The ‘Family of Languages’ Approach to Semantic Interoperability

Jérôme Euzenat[1]   Heiner Stuckenschmidt[2]

[1] INRIA Rhône-Alpes
655 avenue de l’Europe, Montbonnot Saint Martin
38334 Saint-Ismier, France
Jerome.Euzenat@inrialpes.fr

[2] Vrije Universiteit Amsterdam
De Boelelaan 1081a
1081 HV Amsterdam, the Netherlands
heiner@cs.vu.nl

Abstract. Different Semantic Web applications can use different knowledge representation languages. Exchanging knowledge thus requires techniques for ensuring semantic interoperability across languages. We present the ‘family of languages’ approach based on a set of knowledge representation languages whose partial ordering depends on the transformability from one language to another by preserving a particular formal property such as logical consequence. For the same set of languages, there can be several such structures based on the property selected for structuring the family. Properties of different strength allow performing practicable but well founded transformations. The approach offers the choice of the language in which a representation will be imported and the composition of available transformations between the members of the family.

1 Motivation

The World Wide Web is the largest information system that ever existed. Its size and heterogeneity makes ontology-based search and integration even more important than in other information systems. The “Semantic Web” [1] is supported by the annotation of web pages, containing informal knowledge as we know it now, with formal knowledge. These documents can reference each other and depend on ontologies and background knowledge. Taking advantage of the Semantic Web requires to be able to gather, compare, transform and compose these annotations. For several reasons (legacy knowledge, ease of use, timelessness, heterogeneity of devices and adaptability), it is unlikely that this formal knowledge will be encoded in the same language. Thus, the interoperability of formal knowledge languages must be studied to interpret the knowledge acquired through the Semantic Web. The problem of comparing languages is well known from the field of formal logic, but it takes a greater importance in the context of the Semantic Web.

We refer to the problem of comparing and interpreting the annotations at the semantic level, i.e., to ascribe to each imported piece of knowledge the correct interpretation, or set of models, as *semantic interoperability*. We will further characterize it below. There are several reasons for non interoperability and several approaches to semantic interoperability using different techniques [2, 3, 4]. In this chapter, we emphasize the mismatch between knowledge representation languages, leaving aside other important problems (e.g., axiomatization mismatches).

Consider a company developing applications involving printer maintenance that is neither a printer specialist nor a technical support specialist. It might have great interest in taking advantage of readily available and acknowledged ontologies. There is not a printer support ontology available so the company will have to merge different knowledge sources. Fortunately, the library of DAML (DARPA Agent Markup Language) contains an ontology describing a technical support application (http://www.daml.org/ontologies/69) and a printer ontology can be found at http://www.ontoknowledge.org/oil/case-studies/. However, the first ontology is encoded in DAML-ONT [5] and the second one in the OIL language [6].

The company wants to merge both representations for its own business but it also wants to check the consistency of the result. It thus requires an integration process through transformations that preserve the consequences and a path from that representation to a consistency checker that preserves consistency (so that, if the target representation is found inconsistent, then the source representation was too).

We discuss an approach that helps achieving semantic interoperability through a structured set of knowledge representation languages for which the properties of transformations from one language to another are known. The transformation of representations from one language to another (e.g., the initial languages in which the ontologies were formulated to the language used by the consistency checker) can take advantage of these characterized transformations in the family, minimizing the effort. This chapter first contrasts the 'family of languages' approach with other known approaches (§2). It then puts forth several structures for a 'family of languages' based on different properties (§3). We show that all these properties concur to semantic interoperability. Then, a concrete implementation of this approach is presented (§4) that we used to integrate languages in the example described above.

2 Approaches to language interoperability

We first give a few definitions of the kind of languages considered in this chapter. Then, we present several approaches for translating from one language to another.

2.1 Languages

For the purpose of the present chapter, a language L will be a set of expressions. A representation (r) is a set of expressions in L.

However, a language can be generated from a set of atomic terms and a set of constructors. A knowledge representations language mainly consists of operators that can be used to form complex terms (or formulas or classes) from simple ones.

For the sake of concreteness, this chapter will take advantage of the results obtained in the field of description logics to illustrate the 'family of languages' approach. This does not mean that the approach only applies to description logics, it can be applied as well to first-order logic [7] or conceptual graphs [8]. In the following we give an abstract definition of such a language:

Example 1 (Abstract description language [9]). *An abstract description language L is the set of L-expressions δ, over a set T of atomic terms (name of atomic classes) and a set F of operators, where L-expressions are recursively defined as follows:*

- *every $t \in T$ is an L-expression*
- *if δ is an L-expression, then $\neg\delta$ is also a L-expression*
- *it δ_1 and δ_2 are L-expressions, then $\delta_1 \wedge \delta_2$ and $\delta_1 \vee \delta_2$ are L-expressions*
- *if $f \in F_L$ in an n-ary operator and $\delta_1, \cdots, \delta_n$ are L-expressions then $f(\delta_1, \cdots, \delta_n)$ is an L-expression*

Note that the set T of atomic terms is independent of a specific language.

