Semantic Web Languages – Towards an Institutional Perspective*

Dorel Lucanu[1], Yuan Fang Li[2], and Jin Song Dong[2]

[1] Faculty of Computer Science
"A.I.Cuza" University
Iaşi, Romania
dlucanu@info.uaic.ro
[2] School of Computing
National University of Singapore, Singapore
{liyf,dongjs}@comp.nus.edu.sg

Abstract. The Semantic Web (SW) is viewed as the next generation of the Web that enables intelligent software agents to process and aggregate data autonomously. Ontology languages provide basic vocabularies to semantically markup data on the SW. We have witnessed an increase of numbers of SW languages in the last years. These languages, such as RDF, RDF Schema (RDFS), the OWL suite of languages, the OWL^- suite, SWRL, are based on different semantics, such as the RDFS-based, description logic-based, Datalog-based semantics. The relationship among the various semantics poses a challenge for the SW community for making the languages interoperable. Institutions provide a means of reasoning about software specifications regardless of the logical system. This makes it an ideal candidate to represent and reason about the various languages in the Semantic Web. In this paper, we construct institutions for the SW languages and use institution morphisms to relate them. We show that RDF framework together with the RDF serializations of SW languages form an indexed institution. This allows the use of Grothendieck institutions to combine Web ontologies described in various languages.

1 Introduction

The family of Semantic Web (SW) languages increased very much in the last years and we guess it will continue to increase in the future. This is somehow surprising for SW community and it contradicts the initial intentions of the SW creators. But it is a reality and we have to live with it. This increase refers specially to the languages describing Web ontologies. Here are several examples: OWL with its three dialects (Lite, DL, and Full) [20], SWRL [15], SWRL FOL [21], DLP [12], OWL^- with its three dialects ($Lite^-$, DL^-, $Full^-$) [5], WRL [1], and the list does not finish here. For these languages, different definitions for their semantics were proposed in the literature:

* This work is partially supported by Singapore MOE project Rigorous Design Methods and Tools for Intelligent Autonomous (R-252-000-201-112) and NUS EERSS Program. The second author would like to thank Singapore Millennium Foundation (SMF) for the financial support.

K. Futatsugi et al. (Eds.): Goguen Festschrift, LNCS 4060, pp. 99–123, 2006.

model-theoretic semantics [20,13], RDF based semantics [1,20], first-order logic based semantics [15,21], frame logic semantics [5], Datalog semantics [5], Z semantics [8,18], and so on. This gives rise to some confusions and debates about the meaning of the hierarchy of SW languages as it has been illustrated in the well-known Tim Berners-Lee's "Semantic Web Stack" diagram (Fig. 1).

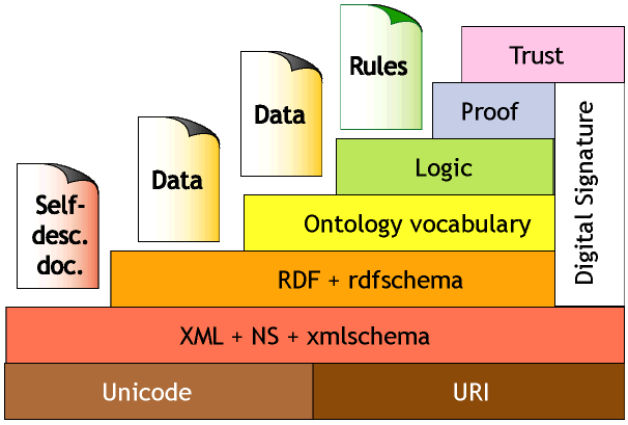


Fig. 1. The Semantic Web stack of languages

In this paper we use the institution theory in order to investigate the exact relationships among these languages.

The notion of institutions [10] was introduced to formalize the concept of "logical systems". Institutions provide a means of reasoning about software specifications regardless of the logical system. Hence, it serves as a natural candidate to study the relationship among the various SW languages, as they are based on different logical systems (semantics).

In this paper, we investigate the relationship among languages RDF [17], RDF Schema [4], OWL suite [20], and OWL$^-$ suite [5] by defining their respective institutions and relating these institutions using morphisms or comorphisms. A main advantage is a better understanding of the semantical relationship among the various SW languages. Here we focus only on the RDF triple-based semantics. We show that RDF framework (RDF and RDF Schema) together with RDF serializations of SW languages form an indexed institution, and hence the whole framework can be organized as a Grothendieck institution [7]. An interesting fact is that the construction of the indexed institution is based on a diagram of RDF theories. We define a method of constructing institutions starting from theories and then we extend it to diagrams of categories and indexed institutions. We believe that we answer in this way the question regarding the layering of SW languages [22]. Semantically, the "stack" of SW languages depicted by Berners-Lee is an indexed institution. This indexed institution produces a Grothendieck

institution which offers a formal framework for combining ontologies written in various languages. In this way, all SW languages can "live" together.

The rest of the paper is organized as follows. In Section 2, we briefly present the background information on SW languages and institutions. In Section 3, we define the institutions of (bare) RDF and RDF Schema languages. These institutions are used as the basis on which one of the semantics for SW languages is constructed using a method presented in Section 2. Section 4 is devoted to the construction of the institutions defining SW languages. In Section 5, we construct an indexed institution based on a diagram of RDF theories and we show that the RDF-based semantics of SW languages can be defined as institution comorphisms from these languages to the indexed institution. Section 6 concludes the paper and discusses future work directions.

Acknowledgment

This paper is dedicated, warmly and respectfully, to Professor Joseph Goguen on the occasion of his 65th birthday. Joseph has determinative contributions in promoting Algebra from the status of an abstract notation to that of a practical specification language, widely used today in software engineering, and promoting logical systems as the "institution of the specification languages". The research and teaching activity of the first author is definitely guided by Joseph's papers on these two issues. We wish him success and happiness in his future.

2 Preliminaries

2.1 Semantic Web Languages

The Semantic Web is a vision as the new generation of the current World Wide Web in which information is semantically marked-up so that intelligent software agents can autonomously understand, process and aggregate data. This ability is realized through the development of a "stack" of languages, as depicted by Berners-Lee in Fig. 1.

Based on mature technologies such as XML, Unicode and URI (Uniform Resource Identifier), The Resource Description Framework (RDF) [17] is the foundation of later languages in the SW. RDF is a model of metadata defining a mechanism for describing resources that makes no assumptions about a particular application domain. It provides a simple way to make *statements* about Web resources. An RDF document is a collection of *triples*: statements of the form $\langle subject,\ predicate,\ object\rangle$, where *subject* is the resource we are interested in, *predicate* specifies the property or characteristic of the subject and *object* states the value of the property. This is the basic structure of the subsequent ontology languages. RDF also defines vocabularies for constructing containers such bags, sequences and lists.

RDF Schema [4] provides additional vocabularies for describing RDF documents. It defines semantical entities such as *Resource*, *Class*, *Property*, *Literal* and various properties about these entities, such as *subClassOf*, *domain*, *range*, etc. In RDF Schema, *Resource* is the universe of description. It can be further categorized as classes, properties, datatypes or literals. With these semantical constructs, RDF Schema can be regarded as the basic ontology language.

The Web resources are represented by full URIs, consisting of a prefix, representing a namespace, and a name representing the actual resource that is being described. In its full form, the prefix and the resource name are separated by a #. In shorthand form, the prefix can be represented by a shorter name and it is separated from the actual name by a colon (:), as the following example shows. After a resource has been introduced by an `rdf:ID` construct (in shorthand form of the URI), it can be subsequently accessed and augmented by the `rdf:about` constructs. When there is no possibility of confusion, the prefix can be omitted (but not the separator #).

Example 1. The following RDF fragment defines an RDFS class *Carnivore*, which is a sub class of *Animal*.

```
<rdfs:Class rdf:ID="Animal"/>
<rdfs:Class rdf:ID="Carnivore">
   <rdfs:subClassOf rdf:resource=#Animal"/>
</rdfs:Class>
```

In this example, the namespace is the URI http://ex.com/animals. The full URI for the class `Animal` is http://ex.com/animals#Animal.

The ability to organize and categorize domain knowledge is a necessity for software agents to process and aggregate Web resources. Domain knowledge is usually organized as inter-related conceptual entities in a hierarchy. The RDF language is not expressive enough to tackle such complexity.

In 2003, W3C published a new ontology language, the Web Ontology Language (OWL) [20]. Based on description logics and RDF Schema, the OWL suite consists of three sublanguages: Lite, DL and Full, with increasing expressiveness. The three sublanguages are meant for user groups with different requirements of expressiveness and decidability. OWL Lite and DL are decidable whereas OWL Full is generally not.

By saying that an ontology language is decidable, it actually means that the core reasoning problems, namely, concept subsumption, concept/ontology satisfiability and instantiation checking, are decidable [16].

One of the major extensions of OWL over RDF Schema is the ability to define restrictions using existing classes and properties. By using restrictions, new classes can be built incrementally. In OWL, conceptual entities are organized as classes in hierarchies. Individuals are grouped under classes and are called instances of the classes. Classes, properties and individuals can be related by properties.

Example 2. The following OWL fragment shows the definition of an object property *eats* and a class *carnivore*, which is further defined as an animal that only eats animals. This is achieved through the use of an `allValuesFrom` restriction in OWL.

```
<owl:ObjectProperty rdf:ID="eats"/>
<owl:Class rdf:about="#Carnivore">
   <rdfs:subClassOf>
      <owl:Restriction><owl:allValuesFrom>
         <owl:Class rdf:resource="#Animal"/>
      </owl:allValuesFrom>
```

```
      <owl:onProperty>
         <owl:ObjectProperty rdf:resource=
              "http://ex.com/animals#eats"/>
      </owl:onProperty>
   </owl:Restriction></rdfs:subClassOf>
</owl:Class>
```

The class *Carnivore* is defined to be a sub class of an OWL restriction that defines an anonymous class which only *eats Animal*s.

The OWL$^-$ [5] suite of languages, namely Lite$^-$, DL$^-$ and Full$^-$, is a restricted variant of OWL languages. OWL Lite$^-$ and DL$^-$ are strict subsets of OWL Lite and DL respectively and they can be directly translated into Datalog. According to [5], the main advantages of the OWL$^-$ include the following. Firstly, by translating OWL$^-$ to Datalog, highly efficient deductive database querying capabilities can be used; Secondly, rules extension and query languages can be easily implemented on top of OWL$^-$.

In order to expand the expressiveness of SW languages, several rules extensions have been proposed. SWRL [15] is a direct *extension* of OWL DL that incorporates Horn-style rules. Among other things, it supports (universally quantified) variables and built-in predicates/ functions for various data types.

On the contrary, the Web Rules Language (WRL) [1] is a rule-based ontology language. Based on deductive databases and logic programming, it is designed to be complementary to OWL, which is strong at checking subsumption relationships among concepts. WRL focuses on checking instance data and the specification of and reasoning about arbitrary rules. Moreover, WRL assumes a “Closed World Assumption”, whereas OWL and SWRL assume an ”Open World Assumption”.

2.2 Institutions

Institutions supply a uniform way for structuring the theories in various logical systems. Many logical systems have been proved to be institutions. Recent research showed that institutions are useful in designing tools supporting verification over multiple logics. The basic reference for institutions is [10]. A comprehensive overview on institutions and their applications can be found in [6]. A well structured approach of the various institution morphisms and many other recent constructions can be found in [11]. A recent application of institutions in formalizing the information integration is given in [9]. The Grothendieck institution construction we use in this paper follows the line from [7]. The institutions use intensively category theory; we recommend [2] for a detailed presentation of categories and their applications in computer science.

In this section we recall the main definitions for institutions and we introduce two new constructions. The first is simple and it generalizes the notion of theoroidal comorphism by allowing to encode sentences from the source institution by conjunctions of sentences in the target institution. The second is more complex and is used to construct indexed institutions starting from diagrams of semantically constrained theories from a basic institution. We use this construction to define the indexed institutions based on RDF triples corresponding to SW languages.

