A Semantics Definition Metamodel

Ma. Laura Caliusco y César Maidana
GIDSATD - UTN - FRSF
Lavaise 610 – Santa Fe
Santa Fe - Argentina
+54 0342 4608585 – Int 222
{mcaliusc, cmaidana}@frsf.utn.edu.ar

y

Ma. Rosa Galli y Omar Chiotti
INGAR-CONICET
Avellaneda 3657
Santa Fe - Argentina
+54 0342 4534451
{mrgalli,chiotti}@ceride.gov.ar

Abstract
New information technologies provide new opportunities for allowing collaborative business-to-business (B2B) relationships. An effective B2B relationship requires a right modeling of collaborative processes and each message within these processes.
The XML (eXtensible Markup Language) is becoming widely used for representing both the processes and the business documents. However, the use of XML is insufficient for implementing an effective B2B relationship because of semantics heterogeneity that takes place in a collaborative process.
In order to represent semantics ontology specification languages from Artificial Inteligence (AI) area have arisen. However, the main disadvantage of these languages is they are mostly based on logic formalisms to support machine reasoning. This makes the language syntax unfamiliar for business analysts who define the collaborative process and model the business documents to be exchanged.
To fill the gap between people involved in the business documents definition and ontology specification languages, there are some proposals for the use of the Unified Modeling Language (UML) in ontology development. But, UML does not satisfy all requirement for ontology modeling.
In this paper we present a metamodel for ontology definition. The objective of this metamodel is to overcome the gap between B2B area and AI techniques to model semantics associated to XML-based B2B documents.

Keywords

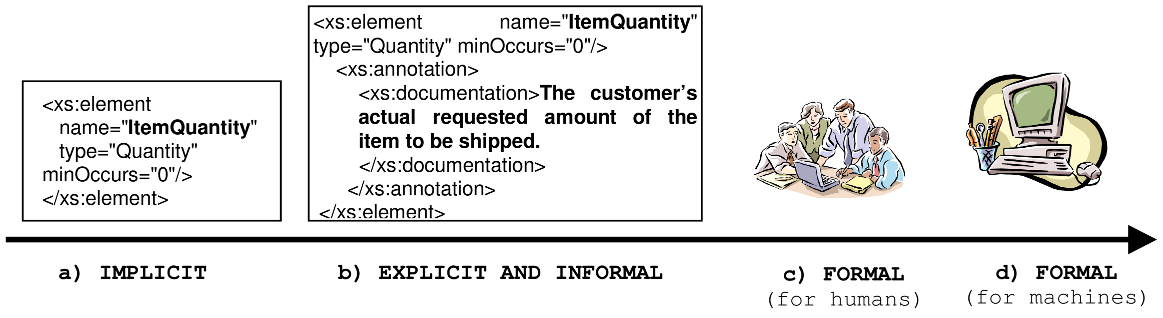
Ontology, Metamodel, XML, business-to-business, aritificial intelligence, software engineering.

1. INTRODUCTION

New information technologies, such as Internet, web-based applications and distributed systems provide new opportunities for allowing collaborative business-to-business (B2B) relationships. In these relationships companies can operate as a single entity and make joint decisions focusing on adding value for their customers. An effective B2B relationship requires a right modeling of collaborative processes and each message within these processes. Each message contains business information that may be defined by a vocabulary that is shared by the parties engaged in the B2B relationship. This business information is contained by a business document, like Purchase Order, Catalog and so on.

The XML (eXtensible Markup Language) is becoming widely used for representing both the process and the business documents [6]. XML was created to facilitate data interchange among different applications, data sources, and operating systems. Due to the data within an XML document are tagged, the document carries with the information necessary to recognize, extract, and manipulate those data. Many XML-based specifications for B2B area have been defined, such as RossettaNet[1], ebXML[2] and OAGIS[3]. However, the use of XML is insufficient for implementing an effective B2B relationship because of semantics heterogeneity [4].

