First International Semantic Web Conference (ISWC2002),
Sardinia, Italy, June 2002.

Layering the Semantic Web: Problems and Directions

Peter F. Patel-Schneider[1] and Dieter Fensel[2]

[1] Bell Labs Research
Murray Hill, NJ, U.S.A.
E-mail: pfps@research.bell-labs.com

[2] Vrije Universiteit Amsterdam (VU)
Amsterdam, the Netherlands
E-mail: dieter@cs.vu.nl

Abstract. The Resource Description Framework and the Resource Description Framework Schema Specification are supposed to be the foundations of the Semantic Web, in that all other Semantic Web languages are to be layered on top of them. It turns out that such a layering cannot be achieved in a straightforward way. This paper describes the problem with the straightforward layering and lays out several alternative layering possibilities. The benefits and drawbacks of each of these possibilities are presented and analyzed.

1 Introduction

The World Wide Web (WWW) has drastically changed the availability of electronically accessible information. The WWW currently contains some 3 billion static documents, which are accessed by over 300 million users internationally. At the same time, this enormous amount of data has made it increasingly difficult to find, access, present, and maintain the information required by a wide variety of users. This is because information content is presented primarily in natural language. Thus, a wide gap has emerged between the information available for tools aimed at addressing the problems above and the information maintained in human-readable form.

In response to this problem, many new research initiatives and commercial enterprises have been set up to enrich available information with machine-processable semantics. Such support is essential for "bringing the web to its full potential". The Semantic Web activity of the World Wide Web Consortium is chartered to design this future "semantic web"—an extended web of machine-readable information and automated services that extend far beyond current capabilities [3, 9, 10]. The explicit representation of the semantics underlying data, programs, pages, and other web resources, will enable a knowledge-based web that provides a qualitatively new level of service. Automated services will improve in their capacity to assist humans in achieving their goals by "understanding" more of the content on the web, and thus providing more accurate filtering, categorization, and searches of information sources. This process will ultimately lead to an extremely knowledgeable system that features various specialized reasoning services. These services will support us in nearly all aspects of our daily life—making access to information as pervasive, and necessary, as access to electricity is today.

