# 10

# Formal and Conceptual Comparison of Ontology Mapping Languages

Saartje Brockmans[1], Peter Haase[1], Luciano Serafini[2], and Heiner Stuckenschmidt[3]

[1] Institute AIFB, University of Karlsruhe, Germany
{sbr,pha}@aifb.uni-karlsruhe.de
[2] Center for Information Technology
Fondazione Bruno Kessler
Trento, Italy
serafini@itc.it
[3] Heiner Stuckenschmidt, Computer Science Institute University of Mannheim,
A5, 6 68159 Mannheim, Germany
heiner@informatik.uni-mannheim.de

**Summary.** The compositional approach where several existing ontologies are connected to form a large modular ontology relies on the representation of mappings between elements in the different participating ontologies. A number of languages have been proposed for this purpose that extend existing logical languages for ontologies in a non-standard way. In this chapter, we compare different proposals for such extensions on a formal level and show that these approaches exhibit fundamental differences with respect to the assumptions underlying their semantics. In order to support application developers to select the right mapping language for a given situation, we propose a mapping metamodel that allows us to encode the formal differences on the conceptual level and facilitates the selection of an appropriate formalism on the basis of a formalism-independent specification of semantic relations between different ontologies by means of a graphical modelling language.

## 10.1 Motivation

The compositional approach to modular ontologies relies on appropriate definitions of interfaces between different modules to be connected. In an ideal case, these interfaces are defined at design time when modules are created in a modular fashion. In reality, we are faced with a situation where no interfaces are defined and relevant connections between ontologies have to be discovered and represented at composition time. There are two main lines of research addressing this problem. The first line is concerned with the development of methods for identifying semantic relations between elements in different ontologies. The second line of research is concerned with

H. Stuckenschmidt et al. (Eds.): Modular Ontologies, LNCS 5445, pp. 267–291, 2009.

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 [3, 7, 6, 14]. In this chapter we compare these approaches and show that they are mostly orthogonal in terms of assumptions made about the right interpretation of mapping relations. 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.

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. 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 from 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. When building the metamodel there is a natural tradeoff between coverage and precision of the metamodel: We focus on approaches that connect description logic based ontologies where mappings are specified in terms of logical axioms. This allows us to be more precise with respect to the nature and properties of mappings. At the same time, we cover a number of relevant mapping approaches that have been developed that satisfy these requirements, including the approaches mentioned in [19]. Further, the restriction to description logics allows us to use previous work on a meta-modeling approach for OWL DL [5] as a starting point.

### 10.1.1 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 [18]. 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 [17] describe an approach that is 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 [8] 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.
- Our approach is based on a sound formal foundation in terms of an encoding of different mapping formalisms in distributed first-order logic.
- We base our meta-modeling on widely used standards in the area of model-driven architectures, in particular MOF and UML.
- Our approach was designed to be able to cover most existing proposals for formal mapping approaches
- 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.

### 10.1.2 Outline

We start our investigation with an informal discussion of several aspects of mapping languages including the kind of semantic relations supported, the kinds of ontology elements connected and some assumption underlying the semantics of different mapping formalisms in 10.2. In sections 10.3 and 10.4 we compare a number of mapping languages on a more formal level. For this purpose, we first introduce distributed first order logic as a unifying framework for encoding different mapping languages in section 10.3. In section 10.4, we encode different mapping language in distributed first order logic. The encoding shows that differences between mapping languages can be expressed in terms of types of axioms used to connect elements in two ontologies and basic assumptions about the relation of the domains under consideration that can be expressed in terms of a set of axioms in Distributed First Order Logic (DFOL). As these results are of purely theoretical interest so far, sections 10.5 and 10.6 are devoted to the problem of providing practical support for the selection of an appropriate mapping formalism. In particular, we propose a metamodel for ontology mappings based on an existing metamodel for OWL ontologies that captures differences and commonalities between different mapping languages on a conceptual level and can be refined to model a particular mapping language by adding constraints to the general model. Section 10.6 presents a graphical modelling language that is based on the metamodel. The language supports the representation of mappings independent of a specific formalism. We close with a discussion of the approach and topics for future work.

## 10.2 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 [20]. 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.

*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. 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 strong form of equivalence is equality, if the connected elements represent exactly the same 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 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 ($\not o$). Adding these negative versions of the relations leaves us with eight semantic relations to 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: In *extensional* mapping definitions, the semantic relations are interpreted as set-relations between the sets of objects represented by elements in the ontologies. In the case of *intensional* mappings, the semantic relations relate the elements directly, i.e. considering the properties of the element itself.

*What are the formal properties of mappings ?*

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 [19]. The assumptions identified in the referred paper are:

- The naming of instances (are instances with the same name assumed to denote the same object?)
- The way inconsistency affects the overall system (does an inconsistency in one ontology also cause the mapped ontologies to become inconsistent)
- The assumptions about the relationships between the mapped domains (where with the *global domain assumption* both ontologies describe exactly the same set of objects, while with the *local domain assumption* the sets of objects may also be completely disjoint or overlap each other)

In [19] it has been shown that the differences between existing proposals of mapping languages for description logics can completely be described in terms of the kinds of semantic relations than can be defined and the assumptions mentioned above. This means that including these aspects in the metamodel ensures that we can model all currently existing mapping approaches and that we are able to distinguish them based on specifications that instantiate the metamodel.

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.

*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 [5]. 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 queries to connect models, which leads to a considerably increased expressiveness.

*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, a mapping is a set of mapping assertions that consist of a semantic relation between elements in different ontologies. Further mappings are first-class objects that exist independently of the ontologies. Mappings are directed and there can be more than one mapping between two ontologies. These choices allow considerable 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.

## 10.3 Distributed First-Order Logic

This section introduces distributed first order logic as a basis for modeling distributed knowledge bases. More details about the language including a sound and complete calculus can be found in [12].

Let $\{L_i\}_{i \in I}$ (in the following $\{L_i\}$) be a family of first order languages with equality defined over a non empty set $I$ of indexes. Each language $L_i$ is the language used by the $i$-th knowledge base (ontology). The signature of $L_i$ is extended with a new set of symbols used to denote objects which are related with other objects in different ontologies. For each variable, and each index $j \in I$ with $j \neq i$ we have two new symbols $x^{\rightarrow j}$ and $x^{j \rightarrow}$, called *arrow variables*. Terms and formulas of $L_i$, also called *$i$-terms* and *$i$-formulas* and are defined in the usual way. Quantification on arrow variables is not permitted. The notation $\phi(\mathbf{x})$ is used to denote the formula $\phi$ and the fact that the free variables of $\phi$ are $\mathbf{x} = \{x_1, \ldots, x_n\}$. In order to distinguish occurrences of terms and formulas in different languages we label them with their index. The expression $\phi : i$ denotes the formula $\phi$ of the $i$-th knowledge base.

The semantics of DFOL is an extension of Local Models Semantics defined in [10]. Local models are defined in terms of first order models. To capture the fact that certain predicates are completely known by the $i$-th sub-system we select a sub-language of $L_i$ containing the equality predicate, denoted as $L_i^c$ we call the *complete fragment* of $L_i$. *Complete terms* and *complete formulas* are terms and formula of $L_i^c$ and vice versa.