According to Uschold (2003), semantics means something that carries meaning and can be implicit, explicit and informal, explicit and formal for human processing and explicit and formal for machine processing. He defines a semantic continuum applying these concepts to Semantic Web. In Figure 1 we represent this idea for XML-based B2B specifications. In Figure 1 (a), it can see that XML specifications contain implicit semantics due to the meaning of each tag exists only in the minds of the humans who have defined them. For example, the tag "*ItemQuantity*" has only meaning for the persons who have defined it and its meaning is implicitly encoded in the application that use it. There are specifications that contain explicit semantics informally defined. For example, OAGIS specifications contain explicit semantics defined between *annotation* elements in natural language, as it can see in Figure 1 (b). This helps the implementation of B2B XML-specifications but it is not enough for determining in unambiguous way the meaning of the data at run time [4].



```
<xs:element
  name="ItemQuantity"
  type="Quantity"
minOccurs="0"/>
</xs:element>
```

```
<xs:element          name="ItemQuantity"
type="Quantity" minOccurs="0"/>
   <xs:annotation>
      <xs:documentation>The customer's
      actual requested amount of the
      item to be shipped.
      </xs:documentation>
   </xs:annotation>
</xs:element>
```

Figure 1. Semantic Continuum.

To process the semantics at run time, it has to be expressed in a machine processsable language. A lot of work has been done in the Semantic Web area, where computers will be able to use the data on the web not just for display purposes, but for automation, integration and reuse of data across various applications [2]. Klein (2003) has defined a procedure that can be used to turn XML documents into knowledge structures specified in an ontology specification language for Internet. From B2B perspective, the main disadvantage of ontology specification languages is they are mostly based on logic formalisms to support machine reasoning. This makes the language syntax unfamiliar for business analysts who define the collaborative processes and model the business documents to be exchanged. Furthermore, the collaborative processes and the business documents have to be implemented by software engineers who have to interpret the information model.

To fill the gap between people involved in the business documents definition and ontology specification languages, there are proposals for UML use in ontology development [8]. But, UML itself does not satisfy needs for representation of ontology concepts that are borrowed from Descriptive Logic and that are included in ontology specification languages [9].

The objective of this paper is to present a metamodel for modeling explicit and formal semantics, for human processing, associated to XML-based business documents. Firstly, we analyze the UML role in ontology modeling.

[1] www.rosettanet.com
[2] www.ebxml.org
[3] www.openapplications.org

Then, we present a metamodel for ontology modeling using "Unified Modeling Language: Infrastucture" specification, version 2 [17] and analyze the relationship between the XML specifications and ontologies in order to add formal and explicit semantics, for human processing, to business documents. Finally, we present future directions and conclusions.

2. UML For Ontology Modeling

The Unified Modeling Language (UML) is a standard defined by the Object Management Group [14]. UML defines an abstract language for describing the structure and behavior of software systems. A standard graphical notation is also defined for creating views of the model elements in this language.

UML has been used to model ontology due to UML class diagram can be used to express concepts in term of classes and relationships among them. Cranefield (2001) proposed an ontology representation formalism based on a subset of the UML together with its associated Object Constraint Language (OCL) for agent software communication.

One advantage of using UML for ontology modeling is that it is easy to understand for business experts. Another advantage is that there are many tools for creating and editing UML models that can be used for ontology modeling. However, UML and OCL have some limitations to represent an ontology. In the next section we briefly discuss these limitations.

2.1 UML and OCL Limitations

The main obstacle for using UML as an ontology modeling language is that the notion of Property in UML metamodel is not the same as the notion of Property in ontology [10]. *Property* in ontology corresponds to the notion of *association* in UML.

In addition, UML has not the appropriate support for describing restrictions. To represent constraints and restrictions into an UML diagram OCL could be used. However, it is difficult to translate axioms to other languages due to OCL has not explicit and unambiguous semantics. For example, a constraint may be encoded by several expressions [8].

Some approaches propose extending UML metamodel to tackle these problems adding the notion of *Property* and *Restriction* as UML *Classifier* [1]. The inclusion of this extension into UML specification causes an important impact on existing tools because it implies to change the data model of these tools.

Furthermore, UML does not provide a formal way to model relations between concepts from semantic point of view. For example, if we want to express that two concepts are synomyns we have to describe this relation by using stereotypes and add axioms with comments in natural language which is ambiguous.

So, instead of extending UML class diagrams to represent information semantics we propose to define a Metamodel based on MOF. In the next section, we analyze the propused metamodel and following an example we use it to model a semantics associated to an XML-based business document.