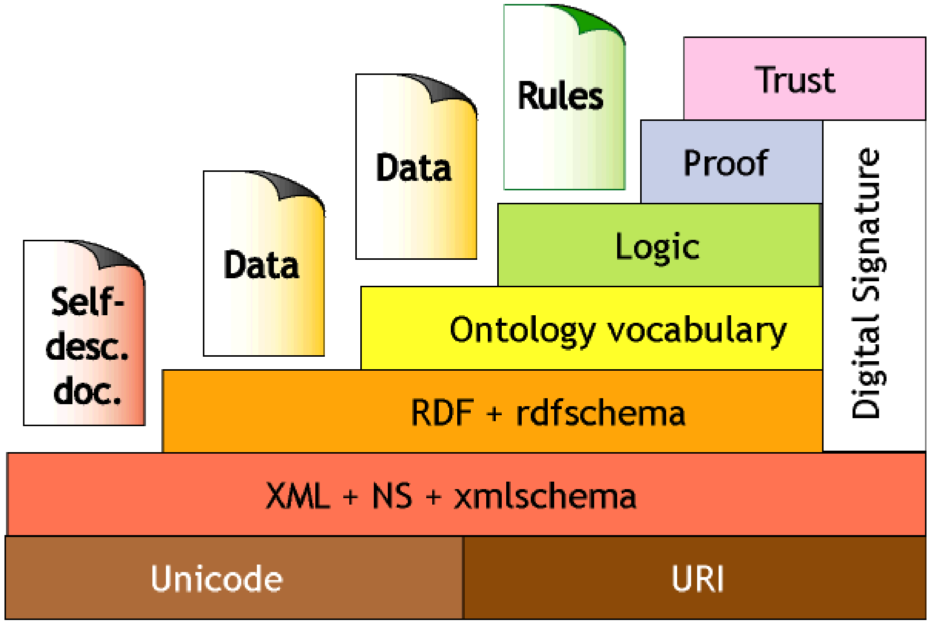

Fig. 1. The Semantic Web Tower

Realizing this vision of the semantic web requires a number of intermediate and related steps. Tim Berners-Lee envisioned a semantic web language tower [http://www.w3c.org/2000/Talks/1206-xml2k-tbl/slide10-0.html] (see Figure 1) that provides intermediate language standards. Based on XML as the universal syntax carrier, a tower of successively more powerful languages are defined. The Resource Description Framework (RDF) [15] and Resource Description Framework Schema Specification (RDFS) [7] define a standard for representing simple metadata—shallow machine-processable semantics of information. The next step is is currently under development by the W3C Web Ontology Working Group (http://www.w3.org/2001/sw/WebOnt/). which is producing a Web Ontology Language, tentatively called OWL. Further extensions will provide rule languages and finally a full-fledged machine processable semantic web.

We are currently working on a small piece of the overall puzzle. This paper reports lessons learned when trying to layer OWL on top of RDF and RDFS.[3] It turns out that OWL cannot be defined as semantic extension of RDF(S) while retaining the syntax of RDF. We will explain the problems we encountered when trying this and we also discuss several alternative workarounds. Each of these have their advantages and drawbacks. The main contribution of the paper is to provide deeper insights on how the language tower of the semantic web can be organized.

Although this paper concentrates on a particular problem, that of layering OWL on top of RDF(S), there are layering decisions to be made whenever a new representation formalism has to be compatible with an existing representation formalism. This

[3] We will use RDF(S) to refer to the combination of RDF and RDFS.

will occur at other points of the semantic web tower, has occured in the past, and will undoubtably occur in the future.

The content of the paper is organized as follows. In Section 2, we describe the context of our problem. We discuss the semantic web language tower as sketched by Tim Berners-Lee. Section 3 sketches four different ways on how to layer OWL on top of RDF. Section 4 discusses one of these solutions in more details. Actually this "solution" is rather a characterization of the problem we encountered in layering the different languages properly. Section 5 discusses three possible solutions. Instead of indicating one solution we elaborate the entire solution space and characterize the alternative in terms of their advantages and draw-backs. Finally, Section 6 provides conclusions.

2 The Context: The Semantic Web Language Tower

Giving a real semantics to the semantic web language tower as sketched by Tim Berners-Lee in Figure 1 requires serious work to clarify many of the present technical issues.

URIs and Unicode provide standard ways to define references to entities and to exchange symbols. XML provides a standard way to represent labeled trees and XML Schema provides a mechanism to define grammars for legal XML documents. Finally, the name space mechanism of XML (NS) allows the combination of documents with heterogeneous vocabulary. These concepts provide the syntactic underpinnings of the semantic web.

The first layer of the semantic web is provided by RDF.

The Resource Description Framework (RDF) "is a foundation for processing metadata; it provides interoperability between applications that exchange machine-understandable information on the Web." [15]. Basically, RDF defines a data model for describing machine processable semantics of data. As stated in the RDF Model and Syntax Specification [15], the basic data model for RDF consists of three object types:

- Resources: A resource may be an entire Web page; a part of a Web page; a whole collection of pages; or an object that is not directly accessible via the Web; e.g. a printed book. Resources are always named by URIs.
- Properties: A property is a specific aspect, characteristic, attribute, or relation used to describe a resource.
- Statements: A specific resource together with a named property plus the value of that property for that resource is an RDF statement.

In a nutshell, RDF defines object-property-value-triples as basic modeling primitives and introduces a standard syntax for them. An RDF document will define properties in terms of the resources to which they apply.

RDF Schema [7] defines a simple modeling language on top of RDF. RDF Schema introduces classes, is-a relationships between classes and properties, and domain and range restrictions for properties as modeling primitives.

RDF and RDF schema (RDF(S) hereafter) use XML as a carrier syntax but do not employ any semantics expressed by XML, instead using their own semantics for XML syntax.