**Definition 1 (Set of local Models).** *A* set of local models *of $L_i$ are a set of first order interpretations of $L_i$, on a domain $\mathbf{dom}_i$, which agree on the interpretation of $L_i^c$, the complete fragment of $L_i$.* ◇

As noted in [9] there is a foundational difference between approaches that use epistemic states and approaches that use a classical model theoretic semantics. The two approaches differ as long as there is more than one model m. Using the notion of complete sublanguage $L_c$, however, we can force the set of local models to be either a singleton or the empty set by enforcing that $L^c = L$. Under this assumption the two ways of defining the semantics of submodels are equivalent. Using this assumption, we are therefore able to simulate both kinds of semantics in DFOL.

Two or more models can carry information about the same portion of the world. In this case we say that they *semantically overlap*. Overlapping is unrelated to the fact that the same constant appears in two languages, as from the local semantics we have that the interpretation of a constant $c$ in $L_i$ is independent from the interpretation of the very same constant in $L_j$, with $i \neq j$. Overlapping is also unrelated to the intersection between the interpretation domains of two or more contexts. Namely if $\mathbf{dom}_1 \cap \mathbf{dom}_2 \neq \emptyset$ it does not mean that $L_1$ and $L_j$ overlap. Instead, DFOL explicitly represent semantic overlapping via a domain relation.

**Definition 2 (Domain relation).** *A* domain relation *from* $\mathbf{dom}_i$ *and* $\mathbf{dom}_j$ *is a binary relations* $r_{ij} \subseteq \mathbf{dom}_i \times \mathbf{dom}_j$. ◇

Domain relation from $i$ to $j$ represents the capability of the $j$-th sub-system to represent in its domain the domain of the $i$-th subsystem. A pair $\langle d, d' \rangle$ being in $r_{ij}$ means that, from the point of view of $j$, $d$ in $\mathbf{dom}_i$ is the representation of $d'$ in $\mathbf{dom}_j$. We use the functional notation $r_{ij}(d)$ to denote the set $\{d' \in \mathbf{dom}_j | \langle d, d' \rangle \in r_{ij}\}$. The domain relation $r_{ij}$ formalizes $j$'s subjective point of view on the relation between $\mathbf{dom}_i$ and $\mathbf{dom}_j$ and not an absolute objective point of view. Or in other words $r_{ij} \neq r_{ji}$ because of the non-symmetrical nature of mappings. Therefore $\langle d, d' \rangle \in r_{ij}$ must not be read as if $d$ and $d'$ were the same object in a domain shared by $i$ and $j$. This facts would indeed be formalized by some observer which is external (above, meta) to both $i$ and $j$. Using the notion of domain relation we can define the notion of a model for a set of local models.

**Definition 3 (DFOL Model).** *A DFOL* model, $\mathcal{M}$ *is a pair* $\langle \{\mathcal{M}_i\}, \{r_{ij}\} \rangle$ *where, for each* $i \neq j \in I$: $\mathcal{M}_i$ *is a set of local models for* $L_i$, *and* $r_{ij}$ *is a domain relation from* $\mathbf{dom}_i$ *to* $\mathbf{dom}_j$. ◇

We extend the classical notion of assignment (e.g., the one given for first order logic) to deal with arrow variables using domain relations. In particular, an assignment $a$, provides for each system $i$, an interpretation for all the variable, and for *some* (by not necessarily all) arrow variables as the domain relations might be such that there is no consistent way to assign arrow variables. For instance if $a_i(x) = d$ and $r_{ij}(d) = \emptyset$, then $a_j$ cannot assign anything to $x^{i\rightarrow}$.

**Definition 4 (Assignment).** *Let* $\mathcal{M} = \langle \{\mathcal{M}_i\}, \{r_{ij}\} \rangle$ *be a model for* $\{L_i\}$. *An* assignment $a$ *is a family* $\{a_i\}$ *of partial functions from the set of variables and arrow variables to* $\mathbf{dom}_i$, *such that:*

1. $a_i(x) \in \mathbf{dom}_i$;
2. $a_i(x^{j\rightarrow}) \in r_{ji}(a_j(x))$;
3. $a_j(x) \in r_{ij}(a_i(x^{\rightarrow j}))$;

*An assignment* $a$ *is* admissible for a formula $\phi : i$ *if* $a_i$ *assigns all the arrow variables occurring in* $\phi$. *Furthermore,* $a$ *is admissible for a set of formulas* $\Gamma$ *if it is admissible* *for any* $\phi : j \in \Gamma$. *An assignment* $a$ *is* strictly admissible *for a set of formulas* $\Gamma$ *if it is admissible for* $\Gamma$ *and assigns only the arrow variables that occurs in* $\Gamma$. ◇

Using the notion of an admissible assignment given above, satisfiability in distributed first order logic is defined as follows:

**Definition 5 (Satisfiability).** *Let* $\mathcal{M} = \langle\{\mathcal{M}_i\}, \{r_{ij}\}\rangle$ *be a model for* $\{L_i\}$, $m \in \mathcal{M}_i$, *and* $a$ *an assignment. An* $i$*-formula* $\phi$ *is* satisfied *by* $m$, *w.r.t,* $a$, *in symbols* $m \models_D \phi[a]$ *if*

1. $a$ *is admissible for* $\phi : i$ *and*
2. $m \models \phi[a_i]$, *according to the definition of satisfiability for first order logic.*

$\mathcal{M} \models \Gamma[a]$ *if for all* $\phi : i \in \Gamma$ *and* $m \in \mathcal{M}_i$, $m \models_D \phi[a_i]$[1]. ◇

Mappings between different knowledge bases are formalized in DFOL by a new form of constraints that involves more than one knowledge bases. These formulas that will be the basis for describing different mapping approaches are called interpretation constraints and defined as follows:

**Definition 6 (Interpretation constraint).** *An* interpretation constraint from $i_1, \ldots, i_n$ to $i$ *with* $i_k \neq i$ *for* $1 \leq k \leq n$ *is an expression of the form*

$$\phi_1 : i_1, \ldots, \phi_n : i_n \to \phi : i \tag{10.1}$$

◇

The interpretation constraint (10.1) can be consider as an axiom that restrict the set of possible DFOL models to those which satisfies it. Therefore we need to define when a DFOL model satisfies an interpretation constraint.

**Definition 7 (Satisfiability of interpretation constraints).** *A model* $\mathcal{M}$ satisfies *the interpretation constraint (10.1), in symbols* $\mathcal{M} \models \phi_1 : i_1, \ldots, \phi_n : i_n \to \phi : i$ *if for any assignment* $a$ *strictly admissible for* $\{\phi_1 : i_1, \ldots, \phi_n : i_n\}$, *if* $\mathcal{M} \models \phi_k : i_k[a]$ *for* $1 \leq k \leq n$, *then* $a$ *can be extended to an assignment* $a'$ *admissible for* $\phi : i$ *and such that* $\mathcal{M} \models \phi : i[a']$. ◇

Notice that, depending on the fact that an arrow variable $x^{\rightarrow}$ occurs on the left or on the right side of the constraint, $x^{\rightarrow}$ has a universal or an existential reading. Figure 10.1 summarizes the different possible readings that will reoccur later. Notationally for any predicate $P$, $\|P\|_i = \bigcap_{m \in \mathcal{M}_i} m(P)$, where $m(P)$ is the interpretation of $P$ in $m$.