3. A Metamodel For Ontology Modeling

The use of ontologies to solve the semantic problem in business integration is not new. However, quite often ontologies are used as simple or structured vocabularies and in this role they do not provide any substantial benefit comparing to existing techniques (Omelayenko, 2002). The most commonly used definition, offered by Grüber (1993) states that: ''an ontology is an explicit specification of a conceptualization''. In this context, it should be clear that "explicit" object is a concrete, symbol-level object. But "conceptualization" is not clear and sometimes "conceptualization" is defined as an abstract model of some phenomenon in the world by having identified the relevant concepts of that phenomenon.

An ontology can be classified as either a lightweight or heavyweight ontology. On the one hand, lightweight ontologies include concepts, concept taxonomies, relationships between concepts and properties that describe concepts. On the other hand, heavyweight ontologies add axioms and constraints to lightweight ontologies [7].

In B2B area, it is common to view lightweight ontologies that describe simple taxonomies of products, catalogs and purchase orders. Our purpose is to define a metamodel that assists to business analysts in the modeling of more enrich ontologies. In order to define the metamodel we have used the Core Package of the "Unified Modeling Language: Infrastucture" specification, version 2 [17].

First, we have to make clear what we have in mind when we refer to ontology. An ontology can be defined as a set of concepts. Concepts imply a set of *terms* and *relations* between them. Furthermore, it is suitable to add *properties* and *axioms* to enrich the ontology. The set of properties define the characteristics of *terms*. *Axioms* are properties of the relation between ontology terms. For example, *PurchaseOrder* is a sub-class of *Order*, and this relation is not a symmetric one (axiom). Following we formalize this definition.

Definition 1. *An ontology O_i is a 4-tuple $<T_i, P_i, R_i, A_i>$ where:*

i identifies the domain or source that an ontology is associated with.
T_i is a set of terms t_j of O_i
P_i is a set of properties of terms $t_j \in T_i$
R_i is a set of relations between t_j and $t_x \in T_i$
A_i is a set of axioms that characterizes each relation of R_i

The typical role of a metamodel is to define the semantics for how a model elements in a model gets intantiated. The main design principles of the propused metamodel are: easy of use in rapid development of ontologies from business documents (standard or ad-hoc) and modularity.

In order to fill the modulariy design principle, the metamodel constructs were grouped into packages according to the elements needed to define an ontology. These packages are:

1. *Kernel Package*: contains classes and associations that form the kernel of the metamodel, wich are used by all other packages. This package imports and specializes elements from InfrastructureLibrary::Core [17].
2. *DataType Package*: contains classes and associations that can be used to create data types and data values for use in defining an ontology.
3. *Ontology Package*: contains classes and associations that can be used to define an ontology.
4. *Terms and Properties Package*: contains classes and associations that can be used to model terms and their properties.
5. *Relations Package*: contains classes and associations that can be used to model relations between terms belonging to an ontology.
6. *Axioms Package*: contains classes and associations that can be used to describe axioms about relations.

Following we define the last four packages and present some issues to model the semantics associated to a XML-based business document. This business document, showed in Figure 2, represents the information that a customer sends to the supplier in order to agree on a supply plan. The XML element *documentation* allows us to instantiate an Annotation class from the metamodel and to associate it to the corresponding element.

```
<xs:complexType name="PlanningSchedule">
    <xsd:element name = "Date" type = "xsd:date"/>
    <xsd:simpleType name = "Description">
      <xsd:restriction base = "xsd:string">
            <xsd:maxLength value = "40"/>
      </xsd:restriction>
   </xsd:simpleType>
   <xsd:element name = "Item" type = "Items"/>
   <xs:element name="ItemQuantity" type="Quantity"  minOccurs="0"/>
      <xs:annotation>
         <xs:documentation> The customer's   actual requested amount of the item to be shipped.
         </xs:documentation>
      </xs:annotation>
   </xs:element>
   <xsd:complexType name = "LocalAddress">
      <xsd:complexContent>
        <xsd:extension base = "Address">
           <xsd:element name= "zip" type = "xsd:string"/>
        </xsd:extension>
      </xsd:complexContent>
   </xsd:complexType>
</xsd:complexType>
```

Figure 2. XML-based business document.

3.1 Ontology Package

In Figure 3 are represented the classes and associations of the Ontology Package needed to model an ontology.

The *Element* is an abstract metaclass with no superclass. An element is a constituent of a model. The *NamedElement* is an abstract metaclass that represents elements that may have a name. The name is used for identification of the named element within the namespace in which it is defined. These classes are defined in the UML infrastructure library [17].