The OIL definition:

```
<rdfs:Class rdf:ID="Herbivore">
  <rdfs:type rdf:resource="oil:DefinedClass">
  <rdfs:subClassOf rdf:resource="Animal" />
  <rdfs:subClassOf>
    <oil:NOT>
      <oil:hasOperand rdf:resource="Carnivore" />
    </oil:NOT>
  </rdfs:subClassOf>
</rdfs:Class>
```

What an RDF Schema system understands:

```
<rdfs:Class rdf:ID="Herbivore">
  <rdfs:type rdf:resource="unknown:DefinedClass">
  <rdfs:subClassOf rdf:resource="Animal" />
  <rdfs:subClassOf>
    <rdf:Description />
  </rdfs:subClassOf>
</rdfs:Class>
```

Fig. 2. OIL versus RDF Schema

The term Ontology vocabulary (and data based on top of it) in the semantic web tower may cause irritation. We guess that the author refers to an ontology language that is a restricted subset of logic to define terminology. Currently, a language called OWL is under development for this purpose. Most of this paper is about on how to define OWL on top of XML and RDF(S). Whereas the relationship between XML and RDF is a simple syntactic one, the relationship between RDF(S) and OWL necessarily has semantic components as well. The details of this relationship involve a rich decision space of proper syntactic and semantic layering.

The semantic web tower figure mentions Logic on top of Ontology layer. On the one hand, this may be misleading as any Ontology language should be properly grounded in logical notions. On the other hand, this may be taken as a hint that on top of an Ontology language a richer logical language should be provided. Again, we may expect complicated discussions on an appropriate layering. For example, proposals for web logic languages may employ a special semantics, such as minimal model semantics, to make inference more amenable to computer implementation.

Proof and trust seem to be rather applications than a new language level. Anyway they are far beyond current efforts.

Currently many layering ideas oriented to syntactical and semantical extensions compete which each other (http://lists.w3.org/Archives/Public/www-webont-wg/) [6, 8]. Working out the proper relationship seems to be much more challenging than just developing one layer. So even before we design the ontology language, we need a vision of how the levels in the semantic web tower relate to each other.

OIL [11, 12], a predecessor of OWL, was defined as a syntactic extension of RDF-Schema which means that every RDF-Schema ontology is a valid ontology in the new language (i.e., an OIL processor will also understand RDF Schema). However, the other direction is also available: defining an OIL extension as closely as possible to RDF Schema allows maximal reuse of existing RDF Schema-based applications and tools. However, since the ontology language usually contains new aspects (and therefore new vocabulary, which an RDF Schema processor does not know), 100% compatibility is not possible. Let us examine an example. The OIL expression in Figure 2 defines herbivore as a class, which is a sub-class of animal and disjunct to all carnivores. An application limited to pure RDF Schema is still able to capture some aspects of this definition. It sees that herbivore is a subclass of animal and a subclass of a second class, which it cannot understand properly. This seems to be a useful way to preserve complicated semantics for simpler applications.

Recently, a interesting feature of this approach become visible. In general one would assume that an OIL/OWL agent can draw more conclusions than an RDF Schema aware agent. This we will call the model theoretic interpretation of the semantic web language tower. However, based on the RDF model theory [13] this turned out to be not true. Because RDF "sees" the syntactical definition of an ontology it can draw conclusions that OWL, which is situated at a logical level, cannot. That is, not every model for an OWL ontology is also a model for the underlying RDF representation. (http://lists.w3.org/Archives/Public/www-webont-wg/2001Dec/0116.html) Therefore, defining the model theory of OWL as an extension of the model theory of RDF and representing OWL constructs syntactically in RDF leads to paradoxical situations, i.e., ill-defined model theories for OWL. We will explain these problems in more detail in the next Sections.

3 Layering in the Semantic Web

What is the relationship between the various layers in this Semantic Web tower? Well this does depend somewhat on which layers are being considered, but there are some principles and some basic kinds of relationships that can be considered.

First, the various layers are languages, so they have a syntax. The syntax of one layer can be an extension of the previous layer, the same as the previous layer, a subset of the previous layer, or just different from that of the previous layer. For example, the syntax of RDF is a subset of the syntax of XML, as RDF uses XML syntax, but not all XML documents are valid RDF documents.