An *institution* is a quadruple $\Im = (\mathsf{Sign}, \mathsf{Mod}, \mathsf{sen}, \models)$, where Sign is a category whose objects are called *signatures*, $\mathsf{Mod} : \mathsf{Sign}^{op} \to \mathsf{Cat}$ is a functor which associates with each signature Σ a category whose objects are called Σ-*models*, sen is a functor $\mathsf{sen} : \mathsf{Sign} \to \mathsf{Set}$ which associates with each signature Σ a set whose elements are called Σ-*sentences*, and $\models$ is a function which associates with each signature Σ a binary relation $\models_\Sigma \subseteq |\mathsf{Mod}(\Sigma)| \times \mathsf{sen}(\Sigma)$, called *satisfaction relation*, such that for each signature morphism $\phi : \Sigma \to \Sigma'$ the *satisfaction condition*

$$\mathsf{Mod}(\phi^{op})(M') \models_\Sigma \varphi \Leftrightarrow M' \models_{\Sigma'} \phi(\varphi)$$

holds for each model $M' \in \mathsf{Mod}(\Sigma')$ and each sentence $\varphi \in \mathsf{sen}(\Sigma)$. The functor sen abstracts the way the sentences are constructed from signatures (vocabularies). The functor Mod is defined over the opposite category Sign^{op} because a "translation between vocabularies" $\phi : \Sigma \to \Sigma'$ defines a forgetful functor $\mathsf{Mod}(\phi^{op}) : \mathsf{Mod}(\Sigma') \to \mathsf{Mod}(\Sigma)$ such that for each Σ'-model M', $\mathsf{Mod}(\phi^{op})(M')$ is M' viewed as a Σ-model. The satisfaction condition may be read as "M' satisfies the ϕ-translation of φ iff M', viewed as a Σ-model, satisfies φ", i.e., the meaning of φ is not changed by the translation ϕ.

We often use $\mathsf{Sign}(\Im)$, $\mathsf{Mod}(\Im)$, $\mathsf{sen}(\Im)$, $\models_\Im$ to denote the components of the institution $\Im$. If $\phi : \Sigma \to \Sigma'$ is a signature morphism, then the Σ-model $\mathsf{Mod}(\phi^{op})(M')$ is also denoted by $M'{\restriction_\phi}$ and we call it *the ϕ-reduct of M'*. We also often write $\phi(\varphi)$ for $\mathsf{Mod}(\phi)(\varphi)$.

If F is a set of Σ-sentences and M a Σ-model, then $M \models \mathsf{F}$ denotes the fact that M satisfies all the sentences in F. Let $\mathsf{F}^\bullet$ denote the set $\mathsf{F}^\bullet = \{\varphi \mid (\forall\, M \text{ a } \Sigma \text{ model}) M \models_\Sigma \mathsf{F} \Rightarrow M \models_\Sigma \varphi\}$. A sentence φ is *semantical consequence* of F, we write $\mathsf{F} \models_\Sigma \varphi$, iff $\varphi \in \mathsf{F}^\bullet$.

A specification (presentation) is a way to represent the properties of a system independent of model (= implementation). Formally, a *specification* is a pair (Σ, F), where Σ is a signature and F is a set of Σ-sentences. A (Σ, F)-*model* is a Σ-model M such that $M \models_\Sigma \mathsf{F}$. We sometimes write $(\Sigma, \mathsf{F}) \models \varphi$ for $\mathsf{F} \models_\Sigma \varphi$. A *specification morphism* from (Σ, F) to (Σ', F') is a signature morphism $\phi : \Sigma \to \Sigma'$ such that $\phi(\mathsf{F}) \subseteq \mathsf{F}'^\bullet$. We denote by Spec the category of the specifications. A *theory* is a specification (Σ, F) with $\mathsf{F} = \mathsf{F}^\bullet$; the full subcategory of theories in Spec is denoted by Th. The inclusion functor $U : \mathsf{Th} \to \mathsf{Spec}$ is an equivalence of categories, having a left-adjoint-left-inverse $F : \mathsf{Spec} \to \mathsf{Th}$ given by $F(\Sigma, \mathsf{F}) = (\Sigma, \mathsf{F}^\bullet)$ on objects and identity on morphisms.

Given an institution $\Im = (\mathsf{Sign}, \mathsf{Mod}, \mathsf{sen}, \models)$, the *theoroidal institution* $\Im^{\mathsf{th}}$ of $\Im$ is the institution $\Im^{\mathsf{th}} = (\mathsf{Th}, \mathsf{Mod}^{\mathsf{th}}, \mathsf{sen}^{\mathsf{th}}, \models^{\mathsf{th}})$, where $\mathsf{Mod}^{\mathsf{th}}$ is the extension of Mod to theories, $\mathsf{sen}^{\mathsf{th}}$ is $\mathsf{sign}; \mathsf{sen}$ with $\mathsf{sign} : \mathsf{Th} \to \mathsf{Sign}$ the functor which forgets the sentences of a theory, and $\models^{\mathsf{th}} = |\mathsf{sign}|; \models$.

Let $\Im = (\mathsf{Sign}, \mathsf{Mod}, \mathsf{sen}, \models)$ and $\Im' = (\mathsf{Sign}', \mathsf{Mod}', \mathsf{sen}', \models')$ be two institutions. An *institution morphism* $(\Phi, \beta, \alpha) : \Im \to \Im'$ consists of:

1. a functor $\Phi : \mathsf{Sign} \to \mathsf{Sign}'$,
2. a natural transformation $\beta : \mathsf{Mod} \Rightarrow \Phi^{op}; \mathsf{Mod}'$, i.e., a natural family of functors $\beta_\Sigma : \mathsf{Mod}(\Sigma) \to \mathsf{Mod}'(\Phi(\Sigma))$, and
3. a natural transformation $\alpha : \Phi; \mathsf{sen}' \Rightarrow \mathsf{sen}$, i.e., a natural family of functions $\alpha_\Sigma : \mathsf{sen}'(\Phi(\Sigma)) \to \mathsf{sen}(\Sigma)$,

such that the following satisfaction condition holds:

$$M \models_{\Sigma} \alpha_{\Sigma}(\varphi') \text{ iff } \beta_{\Sigma}(M) \models'_{\Phi(\Sigma)} \varphi'$$

for any Σ-model M in $\Im$ and $\Phi(\Sigma)$-sentence φ' in $\Im'$. Usually, the institution morphisms are used to express the embedding relationship. An example of institution morphism is $(\Phi, \beta, \alpha) : \Im^{\mathsf{th}} \to \Im$ which express the embedding of $\Im$ in $\Im^{\mathsf{th}}$. $\Phi : \mathsf{Th} \to \mathsf{Sign}$ is given by $\Phi(\Sigma, \mathsf{F}) = \Sigma$, $\beta : \mathsf{Mod}^{\mathsf{th}} \Rightarrow \Phi^{op}; \mathsf{Mod}$ is defined such that $\beta_{(\Sigma,\mathsf{F})}$ is the identity, and $\alpha : \Phi; \mathsf{sen} \Rightarrow \mathsf{sen}^{\mathsf{th}}$ is defined such that $\alpha_{(\Sigma,\mathsf{F})}$ is the identity.

An *institution comorphism* $(\Phi, \beta, \alpha) : \Im \to \Im'$ consists of:

1. a functor $\Phi : \mathsf{Sign} \to \mathsf{Sign}'$,
2. a natural transformation $\beta : \Phi^{op}; \mathsf{Mod}' \Rightarrow \mathsf{Mod}$, i.e., a natural family of functors $\beta_{\Sigma} : \mathsf{Mod}'(\Phi(\Sigma)) \to \mathsf{Mod}(\Sigma)$, and
3. a natural transformation $\alpha : \mathsf{sen} \Rightarrow \Phi; \mathsf{sen}'$, i.e., a natural family of functions $\alpha_{\Sigma} : \mathsf{sen}(\Sigma)) \to \mathsf{sen}'(\Phi(\Sigma))$,

such that the following satisfaction condition holds:

$$\beta_{\Sigma}(M') \models_{\Sigma} \varphi \text{ iff } M' \models_{\Phi(\Sigma)} \alpha_{\Sigma'}(\varphi)$$

for any $\Phi(\Sigma)$-model M' in $\Im$ and Σ-sentence φ in $\Im$. If β_{Σ} is surjective for each signature Σ, then we say that (Φ, β, α) is *conservative*. Usually, the institution comorphisms are used to express the representation (encoding) relationship. A simple example of comorphism is $(\Phi, \beta, \alpha) : \Im \to \Im^{\mathsf{th}}$, where $\Phi : \mathsf{Sign} \to \mathsf{Th}$ is given by $\Phi(\Sigma) = (\Sigma, \emptyset)$, $\beta : \Phi; \mathsf{Mod}^{\mathsf{th}} \Rightarrow \mathsf{Mod}$ is defined such that β_{Σ} is the identity, and $\alpha : \mathsf{sen} \Rightarrow \Phi; \mathsf{sen}^{\mathsf{th}}$ is defined such that α_{Σ} is the identity.

In many practical examples, we have to represent (encode) a sentence from the source institution with a conjunction of sentences from the target institution. A simple example is the representation of the equivalence $\varphi \leftrightarrow \varphi'$ by the conjunction of two Horn rules: $\varphi \to \varphi' \wedge \varphi' \to \varphi$. Hence the following construction. The *conjunction extension* of $\Im$ is the institution $\Im^{\wedge} = (\mathsf{Sign}, \mathsf{Mod}, \mathsf{sen}^{\wedge}, \models^{\wedge})$, where $\mathsf{sen}^{\wedge}(\Sigma) = \mathsf{sen}(\Sigma) \cup \{\varphi_1 \wedge \cdots \wedge \varphi_k \mid \varphi_1, \ldots, \varphi_k \in \mathsf{sen}(\Sigma)\}$, $M \models^{\wedge}_{\Sigma} \varphi$ iff $M \models_{\Sigma} \varphi$ for all $\varphi \in \mathsf{sen}(\Sigma)$, and $M \models^{\wedge}_{\Sigma} \varphi_1 \wedge \cdots \wedge \varphi_k$ iff $M \models_{\Sigma} \varphi_i$ for $i = 1, \ldots, k$. There is an institution morphism $(\Phi, \beta, \alpha) : \Im^{\wedge} \to \Im$ expressing the embedding of $\Im$ in $\Im^{\wedge}$. This embedding can also be represented by a comorphism from $\Im$ to $\Im^{\wedge}$.

An *indexed category* is a functor $G : I^{op} \to \mathsf{Cat}$, where I is a category of indices. The *Grothendieck category* $G^{\#}$ of an indexed category $G : I^{op} \to \mathsf{Cat}$ has pairs $\langle i, \Sigma \rangle$, with i an object in I and Σ an object in $G(i)$, as objects, and $\langle u, \varphi \rangle : \langle i, \Sigma \rangle \to \langle i', \Sigma' \rangle$, with $u : i \to i'$ an arrow in I and $\varphi : \Sigma \to G(u)(\Sigma')$ an arrow in $G(i)$, as arrows.

The *Grothendieck institution* $\Im^{\#}$ of an indexed institution $\Im : I^{op} \to \mathsf{Ins}$ has

1. the Grothendieck category $\mathsf{Sign}^{\#}$ as its category of signatures, where $\mathsf{Sign} : I^{op} \to \mathsf{Cat}$ is the indexed category of signatures of $\Im$;
2. $\mathsf{Mod}^{\#} : (\mathsf{Sign}^{\#})^{op} \to \mathsf{Cat}$ as its model functor, where $\mathsf{Mod}^{\#}\langle i, \Sigma \rangle = \mathsf{Mod}^{i}(\Sigma)$ and $\mathsf{Mod}^{\#}\langle u, \varphi \rangle = \beta^{u}_{\Sigma'}; \mathsf{Mod}^{i}(\varphi)$;
3. $\mathsf{sen}^{\#} : \mathsf{Sign}^{\#} \to \mathsf{Set}$ as its sentence functor, where $\mathsf{sen}^{\#}\langle i, \Sigma \rangle = \mathsf{sen}^{i}(\Sigma)$; and
4. $M \models^{\#}_{\langle i,\Sigma \rangle} \varphi$ iff $M \models^{i}_{\Sigma} \varphi$ for all $i \in |I|$, $\Sigma \in |\mathsf{Sign}^{i}|$, $M \in |\mathsf{Mod}^{\#}(i, \Sigma)|$, and $\varphi \in \mathsf{sen}^{\#}(i, \Sigma)$;

where $\Im(i) = (\mathsf{Sign}^i, \mathsf{Mod}^i, \mathsf{sen}^i, \models^i)$ for each index $i \in |I|$ and $\Im(u) = (\phi^u, \beta^u, \alpha^u)$ for each index morphism $u \in I$.