The main component of this package is *Ontology* class that includes definition of concepts used to describe and represent a domain. This class is associated with the class *OntologyElements* which is an abstract metaclass that gruops the objects of an ontology metamodel. If an ontology is removed, so are the elements owned by it. The association *imports* represents that an ontology could contains definitions whose meaning are defined in other ontologies. The association *prior_Version* identifies the referred ontology as a prior version of one ontology.

Each ontology element could be describe by a comment, represented by the *Comment* class. The *Body* atribute specifies a string that is the comment. This class intend to model, for example, the *xsd:annotations* elements from XML-based documents.

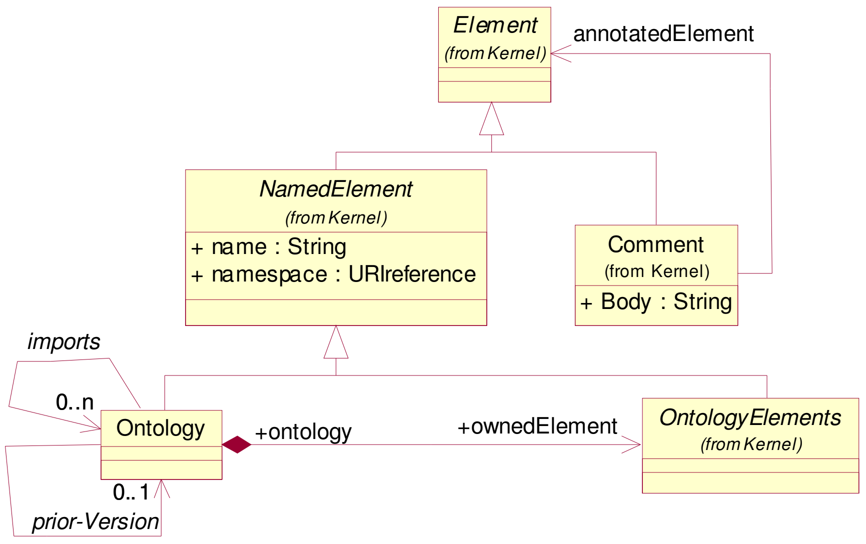


Figure 3. Ontology package.

3.2 Properties and Terms Package

Terms represent the set of concepts that a business analyst wants to represent in an ontology. A term can be simple or complex. A simple term has associated a value and a complex term is composed by other terms. We can state that $T_i = T_i^S \cup T_i^C$, where T_i^S is the set of simple terms and T_i^C is the set of complex terms.

Properties describe the features of a term. For example, allowed values, the number and other features of the values that a simple or complex term could take.

The metamodel that represents the relation between *Properties* and *Terms* is presented in Figure 4. In the proposed metamodel, the class *Properties* defines the features of a term so this class has to be associated at least one instance of *Terms*. It is indicated with the label 0..1 in the association end.

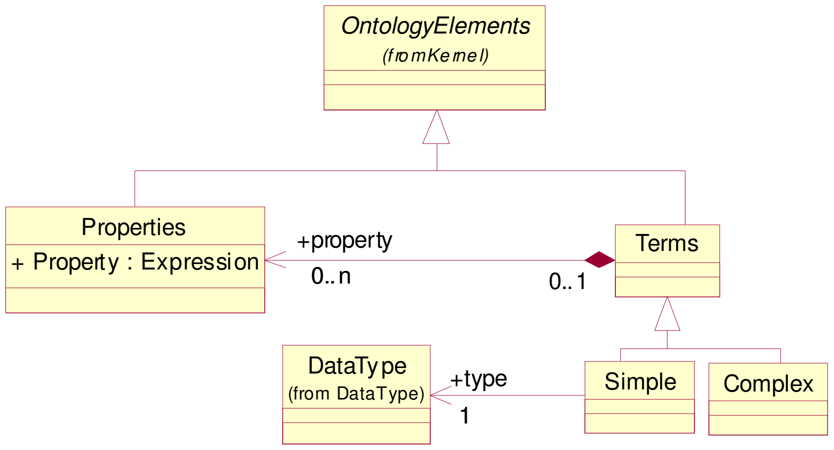


Figure 4. Properties and Terms Package.

