Reasoning in Metamodeling Enabled Ontologies

Nophadol Jekjantuk[1], Gerd Groener[2], and Jeff. Z. Pan[1]

[1] University of Aberdeen, United Kingdom
[2] University of Koblenz-Landau, Germany

Abstract. Ontologies are expected to play an important role in many application domains, as well as in software engineering in general. One problem with using ontologies within software engineering is that while UML, a widely used standard for specifying and constructing the models for a software-intensive system, has a four-layer metamodeling architecture, the standard OWL Web Ontology Language does not support reasoning over layered metamodels. OWL2 provides simple metamodeling by using a punning approach, however, the interpretation function is different based on the context, which leads to non-intuitive results. The OWL FA Language has a well defined metamodeling architecture. However, there is no study and tool for support reasoning over OWL FA. In this paper, we briefly discuss some reasoning tasks in OWL FA. We also provide OWL FA Tool kit, a simple tool kit for manipulating and reasoning with OWL FA.

1 Introduction

Metamodeling appeals in many application areas (such as UML [10], Model Driven Architecture [2], XML [12] and E-Commerce). It is not only the underpinning of modeling languages such as UML, but also central to OMG's MDA-based computing.

The W3C Web Ontology Language (OWL) [9] in combination with reasoning is already used in various other research areas like in model-driven software engineering in order to exploit the expressiveness of OWL and the usage of inference. However, the lack of a formal OWL language or OWL extension which supports metamodeling is an obstacle for the usage of OWL in other complex application areas.

The Resource Description Framework (RDF) and OWL Full support metamodeling by allowing users to use the built-in vocabulary without restriction, which leads to either undecidability or a redefinition of the semantics of the language. For example, given: `rdfs:domain` `rdfs:subPropertyOf` `rdf:type` and R `rdfs:domain` C, we can infer that R `rdf:type` C. It obvious that the semantics of `rdfs:domain` has changed. OWL [9] provides formal semantics focused on conceptual modeling and adaptability of inference using DL reasoners and reasoning algorithms, but OWL does not support layered reasoning. OWL2 provides simple metamodeling with semantics which correspond to the contextual semantics defined in [5], however, it has been shown in [8] that these can lead to non-intuitive results.

Proceedings of OWL: Experiences and Directions 2009 (OWLED 2009),
Rinke Hoekstra and Peter F. Patel-Schneider, editors. http://www.webont.org/owled/2009

For example, the following axioms state that *GiantPanda* is an *Endangered* species, and that *Buddy* is a *GiantPanda*:

$ClassAssertion(GiantPanda\ Buddy)$ ...(1)
$ClassAssertion(Endangered\ GiantPanda)$ ...(2)

The axioms (1), (2) could be interpreted by DL reasoner as follows:

$ClassAssertion(Cls-GiantPanda\ Ind-Buddy)$ ...(3)
$ClassAssertion(Cls-Endangered\ Ind-GiantPanda)$ ...(4)

Because the names of concepts and individuals are not interact with each other even they share the same name, this type of metamodeling is often referred to as punning. Let us consider the following axioms:

$EquivalentClasses(GiantPanda\ Panda)$ ...(5)
$DifferentIndividuals(GiantPanda\ Panda)$ ...(6)

The axioms (5), (6) could be safely added to the ontology in contextual semantics, but under layered semantics this ontology is inconsistent because (5) indicates two concepts *GiantPanda* and *Panda* are equivalent but (6) indicates that two meta-individuals are different.

In this paper, we present modeling and reasoning algorithms for OWL FA knowledge bases. For the reasoning service, An OWL FA ontology is transformed to a set of OWL DL ontologies then existing DL reasoners are applied to the transformed knowledge base. The syntax and semantics of OWL FA is described in Section 2. Modeling in OWL FA is demonstrated in Section 3 In Section 4 reasoning in OWL FA is described. This contains a reduction to OWL DL knowledge bases and reasoning algorithms in order to propagate conditions between different layers.

