Formalism-Independent Specification of Ontology Mappings - A Metamodeling Approach

Saartje Brockmans[1], Peter Haase[1], Heiner Stuckenschmidt[2]

[1] AIFB, Universität Karlsruhe (TH), Germany
[2] University of Mannheim, Germany

Abstract. Recently, the advantages of metamodeling as a foundation for the graphical specification of ontologies have been recognized by the semantic web community. This has lead to a number of activities concerned with the development of graphical modeling approaches for the Web Ontology Language based on the Meta Object Facility (MOF) and the Unified Modeling Language (UML). An aspect that has not been addressed by these approaches so far is the need to specify mappings between heterogenous ontologies. With an increasing number of ontologies being available, the problem of specifying mappings is becoming more important and the rationales for providing model based graphical modeling support for mappings is the same as for the ontologies themselves. In this paper, we propose a MOF-based metamodel for mappings between OWL DL ontologies as well as a UML based graphical language for modeling mappings independent of a specific mapping formalism.

1 Motivation

Initially, ontologies have been introduced as a solution for the problem of semantic heterogeneity and as a facilitator of semantic integration. With the increasing use of ontologies, however, it has turned out that the problem of semantic integration has only been lifted to a different level of abstraction at which different ontologies describing the same domain have to be aligned. There are two main lines of research addressing the problem of ontology alignment. The first line is concerned with the development of heuristic and statistical methods for identifying semantic relations between elements in different ontologies. Examples of such methods are [6, 7, 10, 11]. The second line of research is concerned with formalisms for encoding and using semantic relations (mappings) between ontologies. These formalisms are often based on non-standard extensions of the logics used to encode the ontologies. Examples of such mapping formalisms are [1, 3, 4, 8]. In a recent comparison of these approaches, it has been shown that these approaches are mostly orthogonal in terms of assumptions made about the right interpretation of mapping relations [14]. This means that the approaches cover a large variety of possible interpretations of semantic relations, but it also means that they are incompatible with each other and that the choice of a particular formalism is an important decision with significant influence on remaining options for interpreting and using mappings. Further, making the right decision with respect to a mapping formalism requires in depth knowledge of the corresponding logics and the hidden assumptions made as well as the specific needs of the application.

1.1 Problem Definition

In order to make an informed decision about which mapping formalism to use, this decision should be made as late as possible in the modeling process because it is often not possible to decide whether a given mapping formalism is suitable for specifying all relevant connections. Therefore, mappings should first be specified on a purely informal level by just marking parts of the ontologies that are somehow semantically related. In a next step, the kind of semantic relation that exists between the elements should be specified. A complete set of such semantic relations between elements in the different models provides a specification of the kinds of expressions the mapping formalism has to support. In order to support this process, we need a formalism-independent format for specifying mappings. On the other hand, we have to make sure that concrete mapping representations can be derived automatically from this model in order to support the implementation and use of the mappings. In order to meet these requirements, we propose a metamodel based approach to specifying ontology mappings independent on the concrete mapping formalism. In particular, we propose a Meta Object Facility-based metamodel for describing mappings between OWL DL ontologies as well as a UML profile that defines a graphical format for mapping modeling. As a starting point, we use previous work on a meta-modeling approach for OWL DL [2].

1.2 Related Work and Contributions

There is some related work on meta-modeling and formalism independent modeling of mappings between conceptual models. Omelayenko introduces a model for specifying relations between heterogeneous RDF schema models for the purpose of data transformation in e-commerce [13]. The idea is to construct a separate RDF model that defines the relations in terms of so-called bridges. These bridges are accompanied by transformations that execute the translation. Maedche and others [9] describe an approach that is very similar to the one of Omelayenko. They also define 'bridges' between elements of the different models and add transformation descriptions. As in the work of Omelayenko, the semantics of the bridges is only specified in terms of an RDF schema. The *mapping ontology* by Crubézy and colleagues [5] defines the structure of specific mappings and the transformation functions to transfer instances from one ontology to another. This ontology can then be used by tools to perform the transformations. The ontology provides different ways of linking concepts from the source ontology to the target ontology, transformation rules to specify how values should be changed, and conditions and effects of such rules. Our work extends and improves these approaches with respect to various aspects:

- Our approach addresses state of the art standards in the area of ontology technology, in particular OWL and rule extensions.
- We base our meta-modeling on widely used standards in the area of model-driven architectures, in particular MOF and UML.
- Our approach includes new insights about hidden assumptions of ontology mapping formalisms and can therefore more easily be linked to different formalisms for the sake of implementing modeled mappings.

