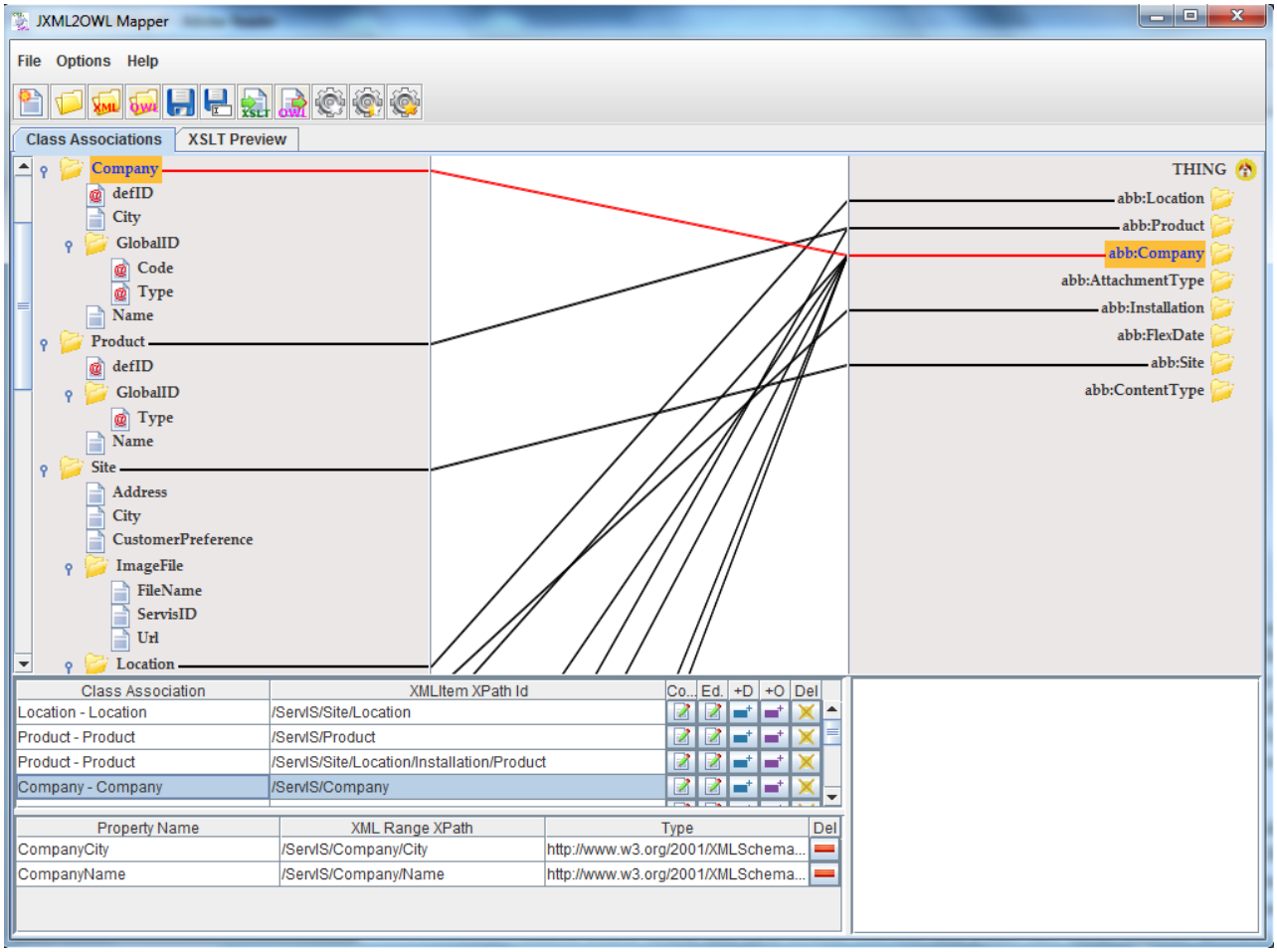

Figure 4: The final prototype with semi-automatic mapping suggestions.

The JXML2OWL application provides a graphical user interface to manually create mappings between an XML document as well as an XML Schema and an OWL ontology by drag and drop. For realizing such a mapping, the application uses the so-called JXML2OWL API developed by Rodrigues et al. The API allows the developer to create mappings between XML elements and OWL ontology entities in a comfortable manner. Thereby, XML elements are addressed by XPaths. For OWL entities unique identifiers according to the W3C OWL Recommendation are used. Regarding an automatic instance transformation, the API creates XSLT rules for all created mappings. Based on the created XSLT script, the application enables such an automatic instance transformation by an integrated XSLT processor. Nevertheless, the mapping at the schema layer has to be performed completely manually by the user with the existing application. Hence, our goal was to extend the application by integrating our semi-automatic matching approach.