We show how a theory (Σ_0, F_0) and a model constraint can define an institution $\underline{\widehat{(\Sigma_0, \mathsf{F}_0)}}$. A *model constraint* is a map $[\![\_]\!]_c$ which associates a subcategory $[\![\Sigma, \mathsf{F}]\!]_c \subseteq \mathsf{Mod}^{\mathsf{th}}(\Sigma, \mathsf{F})$ with each theory (Σ, F), such that $M'|_\phi \in [\![\Sigma, \mathsf{F}]\!]_c$ for all $\phi : (\Sigma, \mathsf{F}) \to (\Sigma', \mathsf{F}')$ and $M' \in [\![\Sigma', \mathsf{F}']\!]_c$. Moreover, a model constraint implies in fact a *semantical extension* in the following sense. $[\![\Sigma, \mathsf{F}]\!]_c \subseteq \mathsf{Mod}^{\mathsf{th}}(\Sigma, \mathsf{F})$ implies $[\![\Sigma, \mathsf{F}]\!]_c^\bullet \supseteq \mathsf{Mod}^{\mathsf{th}}(\Sigma, \mathsf{F})^\bullet$, where $\mathcal{M}^\bullet$ denotes the set of sentences satisfied by all models in $\mathcal{M}$. In other words, in the presence of model constraints we can prove more properties. The constraints defined in [10] are a particular case of model constraints when the subcategory can be syntactically represented. The institution $\underline{\widehat{(\Sigma_0, \mathsf{F}_0)}}$ is defined as follows:

1. the category of signatures is the comma category $(\Sigma_0, \mathsf{F}_0)\downarrow\mathsf{Th}$, where the objects are theory morphisms $f : (\Sigma_0, \mathsf{F}_0) \to (\Sigma, \mathsf{F})$, and the arrows $\phi : f \to f'$ are consisting of theory morphisms $\phi : (\Sigma, \mathsf{F}) \to (\Sigma', \mathsf{F}')$ such that $f; \phi = f'$,
2. the model functor $\mathsf{Mod}(\underline{\widehat{(\Sigma_0, \mathsf{F}_0)}})$ maps each signature $f : (\Sigma_0, \mathsf{F}_0) \to (\Sigma, \mathsf{F})$ into the subcategory $[\![\Sigma, \mathsf{F}]\!]_c$,
3. the sentence functor $\mathsf{sen}(\underline{\widehat{(\Sigma_0, \mathsf{F}_0)}})$ maps a signature $f : (\Sigma_0, \mathsf{F}_0) \to (\Sigma, \mathsf{F})$ into the set of Σ-sentences,
4. the satisfaction relation is defined by $M \models_f \varphi$ iff $M \models_\Sigma \varphi$.

Note that the model constraint is required only for theories (Σ, F) for that there exists a theory morphism $f : (\Sigma_0, \mathsf{F}_0) \to (\Sigma, \mathsf{F})$. We extend the above construction to diagrams of theories and indexed institutions. Let $D : I \to \mathsf{Th}$ be a diagram of theories and $([\![\_]\!]_i \mid i \in |I|)$ a model constraint such that if $u : i \to j$ is an arrow in I, then $M'|_u \in [\![D(i)]\!]_i$ for each $M' \in [\![D(j)]\!]_j$. We denote $D(i)$ by (Σ_i, F_i), $i \in |I|$. If $u : j \to i$ is an arrow in I^{op}, then there is an institution morphism $(\Phi, \beta, \alpha) : \underline{\widehat{(\Sigma_j, \mathsf{F}_j)}} \to \underline{\widehat{(\Sigma_i, \mathsf{F}_i)}}$, where

1. Φ maps a signature $f : (\Sigma_j, \mathsf{F}_j) \to (\Sigma, \mathsf{F})$ into the signature $D(u); f : (\Sigma_i, \mathsf{F}_i) \to (\Sigma, \mathsf{F})$;
2. $\beta : \mathsf{Mod}(\underline{\widehat{(\Sigma_j, \mathsf{F}_j)}}) \to \Phi; \underline{\widehat{(\Sigma_i, \mathsf{F}_i)}}$ is as follows: if $f : (\Sigma_j, \mathsf{F}_j) \to (\Sigma, E)$ is a signature in $\underline{\widehat{(\Sigma_j, \mathsf{F}_j)}}$, then β_f is the identity because $M|_f|_{D(u)}$ is a Σ_i-model by functoriality of $\mathsf{Mod}(\Im)$ and by the fact that $D(u)$ and f are theory morphisms;
3. $\alpha : \Phi; \mathsf{sen}(\underline{\widehat{(\Sigma_j, \mathsf{F}_j)}}) \to \underline{\widehat{(\Sigma_i, \mathsf{F}_i)}}$ is as follows: if $f : (\Sigma_j, \mathsf{F}_j) \to (\Sigma, E)$ is a signature in $\underline{\widehat{(\Sigma_j, \mathsf{F}_j)}}$, then $\alpha_{\Phi(f)}$ is identity.

The diagram $D : I \to Th$ together with the model constraint $([\![\_]\!]_i \mid i \in |I|)$ produces an indexed institution $\mathcal{D} : I^{op} \to \mathsf{Ins}$, where Ins is the category of institutions and the arrows are institution morphisms. We can define now the Grothendieck institution $\mathcal{D}^\#$, where

1. the category of signatures $\mathsf{Sign}(\mathcal{D}^\#)$ is the Grothendieck construction $\mathsf{Sign}(\mathcal{D})^\#$ corresponding to $\mathsf{Sign}(\mathcal{D}) : I^{op} \to \mathsf{Cat}$, which maps each index i into $\mathsf{Sign}(\underline{\widehat{(\Sigma_i, \mathsf{F}_i)}})$;

2. the model functor $\mathsf{Mod}(\mathcal{D}^{\#}) : \mathsf{Sign}(\mathcal{D}^{\#}) \to \mathsf{Cat}$ is given by:
   $\mathsf{Mod}(\mathcal{D}^{\#})(\langle i, f : (\Sigma_i, \mathsf{F}_i) \to (\Sigma, \mathsf{F})\rangle)$ is $\mathsf{Mod}(\widehat{\underline{\Sigma_i, \mathsf{F}_i}})(f)$ (that is equal to $[\![\Sigma, \mathsf{F}]\!]_i$), and if $\langle u, \phi\rangle : \langle i, f{:}(\Sigma_i, \mathsf{F}_i) \to (\Sigma, \mathsf{F})\rangle \to \langle j, f'{:}(\Sigma_j, \mathsf{F}_j) \to (\Sigma', \mathsf{F}')\rangle$, then $\mathsf{Mod}(\mathcal{D}^{\#})(\langle u, \phi\rangle) = \beta_{f'}(u); \mathsf{Mod}(\widehat{\underline{\Sigma_i, \mathsf{F}_i}})(\phi)$, $(\Phi, \beta, \alpha) : \underline{(\widehat{\Sigma_j, \mathsf{F}_j})} \to \underline{(\widehat{\Sigma_i, \mathsf{F}_i})}$;

3. the sentence functor $\mathsf{sen}(\mathcal{D}^{\#}) : \mathsf{Sign}(\mathcal{D}^{\#}) \to \mathsf{Set}$ is given by:
   $\mathsf{sen}(\mathcal{D}^{\#})(\langle i, f : (\Sigma_i, \mathsf{F}_i) \to (\Sigma, \mathsf{F})\rangle)$ is $\mathsf{sen}(\widehat{\underline{\Sigma_i, \mathsf{F}_i}})(f)$ (that is equal to $\mathsf{sen}(\Im)(\Sigma)$), and if $\langle u, \phi\rangle : \langle i, f : (\Sigma_i, \mathsf{F}_i) \to \Sigma, \mathsf{F})\rangle \to \langle j, f' : (\Sigma_j, \mathsf{F}_j) \to \Sigma', \mathsf{F}')\rangle$, then $\mathsf{sen}(\mathcal{D}^{\#})(\langle u, \phi\rangle) = \mathsf{sen}(\widehat{\underline{\Sigma_i, \mathsf{F}_i}})(\phi); \alpha_f(u)$, where $(\Phi, \beta, \alpha) : \underline{(\widehat{\Sigma_j, \mathsf{F}_j})} \to \underline{(\widehat{\Sigma_i, \mathsf{F}_i})}$;

4. if $f : (\Sigma_i, \mathsf{F}_i) \to (\Sigma, \mathsf{F})$, $M \in \mathsf{Mod}(\mathcal{D}^{\#})(\langle i, f\rangle)$ and $\varphi \in \mathsf{sen}(\mathcal{D}^{\#})(\langle i, f\rangle)$, then $M \models_{\langle i,f\rangle} \varphi$ iff $M \models_f \varphi$.

This construction will be used to formalize the RDF triple-based logics underlying SW languages. For instance, it is useful to combine ontologies described in various SW languages.

3 RDF and RDF Schema Logics

In this section, we define the institutions for the languages RDF and RDF Schema. The construction of these institutions is divided into three steps. Firstly, we construct a bare-bone institution for RDF logic, capturing only the very essential concepts in RDF, namely the resource references and the triples format. This logic then serves as the basis on which the institutions of the actual RDF and RDF Schema are constructed. In turn, the institutions defined in this section serve to define the RDF serialization of ontology languages defined in Section 4.

3.1 Bare RDF Logic $\widehat{\underline{\text{BRDF}}}$

As introduced in Section 2.1, the Resource Description Framework (RDF) is the foundation language of the Semantic Web and all upper layer languages are based on it. Hence, they are all based on the syntax defined in RDF, which is, the *triples* format. Together with the use of URI for resource referencing, these two features of the RDF language are common to all other languages. Hence, we extract them from RDF language and define an institution, the bare RDF logic $\widehat{\underline{\text{BRDF}}}$.

Example 3. Since resource references are the only signatures in $\widehat{\underline{\text{BRDF}}}$, any triple will be part of the sentences. As $\widehat{\underline{\text{BRDF}}}$ is a bare-bone RDF institution, it does not define the XML serialization presented in the previous two examples. Therefore, we will use the informal syntax in this example. Note that the separator # is replaced by a : in this notation. The following triple is a legal sentence in $\widehat{\underline{\text{BRDF}}}$, stating that carnivores eat animals.

(animals:Carnivore, animals:eats, animals:Animal)

The Bare RDF logic $\widehat{\underline{\text{BRDF}}}$ is a bare-bone institution with resource references as the only signatures. The sentences are triples. $\widehat{\underline{\text{BRDF}}}$ is not expressive at all. We use it as a basis upon which we develop the Grothendieck institution of the triples-based logics underlying SW languages.

A signature RR in $\widehat{\underline{\text{BRDF}}}$ is a set of resource references. A signature morphism $\phi : RR \to RR'$ is an arrow in Set. The RR-sentences are triples of the form (sn, pn, on), where $sn, pn, on \in RR$. Usually, sn is for subject name, pn is for property (predicate) name, and on is for object name. RR-models $\mathbb{I}$ are tuples $\mathbb{I} = (R_\mathbb{I}, P_\mathbb{I}, S_\mathbb{I}, \text{ext}_\mathbb{I})$, where $R_\mathbb{I}$ is a set of resources, $P_\mathbb{I}$ is a subset of $R_\mathbb{I}$ ($P_\mathbb{I} \subseteq R_\mathbb{I}$) - the set of properties, $S_\mathbb{I} : RR \to R_\mathbb{I}$ is a mapping function that maps each resource reference to some resource, and $\text{ext}_\mathbb{I} : P_\mathbb{I} \to \mathcal{P}(R_\mathbb{I} \times R_\mathbb{I})$ is an extension function mapping each property to a set of pairs of resources that it relates. An RR-homomorphism $h : \mathbb{I} \to \mathbb{I}'$ between two RR-models is a function $h : R_\mathbb{I} \to R_{\mathbb{I}'}$ such that $h(P_\mathbb{I}) \subseteq P_{\mathbb{I}'}$, $S_\mathbb{I}; h = S_{\mathbb{I}'}$, and $\text{ext}_\mathbb{I}; \mathcal{P}(h \times h) = h|_{P_\mathbb{I}}; \text{ext}_{\mathbb{I}'}$. The satisfaction is defined as follows:

$$\mathbb{I} \models_{RR} (sn, pn, on) \textit{ iff } (S_\mathbb{I}(sn), S_\mathbb{I}(on)) \in \text{ext}_\mathbb{I}(S_\mathbb{I}(pn)),$$

that (sn, pn, on) is satisfied if and only if the pair consisting of the resources associated with the subject name sn and the object name on is in the extension of pn.