1.3 Outline

In Section 2, we summarize existing proposals of mapping formalisms and explain the different aspects in which these approaches can vary. Section 3 introduces the principles of metamodeling and model-driven architecture that provides the basis for our work. In Section 4, we propose a MOF-based metamodel for OWL DL ontology mappings and explain its relation to the metamodels for OWL DL and rule extensions that have been proposed in our previous work. Section 5 presents a graphical notation for ontology mappings in terms of a UML profile that specifies the presentation of the different metamodel components based on existing profiles for OWL DL and rule extensions. We close with a discussion of the approach and future work.

2 Ontology Mapping Formalisms

In contrast to the area of ontology languages where the Web Ontology Language OWL has become a de facto standard for representing and using ontologies, there is no agreement yet on the nature and the right formalism for defining mappings between ontologies. In a recent discussion on the nature of ontology mappings, some general aspects of mapping approaches have been identified [15]. We briefly discuss these aspects in the following and clarify our view on mappings that is reflected in the proposed metamodel with respect to these aspects.

2.1 What do mappings define ?

In this paper, we restrict our attention to declarative mapping specifications. In particular, we see mappings as axioms that define a semantic relation between elements in different ontologies.

A number of different kinds of semantic relations have been proposed. Most common are the following kinds of semantic relations:

Equivalence ($\equiv$) Equivalence states that the connected elements represent the same aspect of the real world according to some equivalence criteria. A very strong form of equivalence is equality, if the connected elements represent exactly the same real world object.

Containment ($\sqsubseteq, \sqsupseteq$) Containment states that the element in one ontology represents a more specific aspect of the world than the element in the other ontology. Depending on which of the elements is more specific, the containment relation is defined in the one or in the other direction.

Overlap (o) Overlap states that the connected elements represent different aspects of the world, but have an overlap in some respect. In particular, it states that some objects described by the element in the one ontology may also be described by the connected element in the other ontology.

In some approaches, these basic relations are supplemented by their negative counterparts. The corresponding relations can be used to describe that two elements are *not* equivalent ($\not\equiv$), *not* contained in each other ($\not\sqsubseteq$) or *not* overlapping or disjoint respectively ($\emptyset$). Adding these negative versions of the relations leaves us with eight semantic relations that cover all existing proposals for mapping languages. In addition to the type of semantic relation, an important distinction is whether the mappings are to be interpreted as extensional or as intensional relationships.

Extensional In extensional mapping definitions, the semantic relations are interpreted as set-relations between the sets of objects represented by elements in the ontologies. Intuitively, elements that are extensionally the same have to represent the same set of objects.

Intensional In the case of intensional mappings, the semantic relations relate the elements directly, i.e. considering the properties of the element itself. In particular, if two elements are intensionally the same, they refer to exactly the same real world object.

2.2 What do mappings preserve ?

It is normally assumed that mappings preserve the 'meaning' of the two models in the sense that the semantic relation between the intended interpretations of connected elements is the one specified in the mapping. A problem with this assumption is that it is virtually impossible to verify this property. Instead, there are a number of verifiable formal properties that mappings can be required to satisfy. Examples of such formal properties are the satisfiability of the overall model, the preservation of possible inferences or the preservation of answers to queries. Often, such properties can only be stated relative to a given application context, such as a set of queries to be answered or a set of tasks to be solved.

The question of what is preserved by a mapping is tightly connected to the hidden assumptions made by different mapping formalisms. A number of important assumptions that influence this aspect have been identified and formalized in [14]. A first basic distinction concerns the relationship between the sets of objects (domains) described be the mapped ontologies. Generally, we can distinguish between a global domain and local domain assumption:

Global Domain It is assumed that both ontologies describe exactly the same set of objects. As a result, semantic relations are interpreted in the same way as axioms in the ontologies. There are special cases of this assumption, where one ontology is regarded as a 'global schema' and describes the set of all objects, other ontologies are assumed to describe subsets of these objects.

Local Domains It is not assumed that ontologies describe the same set of objects. This means that mappings and ontology axioms normally have different semantics. There are variations of this assumption in the sense that sometimes it is assumed that the sets of objects are completely disjoint and sometimes they are assumed to overlap each other.