Following we present some issues to model the terms belonging to a XML-based business document:

1. The *xsd:complexType* element has to be model by the *Complex* class. For example, from the document defined in Figure 2, line 1, we can model "**PlanningSchedule**" and "**LocalAddress**" terms as complex one.
2. The *xsd:simpleType* element has to be model by the *Simple* class. For example, from the document presented in Figure 2, line 3, we can model "**Description**" element as a simple term.
3. Then the elements defined by *xsd:element* tag could be simple or complex depending on its type definition. That is, if they are defined as a base xsd type, they are simple terms. For example, in line 2 the "**Date**" element is defined as "xsd:date" and in line 19 the "**zip**" element is defined as "xsd:string". Both elements have to be modeled as simple terms. Then, in line 8 and 9, the "**Item**" and "**ItemQuantiy**" elements are defined as *Items* and *Quantity* respectively. In order to determine if they are simple or complex terms we have to analyze if *Items* and *Quantity* have been defined as simple or complex terms. These difinitions are not in this document. So, we suppose that *Items* was defined as complex term and *Quantity* was defined as simple one.
4. Furthermore, in this document (line 3) the "**Description**" element is restricted by using xsd:restriction definition. This characteristic has to be modeled as a property of the "**Description**" term. That is, the xsd:restriction element has to be modeled by *Properties* class.

Figure 5 presents the model of the terms belonging to the XML-based document represented in Figure 2. This model was obtained by applying the issues defined aboved. The notation used to graphically represent this model is based on the UML notation by using UML stereotypes.

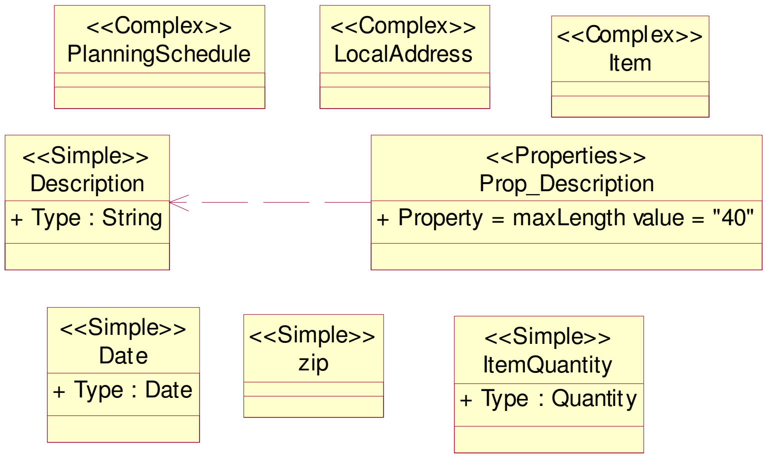


Figure 5. Model of business document terms.

3.3 Relations Package

Relations R_i represent how terms belonging to O_i are related to. Relations can be divided into hierarchical relations (R_i^H), conceptual relations (R_i^C) and particular relations (R_i^P). That is, $R_i = R_i^H \cup R_i^C \cup R_i^P$. We do not put any restrictions on the set R_i^P because it intends to model the relations defined by the ontology modeler. Whereas we require that $R_i^H = \{\text{is-a, part-of, inst-of}\}$. Furthermore, we propose that $R_i^C = \{\text{synonym, antonym}\}$, but it can be extended to add other conceptual characteristics.

The relations metamodel is presented in Figure 6. *Terms* and *Relations* classes are associated via the *RelationEnd* class. An instance of *Relations* class has to be associated at least with two instances of *RelationEnd* class, it is indicated with the label 2..n. *RelationEnd* class is associated with one *Terms* class and contains the information about cardinality and the role of terms. Furthermore, this class has the *Navigable* attribute to represent the direction of the relation.

One important ontologies requirement is its ability to structure the relations into hierarchies. That is, to define sub-relations of a relation. Furthermore, it is suitable to define equivalent relations and inverse relations. These are modeled by the relations *subrelationof*, *inversof* and *equivalentto*.