The implementation of the semi-automatic matching approach requires the extension of the JXML2OWL API and the adjustment of the JXML2OWL Mapper. Thus, besides the manual mapping mode, the application was extended for the three step semi-automatic mapping mode. Three buttons to control the automatic modes were necessary. Furthermore, a solution for defining relations between XML elements manually by identifiers was required. Hence, a new dialog mode should take such a conditional mapping into account.

During our implementation we noticed the limits of information available solely from the XML Schema. Thus, we have recognized the need of the XML instance document to define conditional mappings. As a result, we have adapted our concept to enable a semi-automatic matching in the XSD as well as in the XML mode. Nevertheless, both modes have advantages and disadvantages, which were considered in the evaluation in more detail.

Besides the heuristic approach of our semiautomatic matching, the correspondence computation is grounded on the term-based matching techniques. For realizing the string comparison we have used the SimMetric Library of the University of Sheffield [3]. By using the Java WordNet Similarity Library [9] we consider in a further step the lexical semantic relations between the labels as an additional criterion for a mapping.

In order to implement our three matching algorithms (dependent from each other), we had to take adjustments and

extensions in the JXML2OWL API. When a mapping is created automatically a similarity value is computed. The API does not support such a value. Hence, we extended the API methods and the processing for handling and serializing the similarity value in the project file. Furthermore, the API received methods for enabling the conditional mappings.

In the following we summarize the new mapping procedure in the JXML2OWL Mapper: The starting point is the same as before. An XML schema or an XML instance document together with the OWL schema is loaded. Rather than dragging the mapping line between represented XML structure in the left frame and the OWL structure in the right, the user pushes the Class Matching button. The first algorithm computes the mappings and draws the lines between the corresponding elements. Afterwards, the user is invoked to control the automatic created mappings and if necessary to add missing mappings. Additionally, at this stage, the user obtained the opportunity to construct conditional mappings. When this is done, the Dataype Property Matching can be started using the second button. The results of this automatic algorithm are then presented in the second table underneath the class association table. Even after this stage, the user should control the results. In the last step the relation between concepts may be calculated by using the appropriate button. After the automatic computation follows an optional manual control phase.

6. EVALUATION

We evaluated the extended tool on two datasets, a real industry data from our project partner ABB and a research example about tourism that has also been used by Rodrigues et al.

However, for a well-performing semi-automatic matching tool, we had to determine at first appropriate thresholds for our two term-based matching techniques. Based on these two thresholds the correspondence computation has to decide, when an XML element candidate should be mapped to an OWL entity. Hence, we pursued the goal to investigate the threshold with the best compromise between precision and recall for the Monge Elkan metric as well as for Pirrï£¡ and Secco metric.

In order to do this, we used the same datasets which we applied for the tool evaluation. As a result, we computed the F1 score of 80% for the Monge Elkan metric in the ABB scenario. Further studies with a reduced as well as an extended amount of test entities in the ABB scenario revealed the same threshold. Hence, we could assume that this evaluated threshold for string-based comparison is independent of the scenario domain. Our assumption may be underpinned by the characteristic of such string comparisons, which only consider syntactical aspects of the label, rather than the semantics. Finally, for the Monge Elkan metric we do not have to evaluate the second scenario. Furthermore, we constituted the advantage, that no threshold adjustments concerning other application domains are required.

However, the situation is different with the WordNet-based metric of Pirrï£¡ and Secco. If lexical semantic information are considered, the threshold is not domain independent. As a result we evaluated two different thresholds for our

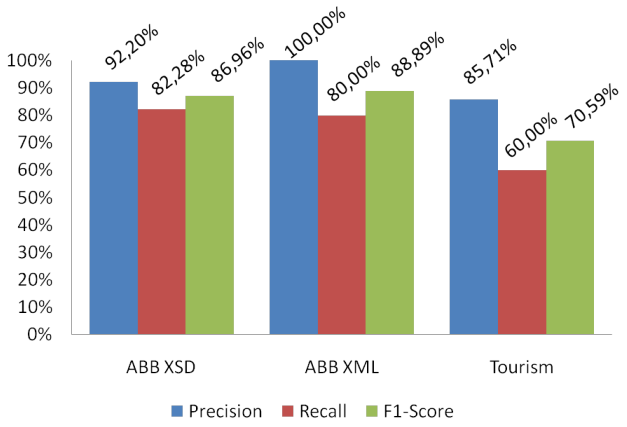

Figure 5: Evaluation of the semi-automatic mapping in two scenarios.

two datasets. The F1 score for the ABB scenario results in 59.13%. For the tourism use case we achieved 76% as the best compromise between precision and recall. That is, depending on the scenario, we have to re-calculate the threshold.