In order to simplify the notation, we often write $\text{ext}_\mathbb{I}(pn)$ instead of $\text{ext}_\mathbb{I}(S_\mathbb{I}(pn))$.

3.2 RDF Logic $\widehat{\underline{\text{RDF}}}$

The RDF logic $\widehat{\underline{\text{RDF}}}$ is constructed with $\widehat{\underline{\text{BRDF}}}$ as the basis. The addition in $\widehat{\underline{\text{RDF}}}$ is the built-in vocabularies of the RDF language and the semantics of these language constructs. Hence, as shown below, we denote the signatures of $\widehat{\underline{\text{RDF}}}$ using theories, which consist of these built-in vocabularies and sentences giving them semantics. We also add some weak model constraints. More precisely, $\widehat{\underline{\text{RDF}}}$ is defined using the construction we defined in Section 2.2 starting from a theory RDF and a model constraint $[\![\_]\!]_{RDF}$.

The *RDF theory* is $\texttt{RDF} = (\texttt{RDFVoc}, T_{\texttt{RDF}})$, where the RDF vocabulary RDFVoc includes the following items:

```
rdf:type, rdf:Property, rdf:value,
rdf:Statement, rdf:subject, rdf:predicate, rdf:object,
rdf:List, rdf:first, rdf:rest, rdf:nil,
rdf:Seq, rdf:Bag, rdf:Alt, rdf:_1 rdf:_2 ...
```

and $T_{\texttt{RDF}}$ consists of triples expressing properties of the vocabulary symbols:

```
(rdf:type, rdf:type, rdf:Property),
(rdf:subject, rdf:type, rdf:Property),
(rdf:predicate, rdf:type, rdf:Property),
(rdf:object, rdf:type, rdf:Property),
(rdf:value, rdf:type, rdf:Property),
(rdf:first, rdf:type, rdf:Property),
(rdf:rest, rdf:type, rdf:Property),
(rdf:nil, rdf:type, rdf:List),
(rdf:_1, rdf:type, rdf:Property),
(rdf:_2, rdf:type, rdf:Property),
...
```

Note that the above vocabularies such as `rdf:type` are all shorthands of legal URIs, as described in Section 2. All the above triples are self explanatory. For instance, the triple `(rdf:value, rdf:type, rdf:Property)` states that `rdf:value` is a property.

We suppose that there is a given set $R_{\mathbb{RDF}}$ of RDF resources and a function $S_{\mathbb{RDF}}$: RDFVoc $\rightarrow R_{\mathbb{RDF}}$ which associates a resource with each RDF symbol. It is easy to see that $R_{\mathbb{RDF}}$ and $S_{\mathbb{RDF}}$ can be extended to an RDF-model $\mathbb{RDF}$.

For each theory such that there is a theory morphism f : RDF $\rightarrow (RR, T)$, we consider the model constraint $[\![RR, T]\!]_{\text{RDF}}$ as consisting of those (RR, T)-models $\mathbb{I}$ such that

- $R_{\mathbb{I}}$ includes $R_{\mathbb{RDF}}$ and the restriction of $S_{\mathbb{I}}$ to RDFVoc coincides with $S_{\mathbb{RDF}}$,
- if $p \in P_{\mathbb{I}}$, then $(p, S_{\mathbb{I}}(\texttt{rdf:Property})) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type})$.

Since f is a theory morphism, the restriction of $\mathbb{I}$ to RDFVoc is an RDF-model. We denote by $\widehat{\underline{\text{RDF}}}$ the institution defined by the theory RDF together with the model constraint $[\![\text{–}]\!]_{\text{RDF}}$ using the method presented in Section 2.2.

If we denote by $\widehat{\underline{(\emptyset, \emptyset)}}$ the institution defined by the theory $(\emptyset, \emptyset)$ and the model constraint $[\![RR, T]\!]_{\emptyset} = \mathsf{Mod}(\widehat{\underline{\text{BRDF}}})^{\mathsf{th}}(RR, T)$, then $\widehat{\underline{\text{BRDF}}}^{\mathsf{th}}$ is isomorphic to $\widehat{\underline{(\emptyset, \emptyset)}}$ and we have the institution morphisms $\widehat{\underline{\text{RDF}}} \rightarrow \widehat{\underline{\text{BRDF}}}^{\mathsf{th}} \rightarrow \widehat{\underline{\text{BRDF}}}$.

3.3 The RDF Schema Logic $\widehat{\underline{\text{RDFS}}}$

RDF Schema defines additional language constructs for the RDF language. It expands the expressiveness of RDF by introducing the concept of universe of resources (`rdfs:Resource`), the classification mechanism (`rdfs:Class`) and a set of properties that relate them (`rdfs:subClassOf`, `rdfs:domain`, `rdfs:range`). Hence, it is natural for the RDFS institution $\widehat{\underline{\text{RDFS}}}$ to be developed on top of $\widehat{\underline{\text{RDF}}}$, with some more model constraints added to capture the semantics of RDFS language constructs.

Example 4. Example 1 defines the sub class relationship between two RDF Schema classes. In the shorthand, it can be represented in the informal syntax as follows.

```
(animals:Carnivore, rdfs:subClassOf, animals:Animal)
```

The *RDF Schema theory* RDFS $=$ (RDFSVoc, T_{RDFSVoc}) is composed of the RDF Schema vocabulary RDFSVoc including RDFVoc together with

```
rdfs:domain, rdfs:range, rdfs:Resource,
rdfs:Literal, rdfs:Datatype, rdfs:Class,
rdfs:subClassOf, rdfs:subPropertyOf, rdfs:member,
rdfs:Container, rdfs:ContainerMembershipProperty
```

and the sentences T_{RDFS} including T_{RDFS} together the triples setting the properties of the new symbols (for the whole list of triples see [13]):

```
(rdf:type, rdfs:domain, rdfs:Resource),
(rdfs:domain, rdfs:domain, rdf:Property),
(rdfs:range, rdfs:domain, rdf:Property),
```

```
(rdfs:subPropertyOf, rdfs:domain, rdf:Property),
(rdfs:subClassOf, rdfs:domain, rdfs:Class),
 ...
(rdf:type, rdfs:range, rdfs:Class),
(rdfs:domain, rdfs:range, rdfs:Class),
(rdfs:range, rdfs:range, rdfs:Class),
(rdfs:subPropertyOf, rdfs:range, rdf:Property),
(rdfs:subClassOf, rdfs:range, rdfs:Class),
...
```

We suppose that there is a given set $R_{\mathbb{RDFS}}$ of RDF Schema resources and a function $S_{\mathbb{RDFS}} : \texttt{RDFSVoc} \to R_{\mathbb{RDFS}}$ which associates a resource with each RDF Schema symbol and that satisfies $S_{\mathbb{RDFS}}|_{\texttt{RDFVoc}} = S_{\mathbb{RDF}}$.

For each theory such that there is a theory morphisms $f : \texttt{RDFS} \to (RR, T)$, we define the model constraint $[\![RR, T]\!]_{\text{RDFS}}$ obtained by strengthening $[\![RR, T]\!]_{\text{RDF}}$ with the following conditions. If $\mathbb{I} \in [\![RR, T]\!]_{\text{RDFS}}$, then:

- $R_{\mathbb{I}}$ includes $R_{\mathbb{RDFS}}$ and the restriction of $S_{\mathbb{I}}$ to RDFSVoc coincides with $S_{\mathbb{RDFS}}$
- $\text{ext}_{\mathbb{I}}(\texttt{rdfs:Resource}) = R_{\mathbb{I}}$
- $(\forall\, x, y, u, v : R_{\mathbb{I}})(x, y) \in \text{ext}_{\mathbb{I}}(\texttt{rdfs:domain}) \wedge (u, v) \in \text{ext}_{\mathbb{I}}(x) \Rightarrow u \in \text{ext}_{\mathbb{I}}(y)$
- $(\forall\, x, y, u, v : R_{\mathbb{I}})(x, y) \in \text{ext}_{\mathbb{I}}(\texttt{rdfs:range}) \wedge (u, v) \in \text{ext}_{\mathbb{I}}(x) \Rightarrow v \in \text{ext}_{\mathbb{I}}(y)$
- $(\forall\, x, y : R_{\mathbb{I}})(x, y) \in \text{ext}_{\mathbb{I}}(\texttt{rdfs:subClassOf}) \Rightarrow \text{ext}_{\mathbb{I}}(x) \subseteq \text{ext}_{\mathbb{I}}(y)$
- $(\forall\, x : \text{ext}_{\mathbb{I}}(\texttt{rdf:Class}))(x, S_{\mathbb{I}}(\texttt{rdfs:Resource})) \in \text{ext}_{\mathbb{I}}(\texttt{rdfs:subClassOf}))$
- $(\forall\, x, y : R_{\mathbb{I}})(x, y) \in \text{ext}_{\mathbb{I}}(\texttt{rdfs:subPropertyOf}) \Rightarrow \text{ext}_{\mathbb{I}}(x) \subseteq \text{ext}_{\mathbb{I}}(y)$
- $(\forall\, x : \text{ext}_{\mathbb{I}}(\texttt{rdfs:ContainerMembershipProperty}))$ $(x, S_{\mathbb{I}}(\texttt{rdfs:member})) \in \text{ext}_{\mathbb{I}}(\texttt{rdfs:subPropertyOf})$

In other words, $[\![\_]\!]_{\text{RDFS}}$ gives the intended semantics to syntactic constructions such as domain, range, subClassOf, subPropertyOf, etc.

We denote by $\widehat{\underline{\texttt{RDFS}}}$ the institution such that $\widehat{\underline{\texttt{RDFS}}} \to \widehat{\underline{\texttt{RDF}}}$ is the indexed institution produced by the diagram $\texttt{RDF} \to \texttt{RDFS}$ together with the model constraint $[\![\_]\!]_{\text{RDFS}}$. We have the theory morphisms (inclusions) $(\emptyset, \emptyset) \to \texttt{RDF} \to \texttt{RDFS}$ and $[\![\_]\!]_{\emptyset} \subseteq [\![\_]\!]_{RDF} \subseteq [\![\_]\!]_{RDFS}$. We can formalize now the logics underlying RDF framework as the Grothendieck institution defined by the indexed institution:

$$\widehat{\underline{\texttt{RDFS}}} \longrightarrow \widehat{\underline{\texttt{RDF}}} \longrightarrow \widehat{\underline{\texttt{BRDF}}}^{\,\text{th}} \underset{co}{\rightleftarrows} \widehat{\underline{\texttt{BRDF}}}$$

4 Semantic Web Logics

A number of ontology languages have been proposed in the past years. These include the OWL suite of languages, the OWL$^-$ suite of languages, OWL Flight, etc. They are all based on RDF and RDFS but imposes different restrictions on the usage of RDF(S) language constructs. Hence, their expressiveness is different. In this section,

we construct institutions in an RDF(S)-independent way for some of these languages and inter-relate them using institution morphisms. Then we relate them to RDF(S) by exhibiting the comorphisms defining the RDF serializations. These institutions are incrementally constructed using the same pattern. Therefore we present more details only for the first (smallest) one.

4.1 OWL Lite$^-$ Logic $\widehat{\text{OWLLite}}^-$

OWL Lite$^-$ [5] is a proper subset of OWL Lite (see the next subsection) that can be translated in Datalog. It is obtained from OWL Lite by removing those features considered hard to reason about. OWL Lite$^-$ is the lightest dialect of SW languages and therefore we start with it. We denote by $\widehat{\text{OWLLite}}^-$ the institution of the ontology language OWL Lite$^-$.

Example 5. The class subsumption relationship is allowed in OWL Lite$^-$ as long as neither of the classes is either top ($\top$, the super class of all classes) or bottom ($\bot$, the sub class of all classes, i.e., the empty class). Moreover, `allValuesFrom` restrictions like that mentioned in Example 2 is also allowed in OWL Lite$^-$. Hence, Example 2 is also an OWL Lite$^-$ fragment.

The signatures of $\widehat{\text{OWLLite}}^-$ are triples $\Sigma = (CN, PN, IN)$, where CN is a set of class names, PN is a set of individual property names, and IN is a set of individual names. We assume that CN, PN, and IN are pairwise disjoint. A signature morphism $\phi : \Sigma \to \Sigma'$ is a function $\phi : CN \cup PN \cup IN \to CN' \cup PN' \cup IN'$ such that $\phi(CN) \subseteq CN'$, $\phi(PN) \subseteq PN'$, $\phi(IN) \subseteq IN'$, where $\Sigma' = (CN', PN', IN')$.