By means of interpretation constraints on equality, we can formalize possible relations between heterogeneous domains.

[1] Since it will be clear from the context, in the rest we will use the classical satisfiability symbol $\models$ instead of $\models_D$ and we will write $m \models \phi[a]$ to mean that an $i$-formula $\phi$ is satisfied by $m$. In writing $m \models \phi[a]$ we always mean that of $a$ is admissible for $\phi : i$ (in addition to the fact that $m$ classically satisfies $\phi$ under the assignment $a$).

| | | |
|---|---|---|
| $a)\ \mathcal{M} \models i : P(x^{\rightarrow j}) \rightarrow j : Q(x)$ | iff | For all $d \in \|P\|_i$ and <u>for all $d' \in r_{ij}(d)$</u>, $d' \in \|Q\|_j$ |
| $b)\ \mathcal{M} \models i : P(x) \rightarrow j : Q(x^{i\rightarrow})$ | iff | For all $d \in \|P\|_i$ <u>there is a $d' \in r_{ij}(d)$</u>, s.t., $d' \in \|Q\|_j$ |
| $c)\ \mathcal{M} \models j : Q(x^{i\rightarrow}) \rightarrow i : P(x)$ | iff | For all $d \in \|Q\|_j$ and <u>for all $d'$ with $d \in r_{ij}(d')$</u>, $d' \in \|P\|_i$ |
| $d)\ \mathcal{M} \models j : Q(x) \rightarrow i : P(x^{\rightarrow j})$ | iff | For all $d \in \|Q\|_i$ <u>there is a $d'$</u> <u>with $d \in r_{ij}(d')$</u>, s.t., $d' \in \|P\|_i$ |

**Fig. 10.1.** Implicit Quantification of Arrow Variables in Interpretation Constraints

$$
\begin{aligned}
\mathsf{F}_{ij} &= \left\{x^{\rightarrow j} = y^{\rightarrow j} : i \rightarrow x = y : j\right\} \\
\mathsf{INV}_{ij} &= \left\{\begin{array}{l} x = y^{j\rightarrow} : i \rightarrow x^{i\rightarrow} = y : j \\ x = y^{i\rightarrow} : j \rightarrow x^{j\rightarrow} = y : i \end{array}\right\} \\
\mathsf{OD}_{ij} &= \mathsf{F}_{ij} \cup \mathsf{F}_{ji} \cup \mathsf{INV}_{ij} \\
\mathsf{ED}_{ij} &= \mathsf{OD}_{ij} \cup \{x = x : i \rightarrow x^{i\rightarrow} = x^{i\rightarrow} : j\} \\
\mathsf{ID}_{ij} &= \mathsf{ED}_{ij} \cup \mathsf{ED}_{ji} \\
\mathsf{RD}_{ij} &= \left\{\begin{array}{l} x = c : i \rightarrow x^{i\rightarrow} = c : j \\ x = c : j \rightarrow x^{j\rightarrow} = c : i \end{array} \middle| \, c \in L_i \cap L_j \right\} \\
\mathsf{IP}_{ij} &= \bot : i \rightarrow \bot : j
\end{aligned}
$$

**Proposition 1.** *Let $\mathcal{M}$ be a DFOL model and $i \neq j \in I$.*

1. *$\mathcal{M} \models \mathsf{F}_{ij}$ iff $r_{ij}$ is a partial function.*
2. *$\mathcal{M} \models \mathsf{INV}_{ij}$ iff $r_{ij}$ is the inverse of $r_{ji}$.*
3. *$\mathcal{M} \models \mathsf{OD}_{ij}$ if $r_{ij}(= r_{ji}^{-1})$ is an isomorphism between a subset of $\mathbf{dom}_i$ and a subset of $\mathbf{dom}_j$. I.e., $\mathbf{dom}_i$ and $\mathbf{dom}_j$ (isomorphically) overlap.*
4. *$\mathcal{M} \models \mathsf{ED}_{ij}$ iff $r_{ij}(= r_{ji}^{-1})$ is an isomorphism between $\mathbf{dom}_i$ and a subset of $\mathbf{dom}_j$. I.e., $\mathbf{dom}_i$ is (isomorphically) embedded in $\mathbf{dom}_j$*
5. *$\mathcal{M} \models \mathsf{ID}_{ij}$ iff $r_{ij}(= r_{ji}^{-1})$ is an isomorphism between $\mathbf{dom}_i$ and $\mathbf{dom}_j$. I.e., $\mathbf{dom}_i$ is isomorphic to $\mathbf{dom}_j$.*
6. *$\mathcal{M} \models \mathsf{RD}$, if for every constant $c$ of $L_i$ and $L_j$, if $c$ is interpreted in $d$ for all $m \in \mathcal{M}_i$ then $c$ is interpreted in $r_{ij}(d)$ for all models of $m \in \mathcal{M}_j$, and vice-versa. I.e., the constant $c$ is rigidly interpreted by $i$ and $j$ in two corresponding objects.*
7. *Finally $\mathcal{M} \models \mathsf{IP}_{ij}$ iff $\mathcal{M}_i = \emptyset$ implies that $\mathcal{M}_j = \emptyset$. I.e., inconsistency propagates from $i$ to $j$.*

## 10.4 Modeling Mapping Languages in DFOL

Mapping languages formalisms are based on four main parameters: local languages and local semantics used to specify the local knowledge, and mapping languages and semantics for mappings, used to specify the semantic relations between the local knowledge. In this section we focus on the second pair and as far as local languages and local semantics it is enough to notice that:

**Local languages.** In all approaches local knowledge is expressed by a suitable fragment of first order languages.

**Local semantics.** With the notable exception of [9], where authors propose an *epistemic approach* to information integration, all the other formalisms for ontology mapping assume that each local knowledge is interpreted in a (partial) state of the world and not into an epistemic state. This formally corresponds to the fact that each local knowledge base is associated with *at most one* FOL interpretation.

The first assumption is naturally captured in DFOL, by simply considering $L_i$ to be an adequately restricted FOL language. As far as the local semantics, in DFOL models each $L_i$ is associates with a *set of interpretations*. To simulate the single local model assumption, in DFOL it is enough to declare each $L_i$ to be a *complete* language. This implies that all the $m \in M_i$ have to agree on the interpretation of $L_i$-symbols.

Notationally, $\phi, \psi, \ldots$ will be used to denote both DL expressions and FOL open formulas. If $\phi$ is a DL concept, $\phi(x)$ (or $\phi(x_1, \ldots, x_n)$) will denote the corresponding translation of $\phi$ in FOL as described in [1]. If $\phi$ is a role $R$ then $\phi(x, y)$ denotes its translation $P(x, y)$, and if $\phi$ is a constant $c$, then $\phi(x)$ denote its translation $x = c$. Finally we use $\mathbf{x}$ to denote a set $x_1, \ldots, x_n$ of variables.