In order to evaluate the extended tool, we wanted to determine the degree of automatization. Because of our semiautomatic approach, we have not expected a degree of 100%. On the other hand, we have not found comparable research attempts for a semi-automatic matching between XML and OWL with an attached instance transformation. Hence, we had no minimum reference value to obtain at all. Nevertheless, we first had to establish a gold standard for both datasets by manually mapping the elements. Furthermore, we had to consider the special case of the ABB scenario with their possible relations between certain XML elements, expressed by denoted reference identifiers. As we already mentioned, the treatment of such relations need the semiautomatic matching in the XML mode in order to retrieve the references based on the particular structure of the XML instance document. Hence, we have evaluated the ABB use case in the XSD and XML mode. Reviewing an independent scenario apart from the ABB one, the tourism scenario in the XSD mode has been evaluated without any particular inner XML references.

The investigation of the three test cases revealed a precision of at least 85.71%. Even in the XML mode of the ABB scenario a precision of 100% could be gained. Furthermore, in both of the ABB test cases a recall of about 80% has been assessed. Nevertheless, in both ABB test cases different disadvantages were noticed. Thus, the opportunity in the XML mode to define XML inner references by identifiers is coupled with the lack of information about type definitions, usually gained from the XML schema. Hence, this information is not available for the term-based matching techniques. The XSD mode reverses this coherence, because no XML reference could be created. But all information contained in an XML schema is available for the term-based matching techniques.

Considering the tourism evaluation, we could reveal the limits of our semi-automatic matching. It shows that implicit semantics in an XML document can not be fully extracted automatically, especially when more semantic knowledge is needed in order to decide particular mappings. Hence, manual support by the user is necessary at this stage and explains the rather low recall of 60% in this scenario.

Figure 5 shows our evaluation results for each of the three phases in the two scenarios described.

7. CONCLUSION AND OUTLOOK

In this paper, we were concerned with lifting of purely syntactical described data up to a semantic level in order to enable companies to automatically combine their data from different sources. We first conducted an XML structure analysis. On top of it, we developed a concept for a semi-automatic matching approach. For our proof of concept, we have integrated our lifting solution in the JXML2OWL Mapper of Rodigues. With the created application and its evaluation we were able to emphasize the added value. In the future, our concept could be further developed concerning matching techniques in order to gain better results. Additional consideration could take place concerning implicit semantics in XML.

8. REFERENCES

[1] A. Budanitsky and G. Hirst. Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and Other Lexical Resources*, volume 2, 2001.

[2] C. Bussler. B2B Integration: Concepts and Architecture. Springer, 2003.

[3] S. Chapman, 2006. http://staffwww.dcs.shef.ac.uk/people/S.Chapman/simmetrics.html.

[4] J. Euzenat and P. Shvaiko. A survey of schema-based matching approaches. *Journal on Data Semantics*, 2005.

[5] M. J. G. Reif and H. Gall. Towards semantic web engineering: WEESA-Mapping XML Schema to ontologie. In *Proceedings of the WWW2004 Workshop on Application Design, Development and Implementation Issues in the Semantic Web*, 2004.

[6] C. Z. M. Ferdinand and D. Trastour. Lifting xml schema to owl. In *Web Engineering*, pages 776–777, 2004.

[7] A. Monge and C. Elkan. The field matching problem: Algorithms and applications. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, pages 267–270, 1996.

[8] T. V. N. Seco and J. Hayes. An intrinsic information content metric for semantic similarity in WordNet. In *ECAI*, volume 16, page 1089, 2004.

[9] G. Pirró and N. Seco. Design, implementation and evaluation of a new semantic similarity metric combining features and intrinsic information content. In *On the Move to Meaningful Internet Systems: OTM 2008*, pages 1271–1288, 2008.

[10] P. R. T. Rodrigues and J. Cardoso. Mapping xml to existing owl. In *International Conference WWW/Internet*, pages 72–77, 2006.

[11] M. Wauer, D. Schuster, and J. Meinecke. Aletheia - An Architecture for Semantic Federation of Product Information from Structured and Unstructured Sources. In *12th International Conference on Information Integration and Web-based Applications and Services (iiWAS 2010)*, pages 323–330, 2010.