These assumptions about the relationship between the domains is especially important for extensional mapping definitions, because in cases where two ontologies do not talk about the same set of instances, the extensional interpretation of a mapping is problematic as classes that are meant to represent the same aspect of the world can have disjoint extensions. In such cases, e.g. in C-OWL [1], the relationship is not defined directly as a set relationship between the extensions of the concepts, but can be defined in terms of domain relations that connect the interpretation domains by codifying how elements in one domain map into elements of the other domain.

Other assumptions made by approaches concerns the use of unique names for objects - this assumption is often made in the area of database integration - and the preservation of inconsistencies across mapped ontologies. In order to make an informed choice about which formalism to use, these assumptions have to be represented by the modeler and therefore need to be part of the proposed metamodel.

2.3 What do mappings connect ?

In the context of this work, we decided to focus on mappings between ontologies represented in OWL DL. This restriction makes it much easier to deal with this aspect of ontology mappings as we can refer to the corresponding metamodel for OWL DL specified in [2]. In particular, the metamodel contains the class `OntologyElement`, that represents an arbitrary part of an ontology specification. While this already covers many of the existing mapping approaches, there are a number of proposals for mapping languages that rely on the idea of view-based mappings and use semantic relations between (conjunctive) queries to connect models, which leads to a considerably increased expressiveness.

2.4 How are mappings organized ?

The final question is how mappings are organized. They can either be part of a given model or be specified independently. In the latter case, the question is how to distinguish between mappings and other elements in the models. Mappings can be uni- or bidirectional. Further, it has to be defined whether a set of mappings is normative or whether it is possible to have different sets of mappings according to different applications, viewpoints or different matchers.

In this work, we use a mapping architecture that has the greatest level of generality in the sense that other architectures can be simulated. In particular, we make the following choices:

- A mapping is a set of mapping assertions that consist of a semantic relation between mappable elements in different ontologies
- Mappings are first-class objects that exist independent of the ontologies. Mappings are directed and there can be more than one mapping between two ontologies

These choices leave us with a lot of freedom for defining and using mappings. Approaches that see mappings as parts of an ontology can be represented by the ontology and a single mapping. If only one mapping is defined between two ontologies, this can be seen as normative, and bi-directional mappings can be described in terms of two directed mappings.

3 Metamodeling with MOF

This section introduces the essential ideas of the Meta Object Facility (MOF, [12]) and shows how a metamodel and a UML profile for ontology mappings fit into this more general picture. Next to this, we introduce our existing metamodels and UML profiles for OWL DL ontologies. We have shown in previous work that UML methodology, tools and technology are a feasible approach for supporting the development and maintenance of ontologies and rules. We argue that it would be a good support for the modeling and maintenance of ontology mappings as well. Our proposed metamodel and UML profile for mappings extend our existing metamodels and profiles.

3.1 Meta Object Facility

The Meta Object Facility (MOF) is an extensible model driven integration framework for defining, manipulating and integrating metadata and data in a platform independent manner. The goal is to provide a framework that supports any kind of metadata, and that allows new kinds to be added as required. MOF plays a crucial role in the four-layer metadata architecture of the Object Management Group (OMG). The bottom layer of this architecture encompasses the raw information to be described. The model layer, which is the second layer, contains the definition of the required structures to describe the model for the given domain. The metamodel, which is situated on the third layer, defines the terms in which the model is expressed. Finally, the MOF constitutes the top layer, also called the meta-metamodel layer. Note that the top MOF layer is hard wired in the sense that it is fixed, while the other layers are flexible and allow to express various metamodels such as the UML metamodel or, in our case, the metamodel for ontology mappings.

The general idea of using MOF-based metamodels and UML profiles for the purpose of modeling ontologies and ontology mappings is depicted in Figure 1: A metamodel for OWL ontology mappings as well as a UML profile are grounded in MOF, in that they are defined in terms of the MOF meta-metamodel. The UML profile mechanism is an extension mechanism to tailor UML to specific application areas. Our proposed UML profile defines a visual notation for optimally supporting the specification of OWL ontology mappings. This visual syntax is based on the metamodel and is independent of a concrete mapping formalism. Mappings in both directions between the metamodel and the profile have to be established.

