Construction of a domain ontology from the conceptual data models

Tami Abdelaziz*, Elberrichi Zakaria**

* a_tami@esi.dz, ** elberrichi@gmail.com

EEDIS Laboratory, Djillali Liabes University of Sidi Belabbes, Algeria.

Abstract— Ontologies are explicit conceptualization for logic processing specifications. Building ontology is a difficult task. The ability to obtain semantic concepts directly from a language exchange is a new methodology that can help reduce the complexity of the problem.

In this paper, we propose a new approach to construct domain ontology from the conceptual data models. This approach takes into account the syntactic and semantic aspects of conceptual data models and uses information retrieval techniques to define a similarity measure calculated based on the basic concepts of the conceptual data models. Thus, we present a methodology for extraction and construction of ontology from the conceptual data models.

Key words— Ontology, conceptual data models, semantic concepts, information retrieval, similarity.

1. INTRODUCTION

We propose, in this paper, a semi-automatic approach that helps building domain ontology, based on the resources of knowledge gained from the conceptual data models.

Conceptual data models are very interesting elements in information systems and software engineering. In knowledge engineering, conceptual data models play an important role in representing concepts and relationships between concepts in a particular domain. To conceive an information system is to be able to give a representative scheme of this system, by exploiting the knowledge within the system. In the domain of knowledge, ontology represents the data in terms of syntax and semantics. In fact, ontology are structured in concepts, properties and relationships between concepts. It allows to present, integrate and annotate the knowledge in a more effective and efficient manner.

Ontologies are generally reusable. However, the construction of domain ontology is necessary to minimize ambiguities left during the conception of ontology; in fact, several concepts can define one only term in different uses.

Our approach is consists of three steps: the first step is based on the application of mapping techniques of UML concepts (classes, properties and association), and the application of the techniques of semantic research on these concepts. The second step focuses on the translation of UML concepts to XSD elements, exploiting the definition and representations rules of XSD elements, specifically, complex types, simple types, and data generated types. The third step relates to the detection and extraction of the hidden semantics.

In this paper, we will to explain and justify our solution, so we detail each phase of the proposed approach. We will clearly define the complete architecture of our approach, while explaining the phases and components.

2. RELATED WORKS

Several approaches for the construction of domain ontologies have been proposed. In the continuation, we will present some of these works.

Through the analysis of entities, relationships and attributes [1] defines the mapping rules between the concepts of databases (entity, attributes, relationships and constraints) to the ontological concepts equivalent (class, properties, ...). A process of mapping is established, where relational databases are used for the construction of ontology. Entities transformed into classes, relationships transformed into classes, the attributes transformed into properties in the classes and the tuples transformed into instances in the ontology.

Benslimane and al. [2], propose an approach based on the analysis of HTML forms. The semantics obtained is used to produce the relational schemas in order to enrich them. The transformation rules are applied to build the ontology.

Several works use UML conceptual models and ontologies representation languages to generate the ontology, and propose different approaches based on MDE (Model Driven Engineering). In [3], the generation of ontology is done by annotating UML models, and uses stereotypes in the UML profile. The transformation is expressed in XSLT (eXtensible Stylesheet Language Transformations) style defines the rules for transforming a source tree into a result tree.

The EODM (EMF Ontology Definition Metamodel) project is an implementation using EMF (Eclipse Modeling Framework). EODM is derived from ODM (Ontology Definition Metamodel) of the OMG (Object Management Group), implemented in (EMF). To facilitate software development and implementation EODM includes analysis, serialization, reasoning, and the transformation between RDFS/OWL and other data modeling languages such as UML, Ecore. EODM is an open-source Eclipse project. In [4] some techniques are used as the definition of UML profiles and ODM. The objective is to represent ontologies in RDFS format one using the graphical notation of UML.

3. PROPOSED APPROACH

The idea we propose to build ontology is based on a series of steps and rules of transformation of conceptual data models in XML Schema which is the pivot for other analysis rules and transformations to get ontology based on the basic concepts that carries the XML language. The functional architecture of our approach in Fig. 1.