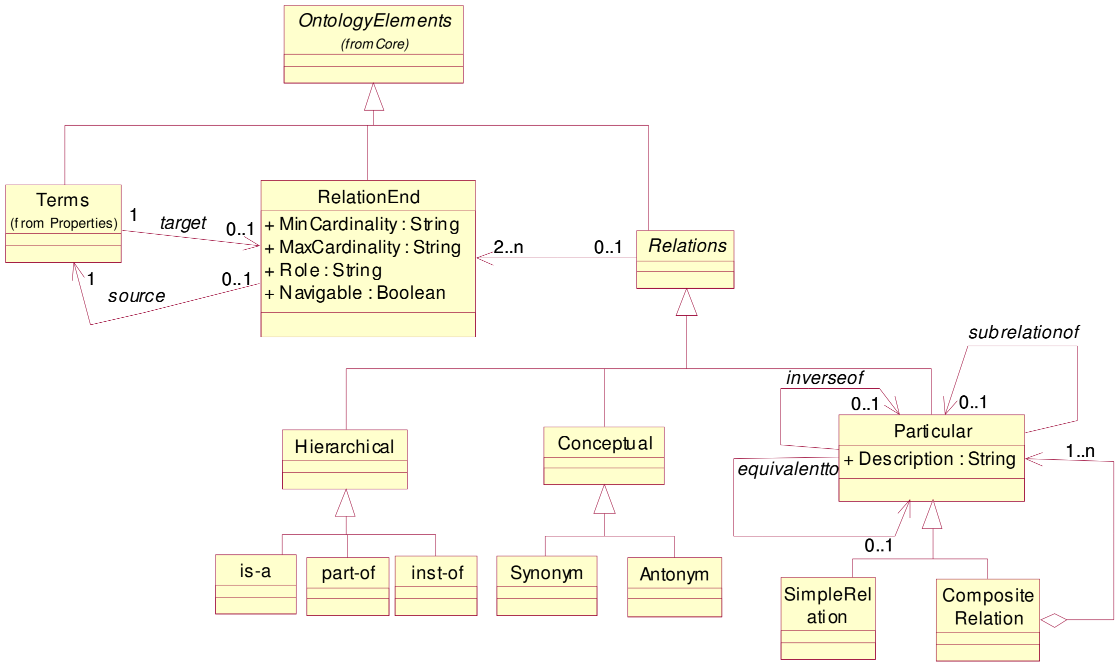


Figure 6. Relations Package.

The R_i^H set can be derived from an XML document [3]. Figure 7 represents the model of terms and relationships between them after applying the following rules to the XML business document presented in Figure 2.

1. The *xsd:extension* element represent the *is-a* relationship between terms. For example, in Figure 2 line 18, the <xsd:extension base = "**Address**"> definition state that the element previously defined (**LocalAddress**) *is-a* **Address**.

2. The combination of both *complexType* and *element* primitives represents the *part-of* relationship between terms. For example, all elements defined into the *complexType* primitive that define the **PlanningSchedule** term are related with it by the *part-of* relation. That is, **Date**, **Description**, **Items**, **ItemQuantity** and **LocalAddress** are *part-of* **PlannningSchedule**. Furthermore, **zip** is *part-of* **LocalAddress**.

3. The *element* primitive represents the *Inst-of* relation between terms.

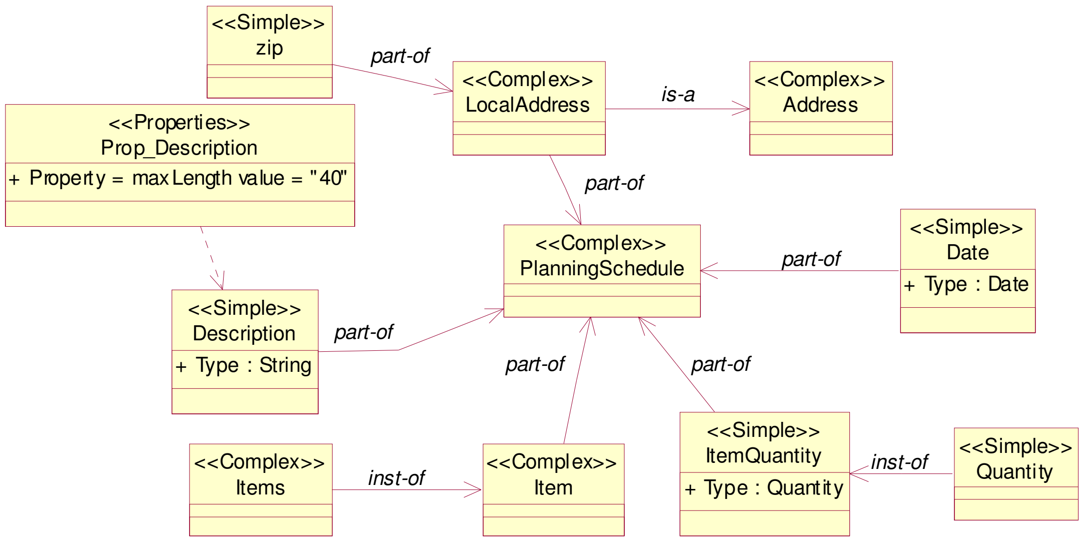


Figure 7. Model of terms and relationships between them.