The concepts in an ontology can be intentionally described by L-expressions. Knowledge representation formalisms are subject to a well-known trade-off between expressiveness of representation and complexity of reasoning [10]. This trade-off leads to identify different formalisms that are suited for different application scenarios. This also holds for ontology languages: there is no language that fits all situations. Several approaches have been proposed for ensuring semantic interoperability. We present them from the standpoint of the transformation ($\tau : 2^L \longrightarrow 2^{L'}$) from one knowledge representation language (L) to another (L').

2.2 The Mapping Approach

The most direct and often used approach maps certain types of expressions in the source language to corresponding expressions in the target language. The formal nature of these mappings vary from purely syntactic matches to “theory interpretations” [7] with well defined properties. Therefore we characterize the mapping approach solely by the existence of a function that maps expressions from one language to another.

$$\exists\tau, (\forall r \subseteq L, \tau(r) \subseteq L') \tag{1}$$

The existence of a transformation τ from L to L' satisfying a property p is denoted by by $L \preceq_p L'$. In general, the property depends on the purpose of the transformation: change of language will require equivalence of meaning, importation will need preservation of consequences, simplification or abstraction will require preservation of models. Semantic properties will be considered more in depth in section 3.

This approach has the drawback of requiring transformations from any language to any other. It is thus not very reusable and requires to check individually the properties of the transformations. A current example of the mapping approach is described in [11].

2.3 The Pivot Approach

In order to reduce the number of transformations necessary to integrate languages, a special transformation architecture can be used. One of the most common is the use of a single pivot language P all other languages are translated to. To preserve semantics, this pivot language has to be able to express all other languages; an adequate property p must then be used. More formally, the pivot approach is characterized by the following assumption:

$$\exists! P, \forall L, (L \preceq_p P) \quad (2)$$

Probably the most prominent example of a pivot architecture is Ontolingua [12]. In this approach the Ontolingua language serves as a pivot language.

However, the approach has often been criticized for information loss during translations from the pivot language to less expressive languages.

2.4 The Layered Approach

A third approach to deal with semantic interoperability is the use of a layered architecture containing languages with increasing expressiveness (denoted by a corresponding p property). This approach has been proposed in order to avoid the problems arising from the need of using a very expressive language and to ensure tractable reasoning with the integrated languages. In such a layered architecture, representations can be translated, without semantic mismatch, into languages higher in the hierarchy. Formally speaking, the languages form a total order induced by the coverage relation.

$$\forall i, j, (i \leq j \Leftrightarrow L_i \preceq_p L_j) \quad (3)$$

A recent example of a layered architecture is the ontology language OIL [6] that has been built onto existing web standards. The idea is to use the W3C Standard RDF Schema as the language on the lowest layer and build additional language features on top of it. Doing this, it should be possible to translate RDF schema definitions into languages of the higher levels in order to enrich it. Recently, the idea of layering different languages has been taken up again by the standardization committee for the web ontology language standard OWL.

2.5 The 'Family of Languages' Approach

The 'family of languages' approach, presented in this chapter, considers a set of languages structured by a partial order ($\preceq_p$). This is more general than a total order, difficult to choose a priori, and more convenient for the users who can find languages closer to their needs (or, for an intermediate language, languages closer to their own languages).

For every two languages in the family a third language should exist that covers both of them.

$$\forall L, L', \exists L'', (L \preceq_p L'' \wedge L' \preceq_p L'') \quad (4)$$

This equation is different from equation (2) because L'' is dependent on L and L'. In fact, the 'family of languages' approach generalizes the pivot approach insofar as the pivot approach fulfills the 'family of languages' property, because the pivot language P can always be used as integration language. It also generalizes the layered approach, because in the layered framework the language that is higher in the hierarchy can be used as the L" language for equation (4). However, the 'family of languages' approach is more flexible, because it does not require a fixed pivot language nor a fixed layering of languages. On the contrary, any language that fulfills certain formal criteria can be used as integration language. We discuss these formal criteria in the following section.

Consequence. *The ‘family of languages’ property generalizes the pivot and the layered approach to language integration, i.e.,* $(2) \Rightarrow (4)$ *and* $(3) \Rightarrow (4)$.

The advantage of this approach is the ability to choose an entry (resp. exit) point into the family that is close to the input (resp. output) language. This enables the use of existing results on the family of languages for finding the best path from one language to another (at least by not choosing a very general pivot language). This path can be found with the help of the coverage relation, i.e. by finding some least upper language.