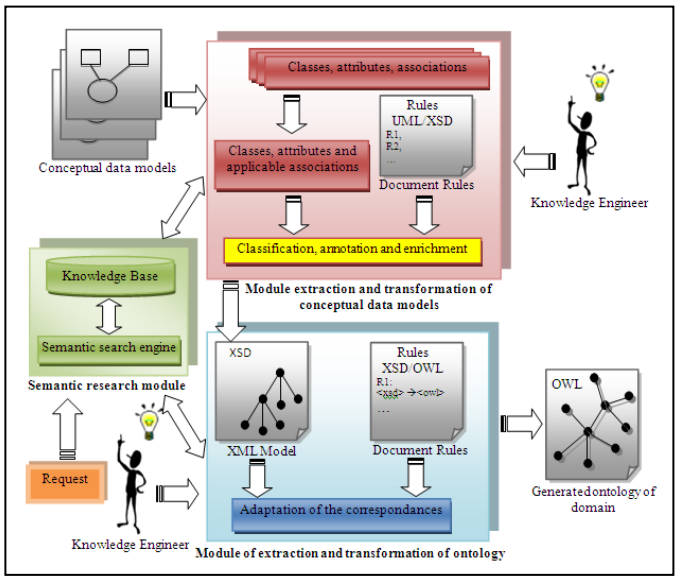

Fig. 1. The approach architecture

As we can see, the approach is composed of 3 main parts explained below.

3.1. Module of extraction and transformation of conceptual data models: This module includes the rules of extraction and transformation of conceptual data models toward a XML schema. These rules have the function to translate the conceptual data model to XSD scheme. These rules take into account the static structure (definition of concepts), and the dynamic aspect concerning the integrity constraints such as primary key and foreign key.

3.2. Module of extraction and transformation of ontology: The xml model generated from the previous phase is used in the module of extraction and transformation of ontology in order to apply transformation rules XSD/OWL. These rules permit the construction of the OWL ontology (classes construction rules, properties, relationships between classes ...). The generated ontology describes the knowledge specific to a given domain and provides a formal understanding, consensus, referenceable of the domain concepts.

3.3. Semantic search module: The semantic search plays an important role for the analysis and the selection of the concepts of the conceptual data models. Information retrieval techniques are used to index and to find the pertinent concepts of the class diagrams. This phase is applied in parallel with the other phases of the approach to facilitate the detection of the pertinent concepts in order to generate the xml model that represents the class diagrams and to generate the domain ontology.

The class diagram is represented by an XML document under XSD format. UML concepts are defined from the senses used. However, the detection of the senses of the concept is not an easy task. For this, the knowledge base can overcome the problem of heterogeneity of the senses of the concept since the knowledge engineer can make the best choice between the possible senses with the help of a semantic search engine.

For the knowledge base, we have chosen to use WordNet. In fact, using WordNet can help apply queries to search for words and senses of words and to follow the hierarchy of senses of words. We can perform lexical disambiguation algorithms. We need a similarity measure in the class diagrams. The based similarity measure on the WordNet linguistic resource may help to know that the concepts are similar or not, and to perform the mappings in our approach[5].

4. Detailed Description of the Approach

4.1. Alignment of conceptual data models: In his approach [6] defines an alignment that takes as input two schemes (S1 and S2) and product in output an alignment between the elements of S1 and S2. The alignment process generally passes by three phases: 1) Conversion schemes S1 and S2 in the internal format of the alignment method, 2) Calculation of similarities for each mapping, 3) Selecting a subset of mappings.

External resources (e.g. Wordnet) and parameters are used to adapt the operation of the alignment process.

In some approaches, the mappings are associated with a similarity value ([0, 1]), and to a type of relationship (similar, more general or different).

In our case the alignment of conceptual data models is defined as the operation that takes as input a number of conceptual data models and product in output a list of correspondences between the concepts of these models. Since the conceptual data models regroup knowledge with similar correspondences, a similarity measure is used to find pertinent concepts in order to produce the XML model.

4.2. Generation of an XML Model: This phase is performed to extract the concepts found in class diagrams, specifically: classes, attributes and associations, these concepts should be grouped for each class diagram. In order to generate an XML model, we follow a series of steps to perform mappings between the concepts of class diagrams.