Ontology mappings in a concrete mapping language instantiate the metamodel. The constructs of the mapping languages have a direct correspondence with those of the metamodel. Analogously, specific UML models instantiate the UML profile. Within the MOF framework, the UML models are translated into concrete mapping definitions based on the above mappings between the metamodel and the UML profile. In this translation step, i.e. after the visual modeling of the mappings, the decision about a concrete mapping formalism is taken, based on the types of the mappings which were modeled.

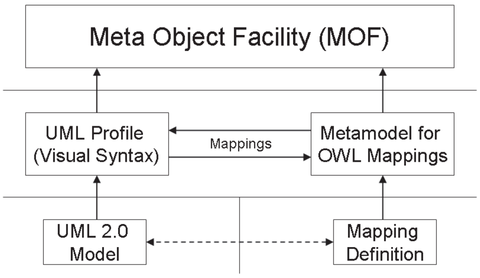

Fig. 1. How a metamodel and a UML profile for ontology mappings fit into the picture of the Meta Object Facility Framework

3.2 A Metamodel for OWL DL Ontologies

We now review our previous work on a metamodel for OWL DL. Figure 2 shows the central part of the OWL DL metamodel. Among others, it shows that every element of an ontology is a subclass of the class `OntologyElement` and hence a member of an `Ontology`. The diagram of Figure 2 is the main part of the OWL DL metamodel but does by far not represent it fully. The metamodel is, just like OWL DL itself, a lot more extensive. However, for lack of space, we refer to [2] for a full specification. Additionally, the metamodel is augmented with constraints, expressed in the Object Constraint Language ([17]), specifying invariants that have to be fulfilled by all models that instantiate the metamodel.

Our metamodel for OWL DL ontologies ([2]) has a one-to-one mapping to the abstract syntax of OWL DL and thereby to its formal semantics.

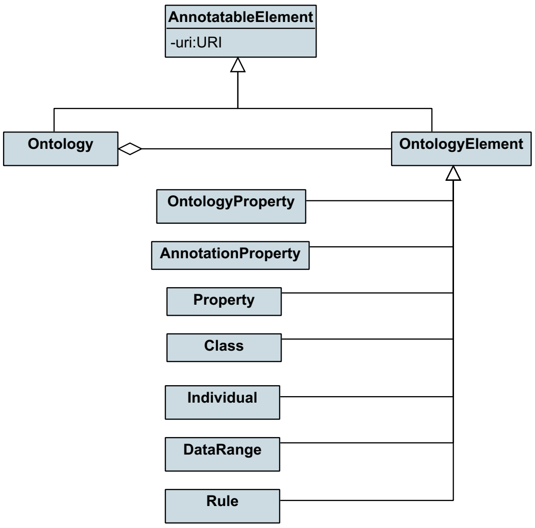

Fig. 2. Main Elements of the Ontology Definition Metamodel

Further, we have defined a metamodel for rule extensions of OWL DL. For the details, we refer the reader to [2]. In our mapping metamodel, we reuse parts of the rule metamodel, as we explain in detail in Section 4.

3.3 A UML Profile for OWL DL Ontologies

Our UML profile is faithful to UML2 as well as to OWL DL, with a maximal reuse of features from the languages. Since the UML profile mechanism supports a restricted form of metamodeling, our proposal contains a set of extensions and constraints to UML2. This tailors UML2 such that models instantiating the OWL DL metamodel can be defined. Our UML profile has a basic mapping, from OWL class to UML class, from OWL property to binary UML association, from OWL individual to UML object, and from OWL property filler to UML object association. Extensions to UML2 consist of custom UML-stereotypes, which usually carry the name of the corresponding OWL DL language element, and dependencies.

Figure 3 shows a small example of an ontology. It contains the definition of two classes `Article` and `Book` as subclasses of `Publication`. The two are defined to be disjoint. The ontology contains another class `Person` and its subclass `Researcher`. An association between `Publication` and `Person` denotes the object property `authorOf`, from which domain and range are defined via an association class. Furthermore, the ontology contains several instances of its

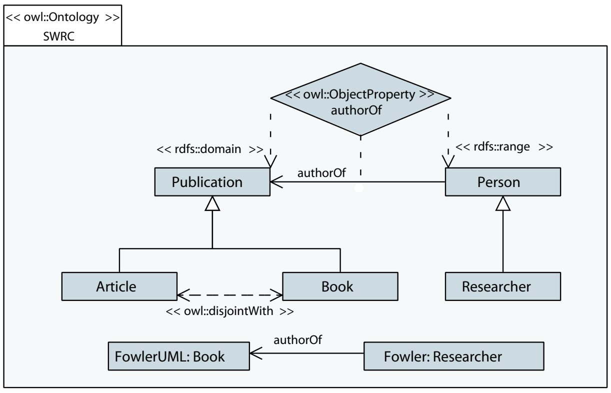

Fig. 3. Sample Ontology Depicted using the UML Profile for the ODM

classes and object property. For a discussion of all details of the UML profile for OWL DL ontologies, we refer to [2].