3 The Semantic Structure of a Family

A ‘family of languages’ is a set $\mathcal{L}$ of languages. The goal of the family is to provide an organization that allows to transform a representation from one language of the family to another. We thus use the notion of a transformation $\tau : 2^L \longrightarrow 2^{L'}$ from one representation into another as the basis of the family structure. It will then be easier to use this structure in transformations. The structure of a family of language is given by ordering this set with regard to available transformations satisfying some constraints (with the covering order $\preceq_p$).

In order to provide a meaningful definition of this ordering, we investigate orders based on the semantics of the languages as provided by model theory. In this framework, an interpretation I is a predicate over the assertions of a language. Naturally, this interpretation can be defined by structural rules such as those used for defining first-order logic interpretations or description logics.

Again, this can be illustrated in the description logics framework.

Example 2 (Abstract description model [9]). *An Abstract description model is of the form:*

$$\Im = \langle W, F^{\Im} = (f_i^{\Im})_{i \in I} \rangle$$

where W *is a nonempty set and* $f_i^{\Im}$ *are functions mapping every sequence* $\langle X_1, \cdots, X_{n_i} \rangle$ *of subsets of* W *to a subset of* W.

We can define the interpretation mapping in two steps. First we assume an assignment $\mathcal{A}$ mapping every $t \in T$ to a subset of W, then we define the interpretation mapping recursively as follows:

Example 3 (Semantics [9]). *Let* L *be a language and* $\Im = \langle W, F^{\Im} \rangle$ *an abstract description model. An assignment* $\mathcal{A}$ *is a mapping from the set of atomic term* T *to* 2^W. *The assignment of a subset of* W *to a term* t *is denoted by* $t^{\mathcal{A}}$. *The extension* $\delta^{\Im,\mathcal{A}}$ *of a* L*-expression is now defined by:*

1. $t^{\Im,\mathcal{A}} := t^{\mathcal{A}}$ *for every* $t \in T$
2. $(\neg\delta)^{\Im,\mathcal{A}} := W - \delta^{\Im,\mathcal{A}}$
3. $(\delta_1 \wedge \delta_2)^{\Im,\mathcal{A}} := \delta_1^{\Im,\mathcal{A}} \cap \delta_2^{\Im,\mathcal{A}}$
4. $(\delta_1 \vee \delta_2)^{\Im,\mathcal{A}} := \delta_1^{\Im,\mathcal{A}} \cup \delta_2^{\Im,\mathcal{A}}$

5. $f(\delta_1, \cdots, \delta_n)^{\Im,\mathcal{A}} := f^{\Im}(\delta_1^{\Im,\mathcal{A}}, \cdots, \delta_n^{\Im,\mathcal{A}})$ *for every* $f \in F$

The semantics definition given above is the basis for deciding whether an expression δ is satisfiable and whether an expression δ_1 follows from another expression δ_2. More specifically, the L-expression δ is satisfiable if $\delta^{\Im,\mathcal{A}} \neq \emptyset$, an L-expression δ_1 is implied by δ_2 (denoted as $\delta_1 \leq \delta_2$) if $\delta_1^{\Im,\mathcal{A}} \subseteq \delta_2^{\Im,\mathcal{A}}$.

A model of a representation $r \subseteq L$, is an interpretation I satisfying all the assertions in r. The set of all models of a representation r in L is denoted by $\mathcal{M}_L(r)$. An expression δ is said to be a consequence of a set of expression r if it is satisfied by all models of r (this is noted $r \models_L \delta$). The considerations below apply to first-order semantics but they can be extended.

The languages of a family $\mathcal{L}$ are interpreted homogeneously. This means that the constraints that apply to the definition of the interpretations are the same across languages of the family (and thus, if languages share constructs, like $\vee$, $\neg$, $\wedge$, they are interpreted in the same way across languages). We generally consider languages defined by a grammar with an interpretation function defined by induction over the structure of formulas (like description logics, first order logic or conceptual graphs). In this case, the homogeneity is provided by having only one interpretation rule per formula constructor.

This section will provide tools for defining the structure of a 'family of languages'. It will focus on a semantic structure that is prone to provide semantic interoperability. The structure is given by the coverage relation ($\preceq_p$ above) that can be established between two languages when a transformation from one to the other exists. In this section, the coverage relation will be characterized with regard to a property that it satisfies. The ultimate goal of these properties is to ensure the possible preservation of the consequences while transforming from a language to another.

3.1 Language inclusion

The simplest transformation is the transformation from one language to another syntactically more expressive one (i.e., which adds new constructors).

Definition 1 (Language inclusion). *A language L is included in another language L' iff $\forall \delta \in L, \delta \in L'$.*

The transformation is then trivial: it is identity. This trivial interpretation of semantic interoperability is one strength of the 'family of languages' approach because, in the present situation, nothing have to be done for gathering knowledge. This first property provides a first relation for structuring a family:

Definition 2 (Language-based Coverage).

$$L \preceq_{synt} L' \Leftrightarrow_{def} (L \subseteq L')$$

Language inclusion can be characterized in a more specific way on languages defined as a term algebra where the inclusion of languages can be reduced to the inclusion of the sets of term constructors.