4.2.1. Selection of pertinent classes and associations: This step permits to measure the pertinence for each class and each association in a class diagram selected from a number (*nd*) of class diagrams. Pertinent classes and associations are represented by vectors. In this step, we do the calculations on the occurrence of any class and any existing relationship to make the mappings of classes and associations, in order to choose pertinent classes and associations, and then we proceed to clean and enrich the xml generated model. However, to improve the results of pertinence, the knowledge engineer can adapt a threshold to select pertinent classes and associations.

4.2.2. Technique of frequency calculation: We used the weighting technique used in information retrieval in the text corpus TF-IDF (term frequency-inverse documents frequency) to calculate the frequency of occurrence of the class in order to present a vector that contains the name of the class UML and calculated frequency.

Formal definition

The frequency of a class $C_{i,j}$ in the class diagram *dj* is defined by: $tf_{i,j} = \frac{n_{Ci,j}}{m_{k,j}}$,

with : $n_{Ci,j}$ is the number of class occurrences $C_{i,j}$ in the diagram d_j.

$m_{k,j}$ is the number of occurrences of all the classes in the diagram d_j. $m_{k,j} = \sum_k n_{k,j}$

The inverse frequency of a class diagram *dj* is defined by:

$$idf_i = \log \frac{nd}{ndj}$$

with : *nd* is the number of diagrams of classes in the corpus.

ndj is the number of class diagram where belongs the class $C_{i,j}$.

Thus, the weight of the class $C_{i,j}$ in the class diagram *dj* is defined by: $tfidf_{i,j} = tf_{i,j} \times idf_i$

4.2.3. Mapping of the concepts: The techniques of information retrieval can solve the problems of ambiguities of the names of classes and associations in class diagrams, two class names are designated the same class in two different class diagrams. Through mapping classes, we can give up the semantic ambiguity between class names, our proposition is to use the concept of calculate of semantic distance to remove this ambiguity. However, the knowledge engineer adjusts a threshold (real value ([0..1])) for calculating the semantic distance between two classes. If two classes are completely similar, then the threshold to a value equal to 1. The threshold value of value 0 indicates that there is no similarity between the two classes.

For the mapping of the concepts we need some treatments between the elements of the XSD documents that represent the class diagrams, these treatments permit to establish operations between classes, associations or properties. These operations are defined in Table 1.

TABLE 1. OPERATIONS USED FOR MAPPING CONCEPTS

Operation	**Significance**
IdentEle ()	Operation used if two elements are identical
RenEle ()	Operation used to reappoint an element of the XML model
AddEle()	Operation used to add an element to the XML model
SuppEle()	Operation used to suppress an element
MergEle ()	Operation used to merge two different elements in only one element
EclatEle ()	Operation used to explode an element in several equivalent elements

Calculations of similarity: The applications basing on the calculation of similarity suffer of several problems as: disambiguated, indexing, extraction of the data, integration of the data... In our work we present a measure of similarity between the classes that permits to use the semantic neighborhood for the calculation of the weights of the class participants in the class diagram.

The semantic similarity is applied in several domains of research of information. To start our propositions, we took advantage of notions of similarity calculation and we indexed the concepts found by the semantic neighborhood.

In the literature, several approaches use the notions of graph (node, arc, path, distance.). Two axes exist to measure the similarity between the concepts ontology:

The first axis articulates on the tree structure, it regroups the approaches based on the arcs of a graph, where the nodes are the concepts and the arcs are links between concepts. The arcs are used to browse the path and to calculate the distance between the concepts.

R. RADA and his colleagues [7] presents a measure based on the calculation of the distance between two concepts; indeed, this distance represents the minimum number of arcs to browse between a concept C_1 who meets in a path joining a concept C_2. This distance is noted by $dist(C_1, C_2)$. The measure of similarity is defined by:

$$sim_{rada}(C_1, C_2) = \frac{1}{1 + dist(C_1, C_2)},$$

with: $dist(C_1, C_2) = che \min_{\min}(C_1, C_2)$

W. Zhibiao and his colleague [8] presents a measure based on the depth of two concepts (C_1 and C_2) in relation to the smallest concept generalizing common (C) (e.g. Fig. 2), indeed, the depth of a concept represents its distance with the root (R) while passing by the concept (C). The measure of similarity is defined by:

$$sim_{Wu\&P}(C_1, C_2) = \frac{2 \times prof(C)}{prof(C_1) + prof(C_2)},$$

with : $prof(Ci) = dist(C_i, R)$

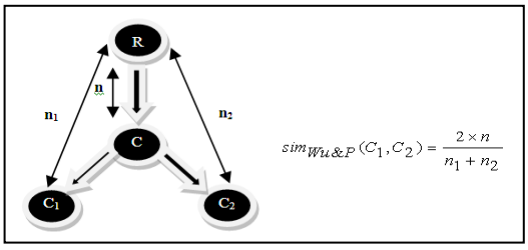

Fig. 2. Example of hierarchy of concepts structure

The second axis articulates on the informational content of the concepts, it regroups the approaches based on the nodes. Several methods are developed in this axis. We present some measures used by these methods. In [9] present a state of the art where different measures are compared.

The informational content is introduced by [10], where the notion of probability is used to select the pertinent concept in a corpus, indeed, the frequency of apparition of a concept (C_1) as well as the frequency of apparition of another concept subsumed (C_2) imply that the concept (C_2) is more specific than the concept (C_1). The informational content is defined by:

CI(C) = - log(P(C)).

Where P(C) represents the probability of the concept C.

If C_1 is similar to C2 (C_1 is-a C_2), then $P(C_1) <= P(C_2)$.

The measure of similarity is defined by:

$$sim_{Resnik}(C_1, C_2) = \max_{C \in subsumers(C_1, C_2)} [-\log P(C)]$$

with $subsumers(C_1, C_2)$: is the set of the concepts that are ancestors of C_1 and C_2 in one of the senses of these concepts.

This equation means that we want a concept, with a maximal value of the informational content, that subsumes the two concepts (C_1 et C_2).

The probability P(C) is defined by the following formula:

$$P(C) = \frac{freq(C)}{N}$$

P(C) represents the probability of apparition of a concept C in the corpus of reference. This probability corresponds to the relative frequency: $freq(C) = \sum_{N \in Words(C)} count(N)$

N = total number of concepts in the corpus.

Words (C) is the set of names that have a meaning that one of the ancestors is the concept C.

In addition, there are hybrid approaches offer the combination of the two main axes. That is to say, these approaches are based on a model that uses techniques based on arcs (distances) and techniques based on the nodes (information content).

[11] presents a measure based on the calculation of the inverse of the semantic distance between the concepts (C_1 et C_2) in order to rephrase the results by based calculations on the nodes. The measure of similarity is defined by:

$$Sim_{Jiac}(C_1, C_2) = \frac{1}{dist(C_1, C_2)},$$

where $dist(C_1, C_2) = CI(C_1) + CI(C_2) - 2CI(C)$

with C : is the concept that maximizes the value of similarity ($C \in subsumers(C_1, C_2)$)

Leacock. C and Chodorow. M [12] presents a measure based on the length of the shortest path in a hierarchy of concepts (is-a) between two synsets of Wordnet, indeed, the measure of similarity is defined by:

$$Sim_{Leac\&Cho} = \max[-\log(\frac{length(C_1, C_2)}{2 \times D})]$$

length(C_1,C_2) represent the length of the shortest path between the concepts C_1 and C_2.

D : represent the length of the longest path between the concept root (R) and the lowest concept.

In our work, we used the measure of Wu&Palmer that follows the first axis. Our work doesn't serve to validate the techniques of measure of similarities, but, one to note the utility to apply a measure of similarity to do the mappings between the concepts of the different conceptual data models.

The pertinent classes and associations are regrouped in order to form an xml document (XML schema), this document will participate in the phase of ontology generation.

4.2.4. Extraction and transformation UML/XSD

4.2.4.1. Indexing of the class diagrams: The main idea is to transform the basic concepts of conceptual data models by the definition of XML schema, as shown in Fig. 3. Our approach is to find a single and standard format as a pivot to generate an ontology format.

Indexing class diagrams is based on the use of existing pertinent concepts in class diagrams. The index structure is important to make correspondences between models. An XSD document permits to define the structure of an XML document, as well as all the tools for processing XML documents can be applied. For this we have chosen to use XSD as an indexing format of the class diagrams.