2 OWL FA syntax and semantics

OWL FA [8] enables metamodeling. It is an extension of OWL DL, which refers to the description logic $\mathcal{SHOIN}(\mathcal{D})$. Ontologies in OWL FA are represented in a layered architecture. This architecture is mainly based on the architecture of RDFS(FA) [7].

OWL FA specifies a layer number in class constructors and axioms to indicate the layer they belong to.

Let $\mathsf{CN} \in \mathbf{V}_{\mathrm{C_i}}$ be an atomic class name in layer i ($i \geq 0$), R an OWL FA-property in layer i, $o \in \mathbf{I}$ an individual, $T \in \mathbf{V}_{\mathrm{DP}}$ a datatype property name, and C, D OWL FA-classes in layer i. Valid OWL FA-classes are defined by the abstract syntax:

$$\begin{aligned} C ::= {} & \top_{\mathrm{i}} \mid \bot \mid \mathsf{CN} \mid \neg_{\mathrm{i}} C \mid C \sqcap_{\mathrm{i}} D \mid C \sqcup_{\mathrm{i}} D \mid \{o\} \mid \exists_{\mathrm{i}} R.C \mid \\ & \mid \forall_{\mathrm{i}} R.C \mid \leqslant_{\mathrm{i}} nR \mid \geqslant_{\mathrm{i}} nR \mid \\ & (\text{if i} = 1)\ \exists_1 T.d \mid \forall_1 T.d \mid \leqslant_1 nT \mid \geqslant_1 nT \end{aligned}$$

The semantics of OWL FA is a model theoretic semantics, which is defined in terms of interpretations. In other words, The semantics of two layers which can be considered as TBox and ABox are same as in OWL DL. The idea of OWL FA is that the interpretation depends on the layer but is still an OWL DL interpretation. Given an OWL FA alphabet $\mathbf{V}$, a set of built-in datatype names $\mathbf{B} \subseteq \mathbf{V}_{\mathrm{D}}$ and an integer k ≥ 1, an *OWL FA interpretation* is a tuple of the form pair $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$, where $\Delta^{\mathcal{J}}$ is the domain (a non-empty set) and $\cdot^{\mathcal{J}}$ is the interpretation. In the rest of the paper, we assume that i is an integer such that $1 \leq i \leq k$. The interpretation function can be extended to give semantics to OWL FA-properties and OWL FA-classes. Let $RN \in \mathbf{V}_{\mathrm{AP_i}}$ be an abstract property name in layer i and R be an abstract property in layer i. Valid OWL FA abstract properties are defined by the abstract syntax: $R ::= RN \mid R^{-}$, where for some $x, y \in \Delta^{\mathcal{J}}_{A_{\mathrm{i}-1}}$, $\langle x, y\rangle \in R^{\mathcal{J}}$ iff $\langle y, x\rangle \in R^{-\mathcal{J}}$. Valid OWL FA datatype properties are datatype property names. The interpretation function is explained in detail in [8].

An interpretation $\mathcal{J}$ satisfies an ontology Σ if it satisfies all the axioms in Σ. Σ is *satisfiable* (*unsatisfiable*) iff there exists (does not exist) such an interpretation $\mathcal{J}$ that satisfies Σ. Let C, D be OWL FA-classes in layer i, C is *satisfiable* w.r.t. Σ iff there exist an interpretation $\mathcal{J}$ of Σ s.t. $C^{\mathcal{J}} \neq \emptyset_{\mathrm{i}}$; C subsumes D w.r.t. Σ iff for every interpretation $\mathcal{J}$ of Σ we have $C^{\mathcal{J}} \subseteq D^{\mathcal{J}}$.

3 Modeling the Metamodeling enabled Ontologies

In this section, we present two ways of a model metamodeling enabled ontologies.