### 10.4.1 Distributed Description Logics/C-OWL

The approach presented in [2] extends DL with a local model semantics similar to the one introduced above and so-called bridge rules to define semantic relations between different T-Boxes. A distributed interpretation for DDL on a family of DL language $\{L_i\}$, is a family $\{\mathcal{I}_i\}$ of interpretations, one for each $L_i$ plus a family $\{r_{ij}\}_{i \neq j \in I}$ of domain relations. While the original proposal only considered subsumption between concept expressions, the model was extended to a set of five semantic relations discussed below. The Semantics of the five semantic relations defined in C-OWL is the following:

**Definition 8 ([4]).** *Let $\phi$ and $\psi$ be either concepts, or individuals, or roles of the descriptive languages $L_i$ and $L_j$ respectively*[2].

1. $\mathfrak{I} \models \phi : i \xrightarrow{\sqsubseteq} \psi : j$ *if* $r_{ij}(\phi^{\mathcal{I}_i}) \subseteq \psi^{\mathcal{I}_j}$;
2. $\mathfrak{I} \models \phi : i \xrightarrow{\sqsupseteq} \psi : j$ *if* $r_{ij}(\phi^{\mathcal{I}_i}) \supseteq \psi^{\mathcal{I}_j}$;
3. $\mathfrak{I} \models \phi : i \xrightarrow{\equiv} \psi : j$ *if* $r_{ij}(\phi^{\mathcal{I}_i}) = \psi^{\mathcal{I}_j}$;
4. $\mathfrak{I} \models \phi : i \xrightarrow{\perp} \psi : j$ *if* $r_{ij}(\phi^{\mathcal{I}_i}) \cap \psi^{\mathcal{I}_j} = \emptyset$;
5. $\mathfrak{I} \models \phi : i \xrightarrow{*} \psi : j$ *if* $r_{ij}(\phi^{\mathcal{I}_i}) \cap \psi^{\mathcal{I}_j} \neq \emptyset$;

[2] In this definition, to be more homogeneous, we consider the interpretations of individuals to be sets containing a single object rather than the object itself.

*An interpretation for a context space is a model for it if all the bridge rules are satisfied.* ◇

From the above satisfiability condition one can see that the mapping $\phi : i \xrightarrow{\equiv} \psi : j$ is equivalent to the conjunction of the mappings $\phi : i \xrightarrow{\sqsubseteq} \psi : j$ and $\phi : i \xrightarrow{\sqsupseteq} \psi : j$. The mapping $\phi : i \xrightarrow{\perp} \psi : j$ is equivalent to $\phi : i \xrightarrow{\sqsubseteq} \neg\psi : j$. And finally the mapping $\phi : i \xrightarrow{*} \psi : j$ is the negation of the mapping $\phi : i \xrightarrow{\sqsubseteq} \psi : j$. Therefore for the translation we will consider only the primitive mappings. As the underlying notion of a model is the same as for DFOL, we can directly try to translate bridge rules into interpretation constraints. In particular, there are no additional assumptions about the nature of the domains that have to be modeled. The translation is the following:

| **C-OWL** | **DFOL** |
|---|---|
| $\phi : i \xrightarrow{\sqsubseteq} \psi : j$ | $\phi(x^{\rightarrow j}) : i \rightarrow \psi(x) : j$ |
| $\phi : i \xrightarrow{\sqsupseteq} \psi : j$ | $\psi(x) : j \rightarrow \phi(x^{\rightarrow j}) : i$ |
| $\phi : i \stackrel{\sqsubseteq}{\not\rightarrow} \psi : j$ | No translation |

We see that a bridge rule basically corresponds to the interpretation a) and d) in Figure 10.1. The different semantic relations correspond to the usual reads of implications. Finally negative information about mappings (i.e., $\phi : i \stackrel{\sqsubseteq}{\not\rightarrow} \psi : j$) is not representable by means of DFOL interpretation constraints.

### 10.4.2 Ontology Integration Framework (OIS)

Calvanese and colleagues in [7] propose a framework for mappings between ontologies that generalizes existing work on view-based schema integration [22] and subsumes other approaches on connecting DL models with rules. In particular, they distinguish global centric, local centric and the combined approach. Differences between these approaches are in the types of expressions connected by mappings. With respect to the semantics of mappings, they are the same and are therefore treated as one.

OIS assumes the existence of a global model $g$ into which all local models $s$ are mapped. On the semantic level, the domains of the local models are assumed to be embedded in a global domain. Further, in OIS constants are assumed to rigidly designate the same objects across domain. Finally, global inconsistency is assumed, in the sense that the inconsistency of a local knowledge makes the whole system inconsistent. As shown in Proposition 1, we can capture these assumptions by the set of interpretation constraints $\mathsf{ED}_{sg}$, $\mathsf{RD}_{sg}$, and $\mathsf{IP}_{sg}$, where $s$ is the index of any source ontology and $g$ the index of the global ontology.

According to these assumptions mappings are described in terms of correspondences between a local and the global model. The interpretation of these correspondences are defined as follows:

**Definition 9 ([7]).** *Correspondences between source ontologies and global ontology are of the following three forms*

1. *$\mathcal{I}$ satisfies $\langle \phi, \psi, sound \rangle$ w.r.t. the local interpretation $\mathcal{D}$, if all the tuples satisfying $\psi$ in $\mathcal{D}$ satisfy $\phi$ in $\mathcal{I}$*
2. *$\langle \phi, \psi, complete \rangle$ w.r.t. the local interpretation $\mathcal{D}$, if no tuple other than those satisfying $\psi$ in $\mathcal{D}$ satisfies $\phi$ in $\mathcal{I}$,*
3. *$\langle \phi, \psi, exact \rangle$ w.r.t. the local interpretation $\mathcal{D}$, if the set of tuples that satisfies $\psi$ in $\mathcal{D}$ is exactly the set of tuples satisfying $\phi$ in $\mathcal{I}$.* ◇

From the above semantic conditions, $\langle \phi, \psi, exact \rangle$ is equivalent to the conjunction of $\langle \phi, \psi, sound \rangle$ and $\langle \phi, \psi, complete \rangle$. It is therefore enough to provide the translation of the first two correspondences. The definitions 1 and 2 above can directly be expressed into interpretation constraints (compare Figure 10.1) resulting in the following translation:

| **GLAV Correspondence** | **DFOL** |
|---|---|
| $\langle \phi, \psi, sound \rangle$ | $\psi(\mathbf{x}) : s \rightarrow \phi(\mathbf{x}^{s\rightarrow}) : g$ |
| $\langle \phi, \psi, complete \rangle$ | $\phi(\mathbf{x}) : g \rightarrow \psi(\mathbf{x}^{\rightarrow g}) : s$ |

The translation shows that there is a fundamental difference in the way mappings are interpreted in C-OWL and in OIS. While C-OWL mappings correspond to a universally quantified reading (Figure 1 a), OIS mappings have an existentially quantified readings (Figure 1 b/d). We will come back to this difference later.

### 10.4.3 DL for Information Integration (DLII)

A slightly different approach to the integration of different DL models is described in [6]. This approach assumes a partial overlap between the domains of the models $M_i$ and $M_j$ rather than a complete embedding of them in a global domain. This is captured by the interpretation constraint $\mathsf{OD}_{ij}$. The other assumptions (rigid designators and global inconsistency) are the same as for OIS.