4 A Metamodel for Ontology Mappings

We propose a formalism-independent metamodel for OWL ontology mappings. The metamodel is a consistent extension of our earlier work on metamodels for OWL DL ontologies and SWRL rules, as explained in Section 3.2. It has constraints defined in OCL [17] as well, which we omit here due to lack of space and instead refer to [2] for a complete reference.

Figure 4 shows the metamodel for mappings. In the figures, darker grey classes denote classes from the metamodels of OWL DL and rule extensions. The central class in the metamodel is the class `Mapping`, having four attributes. The URI, defined by the attribute `uri`, allows to uniquely identify a mapping and refer to it as a first-class object. The assumptions about the use of unique names for objects and the preservation of inconsistencies across mapped ontologies, are defined through the boolean attributes `uniqueNameAssumption` respectively `inconsistencyPreservation`. For the assumptions about the domain, we defined an attribute `DomainAssumption`. This attribute may take specific values that describe the relationship between the connected domains: overlap, containment (in one of the two directions) or equivalence.

A mapping is always defined between two ontologies. An ontology is represented by the class `Ontology` in the OWL DL metamodel. Two associations from `Mapping` to `Ontology`, `sourceOntology` and `targetOntology`, specify the source respectively the target ontology of the mapping. Cardinalities on both associations denote that to each `Mapping` instantiation, there is exactly one `Ontology` connected as source and one as target.

A mapping consists of a set of mapping assertions, denoted by the MOF aggregation relationship between the two classes `Mapping` and `MappingAssertion`.

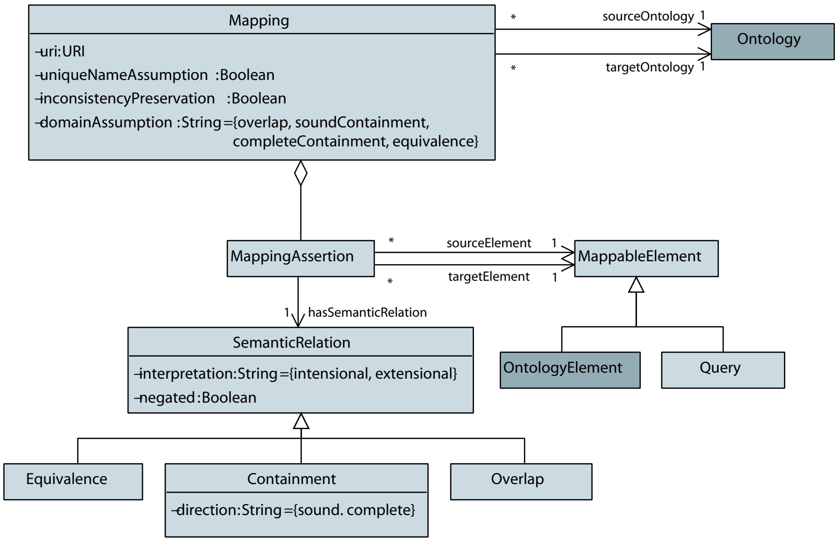

Fig. 4. Metamodel for ontology mappings

The elements that are mapped in a `MappingAssertion` are defined by the class `MappableElement`. A `MappingAssertion` is defined through exactly one `SemanticRelation`, one source `MappableElement` and one target `MappableElement`. This is defined through the three associations starting from `MappingAssertion` and their cardinalities.

In Section 2.1, we have introduced four semantic relations along with their logical negation to be defined in the metamodel. Two of these relationship types are directly contained in the metamodel through the subclasses `Equivalence` and `Overlap` of the class `SemanticRelation`. The other two, containment in either direction, are defined through the subclass `Containment` and its additional attribute `direction`, which can be `sound` ($\sqsubseteq$) or `complete` ($\sqsupseteq$).