3.1 Modeling with OWL FA

One way to model a metamodeling enabled ontology is to use OWL FA syntax. Although the layer numbers can/should be encapsulated by tools, there are two rules of thumb to help users to get the number right. Firstly, the subscript numbers are only used to indicate a sub-ontology (e.g. $\mathcal{O}_2$), a constructor (e.g. $\exists_2$), or axiom symbols (e.g. $\sqsubseteq_2$, $:_2$) in a sub-ontology. Secondly, subscript numbers of the constructors and axiom symbols indicate the sub-ontology that the class descriptions constructed by these constructors and axioms belong to. The following example shows how to model an Endangered Species ontology with the OWL FA functional syntax. The main reason for using functional syntax is that it is obvious to see which layer they belong to. Example 1 shows the Endangered Species expressed with OWL FA syntax.

Example 1. Endangered Species ontology expressed in OWL FA:

$Declaration(Class2(Endangered))$
$ClassAssertion2(Endangered\ GiantPanda)$
$Declaration(Class1(GiantPanda))$
$ClassAssertion1(GiantPanda\ Buddy)$
$EquivalentClasses1(GiantPanda\ Panda)$

3.2 Modeling with OWL DL

Another way to model a metamodeling enabled ontology is to divide an OWL FA ontology $\mathcal{O}$ into a set of sub-ontologies $\mathcal{O}_1, \ldots, \mathcal{O}_k$. In the sub-ontology $\mathcal{O}_i$ (1 $\leq$ i $\leq$ k), (meta-) objects are resources in layer (i $-$ 1), while (meta-) classes and properties are resources in layer i. The advantage of this method is we can use existing OWL editor to model each sub-ontology. However, in order to preserve the OWL FA spirit, users have to make sure that the meta-individuals name in the meta-ontology are the same as class or property names in the domain ontology. Example 2 shows the Endangered Species expressed with OWL DL syntax in separate ontology.

Example 2. Endangered Species ontology expressed in OWL DL:

$\mathcal{O}_2$:
$Declaration(Class(Endangered))$
$ClassAssertion(Endangered\ GiantPanda)$
$\mathcal{O}_1$:
$Declaration(Class(GiantPanda))$
$ClassAssertion(GiantPanda\ Buddy)$
$EquivalentClasses(GiantPanda\ Panda)$

4 Reasoning in OWL FA

Now we briefly discuss some reasoning tasks in OWL FA. According to the layered architecture, the knowledge base Σ in OWL FA is divided into a sequence of knowledge bases $\Sigma_1, \ldots \Sigma_k$, whereas k is the number of layers. Since individuals in layer $i+1$ can be classes and properties in layer i, this also affects the axioms of each layer. Individual axioms in the knowledge base Σi+1 can be considered as class axioms in the knowledge base Σ i.

In an OWL FA knowledge base Σ, the Σ_2, ..., Σ_k are $\mathcal{SHIQ}$ knowledge bases because having a nominal in higher layer can lead to unsatisfiable of the knowledge bases. An interesting feature of Σ is that there could be more interactions between Σ_i and $\Sigma_{\mathrm{i}+1}$.

4.1 Preprocessing

In this section, we discuss how to reduce the reasoning problem in OWL FA into reasoning problem in OWL DL.

Definition 1. *Let* $\Sigma = \langle \Sigma_1, \ldots, \Sigma_\mathrm{k} \rangle$ *be an OWL FA knowledge base, where each of* $\Sigma_1, \ldots, \Sigma_\mathrm{k}$ *is consistent.* $\Sigma^* = \langle \Sigma_1^*, \ldots, \Sigma_\mathrm{k}^* \rangle$*, called the* explicit knowledge base*, is constructed by making all the implicit atomic class axioms, atomic property axioms, individual equality axioms explicit.* $\diamond$

As we have a finite set of vocabulary, we have the following Lemma.