Example 4 (The FaCT Reasoner). *The FaCT description logic reasoner implements two reasoning modules one for the language $\mathcal{SHF}$ and one for the language $\mathcal{SHIQ}$ which extends $\mathcal{SHF}$ with inverse roles and qualified number restrictions. As a consequence, $\mathcal{SHF}$ models can be handled by the $\mathcal{SHIQ}$ reasoner without change.*

3.2 Interpretation preservation

The former proposal is restricted in the sense that it only allows, target languages that contain the source language, though there could be equivalent non-syntactically comparable languages. This applies to the description logic languages $\mathcal{ALC}$ and $\mathcal{ALUE}$ which are known to be equivalent while none has all the constructors of the other[1]. This can be described as the equality of the Tarskian style interpretation for all the expressions of the language.

Definition 3 (Interpretation preservation). *A transformation τ preserves the interpretations iff*

$$\forall \delta \in L, \forall I, I(\tau(\delta)) = I(\delta)$$

In fact, there can be no other interpretations because of the requirement that the languages must be interpreted homogeneously.

Example 5 (Reasoning in Core-OIL). *The lowest layer of the ontology language OIL which has gained significant attention in connection with the Semantic Web is Core-OIL which provides a formal semantics for a part of RDF schema. In order to provide reasoning services, the language is translated into the logic $\mathcal{SHIQ}$ and the FaCT reasoner is used to provide the reasoning services [13]. Core-OIL can contain assertions restricting the applicability of a particular role ($R \leq (\texttt{domain}\, C)$). These assertions must be expressed in $\mathcal{SHIQ}$ which does not offer the domain constructor. It is thus translated into an assertion stating that for any term under $\top$, the range of the inverse of this relation is this particular domain. The translation contains the following interpretation-preserving mapping[2]:*

$$\tau(R \leq (\texttt{domain}\, C)) = \top \leq (\texttt{all}\, (\texttt{inv}\, R)\, C)$$

For that purpose, one can define $L \preceq_{int} L'$ if and only if there exists a transformation from L to L' that preserves the interpretations of the expressions.

Definition 4 (Interpretation-based coverage).

$$L \preceq_{int} L' \Leftrightarrow_{def} \textit{there is an interpretation preserving transformation } \tau : L \rightarrow L'$$

Obviously, language inclusion is stronger than interpretation preservation because the languages are homogeneous and the transformation is then reduced to identity.

Proposition 1 (Language-based coverage entails interpretation-based coverage). *If $L' \preceq_{synt} L$ then $L' \preceq_{int} L$.*

The τ transformation is, in general, not easy to produce (and it can generally be computationally expensive) but we have shown, in [14], how this can be practically achieved.

[1]This is true if we consider that the languages here are those described by their names: $\mathcal{AL}$+negation vs. $\mathcal{AL}$+disjunction+qualified existentials. Of course, because they have the same expressivity all the constructors of each language can be defined in the other. But this equivalence must be proved first.

[2]This is not sufficient for eliminating all occurrences of domain. For instance, `(all (domain C) C')` has to be transformed into `(or (not C) (all anyrelation C'))`. This does not work for *concrete domains* either.

3.3 Expressiveness

The previous property was subordinated to the coincidence of interpretation. In particular, the domain of interpretation has to be the same and the way entities are interpreted must coincide.

Franz Baader [15] has provided a definition of expressiveness of a first-order knowledge representation language in another by considering that a language can be expressed into another if there exists a way to transform any theory of the first into a theory of the second with the same models up to predicate renaming.

His definitions is based on the idea of "abstract models" in which a language is a pair made of a language L and a model selection function Mod_L which filters the acceptable models for the language (which are not all the first order models). Here, we consider as acceptable all the first-order models $\mathcal{M}_L(L)$.

Definition 5 (Expressibility modulo renaming [15]). *A language L is expressible in a language L' if and only if $\forall r \subseteq L$, $\exists$ a transformation $\tau : L \rightarrow L'$, $\exists \nu : Pred(r) \rightarrow Pred(\tau(r))$ such that $\forall m \in \mathcal{M}_L(r), \exists m' \in \mathcal{M}_{L'}(\tau(r)); \forall \delta \in L, m(\delta) = m'(\nu(\delta))$ and $\forall m' \in \mathcal{M}_{L'}(\tau(r)), \exists m \in \mathcal{M}_L(r); \forall \delta \in L, m(\delta) = m'(\nu(\delta))$. $Pred(r)$ is the set of atomic terms T found in the expression r.*

The notion of expressibility modulo renaming can best be explained by an example:

Example 6 (Eliminating undefined concepts axioms in $\mathcal{TF}$). *Bernhard Nebel has shown that the transformation from a T-Box with the introduction of undefined (primitive) concepts can be translated into T-box with additional concepts (primitive component concepts). So, each undefined concept (only specified by implication $\leq$), is introduced by a definition (an equivalence axiom using $\equiv$) as the conjunction (`and`) of its known subsumers and an undefined part (expressed with an overline here):*

$$\tau(Man \leq Human) = [Man \equiv (\texttt{and}\ Human\ \overline{Man})]$$

This transformation preserves expressiveness [15].