The negated versions of all semantic relations are specified through the boolean attribute `negated` of the class `SemanticRelation`. The other attribute of `SemanticRelation`, `interpretation`, defines whether the mapping assertion is assumed to be interpreted intentionally or extensionally.

As discussed in Section 2, a mapping assertion can connect two mappable elements, which may ontology elements or queries. To support this, `MappableElement` has two subclasses `OntologyElement` and `Query`. The former is previously defined in the OWL DL metamodel, as shown in Figure 2. The class `Query` reuses constructs from the SWRL metamodel. The reason for reusing large parts of the rule metamodel lies in the fact that conceptually rules and queries are of very similar nature [16]: A rule consists of a rule body (antecedent) and rule head (consequent), both of which are conjunctions of logical atoms. A query can be considered as a special kind of rule with an empty head. The distinguished variables specify the variables that are returned by the query. Informally, the an-

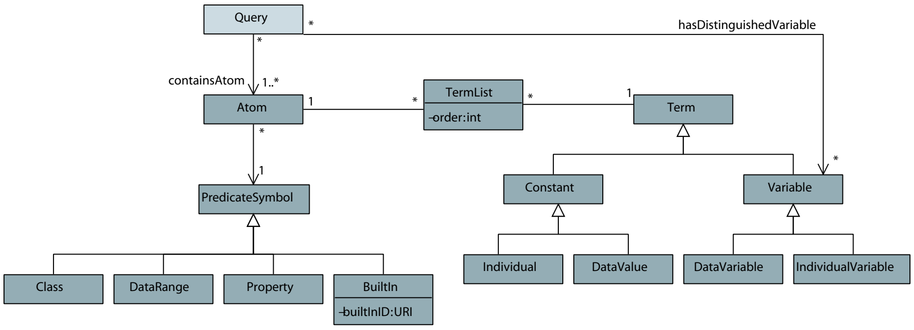

Fig. 5. Metamodel for ontology mappings - definition of a query

swer to a query consists of all variable bindings for which the grounded rule body is logically implied by the ontology. Figure 5 shows this connection and shows how a `Query` is composed. They depict how atoms from the antecedent and the consequent of SWRL rules can be composed. Similarly, a `Query` also contains a `PredicateSymbol` and some, possibly just one, `Terms`. We defined the permitted predicate symbols through the subclasses `Class`, `DataRange`, `Property` and `BuiltIn`. Similarly, the four different types of terms are specified as well. The UML association class `TermList` between `Atom` and `Term` allows to identify the order of the atom terms. Distinguished variables of a query are differentiated through an association between `Query` and `Variable`.

5 A UML Profile for Ontology Mappings

This section describes the UML profile as a visual notation for specifying ontology mappings, based on the metamodel discussed in Section 4. Our goal, as described in Section 1.1, is to allow the user to specify mappings without having decided yet on a specific mapping language or even on a specific semantic relation. This is reflected in the proposed visual syntax which is, like the metamodel, independent from a concrete mapping formalism. The UML profile is consistent with the design considerations taken for the previously defined UML profiles for OWL ontologies and rule extensions.

First of all, users specify two ontologies between which they want to define mappings. The visual notation for this as defined in our profile, is presented in Figure 6. Just as for ontologies as collections of ontology elements, we apply the UML grouping construct of a package to represent mappings as collections of mapping assertions. Attributes of the mapping, like the domain assumption, are represented between curly brackets.

In Figure 7, a source concept `Publication` is defined to be more specific than the target concept `Entry`. The example in Figure 9 relates two properties `authorOf` and `creatorOf` using an extensional containment relationship. Figure 8 models `Researcher Fowler` and `Author MartinFowler` as two equivalent instances.