Lemma 1. *Given an OWL FA knowledge base* $\Sigma = \langle \Sigma_1, \ldots, \Sigma_k \rangle$. *The explicit knowledge base* Σ^* *(OWL DL knowledge base) can be calculated from* Σ *in finite steps.*

Proof. When $k = 1$, we can calculate the explicit knowledge base Σ_1^* in finite steps because the sets of names of classes (in layer 1), roles (in layer 1) and individuals are finite. When $k > 1$, let us assume that we can calculate the explicit knowledge bases $\Sigma_1', ..., \Sigma_i'$ (where $1 \leq i < k$) from $\Sigma_1, \ldots, \Sigma_i$ in finite steps. We add all the class and property equality axioms in Σ_i' to Σ_{i+1}. If the updated Σ_{i+1} is consistent, Then,we can make the implicit individual equality axioms (if any) explicit and add new class and property equality axioms into Σ_i'. Thus, we can calculate Σ_1'', ..., Σ_i'' in finite steps. As the individual names in Σ_{i+1} are finite, we can calculate the explicit knowledge bases $\Sigma_1^*, \ldots, \Sigma_{i+1}^*$ in finite steps.

Note that if a class description is not defined in Σ_i (i.e., if it is not equivalent to any atomic class), it is not represented by any meta-individual in Σ_{i+1}. This suggests the connections between Σ_i and Σ_{i+1} are atomic classes and properties in Σ_i, which are meta-individuals in Σ_{i+1}.

We now present the algorithm `Reduce`, that will reduce an OWL FA knowledge base Σ into a set of OWL DL knowledge bases $\langle \Sigma_1^*, \ldots, \Sigma_k^* \rangle$. This algorithm is based on definition 1 and lemma 1. The algorithm takes OWL FA KB Σ as input and returns a set of OWL DL KB $\langle \Sigma_1, \ldots, \Sigma_k \rangle$. The Algorithm `Reduce` is shown in Figure 1.

The following theorem shows the termination of the algorithm `Reduce`, applied to an OWL FA KB Σ.

Theorem 1. *Given an OWL FA knowledge base* $\Sigma = \langle \Sigma_1, \ldots, \Sigma_k \rangle$, *then* `Reduce`$(\Sigma)$ *terminates.*

Proof. Termination of algorithm `Reduce` is straightforward from Lemma 1, which we can construct $\langle \Sigma_1^*, \ldots, \Sigma_k^* \rangle$ from Σ in finite step and a sets of class, property and individual equality axioms are finite. Thus, algorithm `Reduce` always terminates.

4.2 Consistency Checking

In this section, We present the algorithm `Consistent`, that will check the consistency of OWL FA knowledge base Σ. We can reduce an OWL FA knowledge base to a collection of OWL DL knowledge bases, therefore existing DL reasoner capabilities can be used. Consistency checking for OWL FA is done in two steps: First, we check the syntax of OWL FA. For example, $\Sigma = \{C \sqsubseteq_2 D, C \sqsubseteq_3 E\}$ is non-well formed because in OWL FA we do not allow OWL class construct between layer except an instance-of relationship. Secondly, we check the consistency of each OWL DL-knowledge base that is computed from the OWL FA knowledge base with an existing DL reasoner. The Algorithm `Consistent` is shown in Figure 2.