3.4 Axioms Package

Axioms are properties of relations and they help to constraint the concepts interpretation. Furthermore, they provide guidelines for automated reasoning. In the knowledge engineering area, axioms have been represented using logic languages. In the UML class diagram, axioms could be expressed by OCL. For example, OCL constraints have to be used to declare a transitive property of a relation between terms. However, describing such constrains may involve writing moderately complex OCL expressions that are not immediately understandable to a human reader. In addition, there may be several different expressions encoding the same constraint. An interesting issue is to represent axioms as objects [16].

Axioms can be divided into several subsets. In this work we define $A_i = A_i^{RA} \cup A_i^{P}$, where A_i^{RA} represents the set of axioms for relational algebra and A_i^{P} represents the set of particular axioms, which are defined by the users. We do not put any restrictions on the set A_i^{P}, whereas we required that A_i^{RA}={symmetric, reflexive, transitive, functional}.

Figure 8 represents the metamodel for modeling axioms and their association with the *Relations* class. The *AParticular* class is related to *Period* class for modeling temporal axioms.

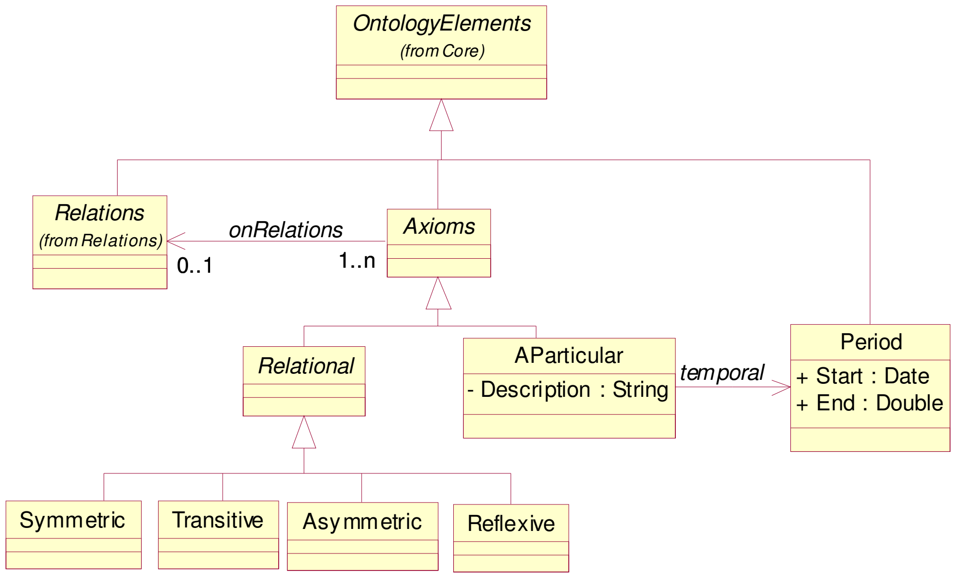


Figure 8. Axioms Package.

4. CONCLUSIONS AND FUTURE DIRECTIONS

To effectively carry out a collaborative B2B relationship, enterprises need to unambigously understand the business information that they interchange. To do that, they have to implement an ontology in order to represent the semantics associated to this information.

The semantics associated to B2B specifications has to evolve by moving along the semantic continuum from implicit semantics to formal semantics in order to support an effective information processing. The metamodel defined in this paper allow to people involved in a B2B relationship modeling the semantic associated to XML-based business documents. The concepts defined in this metamodel for modeling ontology can be used as stereotypes in UML. Furthermore, we have defined a set of rules to model the semantics implicit in XML-based business documents.

The future directions will be focus on the mapping between this metamodel and the ontology specification languages. There are traditional ontology specification languages and other languages created in the context of Internet that exploit the characteristics of the Web. Such languages are usually called *web-based ontology languages* or *ontology markup languages*. These languages are still in a development phase: they are continuously evolving [7].