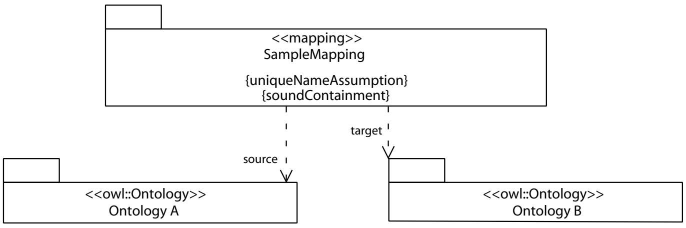

Fig. 6. Visual notation for a mapping between two ontologies

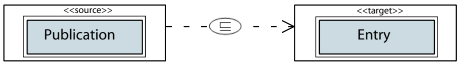

Fig. 7. Sample containment relation between two concepts

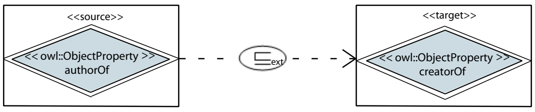

Fig. 8. Sample extensional containment relation between two properties

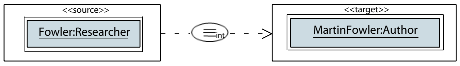

Fig. 9. Sample intensional equivalence relation between two individuals

Both source and target elements of mapping assertions are represented in a box, connected to each other via a dependency with the corresponding symbol of the semantic relation. In the first step of the process, when users just mark elements being semantically related without specifying the type of semantic relation, the dependency does not carry any relation symbol. Stereotypes in the two boxes denote source- and target ontology. Like defined in the metamodel, these mapped elements can be any element of an ontology (metaclass `OntologyElement`) or a query (metaclass `Query`). They are represented like defined in the UML profile for OWL and rules. The parts of the mappable elements which are effectively being mapped to each other, are denoted via a double-lined box, which becomes relevant if the mapped elements are more complex constructs, as explained in the following.

A more complex example mapping assertion is pictured in Figure 10. The example defines that the union of the classes `PhDThesis` and `MasterThesis`, is equivalent to the class `Thesis`.

Figure 11 shows another example of an equivalence relation between two expressions. It specifies that the class which is connected to the class `Publication` via a property `authorOf` with the `someValuesFrom` restriction, is equivalent to the class `Author`.

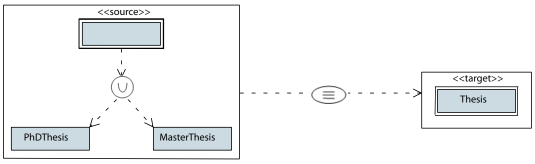

Fig. 10. Sample equivalence relation between complex class descriptions

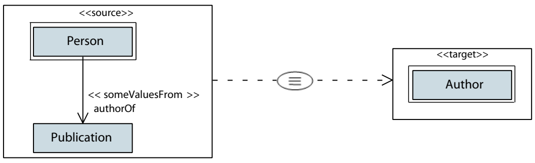

Fig. 11. Sample equivalence relation between complex class descriptions

Figure 12 shows an example of an equivalence relation between two queries. The first query is about a Publication X with a Topic Y named Z. The target query is about an Entry X with subject Z. The mapping assertion defines the two queries to be equivalent. The effective correspondences are established between the the two distinguished variables X and Z, which are again denoted with a double-lined box.

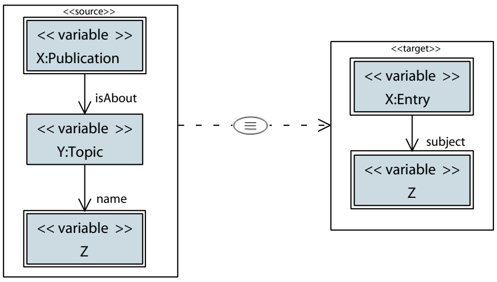

Fig. 12. Sample equivalence relation between two queries

6 Discussion

We have presented a MOF based metamodel as well as a UML profile to support formalism independent graphical modeling of mappings between OWL ontologies. The model ties in with previous work on similar metamodels for OWL DL and rule extensions. While the work presented addresses an important gap in the existing modeling infrastructure, it has to be seen as the basis for a more complete framework for mapping modeling based on MOF and UML. In particular, we only considered a very abstract metamodel that was designed to cover a wide range of existing formalisms for specifying mappings. In order to be able to provide support not only for the acquisition of mappings but also for their implementation in one of the existing formalisms, three additional steps have to be taken.

1. In a first step, we have to link the abstract metamodel presented in this paper to concrete mapping formalisms. This can best be done by creating specializations of the generic metamodel that correspond to individual mapping formalisms. This normally means that restrictions are added to the metamodel in terms of OCL constraints that formalize the specific properties of the respective formalism.
2. In a second step, we have to develop a method for checking the compatibility of a given graphical model with a particular specialization of the metamodel. This is necessary for being able to determine whether a given model can be implemented with a particular formalism. Provided that specializations are entirely described using OCL constraints, this can be done using an OCL model checker.
3. Finally, we have to develop methods for translating a given graphical model into an appropriate mapping formalism. This task can be seen as a special case of code generation where instead of executable code, we generate a formal mapping model that can be operationalized using a suitable inference engine.