Algorithm Reduce(Σ)
Input: OWL FA KB Σ
Output: a set of OWL DL KB$\langle \Sigma_1^*, \ldots, \Sigma_{\mathrm{k}}^* \rangle$.
begin
  Collect axioms from the layer number and store in $L_0, ..., L_n$
  Create knowledge base $\Sigma_i^* = (L_0, L_1), ..., \Sigma_k^* = (L_{n-1}, L_n)$
  repeat
    $CE_i = \emptyset, PE_i = \emptyset$ *and* $OE_i = \emptyset$
    Compute the explicit knowledge Σ_i^* with DL reasoner
    for each $\Sigma_i^* (1 \leq i \leq k)$ **do**
      Identify the new concept equality in Σ_i^* and store in CE_i
      Identify the new property equality in Σ_i^* and store in PE_i
      Identify the new individual equality in Σ_i^* and store in OE_i
      for each CE_i, PE_i and OE_i **do**
        if $i = 1$ **then**
          Add CE_i and PE_i as individual equality into Σ_{i+1}^*
        else if $i =$ k **then**
          Add OE_i as property or concept equality into Σ_{i-1}^*
        else
          Add CE_i and PE_i as individual equality into Σ_{i+1}^*
          Add OE_i as property or concept equality into Σ_{i-1}^*
        end if
      end for
    end for
  until $CE_i = \emptyset$ && $PE_i = \emptyset$ && $OE_i = \emptyset$
  Return $\langle \Sigma_1^*, \ldots, \Sigma_{\mathrm{k}}^* \rangle$.
end

Fig. 1. The algorithm Reduce

Algorithm Consistent(Σ)
Input: OWL FA Knowledge Base $\Sigma = \langle \Sigma_1, \ldots, \Sigma_{\mathrm{k}} \rangle$
Output: *true* if Σ is consistent, *false* otherwise
begin
  Check OWL FA syntax
  Σ^* = Reduce(Σ);
  for each Σ_i^* **do**
    check-dl-consistent(Σ_i^*)
    if Σ_i^* is not consistent **then**
      Return *false*
    end if
  end for
  Return *true*.
end

Fig. 2. The algorithm `Consistent`

We invite the reader to note that *check-dl-consistent* is a function call to a DL Reasoner.

Definition 2. *Given an OWL FA knowledge base* $\Sigma = \langle \Sigma_1, \ldots, \Sigma_k \rangle$. *$\Sigma$ is consistent iff each* Σ_i^* *(*$1 \leq i \leq k$*) is consistent.* $\diamond$

Definition 2 shows that we can reduce the OWL FA-knowledge base consistency problem to the OWL DL-knowledge base consistency problem.

Proof. The consistency check of an OWL FA KB with Consistent(Σ) is straightforward from Lemma 1. We can construct $\langle \Sigma_1^*, \ldots, \Sigma_k^* \rangle$ from Σ in finite steps then we check the consistency for each Σ_i^* with a DL reasoner. Therefore, the OWL FA knowledge base Σ is consistent if and only if each Σ_i^* $(1 \leq i \leq k)$ is consistent.

Let us consider the following axiom by inserting it into OWL FA KB Σ (cf. example 1):

$DifferentIndividuals2(GiantPanda\ Panda\ )$

It obvious to see that this axiom will make Σ_2^* inconsistent, which leading to OWL FA KB Σ become inconsistent based on definition 2.

Due to space limitations, we cannot describe all reasoning tasks for OWL FA in this paper. However, since we can reduce OWL FA into a set of OWL DL ontologies then all existing DL reasoners' capabilities can be used.

5 OWL FA Tool Kit

In this section, we introduce the OWL FA Tool Kit, a simple graphic user interface for modeler to create an OWL FA ontology and perform reasoning over it. The OWL FA tool kit contains features as following:

- Editor - for modeling OWL FA ontology. In this version it supports only functional syntax.
- Query Answering - for retrieving the result set form given an OWL FA ontology $\mathcal{O}$ and a conjunctive query q.
- Ontology Consistency - for checking whether a given an ontology has at least one model.
- Concept Satisfiability - for verifying whether a given OWL FA ontology $\mathcal{O}$ and a class A, there is a model of $\mathcal{O}$ in which the interpretation of A is a non-empty set.
- Reasoning with OWL DL ontology - for reasoning over OWL DL ontologies. The user can easily create ontology and meta-ontology separately.