We do not want to consider renaming here (it involves knowing what to rename and using the $Pred$ function which denotes the set of predicates used in an expression). So, expressibility is redefined by simply using the transformation τ and ignoring ν. We also strengthened this definition by using a global transformation (independent from the theory to be transformed).

Definition 6 (Expressibility modulo transformation). *A language L is expressible in a language L' if and only if $\exists$ a transformation $\tau : L \rightarrow L'$, such that $\forall r \subseteq L$, $\forall m \in \mathcal{M}_L(r), \exists m' \in \mathcal{M}_{L'}(\tau(r)); \forall \delta \in L, m(\delta) = m'(\tau(\delta))$ and $\forall m' \in \mathcal{M}_{L'}(\tau(r)), \exists m \in \mathcal{M}_L(r); \forall \delta \in L, m(\delta) = m'(\tau(\delta))$*

Naturally, expressibility modulo transformation entails expressibility modulo renaming.

Definition 7 (Expressibility-based coverage).

$$L \preceq_{exprt} L' \Leftrightarrow_{def} L \text{ is expressible (modulo transformation) in } L'$$

The following proposition is easily obtained by noting that an interpretation-preserving transformation entails expressibility modulo transformation. So the corresponding model, can be the model itself (or an extension of itself to formulas missing from the initial language).

Proposition 2 (Interpretation-based coverage entails expressivity-based coverage). *If $L \preceq_{int} L'$, then $L \preceq_{exprt} L'$.*

The only asymmetry of these definitions is in the orientation of the transformation. Basically, the two theories are required to have a symmetric correspondence between models. There is certainly room for relaxing this constraint. Our definition of epimorphic transformations goes this way.

3.4 Epimorphic transformations

Full isomorphism between the models of a representation and its transformations is prone to preserve a major part of the meaning. However, an isomorphism would constrain the two sets of models to have the same cardinality. This is relatively artificial. We relax this constraint by asking each model of the transformed representation to be closely related to one model of the source representation. This can be useful when one does want to consider axiomatizations of different nature (e.g., when objects are taken as relations and vice versa as in dual representations of graphs).

Definition 8 (Model epimorphism). *A model epimorphism $\pi : M \to M'$ is a surjective map from a set of models M to another set of models M'.*

Model epimorphisms ensure that all models of the transformed representation are comparable to some model of the source representation.

Definition 9 (Epimorphic transformation). *A transformation τ is epimorphic iff there exists a model epimorphism $\pi : \mathcal{M}_{L'}(\tau(r)) \to \mathcal{M}_L(r)$ such that $\forall r \subseteq L, \forall m' \in \mathcal{M}_{L'}(\tau(r))$ and $\forall \delta \in L$, $\pi(m') \models \delta \Rightarrow m' \models \tau(\delta)$*

This kind of transformation allows the generated representation to have many more very different models than the initial representation, but constrain each of these models to preserve all the consequences of one of the models of the initial representation.

Definition 10 (Correspondance-based coverage).

$$L \preceq_{epi} L' \Leftrightarrow_{def} \textit{there is an epimorphic transformation } \tau : L \to L'$$

This basically ensures that the transformation does not loose information (i.e., does not generate unrelated models). The following proposition is obtained by building the epimorphism from the corresponding models in the second equation of definition 6.

Proposition 3 (Expressibility-based coverage entails correspondance-based coverage). *If $L \preceq_{exprt} L'$, then $L \preceq_{epi} L'$.*

3.5 Consequence preservation

Consequence preservation can be considered the ultimate goal of semantic interoperability: it denotes the fact that the consequences (i.e., the formulas satisfied by all models) of the source and the target representations are preserved (modulo transformation).

Definition 11 (Consequence preservation). *A transformation τ is said consequence-preserving iff $\forall r \subseteq L, \forall \delta \in L, r \models_L \delta \Rightarrow \tau(r) \models_{L'} \tau(\delta)$*

If τ is a consequence-preserving transformation, then for any $r \subseteq L$, it is said that $\tau(r)$ is a conservative extension of r modulo τ.

Example 7 (Translating from $\mathcal{DLR}$ to $\mathcal{SHIF}$). *In order to decide query containment in $\mathcal{DLR}$, the authors of [16] define a mapping from the $\mathcal{DLR}$ logic (which introduces n-ary relations) to $\mathcal{CPDL}$ (Propositional Dynamic Logic with Converse)[3]. These relations are represented by concepts with exactly n features to the components of the relation.*

This transformation is a consequence preserving transformation.