An interpretation $\mathcal{I}$ associates to each $M_i$ a domain $\Delta_i$. These different models are connected by inter-schema assertions. Satisfiability of interschema assertions is defined as follows:[3]

**Definition 10 (Satisfiability of interschema assertions).** *If $\mathcal{I}$ is an interpretation for $M_i$ and $M_j$ we say that $\mathcal{I}$ satisfies the interschema assertion*

[3] To simplify the definition we introduce the notation $\top^{\mathcal{I}}_{nij} = \top^{\mathcal{I}}_{ni} \cap \top^{\mathcal{I}}_{nj}$ for any $n \geq 1$. Notice that $\top^{\mathcal{I}}_{nij} = \Delta^n_i \cap \Delta^n_j$.

$\phi \sqsubseteq_{ext} \psi$, *if* $\phi^{\mathcal{I}} \subseteq \psi^{\mathcal{I}}$ $\quad$ $\phi \not\sqsubseteq_{ext} \psi$, *if* $\phi^{\mathcal{I}} \not\subseteq \psi^{\mathcal{I}}$
$\phi \equiv_{ext} \psi$, *if* $\phi^{\mathcal{I}} = \psi^{\mathcal{I}}$ $\quad$ $\phi \not\equiv_{ext} \psi$, *if* $\phi^{\mathcal{I}} \neq \psi^{\mathcal{I}}$
$\phi \sqsubseteq_{int} \psi$, *if* $\phi^{\mathcal{I}} \cap \top^{\mathcal{I}}_{nij} \subseteq \psi^{\mathcal{I}} \cap \top^{\mathcal{I}}_{nij}$
$\phi \equiv_{int} \psi$, *if* $\phi^{\mathcal{I}} \cap \top^{\mathcal{I}}_{nij} = \psi^{\mathcal{I}} \cap \top^{\mathcal{I}}_{nij}$
$\phi \not\sqsubseteq_{int} \psi$, *if* $\phi^{\mathcal{I}} \cap \top^{\mathcal{I}}_{nij} \not\subseteq \psi^{\mathcal{I}} \cap \top^{\mathcal{I}}_{nij}$
$\phi \not\equiv_{int} \psi$, *if* $\phi^{\mathcal{I}} \cap \top^{\mathcal{I}}_{nij} \neq \psi^{\mathcal{I}} \cap \top^{\mathcal{I}}_{nij}$ ◇

As before $\equiv_{est}$ and $\equiv_{int}$ are definable as conjunctions of $\sqsubseteq_{est}$ and $\sqsubseteq_{int}$, so we can ignore them for the DFOL translation. Furthermore, a distinction is made between extensional and intentional interpretation of inter-schema assertions, which leads to different translations into DFOL.

| **inter-schema assertions** | **DFOL** |
|---|---|
| $\phi \sqsubseteq_{ext} \psi$ | $\phi(\mathbf{x}) : i \to \psi(\mathbf{x}^{i\to}) : j$ |
| $\phi \not\sqsubseteq_{ext} \psi, \phi \not\equiv_{ext} \psi$ | No translation |
| $\phi \sqsubseteq_{int} \psi$ | $\phi(\mathbf{x}^{\to j}) : i \to \psi(\mathbf{x}) : j$ |
| $\phi \not\sqsubseteq_{int} \psi, \phi \not\equiv_{int} \psi$ | No translation |

While the extensional interpretation corresponds to the semantics of mappings in OIS, the intentional interpretation corresponds to the semantics of mappings in C-OWL. Thus using the distinction made in this approach we get an explanation of different conceptualizations underlying the semantics of C-OWL and OIS that use an extensional and an intentional interpretation, respectively.

### 10.4.4 $\mathcal{E}$-Connections

A different approach for defining relations between DL knowledge bases has emerged from the investigation of so-called $\mathcal{E}$-connections between abstract description systems [16]. Originally intended to extend the decidability of DL models by partitioning it into a set of models that use a weaker logic, the approach has recently been proposed as a framework for defining mappings between ontologies [13].

In the $\mathcal{E}$-connections framework, for every pair of ontologies $ij$ there is a set $\mathcal{E}_{ij}$ of *links*, which represents binary relations between the domain of the $i$-th ontology and the domain of the $j$-th ontology. Links from $i$ to $j$ can be used to define $i$ concepts, in a way that is analogous to how roles are used to define concepts. In the following table we report the syntax and the semantics of $i$-concepts definition based on links. ($E$ denotes a link from $i$ to $j$ and $C$ denotes a concept in $j$). The only assumption about the relation between domains is global inconsistency (see above).

In DFOL we have only one single relation between from $i$ to $j$, while in $\mathcal{E}$-connection there are many possible relation. However, we can use a similar technique as used in [2] to map relations on inter-schema relations: each of the relation in $\mathcal{E}_{ij}$ acts as a $r_{ij}$. To represent $\mathcal{E}$-connection it is therefore enough to label each arrow variable with the proper link name. The arrow variable $x^{\xrightarrow{Own} j}$ is read as the arrow variable $x^{\to i}$ where $r_{ij}$ is intended to be the interpretation of $\mathsf{Own}_{ij}$. With this syntactic extension of DFOL concepts definitions based on links (denoted as $E$) can be codified in DFOL as follows:

| $\mathcal{E}$-connections | DFOL |
|---|---|
| $\phi \sqsubseteq \exists E.\psi$ | $\phi(x) : i \rightarrow \psi(x^{i \xrightarrow{E}}) : j$ |
| $\phi \sqsubseteq \forall E.\psi$ | $\phi(x^{\xrightarrow{E} j}) : i \rightarrow \psi(x) : j$ |
| $\phi \sqsubseteq \geq nE.\psi$ | $\bigwedge_{k=1}^{n} \phi(x_1) : i \rightarrow$ $\bigwedge_{k \neq h=1}^{n} \psi(x_k^{i \xrightarrow{E}}) \wedge x_k \neq x_h : j$ |
| $\phi \sqsubseteq \leq nE.\psi$ | $\phi(x) \wedge \bigwedge_{k=1}^{n+1} x = x_k^{\xrightarrow{E} j} : i \rightarrow$ $\bigvee_{k=1}^{n+1} \left(\psi(x_k) \supset \bigvee_{h \neq k} x_h = x_k\right) : j$ |

We see that like OIS, links in the $\mathcal{E}$-connections framework have an extensional interpretation. The fact, that the framework distinguishes between different types of domain relations, however, makes it different from all other approaches.

Another difference to the previous approaches is that new links can be defined on the bases of existing links similar to complex roles in DL. Syntax and semantics for link constructors is defined in the usual way: $(E^{-})^{I} = \left(E^{\mathcal{I}}\right)^{-1}$ (Inverse), $(E \sqcap F)^{I} = E^{\mathcal{I}} \cap F^{\mathcal{I}}$ (Conjunction), $(E \sqcup F)^{I} = E^{\mathcal{I}} \cup F^{\mathcal{I}}$ (Disjunction), and $(\neg E)^{I} = (\Delta_i \times \Delta_j) \setminus E^{\mathcal{I}}$ (Complement). Notice that, by means of inverse link we can define mapping of the b and d type. E.g., the e-connection statement $\phi \sqsubseteq \exists E^{-}\psi$, encodes corresponds to the DFOL bridge rules $i : \phi(x) \rightarrow j : \psi(x^{i\rightarrow})$ which is of type b). Similarly the e-connection $\phi \sqsubseteq \forall E^{-}\psi$ corresponds to a mapping of type d).