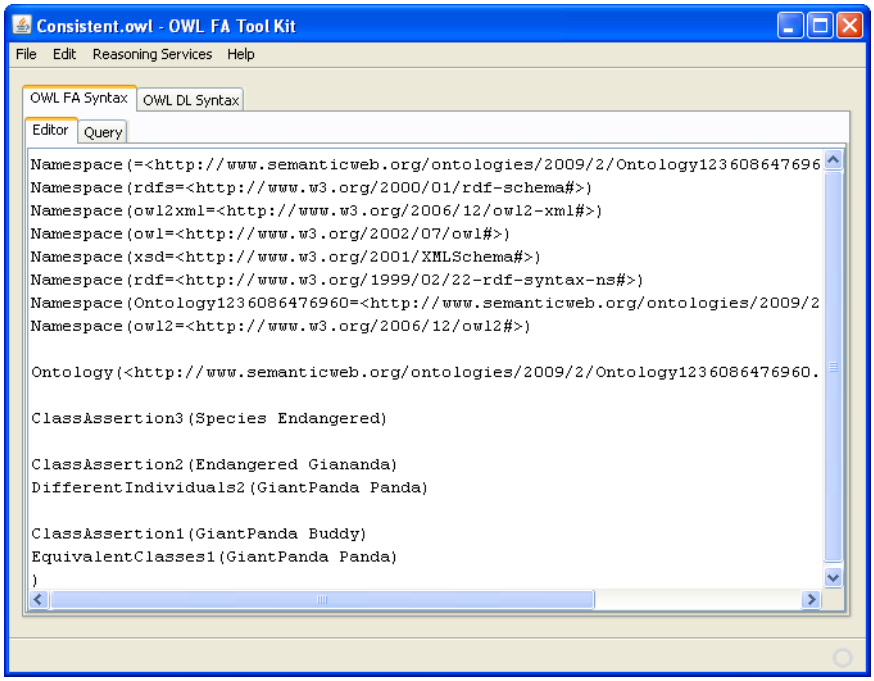

Fig. 3. OWL FA Tool Kit

6 Evaluation

In this section, we compare the metamodeling in OWL FA with OWL 2 as OWL 2 is the only OWL language that can support metamodeling and has tools support.

6.1 Use case 1: Consistency checking

OWL2 provides simple metamodeling with semantics which correspond to the contextual semantics defined in [5], however, it has been shown in [8] that these can lead to non-intuitive results.

As we discussed in section 1, if we use axiom (1),(2),(5) and (6) to create an OWL 2 ontology. This ontology is consistent when we perform consistency checking with any DL reasoner. However, this ontology is inconsistent based on layered architecture in OWL FA as we described in section 4.

6.2 Use case 2: Instance retrieval

In OWL 2, the names of concepts, properties, and individuals are not interact with each other even they share the same name, it is difficult for any existing DL

reasoner to discover those information. for example, we would like to retrieve all objects that belong to *Endangered*. Without adding an axiom to indicate that *Panda* is a *Endangered* then, *Panda* is not included in the answer set. However, OWL FA and our tool kit will return more complete answer set because in the reduction process, we propagate all class and property equalities to be object equalities in the higher layer and propagate all object equalities to be class or property equalities in the lower layer.

7 Related Work

OWL FA was introduced in [8] as a metamodeling extension of OWL. Motik [5] addressed metamodeling in OWL with two different semantics. The contextual semantics (or π-semantics) uses punning, i.e. names are replaced by distinct names for concepts, individuals and roles. This is like the different representation of an object in the OWL DL ontologies Σ_{i} in OWL FA. OWL2 [6] provides simple metamodeling features which is based on the contextual approach. The other semantics is the HiLog semantics (or ν-semantics). The HiLog semantics is stronger than the π-semantics. The concepts and individual interpretations are not independent.

De Giacoma et al. [4] proposed the `HiDL-Lite` language, which adds one layer on top of the DL-Lite$_{\mathcal{R}}$ language. This supports meta-classes and meta-properties and presents the query answering algorithm by reducing `HiDL-Lite` to DL-Lite$_{\mathcal{R}}$ with the intention of using an existing DL-Lite reasoner. In OWL FA the semantics of meta-level are same as domain knowledge unlike `HiDL-Lite` that semantics of the meta-level need to re-define.