A Σ-model $\mathbb{I}$ consists of $(R_\mathbb{I}, S_\mathbb{I}, \text{ext}_\mathbb{I})$, where $R_\mathbb{I}$ is a set of resources, $S_\mathbb{I} : CN \cup PN \cup IN \to R_\mathbb{I}$ is a map such that $S_\mathbb{I}(CN)$, $S_\mathbb{I}(PN)$ and $S_\mathbb{I}(IN)$ are pairwise disjoint, and $\text{ext}_\mathbb{I}$ is a map associating a subset of $R_\mathbb{I}$ with each class name $cn \in CN$, and a subset of $R_\mathbb{I} \times R_\mathbb{I}$ with each property name pn. A Σ-homomorphism $h : \mathbb{I} \to \mathbb{I}'$ between two Σ-models is a function $h : R_\mathbb{I} \to R_{\mathbb{I}'}$ such that $S_\mathbb{I}; h = S_{\mathbb{I}'}$, $\text{ext}_\mathbb{I}|_{CN}; \mathcal{P}(h) = \text{ext}_{\mathbb{I}'}|_{CN}$, and $\text{ext}_\mathbb{I}|_{PN}; \mathcal{P}(h \times h) = \text{ext}_{\mathbb{I}'}|_{PN}$.

For class expressions and Σ-sentences we use a more compact notation:

$$
\begin{aligned}
\mathcal{R}es ::= {} & \texttt{restriction}(pn\ \texttt{allValuesFrom}\ cn) \mid \\
& \texttt{restriction}(pn\ \texttt{minCardinality}(\texttt{0})) \\
\mathcal{C} ::= {} & cn \mid \mathcal{R}es \\
\mathcal{S} ::= {} & \texttt{Class}(cn\ \texttt{partial}\ \mathcal{C}_1 \ldots \mathcal{C}_k) \mid \texttt{Class}(cn\ \texttt{complete}\ cn_1 \ldots cn_k) \mid \\
& \texttt{EquivalentClasses}(cn_1 \ldots cn_k) \mid \\
& \texttt{ObjectProperty}(pn\ \texttt{super}(pn_1) \ldots \texttt{super}(pn_k)) \mid \\
& pn.\texttt{domain}(cn_1 \ldots cn_k) \mid pn.\texttt{range}(cn_1 \ldots cn_k) \mid pn.\texttt{inverseOf}(pn_1) \mid \\
& pn.\texttt{Symmetric} \mid pn.\texttt{Transitive} \mid \\
& \texttt{SubProperty}(pn_1\ pn_2) \mid \texttt{EquivalentProperties}(pn_1 \ldots pn_k) \mid \\
& \texttt{Individual}(in\ \texttt{type}(cn_1) \ldots \texttt{type}(cn_k)) \mid in.\texttt{value}(pn\ in_1)
\end{aligned}
$$

The semantics of expressions is given by:

$$\begin{aligned}&\text{ext}_{\mathbb{I}}(\texttt{restriction}(pn\ \texttt{allValuesFrom}\ cn)) =\\ &\quad \{x \mid (\forall\, y)(x,y) \in \text{ext}_{\mathbb{I}}(pn) \Rightarrow y \in \text{ext}_{\mathbb{I}}(cn)\\ &\text{ext}_{\mathbb{I}}(\texttt{restriction}(pn\ \texttt{minCardinality}(0))) =\\ &\quad \{x \mid \#(\{y \mid (x,y) \in \text{ext}_{\mathbb{I}}(pn)\}) \geq 0\}\end{aligned}$$

The satisfaction relation between OWL Lite$^-$ Σ-models $\mathbb{I}$ and OWL Lite$^-$ Σ-sentences is defined as it is intuitively suggested by syntax. For instance, we have:
$\mathbb{I} \models_\Sigma \texttt{Class}(cn\ \texttt{partial}\ \mathcal{C}_1 \ldots \mathcal{C}_k)$ iff $\text{ext}_{\mathbb{I}}(cn) \subseteq \text{ext}_{\mathbb{I}}(cn_1) \cap \cdots \cap \text{ext}_{\mathbb{I}}(cn_k)$
$\mathbb{I} \models_\Sigma \texttt{ObjectProperty}(pn\ \texttt{super}(pn_1) \ldots \texttt{super}(pn_k))$ iff
$\text{ext}_{\mathbb{I}}(pn) \subseteq \text{ext}_{\mathbb{I}}(pn_1) \cap \cdots \cap \text{ext}_{\mathbb{I}}(pn_k)$
$\mathbb{I} \models_\Sigma pn.\texttt{domain}(cn_1 \ldots cn_k)$ iff $\text{dom}\,\text{ext}_{\mathbb{I}}(pn) \subseteq \text{ext}_{\mathbb{I}}(cn_1) \cap \cdots \cap \text{ext}_{\mathbb{I}}(cn_k)$
$\mathbb{I} \models_\Sigma \texttt{SubProperty}(pn_1\ pn_2)$ iff $\text{ext}_{\mathbb{I}}(pn_1) \subseteq \text{ext}_{\mathbb{I}}(pn_2)$
$\mathbb{I} \models_\Sigma \texttt{Individual}(in\ \texttt{type}(cn_1) \ldots \texttt{type}(cn_k))$ iff $S_{\mathbb{I}}(in) \in \text{ext}_{\mathbb{I}}(cn_1) \cap \cdots \cap \text{ext}_{\mathbb{I}}(cn_k)$
$\mathbb{I} \models_\Sigma in.\texttt{value}(pn\ in_1)$ iff $(S_{\mathbb{I}}(in), S_{\mathbb{I}}(in_1)) \in \text{ext}_{\mathbb{I}}(pn)$
...

4.2 OWL Lite Logic $\widehat{\text{OWLLite}}$

OWL Lite is the least expressive species of the OWL suite. It is obtained by imposing some constraints on OWL Full. These constraints include, for example, that the sets of classes, properties and individuals are mutually disjoint; that min and max cardinality restrictions can only be applied on numbers 0 and 1; that value restrictions such as `allValuesFrom` and `someValuesFrom` can only be applied to named classes. Compared with OWL Lite$^-$, OWL Lite is more expressive since it removes some constraints that are imposed on the latter. The details are discussed in the following. We denote by $\widehat{\text{OWLLite}}$ the institution of OWL Lite.

Example 6. For example, OWL Lite$^-$ does not support the relationship between OWL individuals, namely `sameAs` and `differentFrom`, whereas these features are present in OWL Lite. Suppose that we have two URI references for carnivores `car1` and `car2`, which are actually referring to the same animal. We use the following code fragment to represent this piece of knowledge:

```
<animals:Carnivore rdf:ID="car1"/>
<animals:Carnivore rdf:ID="car2"/>
<animals rdf:about="#car1>
   <owl:sameAs rdf:resource="http://ex.com/animals#car2/>
</animals>
```

$\widehat{\text{OWLLite}}$ is obtained from $\widehat{\text{OWLLite}}^-$ by replacing the definition of expressions with

$$\begin{aligned}\mathcal{R}es ::= &\ \texttt{restriction}(pn\ \texttt{allValuesFrom}\ cn) \mid\\ &\ \texttt{restriction}(pn\ \texttt{someValuesFrom}\ cn) \mid\\ &\ \texttt{restriction}(pn\ \texttt{minCardinality}(\texttt{n})) \mid\\ &\ \texttt{restriction}(pn\ \texttt{maxCardinality}(\texttt{n}))\\ \mathcal{C} ::= &\ cn \mid \texttt{owl:Thing} \mid \texttt{owl:Nothing} \mid \mathcal{R}es\end{aligned}$$

where $n \in \{0, 1\}$, and adding the following sentences:

$$pn.\texttt{Functional} \mid pn.\texttt{InverseFunctional} \mid$$
$$\texttt{SameIndividual}(in_1, \ldots, in_k) \mid \texttt{DifferentIndividuals}(in_1, \ldots, in_k)$$

The semantics of the new expressions is as follows:

$\text{ext}_{\mathbb{I}}(\texttt{owl:Thing})$ is a subset of $R_{\mathbb{I}}$ s.t. $(\forall\, cn \in CN)\text{ext}_{\mathbb{I}}(cn) \subseteq \text{ext}_{\mathbb{I}}(\texttt{owl:Thing})$,
$\text{ext}_{\mathbb{I}}(\texttt{owl:Nothing}) = \emptyset$,
$\text{ext}_{\mathbb{I}}(\texttt{restriction}(pn\ \texttt{someValuesFrom}\ cn)) =$
$\quad \{x \mid (\exists\, y)(x, y) \in \text{ext}_{\mathbb{I}}(pn) \wedge y \in \text{ext}_{\mathbb{I}}(cn)\}$,
$\text{ext}_{\mathbb{I}}(\texttt{restriction}(pn\ \texttt{minCardinality}(1)\ cn)) =$
$\quad \{x \mid \#\{y \mid (x, y) \in \text{ext}_{\mathbb{I}}(pn)\} \geqslant 1\}$,
$\text{ext}_{\mathbb{I}}(\texttt{restriction}(pn\ \texttt{maxCardinality}(1)\ cn)) =$
$\quad \{x \mid \#\{y \mid (x, y) \in \text{ext}_{\mathbb{I}}(pn)\} \leqslant 1\}$.

The satisfaction of the new sentences is intuitive and straightforward and we omit its formal definition.

Proposition 1. *There is a conservative comorphism from* $\widehat{\text{OWLLite}}^{-}$ *to* $\widehat{\text{OWLLite}}$.

4.3 OWL DL$^-$ Logic $\widehat{\text{OWLDesLog}}^{-}$

OWL DL$^-$ [5] is an extension of OWL Lite$^-$ and a subset of OWL DL (see the next subsection) which can also be translated in Datalog. We denote by $\widehat{\text{OWLDesLog}}$ the institution of OWL DL$^-$. Compared to OWL Lite$^-$, OWL DL$^-$ allows additional language constructs `value`, `someValuesFrom` and `oneOf`, albeit that the latter two are only allowed as the first argument of `subClassOf` (left hand side).

Example 7. In OWL DL$^-$, the `value` restriction not present in OWL Lite$^-$ is allowed in OWL DL$^-$. This restriction constructs a class that for a given property, each of whose instances must have (among others) a particular individual as the value mapped by this property. Suppose that we want to model the fact that the ancestor of all humans is Adam (among all his/her other ancestors), assuming that we have defined an individual `Adam` and a property `hasAncestor`. Here is the definition in OWL DL$^-$.

```
<owl:Class rdf:ID="Human">
   <rdfs:subClassOf><owl:Restriction>
      <owl:onProperty rdf:resource="#hasAncestor"/>
      <owl:hasValue rdf:resource="#Adam"/>
   </owl:Restriction></rdfs:subClassOf>
</owl:Class>
```

$\widehat{\text{OWLDesLog}}^{-}$ is obtained from $\widehat{\text{OWLLite}}^{-}$ by replacing the definition of expressions with

$$
\begin{aligned}
\mathcal{C} &::= cn \mid \mathcal{R}es \mid \texttt{intersectionOf}(\mathcal{C}_1, \ldots, \mathcal{C}_n) \\
\mathcal{L}hs\_D &::= \mathcal{C} \mid \mathcal{L}hs\_Res \mid \texttt{unionOf}(\mathcal{L}hs\_D, \ldots, \mathcal{L}hs\_D) \mid \texttt{oneOf}(in_1, \ldots, in_k) \\
\mathcal{R}hs\_D &::= \mathcal{C} \mid \mathcal{R}hs\_Res \\
\mathcal{R}es &::= \texttt{restriction}(pn\ \texttt{value}(in)) \\
\mathcal{L}hs\_Res &::= \mathcal{R}es \mid \texttt{restriction}(pn\ \texttt{someValuesFrom}(\mathcal{L}hs\_D)) \\
&\quad \mid \texttt{restriction}(pn\ \texttt{minCardinality}(1)) \\
\mathcal{R}hs\_Res &::= \mathcal{R}es \mid \texttt{restriction}(pn\ \texttt{allValuesFrom}(\mathcal{R}hs\_D))
\end{aligned}
$$

and replacing the class-related sentences with the following:

$$
\begin{aligned}
&\texttt{Class}\,(cn\ \texttt{partial}\,\mathcal{R}hs\_D) \mid \texttt{Class}\,(cn\ \texttt{complete}\,\mathcal{C}) \mid \\
&\texttt{EquivalentClass}(\mathcal{C}_1, \ldots, \mathcal{C}_n) \mid \texttt{subClassOf}(\mathcal{L}hs\_D, \mathcal{R}hs\_D)
\end{aligned}
$$

Note the use of class expressions instead of named classes. $\mathcal{L}hs\_D$ and $\mathcal{R}hs\_D$ represent class descriptions that can only appear in the left hand side and right hand side of the `subClassOf` sentence, respectively.