This definition allows the definition of a consequence-based coverage as usual:

Definition 12 (Consequence-based coverage).

$$L \preceq_{csq} L' \Leftrightarrow_{def} \text{there is a consequence preserving transformation } \tau : L \rightarrow L'$$

Correspondance-based coverage is stronger than consequence-based coverage because it already includes the notion of consequence-preservation. The point is that there can be "more" models in L' than in L, but they satisfy the same assertions as one model in L, thus they cannot inhibit any consequence.

Proposition 4 (Correspondance-based coverage entails consequence-based coverage). *If $L \preceq_{epi} L'$, then $L \preceq_{csq} L'$.*

It is known that expressivity modulo renaming alone does not necessarily entail consequence preservation [15].

3.6 Consistency preservation

Preserving consistency is a very weak property (it is true of any transformation that forgets knowledge). However, transformations that preserve consistency can be used for checking the inconsistency of a knowledge base: if the target knowledge base is inconsistent, then the source was inconsistent too.

Definition 13 (Consistency preservation). *A transformation τ is said to be consistency-preserving iff $\forall r \subseteq L, \mathcal{M}_L(r) \neq \emptyset \Rightarrow \mathcal{M}_{L'}(\tau(r)) \neq \emptyset$*

Example 8 (Reasoning in Standard-OIL). *The second layer of the OIL language called Standard-OIL provides an expressive language for building ontologies. Again, the language is translated into $\mathcal{SHIQ}$ in order to provide inference services. Standard-OIL also includes capabilities for expressing assertional knowledge and instances in concept definitions. As the FaCT reasoner does not support instance reasoning, the translation from Standard-OIL to $\mathcal{SHIQ}$ includes some mappings that do not preserve the complete semantics, but preserve satisfiability [13].*

$$\tau((\texttt{one}-\texttt{of}\, i_1\, i_2)) = (\texttt{or}\, I_1\, I_2)$$

This transformation replaces the enumeration of instances by a disjunction of concepts with the same name (here capitalized).

[3] The mapping is defined for the restriction introduced in [17] where $\mathcal{DLR}$ does not contain regular path expressions.

Consistency-based coverage is defined as usual.

Definition 14 (Consistency-based coverage).

$$L \preceq_{sat} L' \Leftrightarrow_{def} \textit{there is a consistency-preserving transformation } \tau : L \rightarrow L'$$

Proposition 5 (Expressivity-based coverage entails consistency-based coverage). *If* $L \preceq_{exprt} L'$, *then* $L \preceq_{sat} L'$.

3.7 Composition of properties

As a consequence, all the coverage relations concur to providing the families of language with a structure which enriches the basic syntactic structure usually proposed for these languages.

This defines a hierarchy of more and more constrained structure for the 'family of languages'. Establishing one of these structures can be more or less difficult, so it is important to be able to find the more adapted to a particular application (the p property of the beginning) and not a more powerful one. This permits to have the best effort in looking for a path from one language of the family to another.

There can be other useful properties (and thus other structures) that anyone can integrate in the structure of a family. These properties do not have to be totally ordered from the strongest to the weakest. However, for being useful to semantic interoperability, new properties should entail some of the properties above.

These structures enable the composition of transformations while knowing their properties. The following table provides the strongest property satisfied by the composition of two transformations given their properties.

	$\preceq_{synt}$	$\preceq_{int}$	$\preceq_{exprt}$	$\preceq_{sat}$	$\preceq_{epi}$	$\preceq_{csq}$
$\preceq_{synt}$	$\preceq_{synt}$	$\preceq_{int}$	$\preceq_{exprt}$	$\preceq_{sat}$	$\preceq_{epi}$	$\preceq_{csq}$
$\preceq_{int}$	$\preceq_{int}$	$\preceq_{int}$	$\preceq_{exprt}$	$\preceq_{sat}$	$\preceq_{epi}$	$\preceq_{csq}$
$\preceq_{exprt}$	$\preceq_{exprt}$	$\preceq_{exprt}$	$\preceq_{exprt}$	$\preceq_{sat}$	$\preceq_{epi}$	$\preceq_{csq}$
$\preceq_{sat}$	$\preceq_{sat}$	$\preceq_{sat}$	$\preceq_{sat}$	$\preceq_{sat}$	$\emptyset$	$\emptyset$
$\preceq_{epi}$	$\preceq_{epi}$	$\preceq_{epi}$	$\preceq_{epi}$	$\emptyset$	$\preceq_{epi}$	$\preceq_{csq}$
$\preceq_{csq}$	$\preceq_{csq}$	$\preceq_{csq}$	$\preceq_{csq}$	$\emptyset$	$\preceq_{csq}$	$\preceq_{csq}$

In summary, the semantic structure of a 'family of languages' provides us with different coverage criteria all based on the notion of transformability. These notions of coverage do not only give us the possibility to identify and prove coverage, they also specify a mechanism for transforming the covered into the covering language. Therefore we can show that a suitable language can be generated and how the generation is being performed. In the next section we present an instantiation of this approach.