Description Logic reasoning is applied to UML models in [1, 11, 3]. The models are transformed into DL representations. Reasoning is used to check consistency of models and between models. However, metamodels are not considered.

8 Conclusion

In this paper, we reintroduced OWL FA language and demonstrated how to model the metamodeling enabled ontology, followed by a description of reasoning in OWL FA for different reasoning tasks. The reduction algorithm is described. These algorithms use standard DL reasoning as a black-box service. Based on the given examples, a metamodeling enables ontology is described in OWL FA.

We have shown that we can make use of the existing DL reasoners to reason over OWL FA knowledge base. As we discussed in section 4, we can calculate the explicit OWL DL knowledge base Σ^* from OWL FA knowledge base Σ.

We have implemented the OWL FA Tool kit for modeler to manipulating and reasoning over OWL FA ontology and plan to incorporate these into the TrOWL[3] reasoning infrastructure.

[3] http://www.trowl.eu

In the future,we would like to enrich OWL FA to increase expressive power of the language such as propagate the explicit inequality and constrain between layers. Moreover, we would like to apply the fix-layer architecture to OWL 2 DL which has more expressive power than OWL DL. And we plan to provide tool along with reasoning mechanism for OWL 2 FA.

Acknowledgements

This work has been partially supported by the European Project Marrying Ontologies and Software Technologies(MOSTICT2008-216691).

References

1. D. Berardi, D. Calvanese, and G. De Giacomo. Reasoning on UML Class Diagrams. *Artificial Intelligence*, 168(1-2):70–118, 2005.
2. A. Brown. An introduction to Model Driven Architecture. IBM Technical Report. `URL http://www-128.ibm.com/developerworks/rational/library/3100.html`, 2004.
3. A. Cali, D. Calvanese, G. De Giacomo, and M. Lenzerini. Reasoning on UML Class Diagrams in Description Logics. In *In Proc. of IJCAR Workshop on Precise Modelling and Deduction for Object-oriented Software Development (PMD*, 2001.
4. G. De Giacomo, M. Lenzerini, and R. Rosati. Towards higher-order DL-Lite (preliminary report). In *Proceedings of the International Workshop on Description Logic (DL-2008), Dresden, Germany, May 13-16*, 2008.
5. B. Motik. On the properties of metamodeling in owl. *J. Log. Comput.*, 17(4):617–637, 2007.
6. B. Motik, P. F. Patel-Schneider, and B. Parsia. Owl 2 web ontology language: Structural specification and functional-style syntax. World Wide Web Consortium, Working Draft WD-owl2-semantics-20081202, December 2008.
7. J. Z. Pan and I. Horrocks. RDFS(FA) and RDF MT: Two Semantics for RDFS. In *Proc. of the 2nd International Semantic Web Conference (ISWC2003)*, 2003.
8. J. Z. Pan, I. Horrocks, and G. Schreiber. OWL FA: A Metamodeling Extension of OWL DL. In *Proceeding of the International workshop on OWL: Experience and Directions (OWL-ED2005)*, 2005.
9. P. F. Patel-Schneider, P. Hayes, and I. Horrocks. OWL Web Ontology Language Semantics and Abstract Syntax. Technical report, W3C, Feb. 2004. W3C Recommendation.
10. UML. Unified Modeling Language. `http://www.uml.org/`.
11. R. Van Der Straeten, J. Simmonds, and T. Mens. Detecting Inconsistencies between UML Models using Description Logic. In *Proceedings of the 2003 International Workshop on Description Logics (DL2003), Rome, Italy*, volume 81, pages 260–264, 2003.
12. W3C. Extensible Markup Language (XML). URL `http://www.w3.org/XML/`, 2001.