The semantics of the value restriction is $\text{ext}_{\mathbb{I}}(\texttt{restriction}(pn\ \texttt{value}\ in)) = \{x \mid (x, S_{\mathbb{I}}(in)) \in \text{ext}_{\mathbb{I}}(pn)\}$. The semantics of the other expressions and the satisfaction relation for the new sentences are defined as expected.

Proposition 2. *There is a conservative comorphism from* $\widehat{\text{OWLLite}}^{-}$ *to* $\widehat{\text{OWLDesLog}}^{-}$.

4.4 OWL DL Logic $\widehat{\text{OWLDesLog}}$

OWL DL is the main ontology language of the OWL suite. It is more expressive than OWL Lite yet still decidable. It relaxes some constraints imposed on OWL Lite and allows more language constructs to describe classes and properties. Still, classes, properties and individuals are mutually disjoint in OWL DL. We denote by $\widehat{\text{OWLDesLog}}$ the institution of OWL DL. Compared to OWL DL^{-}, OWL DL adds a number of language features, such as enumerated class, disjointness classes, functional property, etc.

Example 8. The class `Continents` defines the continents of the Earth, namely Africa, Antarctica, Asia, Australia, Europe, North America and South America. As this class only contains these 7 instances, it is natural to use an enumeration to define it.

```
<owl:Class rdf:ID="Continents">
   <owl:oneOf rdf:parseType="Collection">
      <owl:Thing rdf:about="#Africa"/>
      <owl:Thing rdf:about="#Antarctica"/>
      <owl:Thing rdf:about="#Asia"/>
      <owl:Thing rdf:about="#Australia"/>
      <owl:Thing rdf:about="#Europe"/>
```

```
      <owl:Thing rdf:about="#North America"/>
      <owl:Thing rdf:about="#South America"/>
   </owl:oneOf>
</owl:Class>
```

$\underline{\widehat{\text{OWLDesLog}}}$ is obtained from $\underline{\widehat{\text{OWLLite}}}$ replacing the definition of expressions with

$$\begin{aligned}
\mathcal{R}es ::= {} & \texttt{restriction}(pn\ \texttt{value}\ in) \mid \\
& \texttt{restriction}(pn\ \texttt{allValuesFrom}\ \mathcal{C}) \mid \\
& \texttt{restriction}(pn\ \texttt{someValuesFrom}\ \mathcal{C}) \mid \\
& \texttt{restriction}(pn\ \texttt{minCardinality}(\texttt{n})) \mid \\
& \texttt{restriction}(pn\ \texttt{maxCardinality}(\texttt{n})) \mid \\
\mathcal{C} ::= {} & cn \mid \texttt{owl:Thing} \mid \texttt{owl:Nothing} \mid \mathcal{R}es \\
& \texttt{intersectionOf}(\mathcal{C}_1, \ldots, \mathcal{C}_k) \mid \texttt{unionOf}(\mathcal{C}_1, \ldots, \mathcal{C}_k) \mid \\
& \texttt{complementOf}(\mathcal{C}) \mid \texttt{oneOf}(in_1, \ldots, in_k)
\end{aligned}$$

and adding the following sentences:

$$\begin{aligned}
& \texttt{EnumeratedClass}(cn\ in_1, \ldots, in_k) \mid \texttt{SubClassOf}(\mathcal{C}_1, \mathcal{C}_2) \mid \\
& \texttt{DisjointClasses}(\mathcal{C}_1, \ldots, C_k) \mid \texttt{EquivalentClasses}(\mathcal{C}_1, \ldots, \mathcal{C}_k) \mid \\
& pn.\texttt{domain}(\mathcal{C}_1 \ldots \mathcal{C}_k) \mid pn.\texttt{range}(\mathcal{C}_1 \ldots \mathcal{C}_k) \mid \\
& \texttt{Individual}(in\ \texttt{type}(\mathcal{C}_1), \ldots, \texttt{type}(\mathcal{C}_k)) \mid in.\texttt{value}(pn_1\ in_1)
\end{aligned}$$

The semantics of the new expressions and the satisfaction relation for the new sentences are defined as expected.

Proposition 3. *a) There is a conservative comorphism from* $\underline{\widehat{\text{OWLLite}}}$ *to* $\underline{\widehat{\text{OWLDesLog}}}$.
b) There is a conservative comorphism from $\underline{\widehat{\text{OWLDesLog}}^{-}}$ *to* $\underline{\widehat{\text{OWLDesLog}}}$.

4.5 OWL Full Logic $\underline{\widehat{\text{OWLFull}}}$

OWL Full adds a number of features on top of OWL DL and also removes some restrictions. The vocabulary no longer needs to be separated. This means an identifier can denote a class, an individual and/or a property at the same time.

Let X be a name denoting a class and a property in the same ontology $\Sigma = ((CN, PN, IN), \mathsf{F})$, i.e., $X \in CN \cap PN$, and let $\mathbb{I}$ be a (Σ, F)-model. For the moment, we denote with $X{:}CN$ the occurrence of X as a class and with $X{:}PN$ the occurrence of X as a property. We have $S_{\mathbb{I}}(X{:}CN) = S_{\mathbb{I}}(X{:}PN) = S_{\mathbb{I}}(X)$, $\text{ext}_{\mathbb{I}}(X{:}CN) \subseteq R_{\mathbb{I}}$, and $\text{ext}_{\mathbb{I}}(X{:}PN) \subseteq R_{\mathbb{I}} \times R_{\mathbb{I}}$. Since X denotes just one entity, we relate the two sets by means of a bijection $rdef_{\mathbb{I}}(X) : \text{ext}_{\mathbb{I}}(X{:}PN) \rightarrow \text{ext}_{\mathbb{I}}(X{:}CN)$. We may think that $rdef_{\mathbb{I}}(X)(r_1, r_2)$ is the URL address where the pair (r_1, r_2) is defined as an instance of $\text{ext}_{\mathbb{I}}(X{:}PN)$. If X denotes a class (property) and an individual, then its meaning as an individual is given by $S_{\mathbb{I}}(X)$ and its meaning as class (property) is $\text{ext}_{\mathbb{I}}(X)$.

Also, keywords of the language can be used in place of classes, properties and individuals, and restrictions. For instance, we may assume that `subClassOf` and

subPropertyOf are in PN. Then for X, Y denoting both classes and properties, we have subClassOf(X, Y) iff SubPropertyOf(X, Y); this is semantically expressed by $(S_{\mathbb{I}}(X{:}CN), S_{\mathbb{I}}(Y{:}CN)) \in \text{ext}_{\mathbb{I}}(\texttt{SubClassOf})$ iff $(S_{\mathbb{I}}(X{:}PN), S_{\mathbb{I}}(Y{:}PN)) \in \text{ext}_{\mathbb{I}}(\texttt{SubPropertyOf})$.

We skip the formal definition of OWL Full here. The new features are added in a similar way to other languages. In the definition of the signatures we remove the restriction as the sets CN, PN, IN to be pairwise disjoint. The corresponding restriction from the definition of models is also removed.

The original definition of OWL Full [20] is given directly over RDF Schema. Here we refer an RDF independent definition for OWL Full. The fact that OWL Full is built over RDF Schema is given by the following result.

Proposition 4. *There is a conservative comorphism from* $\widehat{\text{RDFS}}$ *to* $\widehat{\text{OWLFull}}^{\text{th}}$.

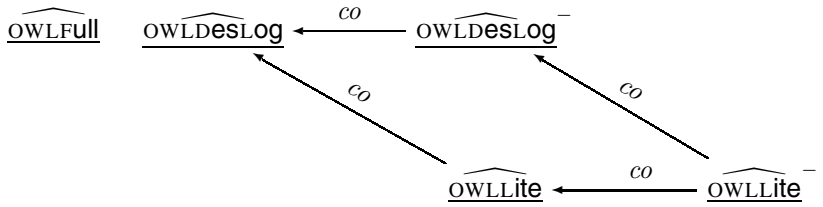
It is easy to see that we cannot embed $\widehat{\text{RDFS}}$ in $\widehat{\text{OWLDesLog}}$. For instance, triples like (rdf:type, rdf:type, rdf:Property) cannot be encoded in $\widehat{\text{OWLDesLog}}$ but can be expressed as a sentence in $\widehat{\text{OWLFull}}$.

If (Σ, F) is an ontology in OWL DL, then, syntactically, it is also an ontology in OWL Full. However, the class of (Σ, F)-models in $\widehat{\text{OWLFull}}$ is richer than that of (Σ, F)-models in $\widehat{\text{OWLDesLog}}$. The reason is that in OWL Full we removed some model constraints which are present in OWL DL. Hence we have the following result:

Proposition 5. *There is not an embedding comorphism from* $\widehat{\text{OWLDesLog}}$ *to* $\widehat{\text{OWLFull}}$.

The theorem above has a drastic consequence: it could be unsound to use OWL Full reasoners for OWL DL ontologies. This is refereed in literature as inappropriate layering [5].

The relationships between SW logics are expressed by the following diagram:



Between $\widehat{\text{OWLFull}}$ and $\widehat{\text{OWLDesLog}}$ we can define only a "syntactical comorphism" consisting of an inclusion functor $\mathsf{Sign}(\widehat{\text{OWLDesLog}}) \to \Phi : \mathsf{Sign}(\widehat{\text{OWLFull}})$ and a natural transformation $\alpha : \Phi; \mathsf{sen}(\widehat{\text{OWLDesLog}}) \to \mathsf{sen}(\widehat{\text{OWLFull}})$.

5 RDF Serialization of Semantic Web Logics

In this section, we define the RDF serialization of the SW languages discussed in the previous section. In terms of institution theory, an RDF serialization is a comorphism (encoding) from the source language to an institution built over an RDF theory as in

Section 2.2. Since the corresponding theories are related by morphisms, we get an indexed institution. This approach results in a much clearer understanding of the relationship among the various languages, as is shown at the end of the section.

5.1 RDF Serialization of OWL Lite$^-$

We define the theory $\texttt{OWLLM} = (\texttt{OWLLMVoc}, T_{\texttt{OWLLM}})$, where OWLLMVoc is RDFSVoc together with an enumerable set of anonymous names and the symbols

```
owl:allValuesFrom, owl:Class, owl:equivalentClass,
owl:equivalentProperty, owl:hasValue, owl:inverseOf,
owl:minCardinality, owl:ObjectProperty,
owl:SymmetricProperty, owl:TransitiveProperty,
owl:Restriction, owl:onProperty,
owl:allValuesFrom, owl:hasValue,owl:minCardinality
```

and $T_{\texttt{OWLLM}}$ is defined as $T_{\texttt{OWLLM}} = T_{\texttt{RDFS}} \quad \cup$

```
{
  (owl:Class, rdfs:subClassOf, rdf:Class),
  (owl:allValuesFrom, rdf:type, rdf:Property),
  (owl:allValuesFrom, rdfs:domain, rdf:Property),
  (owl:equivalentProperties, rdf:type, rdf:Property),
  (owl:equivalentProperties, rdfs:domain, rdf:Property),
  (owl:equivalentProperties, rdfs:subPropertyOf,
                                  rdfs:subPropertyOf),
  (owl:ObjectProperty, rdf:type, rdfs:Class),
  (owl:inverseOf, rdf:type, rdf:Property),
  ...
}
```

The anonymous names are used in translating OWL sentences into conjunctions of triples [20]. As for RDF and RDF Schema, we suppose that there is a given set $R_{\mathbb{OWLLM}}$ of OWL Lite$^-$ resources and a function $S_{\mathbb{OWLLM}} : \texttt{OWLLMVoc} \to R_{\mathbb{OWLLM}}$ which associates a resource to each OWL Lite$^-$ symbol, and satisfies $S_{\mathbb{OWLLM}}|_{\texttt{RDFSVoc}} = S_{\mathbb{RDFS}}$.