XSD uses to support the creation, the manipulation and the retrieval of the concepts while taking advantage of the structure of the XSD documents. The elements of the generated XML model can be defined through the correspondence between UML concepts and XSD elements. Several functions may be presented:

- XSD documents are manipulated by the applications of research of the elements.
- XSD documents are exchanged between applications.
- XSD permits to makes the creation, the treatment, the research and the access to the UML concepts through the hierarchy and the content elements of the XSD document. However, XPath (XML Path Language) permits to navigate the XSD documents.

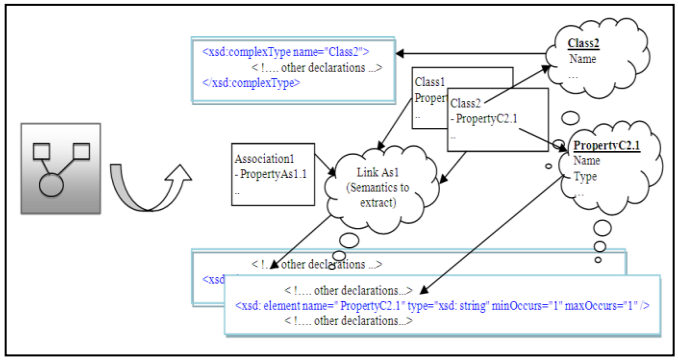

Fig. 3. Index of a class diagram

4.2.4.2. The transformation rules UML/XSD: In this section, we present the transformation rules between the concepts of UML class diagram (classes, attributes, associations ...) and the elements of XML schema. These rules concern the definition of concepts (UML class diagram), as well as integrity constraints acting on primary key and foreign key (e.g. Fig. 4). We propose the following rules for the UML/XSD transformation.

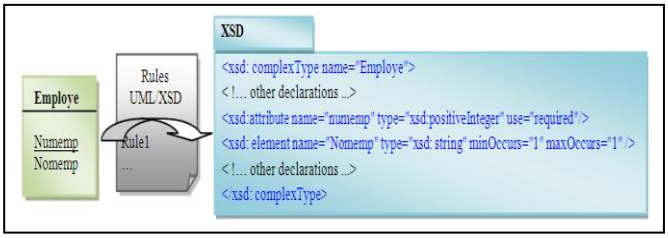

Fig. 4. Example of correspondence UML/XSD

Rule for classes

Rule 1: A class in the conceptual model defines a complex data structure. Each class of a conceptual data model is transformed into an CompexeType element <xsd:complexType>. The element name is the same as the class name. If the attributes of a UML class is not limited to a particular order, the <xsd:all> element is used to contain the UML attributes, otherwise the element "sequence" is used. It is possible to declare abstract type (abstract = "true").

Rules for attributes

In UML there is only one way to implement an attribute, but in XML, there are two ways: elements and attributes.

These last are used to represent an UML attribute of complex type, but for simple values both ways are favored. When we take a decision to choose between XML element and XML attribute, it is essential to understand the nature of data that is assigned to the UML attribute. For our purposes, it is important to support a choice between two ways.

Rule 2: Each non-key attribute is part of a class or an association of a conceptual data model is transformed into a simple element < xsd:element >. The primitive type is the attribute (e.g. type = "xsd: string").

The minOccurs and maxOccurs attributes are used to reflect the cardinality of the attribute of the class appropriate. The attribute can be defined by a default value. If the attribute refers to another class, the element declaration is followed by a definition of complexType, which contains a reference to the appropriate complexType.

Rule 3: Each key attribute is part of a class is transformed into <xsd:attribute> attribute. The name, type and visibility (obligatory or optional) are identical that the name, the type and the visibility of the class attribute.

Rule 4: For a primary key attribute, using a <xsd:key> element is useful to represent this constraint. While to represent the foreign key constraint <keyref> the element is used.

Rule 5: Associations:

Each association of a conceptual data model is transformed in element (xsd:element), the name of element is defined by the role of association, the element type is defined by the name of the targets class. This element is added to the complexType representing the class source. The attributes minOccurs and maxOccurs reflect the cardinality of the association. If the direction of the association is not specified, then use the constraints of multiplicity to determine the direction of the association. For n-ary associations, the primary keys of the participating association classes are added to the element that represents the association.