As the distinctions between different types of links is only made on the model theoretic level, it is not possible to model Boolean combinations of links. Inverse links, however, can be represented by the following axiom:

$$y = x^{\xrightarrow{E} j} : i \rightarrow y^{\xrightarrow{E^{-}} i} = x : j$$
$$y^{\xrightarrow{E^{-}} i} = x : j \rightarrow y = x^{\xrightarrow{E} j} : i$$

Finally the inclusion axioms between links, i.e., axioms of the form $E \sqsubseteq F$ where $E$ and $F$ are homogeneous links, i.e., links of the same $\mathcal{E}_{ij}$, can be translated in DFOL as follows:

$$x = y^{\xrightarrow{E} j} : i \rightarrow x^{i \xrightarrow{F}} = y : j$$

We can say that the $\mathcal{E}$-connections framework significantly differs from the other approaches in terms of the possibilities to define and combine mappings of different types.

### 10.4.5 Summary

The encoding of different mapping approaches in a common framework has two immediate advantages. The first one is the ability to reason across the different frameworks. This can be done on the basis of the DFOL translation of the different approaches using the sound and complete calculus for DFOL [11]. As there are not always complete translations, this approach does not cover all aspects of the different approaches, but as shown above, we can capture the most aspects. There are only two aspects which cannot be represented in DFOL, namely "non mappings" ($\phi : i \xrightarrow{*} \psi : j$ in C-OWL, $\phi \not\sqsubseteq_{int} \psi$ etc. in DLII) and "complex mappings" such as complex links in $\mathcal{E}$-connection. The second benefit is the possibility to compare the expressiveness of the approaches. We have several dimensions along which the framework can differ:

Arity of mapped items[4] C-OWL allows only to align constants, concepts and roles (2-arity relations), $\mathcal{E}$-connection allows to align only 1-arity items, i.e., concepts, while DLII and OIS allow to integrate $n$-arity items.

Positive/negative mappings Most approaches state positive facts about mapping, e.g. that two elements are equivalent. The DLII and C-OWL frameworks also allow to state that two elements do not map ($\phi \not\equiv \psi$).

Domain relations The approaches make different assumptions about the nature of the domain. While C-OWL and $\mathcal{E}$-connections do not assume any relation between the domains, DLII assumes overlapping domains and OIS local domains that are embedded in a global domain.

Multiple mappings Only $\mathcal{E}$-connection approach supports form the definition of different types of mappings between ontologies that partition the inter-domain relations.

Local inconsistency Some approaches provide a consistent semantics also in the case in which some of the ontologies or mappings are inconsistent.

We summarize the comparison in the following table.

| | Int. constr. (cf. fig. 10.1) | | | | Mapping type | | | Domain relation | Arity | Local $\perp$ |
|---|---|---|---|---|---|---|---|---|---|---|
| | a) | b) | c) | d) | Pos. | Neg. | Mult. | | | |
| C-OWL | $\times$ | | | $\times$ | $\times$ | $\times$ | | Het. | 2 | $\times$ |
| OIS | | $\times$ | | $\times$ | $\times$ | | | Incl. | $n$ | |
| DLII | $\times$ | $\times$ | | | $\times$ | $\times$ | | Emb. | $n$ | |
| $\mathcal{E}$-Conn. | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ | $\times$ | Het. | 1 | |

We conclude that existing approaches make choices along a number of dimensions. These choices are obviously influenced by the intended use. Approaches intended for database integration for example will support the mapping of n-ary items that correspond to tuples in the relational model. Despite this fact, almost no work has been done on charting the landscape of choices to be made when designing a mapping approach and for adapting the approach to the requirement of the application. The work reported in this paper provides the basis for this kind of work by identifying the possible choices on a formal level. An important topic of future work is to identify possible combinations of features for mapping languages on a formal level in order to get a more complete picture of the design space of mapping languages.

[4] Due to limited space we did not discuss the encoding of mapped items in this paper.

## 10.5 Conceptual Comparison of Mapping Languages

As we have shown in the previous section, the differences between mapping languages can be described in terms of a fixed set of features including the kinds of semantic mappings and assumptions about the relation between the domains of interest. Other features are the kinds of language elements that can be connected by mappings. We can use these features to lift the comparison of different mapping languages from the formal to the conceptual level. For this purpose, we define a general metamodel of ontology mappings that defines structural aspects of the different formalisms and also includes attributes for defining the formal aspects that have been identified as distinguishing features of different languages.

### 10.5.1 A Metamodel for OWL DL Ontologies

We now review our previous work on a metamodel for OWL DL. Figure 10.2 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 10.2 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. Additionally, the metamodel is augmented with constraints, expressed in the Object Constraint Language ([23]), specifying invariants that have to be fulfilled by all models that instantiate the metamodel. However, for lack of space, we refer to [5] for a full specification. The metamodel for OWL DL ontologies ([5]) has a one-to-one mapping to the abstract syntax of OWL DL and thereby to its formal semantics. Our metamodel is built based on a similar approach as in [15], although the two metamodels have some fundamental differences.

Further, we have defined a metamodel for rule extensions of OWL DL. For the details, we refer the reader to [5]. In our mapping metamodel, we reuse parts of the rule metamodel, as we explain in detail in Section 10.5.2.

### 10.5.2 Extending the Metamodel with Mappings

We propose a formalism-independent metamodel covering OWL ontology mappings as described in Section 10.2. The metamodel is a consistent extension of our earlier work on metamodels for OWL DL ontologies and SWRL rules [5]. It has constraints defined in OCL [23] as well, which we omit here due to lack of space and instead refer to [5] for a complete reference. Figure 10.3 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` with 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

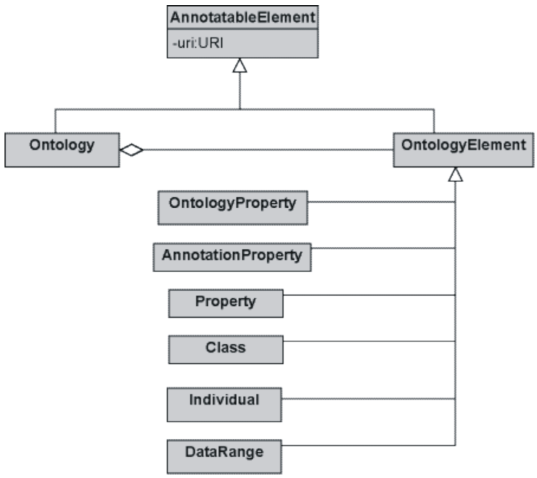


**Fig. 10.2.** Main Elements of the Ontology Definition Metamodel

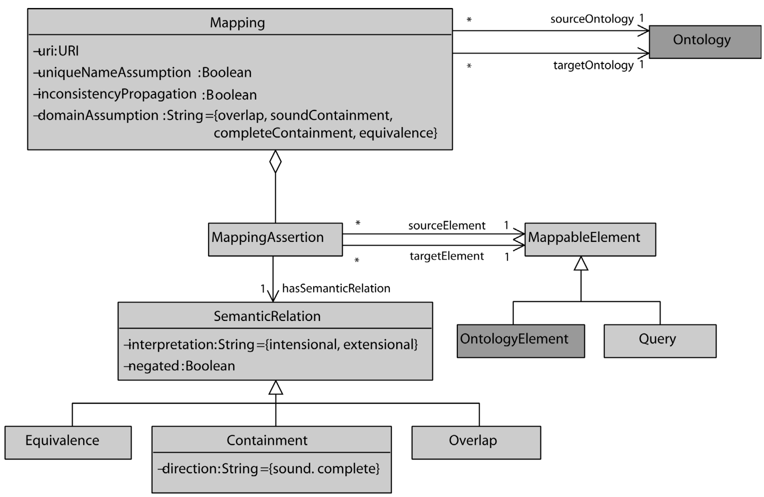


**Fig. 10.3.** Metamodel for ontology mappings

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`. 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.