For each theory such that there is a theory morphisms $f : \texttt{OWLLM} \to (RR, T)$, we define the model constraint $[\![RR, T]\!]_{\text{OWLLM}}$ obtained by strengthening $[\![RR, T]\!]_{\text{RDFS}}$ with the following conditions. If $\mathbb{I} \in [\![RR, T]\!]_{\text{OWLLM}}$, then:

- $R_{\mathbb{I}}$ includes $R_{\mathbb{OWLLM}}$ and the restriction of $S_{\mathbb{I}}$ to OWLLMVoc coincides with $S_{\mathbb{OWLLM}}$.
- $\text{ext}_{\mathbb{I}}(\texttt{owl:Class}) \cap \text{ext}_{\mathbb{I}}(\texttt{owl:ObjectProperty}) = \emptyset$
- $(\forall\, x, y)(x, \texttt{owl:Class}) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type}) \wedge (y, x) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type}) \Rightarrow$
  $((y, \texttt{owl:Class}) \notin \text{ext}_{\mathbb{I}}(\texttt{rdf:type})) \wedge$
  $((y, \texttt{rdf:Property}) \notin \text{ext}_{\mathbb{I}}(\texttt{rdf:type}))$
- $(\forall\, x, y)(x, y) \in \text{ext}_{\mathbb{I}}(\texttt{owl:equivalentClass}) \Rightarrow$
  $(x, S_{\mathbb{I}}(\texttt{owl:Class})) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type}) \wedge$
  $(y, S_{\mathbb{I}}(\texttt{owl:Class})) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type}) \wedge \quad \text{ext}_{\mathbb{I}}(x) = \text{ext}_{\mathbb{I}}(y)$
- $(\forall\, u, w, v)(w, S_{\mathbb{I}}(\texttt{owl:Restriction})) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type}) \wedge$
  $(w, u) \in \text{ext}_{\mathbb{I}}(\texttt{owl:onProperty}) \wedge (w, v) \in \text{ext}_{\mathbb{I}}(\texttt{xowl:allValuesFrom}) \Rightarrow$
  $\text{ext}_{\mathbb{I}}(w) = \{x \mid (x, y) \in \text{ext}_{\mathbb{I}}(u) \Rightarrow y \in \text{ext}_{\mathbb{I}}(v)\}$

- $(\forall\, u, v, w)(w, S_{\mathbb{I}}(\texttt{owl:Restriction})) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type}) \wedge$
  $(w, u) \in \text{ext}_{\mathbb{I}}(\texttt{owl:onProperty}) \wedge (w, v) \in \text{ext}_{\mathbb{I}}(\texttt{owl:hasValue}) \Rightarrow$
  $\text{ext}_{\mathbb{I}}(w) = \{x \mid (x, v) \in \text{ext}_{\mathbb{I}}(u)\}$
- $(\forall\, u, w, y)(w, S_{\mathbb{I}}(\texttt{owl:Restriction})) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type}) \wedge$
  $(w, u) \in \text{ext}_{\mathbb{I}}(\texttt{owl:onProperty}) \wedge (w, 0) \in \text{ext}_{\mathbb{I}}(\texttt{owl:minCardinality}) \Rightarrow$
  $\text{ext}_{\mathbb{I}}(w) = \{x \mid \#(\{(x, y) \in \text{ext}_{\mathbb{I}}(u)\}) \geq 0$
- $(\forall\, v, w, x, y)(x, y) \in \text{ext}_{\mathbb{I}}(\texttt{owl:inverseOf}) \Rightarrow$
  $(x, S_{\mathbb{I}}(\texttt{owl:ObjectProperty})) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type}) \wedge$
  $(y, S_{\mathbb{I}}(\texttt{owl:ObjectProperty})) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type}) \wedge$
  $(w, v) \in \text{ext}_{\mathbb{I}}(x) \Leftrightarrow (v, w) \in \text{ext}_{\mathbb{I}}(y)$
- $(\forall\, u, x, y)(u, S_{\mathbb{I}}(\texttt{owl:SymmetricProperty})) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type}) \wedge$
  $(x, y) \in \text{ext}_{\mathbb{I}}(u) \Rightarrow (y, x) \in \text{ext}_{\mathbb{I}}(u)$
- $(\forall\, u, x, y, z)(u, S_{\mathbb{I}}(\texttt{owl:TransitiveProperty})) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type}) \wedge$
  $(x, y) \in \text{ext}_{\mathbb{I}}(x) \wedge (y, z) \in \text{ext}_{\mathbb{I}}(x) \Rightarrow (x, z) \in \text{ext}_{\mathbb{I}}(x)$
- $(\forall\, x, y)(x, y) \in \text{ext}_{\mathbb{I}}(\texttt{owl:equivalentProperty}) \Rightarrow$
  $(x, S_{\mathbb{I}}(\texttt{rdf:Property})) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type}) \wedge$
  $(y, S_{\mathbb{I}}(\texttt{rdf:Property})) \in \text{ext}_{\mathbb{I}}(\texttt{rdf:type}) \wedge \text{ext}_{\mathbb{I}}(x) = \text{ext}_{\mathbb{I}}(y)$

The second and the third conditions say that the vocabulary is separated: a resource cannot be a class, an individual and/or a property at the same time. The last three conditions give the intended meaning of the symmetric property, transitive property, and equivalent property, respectively. The other conditions give semantics to the new syntactical constructions. We denote by $\widehat{\underline{\texttt{OWLLM}}}$ the institution generated by the theory OWLLM and the model constraint $[\![\_]\!]_{\text{OWLLM}}$ using the method presented in Section 2.2.

Proposition 6. *There is a conservative comorphism* (Φ, β, α) *from* $\widehat{\underline{\text{OWLLite}}}^{-}$ *to* $(\widehat{\underline{\texttt{OWLLM}}}^{\text{th}})^{\wedge}$.

Here is a brief description of the comorphism given by Proposition 6.
$\Phi : \mathsf{Sign}(\widehat{\underline{\text{OWLLite}}}^{-}) \to \mathsf{Sign}((\widehat{\underline{\texttt{OWLLM}}}^{\text{th}})^{\wedge})$ is defined as follows. If $\Sigma = (CN, PN, IN)$ is an OWL Lite^{-} signature, then $\Phi(\Sigma)$ is $\Sigma : \texttt{OWLLM} \to (RR, T)$, where $RR = \texttt{OWLLMVoc} \cup CN \cup PN \cup IN$, and T includes $T_{\texttt{OWLLM}}$ together with:
a triple $(cn, \texttt{rdf:type}, \texttt{owl:Class})$ for each $cn \in CN$,
a triple $(pn, \texttt{rdf:type}, \texttt{owl:ObjectProperty})$ for each $pn \in PN$, and
a triple $(in, \texttt{rdf:type}, \texttt{rdf:Resource})$ for each $in \in IN$.
$\beta : \Phi; \mathsf{Mod}((\widehat{\underline{\texttt{OWLLM}}}^{\text{th}})^{\wedge}) \Rightarrow \mathsf{Mod}(\widehat{\underline{\text{OWLLite}}}^{-})$ is defined as follows. If $\mathbb{I}'$ is a $\Phi(\Sigma)$-model, then $\beta_{\Sigma}(\mathbb{I}') = \mathbb{I}$, where $R_{\mathbb{I}} = R_{\mathbb{I}'}$, $S_{\mathbb{I}}(name) = S_{\mathbb{I}'}(name)$ for each $name \in CN \cup PN \cup IN$, $\text{ext}_{\mathbb{I}}(pn) = \text{ext}_{\mathbb{I}'}(pn)$, and
$\text{ext}_{\mathbb{I}}(cn) = \{x \mid (x, S_{\mathbb{I}'}(cn)) \in \text{ext}_{\mathbb{I}'}(\texttt{rdf:type})\}$.
$\alpha : \mathsf{sen}(\widehat{\underline{\text{OWLLite}}}^{-}) \Rightarrow \Phi; \mathsf{sen}((\widehat{\underline{\texttt{OWLLM}}}^{\text{th}})^{\wedge})$ is given such that α_{Σ} associates with each OWL Lite^{-} syntactical construction a set (conjunction) of triples similar to that defined in [20]. If Σ is an OWL Lite^{-}-signature and M a Σ-model, then M can be extended to $\Phi(\Sigma)$-model by giving semantics to symbols in OWLLMVoc according to triples in $T_{\texttt{OWLLM}}$.

5.2 RDF Serialization of OWL Lite

We define the theory $\texttt{OWLL} = (\texttt{OWLLVoc}, T_{\texttt{OWLL}})$, where OWLLVoc is OWLLMVoc together with

```
owl:Thing, owl:Nothing, owl:FunctionalProperty,
owl:SameIndividual, DifferentIndividuals, owl:someValues,
owl:maxCardinality
```

and $T_{\texttt{OWLL}}$ is defined as $T_{\texttt{OWLL}} = T_{\texttt{OWLLM}} \quad \cup$

```
{
  (owl:Thing, rdf:type, rdfs:Class),
  (owl:Nothing, rdf:type, rdfs:Class),
  (owl:FunctionalProperty, rdf:type, rdfs:Class),
  (owl:InverseFunctionalProperty, rdf:type, rdfs:Class),
  (owl:InverseFunctionalProperty, rdfs:subClassOf,
                                    owl:ObjectProperty),
  (owl:sameAs, rdf:type, rdf:Property),
  ...
}
```

As for OWL Lite$^-$, we suppose that there is given a set $R_{\mathbb{OWLL}}$ of RDF Schema resources and a function $S_{\mathbb{OWLL}} : \texttt{OWLL} \to R_{\mathbb{OWLL}}$ which associates a resource to each OWL Lite symbol, and satisfies $S_{\mathbb{OWLL}}|_{\texttt{OWLLMVoc}} = S_{\mathbb{OWLLM}}$.

For each theory such that there is a theory morphisms $f : \texttt{OWLL} \to (RR, T)$ we define the model constraint $[\![RR, T]\!]_{\text{OWLL}}$ obtained by strengthening $[\![RR, T]\!]_{\text{OWLLM}}$ with the following conditions. If $\mathbb{I} \in [\![RR, T]\!]_{\text{OWLL}}$, then:

- $R_{\mathbb{I}}$ includes $R_{\mathbb{OWLL}}$ and the restriction of $S_{\mathbb{I}}$ to OWLLVoc coincides with $S_{\mathbb{OWLL}}$.
- $\mathbb{I}$ satisfies the restrictions expressing the intended meaning of the new features.

We denote by $\widehat{\underline{\texttt{OWLL}}}$ the institution generated by the theory OWLL and the model constraint $[\![\_]\!]_{\text{OWLL}}$ using the method presented in Section 2.2.

Theorem 1. *There is a conservative comorphism from* $\widehat{\underline{\texttt{OWLLite}}}$ *to* $(\widehat{\underline{\texttt{OWLL}}}^{\,\mathsf{th}})^{\wedge}$.

5.3 RDF Serialization of OWL DL$^-$

We define the theory $\texttt{OWLDLM} = (\texttt{OWLDLMVoc}, T_{\texttt{OWLDLM}})$, where the vocabulary OWLDLMVoc is OWLLMVoc together with

```
owl:SubClassOf, owl:intersectionOf, owl:unionOf, owl:oneOf,
owl:someValues, owl:hasValue
```

and $T_{\texttt{OWLDLM}}$ is defined as $T_{\texttt{OWLDLM}} = T_{\texttt{OWLLM}} \quad \cup$

```
{
  (owl:intersectionOf, rdf:type, rdf:Property),
  (owl:unionOf, rdf:type, rdf:Property),
  (owl:oneOf, rdf:type, rdf:Property),
  (owl:oneOf, rdfs:range, rdf:List),
  ...
}
```

As usual, we suppose that there is a given set $R_{\mathbb{OWLDLM}}$ of RDF Schema resources and a function $S_{\mathbb{OWLDLM}} : \texttt{OWLDLM} \to R_{\mathbb{OWLDLM}}$ which associates a resource to each OWL DL$^-$ symbol, and satisfies $S_{\mathbb{OWLDLM}}|_{\texttt{OWLLMVoc}} = S_{\mathbb{OWLLM}}$.

For each theory such that there is a theory morphisms $f : \texttt{OWLDLM} \to (RR, T)$ we define the model constraint $[\![RR, T]\!]_{\text{OWLDLM}}$ obtained by strengthening $[\![RR, T]\!]_{\text{OWLLM}}$ with the following conditions. If $\mathbb{I} \in [\![RR, T]\!]_{\text{OWLDLM}}$, then:

- $R_{\mathbb{I}}$ includes $R_{\mathbb{OWLDLM}}$ and the restriction of $S_{\mathbb{I}}$ to OWLDLMVoc is $S_{\mathbb{OWLDLM}}$.
- $\mathbb{I}$ satisfies the restrictions expressing the intended meaning of the new features.