4 Implementing the Approach

The 'family of languages' approach can take advantage of many knowledge representation formalisms that have been designed in a modular way. A concrete example of a family is presented below through an example (§4.2) using the DLML encoding of description logics supplied with transformations (§4.1).

4.1 A concrete 'family of languages'

DLML [18] is a modular system of document type descriptions (DTD) encoding the syntax of many description logics in XML. It takes advantage of the modular design of description logics by describing individual constructors separately. The specification of a particular logic is achieved by declaring the set of possible constructors and the logic's DTD is automatically build up by just assembling those of elementary constructors. The actual system contains the description of more than 40 constructors and 25 logics. To DLML is a associated a set of transformations (written in XSLT) allowing to convert a representation from a logic to another.

The first application is the import and export of terminologies from a description logic system. The FaCT system [19] has already developed that aspect by using such an encoding. We also developed, for the purpose of the examples presented here, the transformations from OIL and DAML-ONT to DLML. These transformation are simple XSLT stylesheets.

4.2 Example

Recall the example of the company which needs a printer support ontology and has to merge different knowledge sources mentioned in the introduction: the technical support application ontology in DAML-ONT and the printer ontology written in the OIL language [6]. It also wants to translate the merged ontology into the $\mathcal{SHIQ}$ language in order to check the consistency of the result. The transformation must be consistency preserving. The translation methodology, from one language to another, consists of choosing the input and output languages within the family. The source representation will be translated in the input language and the target representation will be imported from the output language. The input languages are obviously DLML counterparts of OIL and DAML-ONT and the translation is easily carried out because both languages have been inspired by description logics. The target language will be the DLML language corresponding to $\mathcal{SHIQ}$, supported by the FaCT reasoner.

Then, a path from the input language to the output language which satisfies required properties has to be found in the family of languages used. This path is presented below.

The first goal will be achieved by translating the DAML-ONT and OIL representations in a representation language (called G) which encompasses all the constructs of the initial languages. The transformations depend only on the language inclusion property between the two input languages and G.

The second goal will be achieved by composing three DLML transformations that rewrite some representations using a particular construct to representations without it, suitable to be checked for consistency by the FaCT reasoner. This implements transformations already at work in the OIL-based tools [13]. It thus chain the following transformations (summarized by figure 1):

domain2allinv which replaces `domain` restrictions on role definitions by a general constraint applying to the restricted terms (through the restriction of the inverse role codomain): this transformation is interpretation-preserving (see example 5);

oneof2ornot which replaces enumerated domains (`oneof`) by disjunctive concepts whose disjuncts represent the elements of this domain: this transformation is only consistency preserving (see example 8);

cexcl2not which replaces concept exclusion (introduced by the previous transformation) by conjunction with the negated concept. This transformation is also interpretation preserving. (as $(\texttt{disjoint}\,\texttt{C}\,\texttt{D}) \Leftrightarrow C \equiv \neg D$)

$$
\begin{tikzcd}
DAML \arrow[r] & L_{DAML} \arrow[dr, "\preceq_{synt}"'] \arrow[rr, phantom, "\texttt{merge}" below] & & L_{OIL} \arrow[dl, "\preceq_{synt}"] & OIL \arrow[l] \\
& & G = L_{DAML} \cup L_{OIL} \arrow[d, "\texttt{domain2allinv}"', "\preceq_{int}"] & & \\
& & L_{int} \arrow[d, "\texttt{oneof2ornot}"', "\preceq_{sat}"] & & \\
& & L'_{int} \arrow[d, "\texttt{cexcl2not}"', "\preceq_{int}"] & & \\
& & L_{\mathcal{SHIQ}} \arrow[rr] & & \mathcal{SHIQ}
\end{tikzcd}
$$

Figure 1: The transformation flow involved in importing the two ontologies to $\mathcal{SHIQ}$.

Thus the import of OIL and DAML-ONT into $\mathcal{SHIQ}$ described above is consistency preserving.

5 Conclusion

The 'family of languages' approach is one approach for facilitating the exchange of formally expressed knowledge in a characterized way. It generalizes previous proposals for translation architectures and provides a greater flexibility in terms of languages that can be used for the integrated models. This approach is not exclusive to other approaches like direct translation or pivot approaches. But it has several advantages over other solutions to the semantic interoperability problem because it allows users:

- to translate to closer languages among many of them;
- to share and compose many simple transformations for which the properties are known and the transformations available;
- to select the transformations to be used with regard to the kind of properties that are required by the transformation.

The 'family of languages' approach is thus a tool for better 'ontology engineering'.