Rule 6: Generalization / Specialization:

To add additional attribute in a class, an element (xsd: extension) is declared with the attribute (xsd: base) defined in the name of the super-class. For restraint of the attributes of a super-class, an element (xsd: restriction) is declared with the attribute (xsd: base) defined in the name of the super-class. The xsd elements (complexContent, simpleContent) are generated to contain elements that represent the uml attributes of the subclass in the elements of the declared extension or restriction. For multiple inheritances the xsd attributes that represent the primary key of super-class are added to the subclass.

Rule 7: Composition / Aggregation

An element (xsd: element) is generated for each association of aggregation or composition type, the element name is defined by the role of association, the element type is defined by the name of the aggregate class (class component). This element is added to the complexType representing the aggregate class (the composite class). In addition, the element that represents the key attribute of the aggregate class and added to the generated element to represent a composition. The attributes minOccurs and maxOccurs reflect the cardinalities of the association.

Rule 8: Enumeration

A simpleType element is declared to represent an enumeration. A restriction element is generated. Attributes are added to the restriction element as enumeration XSD elements, defined with the UML attribute name value.

Rule 9: Comment

Each UML comment is transformed into <xsd:annotation> element.

Rule 10: The data types

Data types such as (int, double, string ...) are represented in the XML schema with the XSD standard notation (<xsd:int>, <xsd:double> and <xsd:string> ...). The data types defined by the user are represented by the XSD <simpleType> element.

Rule 11: Packages

An element (xsd: element) is generated for each UML package.

4.3. Generation of ontology : The objective of this section is to bridge the gap between XSD and OWL formats. In particular, we present a strategy of the way who's OWL can be generated from XML Schemas. This must be done through the establishment of rules of correspondence between the different data model elements XML and OWL (e.g. Fig. 5). Using the correspondence between tag and concept, we can define rules to identify the elements of XML schema in order to define the transformation between XML Schema elements and the concepts of owl.

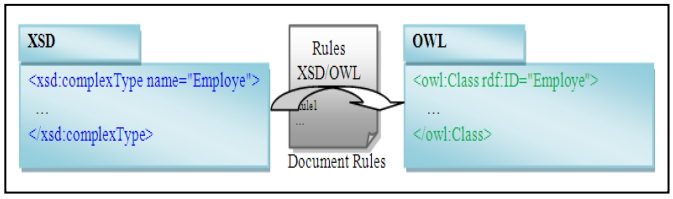

Fig. 5. Example of a correspondence XSD/OWL

We propose the following rules for the creation of the owl format:

The element (xsd: complexType)

Complex types of XML Schema and OWL classes are sets of entities with common characteristics. The complex types of XML Schema are converted to OWL classes.

Rule 1: A complex element (xsd: complexType) is transformed into class under owl (owl: Class).

The element (xsd: simpleType)

A simple type element (xsd: simpleType) is an element that doesn't contain any elements sons, and precise of the constraints and information as text only. The properties of data type (owl: DatatypeProperty) are used to connect individuals to data values. The simple types of XML Schema are transformed to the properties DatatypeProperty of OWL.

Rule 2: Every simple element (xsd:simpleType) is transformed into property owl (owl: DatatypeProperty).

The element (xsd: attribute)

The attributes of XML Schema and the properties of OWL represent the simple type. The attributes of XML Schema are transformed into the properties datatype of OWL.

Rule 3: Each attribute (xsd: attribute) is transformed into property owl (owl: DatatypeProperty).

The element (xsd: element)

Since XML Schema elements and OWL properties represent common characteristics (simple and complex Type). The XML Schema elements are transformed into properties of OWL (Datatypeproperty and ObjectProperty).

Rule 4: Each element (xsd:element) that doesn't contain another element (xsd:element), nor an attribute (xsd:attribute) is transformed into property data type owl (owl:DatatypeProperty).

Rule 5: Each element (xsd: element) that contains another element (xsd:element) or an attribute (xsd:attribute) is transformed into class relationship under owl (owl: ObjectProperty).

Multiplicities (xsd: minOccurs) (xsd: maxOccurs)

Rule 6: The multiplicities (xsd: minOccurs) and (xsd: maxOccurs) are transformed into their equivalent (owl: minCardinality) and (owl: maxCardinality).

Elements (xsd: sequence), (xsd: all), (xsd: choice)

The sequences and the choices of XML Schema are represented by the anonymous classes of OWL formed using the operators of intersection and union.

Rule 7: Each element (xsd: sequence) or (xsd: all) is transformed into (owl: intersectionOf). The element (xsd: choice) is transformed into (unionOf).