We denote by $\underline{\widehat{\texttt{OWLDLM}}}$ the institution generated by the theory OWLDLM and the model constraint $[\![\_]\!]_{\text{OWLDLM}}$ using the method presented in Section 2.2.

Theorem 2. *There is a conservative comorphism from* $\underline{\text{OWL}\widehat{\text{DesLog}}^-}$ *to* $(\underline{\widehat{\texttt{OWLDLM}}}^{\text{th}})^\wedge$.

5.4 RDF Serialization of OWL DL

We define the theory $\texttt{OWLDL} = (\texttt{OWLDLVoc}, T_{\texttt{OWLDL}})$, where OWLDLVoc is OWLLVoc together with

```
owl:DepricatedClass, owl:DisjointClasses, owl:SubClassOf,
owl:Functional,  owl:InverseFunctional, owl:Transitive,
owl:SameIndividual, DifferentIndividuals, owl:someValues,
owl:Thing, owl:Nothing,
owl:intersectionOf, owl:unionOf, owl:complementOf,
owl:oneOf, owl:someValues, owl:hasValue, owl:maxCardinality
```

and $T_{\texttt{OWLDL}}$ is defined as $T_{\texttt{OWLDL}} = T_{\texttt{OWLL}} \quad \cup$

```
{
  (owl:intersectionOf, rdf:type, rdf:Property),
  (owl:intersectionOf, rdfs:domain, owl:Class),
  (owl:equivalentClass, rdf:type, rdf:Property),
  (owl:disjointWith, rdf:type, rdf:Property),
  ...
}
```

As usual, we suppose that there is given a set $R_{\mathbb{OWLDL}}$ of RDF Schema resources and a function $S_{\mathbb{OWLDL}} : \texttt{OWLDL} \to R_{\mathbb{OWLDL}}$ which associates a resource to each OWL DL symbol, and satisfies $S_{\mathbb{OWLDLM}}|_{\texttt{OWLLVoc}} = S_{\mathbb{OWLL}}$.

For each theory such that there is a morphisms $f : \texttt{OWLDL} \to (RR, T)$ we define the model constraint $[\![RR, T]\!]_{\text{OWLDL}}$ obtained by strengthening $[\![RR, T]\!]_{\text{OWLL}}$ with the following conditions. If $\mathbb{I} \in [\![RR, T]\!]_{\text{OWLDL}}$, then:

- $R_{\mathbb{I}}$ includes $R_{\mathbb{OWLDL}}$ and the restriction of $S_{\mathbb{I}}$ to OWLDLVoc coincides with $S_{\mathbb{OWLDL}}$.
- $\mathbb{I}$ satisfies the restrictions expressing the intended meaning of the new features.

We denote by $\underline{\widehat{\texttt{OWLDL}}}$ the institution generated by the theory OWLDL and the model constraint $[\![\_]\!]_{\text{OWLDL}}$ using the method presented in Section 2.2.

Theorem 3. *There is a conservative comorphism from* $\underline{\text{OWL}\widehat{\text{DesLog}}}$ *to* $(\underline{\widehat{\texttt{OWLDL}}}^{\text{th}})^\wedge$.

5.5 RDF Serialization of OWL Full

The theory $\mathtt{OWLF} = (\mathtt{OWLFVoc}, T_{\mathtt{OWLF}})$ is defined as follows. The vocabulary $\mathtt{OWLFVoc}$ includes $\mathtt{OWLDLVoc}$ and the symbols corresponding to the new features. Similarly, $T_{\mathtt{OWLF}}$ includes $T_{\mathtt{OWLDL}}$ together with triples restricting the use of the new symbols as intended and triples expressing the equality of the parts of the OWL universe with their analogues in RDF Schema.

$R_{\mathbb{OWLF}}$ and $S_{\mathbb{OWLF}}$ are defined as usual. The model constraint $[\![\_]\!]_{\mathrm{OWLF}}$ includes:

- The restriction corresponding to $R_{\mathbb{OWLF}}$ and $S_{\mathbb{OWLF}}$.
- Restrictions expressing the intended meaning of all the features.
- Restrictions that force the parts of the OWL universe to be the same as their analogues in RDF.

The vocabulary separation restriction is not included in $[\![\_]\!]_{\mathrm{OWLF}}$.

Theorem 4. *There is a conservative comorphism from* $\widehat{\underline{\mathtt{OWLFull}}}$ *to* $(\widehat{\underline{\mathtt{OWLF}}}^{\mathsf{th}})^{\wedge}$.

5.6 Summing Up

All institutions we defined in this paper and their relationships are represented in Figure 2. The lower side includes the RDF indexed institution and it gives the semantics for RDF layer in the Berners-Lee's stack. It is worth to note that all arrows are institution morphisms; hence we may define the semantics of the layer as being the Grothendieck institution defined by this indexed institution. The upper side includes the institutions corresponding to the SW languages and their relationships expressed as comorphisms. The Grothendieck institution defined by this indexed institution gives the semantics for ontology layer. The semantics of the layering of web ontology languages on the RDF framework is given by a comorphism of Grothendieck institutions. Note that the embedding of RDF Schema in OWL Full is not a component of this comorphism.

6 Conclusion

The multitude of languages causes certain confusion in the Semantic Web community as they are based on different formalisms (description logics, Datalog, RDF Schema, etc.). A careful and thorough investigation of the relationship among the various languages will certainly reveal subtle differences among them.

Institutions and institution morphisms were developed to capture the notion of "logical systems" and relate software systems regardless of the underlying logical system. Hence, it is natural to use institutions to represent the various Semantic Web languages (including RDF and RDF Schema) and study their relationship using institution morphism.

In this paper, based on RDF(S), we define indexed institutions for RDF framework layer and ontology layer. An overall relationship among all these languages can be seen in Fig. 2. The figure shows that the institution approach can precisely capture the

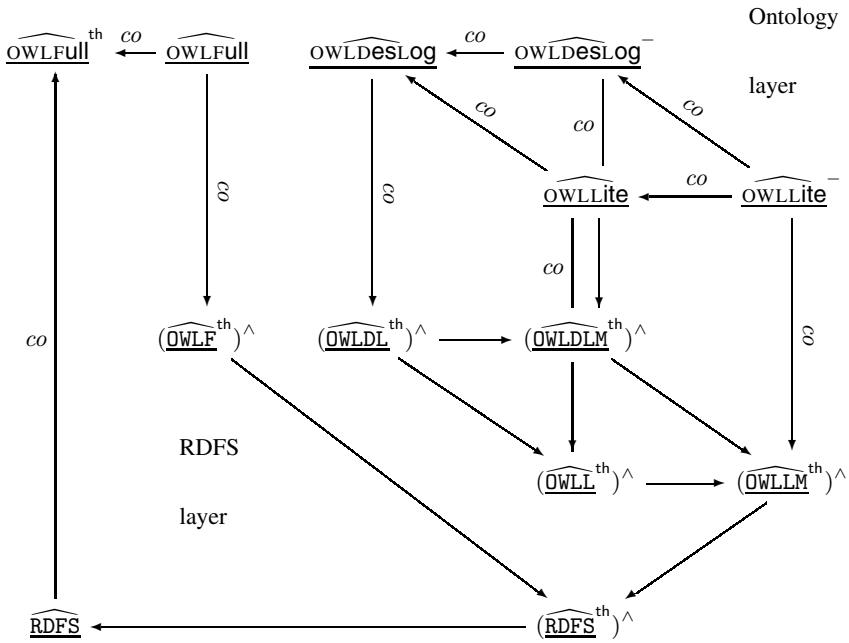


Fig. 2. RDF serialization

relationship among the various languages. The work presented in this paper opens up a new practical application domain for the institutions theory.

One future work direction is to further investigate the relationship of various ontology languages with regard to their respective underlying logical systems. Languages such as the OWL suite (OWL Lite, DL and Full) are based on description logics and they assume an "Open World Assumption". On the other hand, languages such as OWL Flight and WRL are based on logic programming and they assume a "Closed World Assumption". The interoperability of these kinds of languages has been intensively discussed but is still an open question. We believe institution theory can help to clarify this issue by establishing links at the logical level. It is also of interest to investigate the properties of the indexed institution like theory colimits, liberality, exactness, inclusions, and how the design of tools for SW can benefit from these properties.

References

1. J. Angele, H. Boley, J. de Brujin, D. Fensel, P. Hitzler, M. Kifer, R. Krummenacher, H. Lausen, A. Polleres, R.Studer. Web Rule Language (WRL). Version 1.0, 2005. http://www.wsmo.org/wsml/wrl/wrl.html.
2. M. Barr and Ch. Wells. *Category Theory for Computing Science*. Les Publications CRM, Montreal, third edition, 1999
3. T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, May 2001.

4. D. Brickley and R.V. Guha (editors). Resource Description Framework (RDF) Schema Specification 1.0. http://www.w3.org/TR/rdf-schema/, February 2004.
5. J. de Brujin, H. Lausen, and D. Fensel. OWL$^-$. Deliverable D20.1v0.2, WSML, 2004. http://www.wsmo.org/TR/d20/d20.1/v0.2/.
6. R. Diaconescu. *Institution-independent Model Theory*. To appear. http://www.imar.ro/˜diacon/.
7. R. Diaconescu. Grothendieck institutions. *Applied Categorical Structures*, 10:383–402, 2002.
8. J. S. Dong, C. H. Lee, Y. F. Li, and H. Wang. Verifying DAML+OIL and beyond in Z/EVES. In *Proceedings of 26th International Conference on Software Engineering (ICSE'04)*, pages 201–210, Edinburgh, Scotland, May 2004.
9. J. Goguen. Information Integration in Institutions, 2004. To appear in avolume dedicated to Jon Barwise, edited by Larry Moss.
10. J. Goguen and R. Burstall. Institutions: Abstract Model Theory for Specification and Programming. *Journal of the Association for Computing Machinery*, 39(1):95–146, 1992.
11. J. Goguen and G. Roşu. Institution Morphisms. *Formal Aspects of Computing*, 13, pages 274-307, 2002.
12. B.N. Grosof, I. Horrocks, R. Volz, St. Decker. Description Logic Programs: Combining Logic Programs with Description Logic. In *Proc. of the Twelfth International World Wide Web Conference*, pages 48-57, ACM, 2003.
13. P. Hayes. RDF Semantics. http://www.w3.org/TR/rdf-mt/, February 2004
14. I. Horrocks and P. Patel-Schneider. A proposal for an OWL rules language. In Proc. of the Thirteenth International World Wide Web Conference (WWW 2004), ACM, 2004.
15. I. Horrocks, P. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. SWRL: A semantic web rule language combining OWL and RuleML. http://www.daml.org/2004/11/fol/rules-all.html, November 2004.
16. I. Horrocks, P. Patel-Schneider, and F. van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. *J. of Web Semantics*, 1(1):7–26, 2003.
17. G. Klyne, J. Carroll (editors). Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation, 2004. http://www.w3.org/TR/rdf-concepts/.
18. D. Lucanu, Y. F. Li, and J. S. Dong. Web Ontology Verification and Analysis in the Z Framework. Technical Report TR 05-01, University "Alexandru Ioan Cuza" of Iaşi, Romania, January 2005. http://thor.info.uaic.ro/˜tr/tr05-01.ps.
19. B. McBride. Jena: Implementing the RDF Model and Syntax Specification. In 2nd Int'l Semantic Web Workshop, 2001. http://www.hpl.hp.com/personal/bwm/papers/20001221-paper/.
20. P. Patel-Schneider, P. Hayes, and I. Horrocks (editors). OWL Web Ontology Semantics and Abstract Syntax. http://www.w3.org/TR/2004/REC-owl-semantics-20040210/, 2004.
21. P. Patel-Schneider. A proposal for a SWRL extension to forst-order logic. http://www.daml.org/2004/11/fol/proposal, November 2004.
22. P. Patel-Schneider, and D. Fensel, Layering the Semantic Web: Problems and Directions. In *First International Semantic Web Conference (ISWC2002)*, Sardinia, Italy". citeseer.ist.psu.edu/article/patel-schneider02layering.html, 2002.
23. B.C. Pierce. *Basic Category Theory for Computer Science*. MIT, 1991.