We defined 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`. For example, a negated `Overlaps` relation specifies the disjointness of two elements. The other attribute of `SemanticRelation`, `interpretation`, defines whether the mapping assertion is assumed to be interpreted intensionally or extensionally. Please note that the metamodel in principle supports all semantic relations for all mappable elements, including individuals.

A mapping assertion can connect two mappable elements, which may be ontology elements or queries. To support this, `MappableElement` has two subclasses `OntologyElement` and `Query`. The former is previously defined in the OWL DL metamodel. 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 similar nature [21]: 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 answer to a query consists of all variable bindings for which the grounded rule body is logically implied by the ontology.

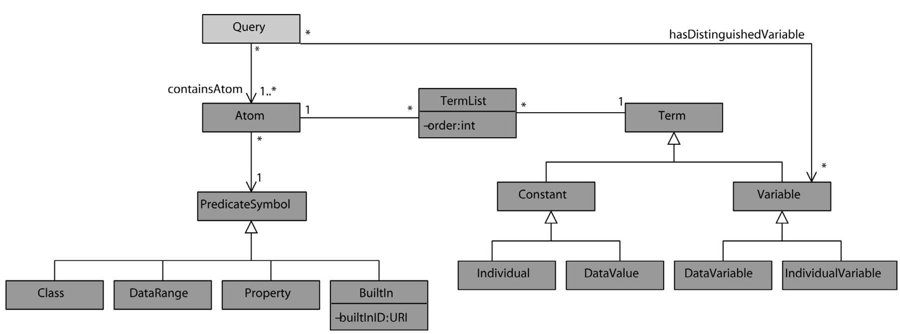


**Fig. 10.4.** Metamodel for ontology mappings - definition of a query

Figure 10.4 shows this connection and shows how a `Query` is composed. It depicts 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]

## 10.6 Formalism Independent Mapping Specification

The metamodel presented in the previous section allowed us to lift the comparison of mapping languages from the formal to a conceptual level and to abstract from formal details. This step does not only ease the comparison of languages it also supports the selection of an appropriate mapping language based on the actual requirements of a given application. We believe that these requirements are best captured by providing the user with the possibility of specifying semantic relations between ontologies independent of a concrete language. In this section, we present a graphical modelling language for mappings that is based on the mapping metamodel presented above. Basing the modelling language on the metamodel ensures that the resulting models can later be linked to constructs in a concrete language via the metamodel. It further allows us to test a given model against the constraints different mapping languages pose on the general metamodel and decide whether a certain language can be used to implement the graphical model.

### 10.6.1 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 10.5 shows a small example of an ontology using the UML profile. It contains the definition of classes `Article`, `Book` and `Thesis` as subclasses of `Publication`. The first two classes are defined to be disjoint. The ontology contains another class `Person` and its subclass `Researcher`. An association between

[5] A variable which is defined as distinguished variable in the source mappable element, must be defined as distinguished variable in the target mappable element as well.

`Publication` and `Person` denotes the object property `authorOf`, from which domain and range are defined via an association class. Furthermore, the ontology defines object properties between `Publication` and `Topic`, and between `Topic` and `Name`. Finally, the ontology contains some instances of its classes and object property. For a discussion of all details of the UML profile for OWL DL ontologies, we refer to [5].

Another small ontology of the same domain is presented in Figure 10.6. In typical use cases such as data translation, data integration, etc. mappings between the two ontologies would have to be defined, as described in the earlier sections. The following sections present a metamodel and UML profile for the definition of these ontology mappings.

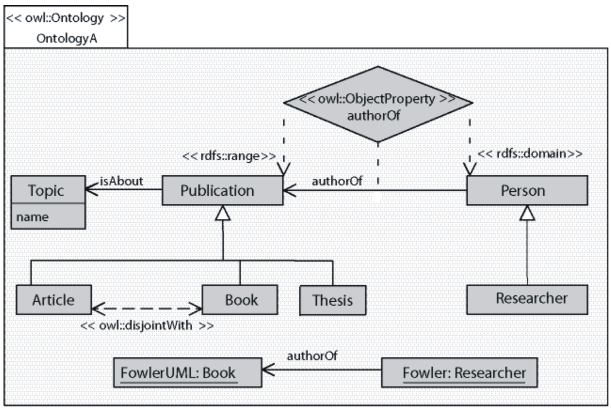


**Fig. 10.5.** A First Sample Ontology Depicted using the UML Profile for the Ontology Metamodel

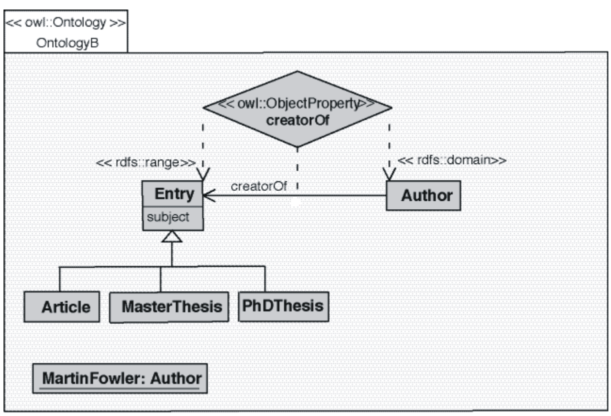


**Fig. 10.6.** A Second Sample Ontology Depicted using the UML Profile for the Ontology Metamodel

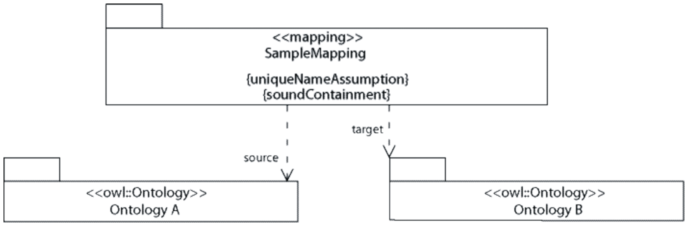


**Fig. 10.7.** Visual notation for a mapping between two ontologies


**Fig. 10.8.** Sample containment relation between two concepts


**Fig. 10.9.** Sample extensional containment relation between two properties

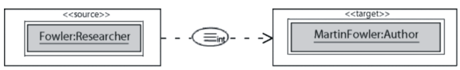


**Fig. 10.10.** Sample intensional equivalence relation between two individuals

### 10.6.2 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 10.5.2. 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 10.7. 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 10.8, a source concept `Publication` is defined to be more specific than the target concept `Entry`. The example in Figure 10.9 relates two properties `authorOf` and `creatorOf` using an extensional containment relationship. Figure 10.10 models `Researcher Fowler` and `Author MartinFowler` as two equivalent instances. 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 10.11. The example defines that the union of the classes `PhDThesis` and `MasterThesis`, is equivalent to the class `Thesis`.