The element (xsd: extension)

It exists in every OWL ontology a superclass named Thing, which all other classes are subclasses. This brings us directly to the concept of inheritance, available using the subClassOf property.

Rule 8: An extension element (xsd: extension) is transformed into subclass under owl (rdfs:subClassOf).

The element (xsd: restriction)

Rule 9: A restriction element (xsd: restriction) is transformed into subclass under owl (owl: Restriction).

The element (xsd: enumeration)

Rule 10: An element (xsd: enumeration) is transformed into subclass under owl (owl: oneOf).

The element (xsd:annotation)

Since XML schema annotations and OWL comments are textual descriptions. The XML schema annotations are transformed into OWL comments.

Rule 11: An element (xsd:annotation) is transformed to comment under owl (rdfs: comment).

5. CONCLUSION

This paper was devoted to the construction of domain ontology; the objective is to develop an approach for the translation of conceptual data models in OWL ontology format. The proposed approach is based on a strategic vision of correspondence between the concepts of the conceptual data model and the ontological concepts.

In this paper, we have presented the architecture of our approach. We also justified our choice of using XML as representation formalism intermediate formats between conceptual data models and ontological models, as well as the set of rules to justify our approach.

The approach we have proposed is to develop a representation (XML Schema) as a pivot, then transform this presentation into another data format (OWL) while adding the syntactic and semantic enrichments to conduct the confirmation and the best realization.

This work will be the subject of an implementation to evaluate the proposed solution. It would be interesting to develop a prototype permitting to automate the transformation of conceptual data models to the ontological model.

REFERENCES

[1] J. Trinkunas and O. Vasilecas, "*Building ontologies from relational databases using reverse engineering methods*", In: CompSysTech '07: Proceedings of the 2007 international conference on Computer systems and technologies New York, NY, USA, pages 1-6, 2007.

[2] SM. Benslimane, M. Malki, MK. Rahmouni, D. Benslimane, "E*xtracting Personalised Ontology from Data-Intensive Web Application: an HTML Forms-Based Reverse Engineering Approach*", INFORMATICA International Journal Vol 18(4), ISSN: 0868-4952, 2007.

[3] D. Gašević, D. Djurić, V. Devedžić, V. Damjanović, "*Converting UML to OWL Ontologies*", In Proceedings of the 13 th International World Wide Web Conference, NY, USA, pages 488-489, 2004.

[4] C. Faucher, F. Bertrand, JY. Lafaye, "*Génération d'ontologie à partir d'un modèle métier UML annoté*", Revue des Nouvelles Technologies de l'Information, vol 12, pages, 65-84, 2008.

[5] A. Amine, Z. Elberrichi, L Bellatreche, M Simonet, *"Concept-based Clustering of Textual Documents Using SOM*." In: proceedings of the 6th ACS/IEEE International Conference on Computer Systems and Applications AICCSA-08, 2008.

[6] JR. Falleri, "*Contributions à l'IDM : reconstruction et alignement de modèles de classes*", Académie de Montpellier, Université Montpellier II -Sciences et Techniques du Languedoc, 2009.

[7] R. RADA, H.,MILI, E.BICKNELL, M.BLETTNER, "*Development and application of a metric on semantic nets*", IEEE Transaction on Systems, Man, and Cybernetics, vol. 19, no 1, pages 17–30, 1989.

[8] W. Zhibiao, M. Palmer, "*Verb semantics and lexical selection*". In Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics, pages 133-138, 1994.

[9] S. Patwardham, "*Incorporating Dictionary and Corpus Information in a Measure of Semantic Relatedness",* M.S. Thesis, August 2003.

[10] Philip Resnik, "*Using Information Content to Evaluate Semantic Similarity in a Taxonomy*", IJCAI, pages 448–453, 1995.

[11] Jay J. Jiang and David W. Conrath, "*Semantic similarity based on corpus statistics and lexical taxonomy*". In Proceedings on International Conference on Research in Computational Linguistics, Taiwan, 1997.

[12] C. Leacock, and M. Chodorow, "*Combining Local Context and WordNet Similarity for Word Sense Identification*". In WordNet: An Electronic Lexical Database, C. Fellbaum, MIT Press, 1998.