In summary, the work presented here is the first step towards a comprehensive, model based approach for modeling and implementing ontology mappings. In contrast to many existing proposals, this approach takes a knowledge-level perspective on mapping modeling and supports an iterative development process where the mapping model is refined in a stepwise manner and the decision for a specific implementation formalism is only taken later in the process.

References

1. P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini, and H. Stuckenschmidt. C-OWL: Contextualizing ontologies. In *Second International Semantic Web Conference ISWC'03*, volume 2870 of *LNCS*, pages 164–179. Springer, 2003.
2. Saartje Brockmans and Peter Haase. A Metamodel and UML Profile for Networked Ontologies – A Complete Reference. Technical report, Universität Karlsruhe, April 2006. http://www.aifb.uni-karlsruhe.de/WBS/sbr/publications/ontology-metamodeling.pdf.
3. D. Calvanese, G. De Giacomo, and M. Lenzerini. A framework for ontology integration. In *Proceedings of the Semantic Web Working Symposium*, pages 303–316, Stanford, CA, 2001.
4. Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Description logics for information integration. In A. Kakas and F. Sadri, editors, *Computational Logic: Logic Programming and Beyond*, volume 2408 of *Lecture Notes in Computer Science*, pages 41–60. Springer, 2002.

5. M. Crubézy and M. A. Musen. Ontologies in support of problem solving. In S. Staab and R. Studer, editors, *Handbook on Ontologies*, pages 321–342. Sringer, 2003.
6. Marc Ehrig and Steffen Staab. QOM - Quick Ontology Mapping. In *3rd International Semantic Web Conference (ISWC2004)*, Hiroshima, Japan, 2004.
7. Fausto Giunchiglia, Pavel Shvaiko, and Mikalai Yatskevich. Semantic matching. In *1st European semantic web symposium (ESWS'04)*, pages 61–75, Heraklion, Greece, 2004.
8. Peter Haase and Boris Motik. A mapping system for the integration of owl-dl ontologies. In *In Proceedings of the ACM-Workshop: Interoperability of Heterogeneous Information Systems (IHIS05)*, November 2005.
9. Alexander Maedche, Boris Motik, Nuno Silva, and Raphael Volz. MAFRA - a mapping framework for distributed ontologies. In *Proceedings of 13th European Conference on Knowledge Engineering and Knowledge Management (EKAW)*, Siguenza, Spain, 2002.
10. S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *18th International Conference on Data Engineering (ICDE-2002)*, San Jose, California, 2002. IEEE Computing Society.
11. N. F. Noy and M. A. Musen. The PROMPT suite: Interactive tools for ontology merging and mapping. *International Journal of Human-Computer Studies*, 59(6):983–1024, 2003.
12. Object Management Group. Meta Object Facility (MOF) Specification. Technical report, Object Management Group (OMG), April 2002. http://www.omg.org/docs/formal/02-04-03.pdf.
13. B. Omelayenko. Rdft: A mapping meta-ontology for business integration. In *Proceedings of the Workshop on Knowledge Transformation for the Semantic Web (KTSW 2002) at the 15-th European Conference on Artificial Intelligence*, pages 76–83, Lyon, France, 2002.
14. L. Serafini, H. Stuckenschmidt, and H. Wache. A formal investigation of mapping languages for terminological knowledge. In *Proc. of the 19th International Joint Conference on Artificial Intelligence - IJCAI05*, Edinburgh, UK, August 2005.
15. Heiner Stuckenschmidt and Michael Uschold. Representation of semantic mappings. In Yannis Kalfoglou, Marco Schorlemmer, Amit Sheth, Steffen Staab, and Michael Uschold, editors, *Semantic Interoperability and Integration. Dagstuhl Seminar Proceedings*, volume 04391, Germany, 2005. IBFI, Schloss Dagstuhl.
16. Sergio Tessaris and Enrico Franconi. Rules and queries with ontologies: a unifying logical framework. In Ian Horrocks, Ulrike Sattler, and Frank Wolter, editors, *Description Logics*, volume 147 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2005.
17. J. Warmer and A. Kleppe. *Object Constraint Language 2.0.* MITP Verlag, 2004.