The more recently defined language is OWL Web Ontology Language [13] which is a W3C (World Wide Web Consortium) proposed recommendation. So, we are interesting in the translation problem between the metamodel propused in this paper for ontology modeling and this language.

REFERENCES

[1] Baclawski, K., Kokar, M., Kogut, P., Hart, L., Smith, J., Holmes,W., Letkowski, J., Aronson, M.: Extending UML to Support Ontology Engineering for the SemanticWeb. In *Fourth International Conference on UML*, Toronto (2001)

[2] Berners-Lee, T.; Hendler, J. and Lassila, O. The Semantic Web. Scientific American **284** (May 2001), 35–43. Available on-line http://www.scientificamerican.com

[3] Bouquet, P, Dona, A, Serafini, L and Zanobini, S. Contextualized local ontologies specification via CTXML. AAAI-02 Workshop on Meaning Negotiation (MeaN-02) July 28, 2002, Edmonton, Alberta, Canada. Available on-line http://sra.itc.it/people/serafini/distribution/aaai-ws-ctxml.pdf

[4] Caliusco, Ma. Laura, Galli, Ma. Rosa and Chiotti, Omar. Ontology and XML-based specifications for collaborative B2B relationships. In *Proceeding of III Jornadas Iberoamericanas de Ingeniería de Software e Ingeniería de Conocimiento (JIISIC)*. Valdivia (Chile). November 2003.

[5] Caliusco, Ma. Laura, Galli, Ma. Rosa and Chiotti, Omar. Ontology Role in Collaborative Business-to-Business Relationships. In *Proceeding of the XXIX Conferencia Latinoamericana de Informática (CLEI 2003)*. La Paz, Bolivia. September, 2003.

[6] Carlson, D. Modeling XML Applications with UML – Practical e-Business Applications. Addison Wesley, 2001.

[7] Corcho, O.; Fernández-López, M.; Gómez-Pérez, A. *Methodologies, tools and Languages for building ontologies*. Where is the meeting point? Data and Knowledge Engineering. 46 (2003) 41–64.

[8] Cranefield, S., Haustein, S. and Purvis, M.. UML as an ontology modelling language. In *Proceedings of the Workshop on Ontologies in Agent Systems, 5th International Conference on Autonomous Agents, Montreal, Canada, 2001.*

[9] Duric, D.; Gasevic, D; Devedzic, V. *A MDA-based Approach to the Ontology Definition Metamodel*. Proceedings of a 4TH Workshop On Computational Intelligence And Information Technologies. October 13, 2003, Faculty of Electronics, Niš, Serbia.

[10] Falkovych, K., Sabou, M., Stuckenschmidt, H.. *UML for the Semantic Web: Transformational Approaches* in B. Omelayenko and M. Klein: Knowledge Transformation for the Semantic Web, IOS Press, 2003.

[11] Grüber, T.R. (1993) A translation approach to portable ontology specification, *Knowledge Acquisition* 5 199–220.

[12] Klein, M. *Interpreting XML via an RDF schema*. chapter in *Knowledge Annotation for the Semantic Web* IOS Press, Amsterdam, 2003.

[13] McGuinness, D and van Harmelen, F. OWL Ontology Web Language – Overview. W3C Recomended Propesed. December 2003. Available on-line http://www.w3.org/TR/owl-features/

[14] Object Management Group. Unified Modeling Language. http://www.uml.org

[15] Omelayenko B. Ontology-Mediated Business Integration. In *Proceedings of the 13-th EKAW 2002 Conference*, Siguenza, Spain, October 1-4, LNAI 2473, 2002, pp. 264-269 © Springer-Verlag

[16] Staab, S; Mädche, A.. Axioms are objects, too - Ontology Engineering Beyond the Modeling of Concepts and Relations. In: V.R. Benjamins, A. Gomez-Perez, N. Guarino (eds.). *Proceedings of the ECAI 2000 Workshop on Ontologies and Problem-Solving Methods*. Berlin, August 21-22, 2000.

[17] UML 2.0 Infrastructure – Final Adopted Specification. September, 2003.

[18] Uschold, M. *Where is the semantics in the Semantic Web?* **AI Magazine.** Volume 24 , Issue 3 (September 2003) Pages: 25 – 36. ISSN:0738-4602. Disponible on-line lsdis.cs.uga.edu/SemWebCourse_files/WhereAreSemantics-AI-Mag-FinalSubmittedVersion2.pdf (2003).