We have presented this approach in a unified framework and proposed a first tower of structure for the 'family of languages' based on the properties that are satisfied by the transformations. Different semantic relations can be used to establish the structure of a 'family of languages' and ensure formal properties of transformations between languages. Many work is still required for characterizing other useful properties, including properties on the syntax of the languages.

As shown, the approach can easily be implemented using existing web technologies such as XML and XSLT, but also provides an infrastructure for ensuring formal properties by proving the formal properties of transformations between concrete languages.

Acknowledgements

This work has benefited from the support of the European Union OntoWeb thematic network (IST-2000-29243) and the PAI Procope program of the Ministère des Affaires Étrangères (MAE) and Deutscher Akademischer Austauschdienst (DAAD).

References

[1] Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. Scientific american **279** (2001) 35–43 http://www.scientificamerican.com/2001/0501issue/0501berners-lee.html.

[2] Masolo, C.: Criteri di confronto e costruzione di teorie assiomatiche per la rappresentazione della conoscenza: ontologie dello spazio e del tempo. Tesi di dottorato, Universit di Padova, Padova (IT) (2000)

[3] Ciocoiu, M., Nau, D.: Ontology-based semantics. In: Proceedings of 7th international conference on knowledge representation and reasoning (KR), Breckenridge, (CO US). (2000) 539–546 http://www.cs.umd.edu/ nau/papers/KR-2000.pdf.

[4] Stuckenschmidt, H., Visser, U.: Semantic translation based on approximate re-classification. In: Proceedings of the KR workshop on semantic approximation granularity and vagueness, Breckenridge, (CO US). (2000) http://www.tzi.de/ heiner/public/ApproximateReClassification.ps.gz.

[5] McGuinness, D., Fikes, R., Stein, L.A., James, H.: DAML-ONT: An ontology language for the semantic web. In Fensel, D., Hendler, J., Lieberman, H., Walhster, W., eds.: The semantic web: why, what, and how? The MIT press (2002) to appear.

[6] Fensel, D., Horrocks, I., Harmelen, F.V., Decker, S., Erdmann, M., Klein, M.: Oil in a nutshell. In: 12th International Conference on Knowledge Engineering and Knowledge Management EKAW 2000, Juan-les-Pins, France (2000)

[7] Enderton, H.: A mathematical introduction to logic. Academic press (1972) revised 2001.

[8] Baget, J.F., Mugnier, M.L.: The $\mathcal{SG}$ family: extensions of simple conceptual graphs. In: Proc. 17th IJCAI, Seattle (WA US). (2001) 205–210

[9] Baader, F., Lutz, C., Sturm, H., Wolter, F.: Fusions of description logics and abstract description systems. Journal of Artificial Intelligence Research **16** (2002) 1–58

[10] Levesque, H., Brachmann, R.: A Fundamental Tradeoff in Knowledge Representation and Reasoning (Revised Version). In: Readings in Knowledge Representation. Morgan Kaufmann Publishers, San Mateo (1985) 31–40

[11] Chalupsky, H.: OntoMorph: a translation system for symbolic knowledge. In: Proceedings of 7th international conference on knowledge representation and reasoning (KR), Breckenridge, (CO US). (2000) 471–482

[12] Gruber, T.: A translation approach to portable ontology specifications. Knowledge Acquisition **5** (1993) 199–220

[13] Horrocks, I.: A denotational semantics for Standard OIL and Instance OIL (2000) http://www.ontoknowledge.org/oil/downl/semantics.pdf.

[14] Euzenat, J.: Towards a principled approach to semantic interoperability. In Gomez Perez, A., Gruninger, M., Stuckenschmidt, H., Uschold, M., eds.: Proc. IJCAI 2001 workshop on ontology and information sharing, Seattle (WA US). (2001) 19–25

[15] Baader, F.: A formal definition of the expresive power of terminological knowledge representation languages. Journal of logic and computation **6** (1996) 33–54

[16] Calvanese, D., De Giacomo, G., Lenzerini, M.: On the decidability of query containment under constraints. In: Proc. of the 17th ACM SIGACT SIGMOD SIGART Sym. on Principles of Database Systems (PODS'98). (1998) 149–158

[17] Horrocks, I., Sattler, U., Tessaries, S., Tobies, S.: Query containment using a $\mathcal{DLR}$ abox - preliminary version. LCTS Report 99-15, RWTH Aachen (1999)

[18] Euzenat, J.: Preserving modularity in XML encoding of description logics. In McGuinness, D., Patel-Schneider, P., Goble, C., Möller, R., eds.: Proc. 14th workshop on description logics (DL), Stanford (CA US). (2001) 20–29

[19] Bechhofer, S., Horrocks, I., Patel-Schneider, P., Tessaris, S.: A proposal for a description logic interface. In: Proc. int. workshop on description logics, Linkping (SE). Number CEUR-WS-22 (1999) http://SunSITE.Informatik.RWTH-Aachen.DE/Publications/CEUR-WS/Vol-22/bechhofer.ps.