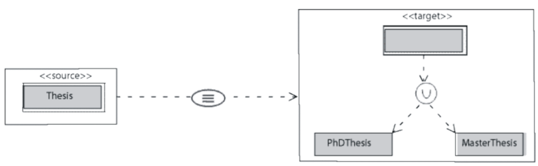


**Fig. 10.11.** Sample equivalence relation between complex class descriptions

Figure 10.12 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`.

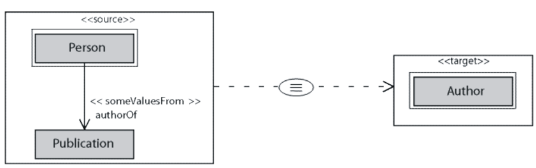


**Fig. 10.12.** Sample equivalence relation between complex class descriptions

Figure 10.13 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 two distinguished variables X and Z, again denoted with a double-lined box.

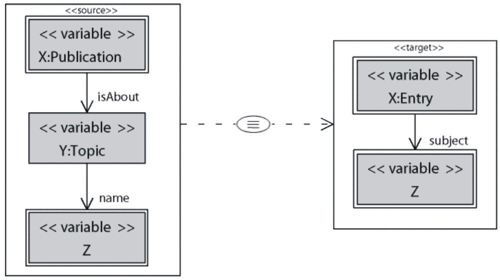


**Fig. 10.13.** Sample equivalence relation between two queries

## 10.7 Discussion and Conclusions

In this chapter, we have presented a comprehensive comparison of ontology mapping languages on a formal and a conceptual level. Based on an encoding in distributed first order logic, we have shown that existing mapping languages differ in a number of quite fundamental assumptions that make them largely incompatible with each other. We have concluded that the choice of a suitable mapping formalism is a critical success factor for a successful composition of ontologies. It is clear that this choice should be based on the formal characteristics of the languages. In order to support the choice of a language based on these characteristics we have presented a metamodel and a graphical modelling language to support formalism independent graphical modeling of mappings between OWL ontologies and their required characteristics. The metamodel ties in with previous work on similar metamodels for OWL DL and rule extensions and the results of the formal analysis of mapping languages. 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. In a first step, we have to link the abstract metamodel 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. 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. 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.

## References


1. Borgida, A.: On the relative expressiveness of description logics and predicate logics. Artificial Intelligence 82, 353–367 (1996) (research note)
2. Borgida, A., Serafini, L.: Distributed description logics: Assimilating information from peer sources. Journal of Data Semantics 1, 153–184 (2003)
3. Bouquet, P., Giunchiglia, F., van Harmelen, F., Serafini, L., Stuckenschmidt, H.: C-OWL: Contextualizing ontologies. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 164–179. Springer, Heidelberg (2003)
4. Bouquet, P., Giunchiglia, F., van Harmelen, F., Serafini, L., Stuckenschmidt, H.: Contextualizing ontologies. Journal on Web Semantics 1(4), 325–343 (2004)
5. Brockmans, S., Haase, P.: 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
6. Calvanese, D., Giacomo, G.D., Lenzerini, M.: Description logics for information integration. In: Kakas, A.C., Sadri, F. (eds.) Computational Logic: Logic Programming and Beyond. LNCS, vol. 2408, pp. 41–60. Springer, Heidelberg (2002)
7. Calvanese, D., Giacomo, G.D., Lenzerini, M.: A framework for ontology integration. In: Cruz, I., Decker, S., Euzenat, J., McGuinness, D. (eds.) The Emerging Semantic Web, pp. 201–214. IOS Press, Amsterdam (2002)
8. Crubézy, M., Musen, M.A.: Ontologies in support of problem solving. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies, pp. 321–342. Springer, Heidelberg (2003)
9. Franconi, E., Tessaris, S.: Rules and queries with ontologies: a unified logical framework. In: Ohlbach, H.J., Schaffert, S. (eds.) PPSWR 2004. LNCS, vol. 3208, pp. 50–60. Springer, Heidelberg (2004)
10. Ghidini, C., Giunchiglia, F.: Local model semantics, or contextual reasoning = locality + compatibility. Artificial Intelligence 127(2), 221–259 (2001)
11. Ghidini, C., Serafini, L.: Distributed first order logics. In: Gabbay, D.M., De Rijke, M. (eds.) Frontiers of Combining Systems 2, pp. 121–139. Research Studies Press Ltd. (2000)
12. Ghidini, C., Serafini, L.: Distributed first order logic - revised semantics. Technical report, ITC-irst (January 2005)
13. Grau, B.C., Parsia, B., Sirin, E.: Working with multiple ontologies on the semantic web. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) ISWC 2004. LNCS, vol. 3298, pp. 620–634. Springer, Heidelberg (2004)
14. Haase, P., Motik, B.: A mapping system for the integration of OWL-DL ontologies. In: Proceedings of the ACM-Workshop: Interoperability of Heterogeneous Information Systems (IHIS 2005) (November 2005)
15. IBM, Sandpiper Software. Ontology Definition Metamodel, Fourth Revised Submission to OMG (November 2005)
16. Kutz, O., Lutz, C., Wolter, F., Zakharyaschev, M.: E-connections of abstract description systems. Artificial Intelligence 156(1), 1–73 (2004)
17. Maedche, A., Motik, B., Silva, N., Volz, R.: MAFRA - a mapping framework for distributed ontologies. In: Gómez-Pérez, A., Benjamins, V.R. (eds.) EKAW 2002. LNCS, vol. 2473, p. 235. Springer, Heidelberg (2002)
18. Omelayenko, B.: 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, Lyon, France, pp. 76–83 (2002)

19. Serafini, L., Stuckenschmidt, H., Wache, H.: A formal investigation of mapping languages for terminological knowledge. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence - IJCAI 2005, Edinburgh, UK (August 2005)
20. Stuckenschmidt, H., Uschold, M.: Representation of semantic mappings. In: Kalfoglou, Y., Schorlemmer, M., Sheth, A., Staab, S., Uschold, M. (eds.) Semantic Interoperability and Integration. Dagstuhl Seminar Proceedings, Germany, vol. 04391. IBFI, Schloss Dagstuhl (2005)
21. Tessaris, S., Franconi, E.: Rules and queries with ontologies: a unifying logical framework. In: Horrocks, I., Sattler, U., Wolter, F. (eds.) Description Logics. CEUR Workshop Proceedings, vol. 147, CEUR-WS.org. (2005)
22. Ullman, J.D.: Information integration using logical views. In: Afrati, F.N., Kolaitis, P.G. (eds.) ICDT 1997. LNCS, vol. 1186, pp. 19–40. Springer, Heidelberg (1997)
23. Warmer, J., Kleppe, A.: Object Constraint Language 2.0. MITP Verlag (2004)