Second, the various layers, or at least most of them, have a semantics or meaning. As the Semantic Web is based on meaning, we should expect that the meanings provided by one layer form the foundation for the next layer. Otherwise, how can this be called a semantic web? So we should expect that one layer semantically extends the previous layer, in that all meanings of the previous layer are maintained by the next layer, and that extra meanings are provided by that layer. For example, this is the relationship between RDF and RDF Schema, where RDF Schema maintains all the meanings of RDF, but adds new meanings of its own.

However, there is a point where the Semantic Web moves out of the semantic realm and into the syntactic realm. At this point we do not expect that the semantics of the layer, if any, are preserved. For example, the foundation of the Semantic Web is Unicode strings and URIs. However, not all of the Unicode strings in an RDF document are strings; instead these strings are given a different meaning in the Semantic Web.

In fact, the Semantic Web treats XML as part of the syntactic realm. The semantics of XML documents are not retained in RDF. Instead RDF provides its own meaning for XML documents; a meaning that is not compatible with the XML data model meaning of the document.

In fact, the only semantic layering currently in the W3C-approved semantic web is the layering between RDF and RDF Schema. RDF Schema uses precisely the syntax of RDF. That is, all RDF documents are syntactically-valid RDF Schema documents. RDF Schema is also a semantic extension of RDF. That is, the RDF Schema meaning of RDF constructs incorporates the RDF meaning of these constructs.

There are at least four proposed layerings of OWL on top of RDF(S).

1. Same-syntax semantic extension: In this proposed layering, which is the same layering relationship as that between RDF and RDF Schema, the syntax of OWL would be the same as the syntax of RDF and the semantics of OWL would be an extension of the semantics of RDF Schema. This looks like the most natural proposal. However, as we will show below, this approach leads to an ill-defined model theory of OWL. Therefore, we describe this solution rather as an enumeration of potential problems than as an actual solution. This “solution” is the point where our analysis departs.
2. Syntax and semantic extension: In this proposed layering the semantics of OWL is defined as an extension of the semantics of RDF Schema. The syntax of OWL is also an extension of the syntax of RDF. In this proposal many syntactical constructs of OWL would not be defined in RDF but instead would use non-RDF XML syntax. This proposal avoids the paradoxical situations of the previous layering proposal. However, new parsers would be required to process the OWL language and an RDF(S) aware agent has a very limited understanding of an OWL ontology.
3. Same-syntax, but diverging semantics: In this proposal layering, OWL syntax would again be RDF syntax, or a subset of RDF syntax, but the meaning of some constructs would be different from their meaning in RDF or RDF Schema. In essence, OWL would treat RDF as a syntax carrier, just as RDF treats XML as a syntax carrier. Actually, most reasonable versions of this approach would not completely discard the RDF and RDF Schema meanings, and considerable overlap is possible. Here an RDF(S) aware agent may understand many aspects of an OWL ontology (as far as they are not beyond his RDF(S) horizon), however, would give some of them a different meaning from his RDF(S) point of view.
4. Differing syntax and semantics: In this proposal layering, OWL differs from RDF and RDF Schema in both syntax and semantics. Again, although the formalisms would diverge, considerable overlap is possible and even desirable.

In the next section we will explain further the first “solution” which is actually not a solution but a way to describe the problem we encountered. In Section 5, we describe the other three solutions in more detail.

4 The Problem When Layering OWL On Top Of RDF(S)

As stated above, the most attractive way to layer OWL on top of RDF(S), at least at first glance, is to use the same layering relationship as that between RDF and RDF Schema. That is, OWL would have the same syntax as RDF and the semantics of OWL would be an extension of the semantics of RDF(S). This layering relationship was the one expected to be used by the designers of RDF(S), at least we so believe based on various statements about RDF(S). However, as we explain in this section, it is just not possible to extend this layering relationship to the ontology level because it leads to semantic paradoxes.

Naïve users may argue that semantic paradoxes are not important in the semantic web. After all, should the semantic web not be able to represent contradictions, and maybe even reason effectively in the presence of contradictions, and semantic paradoxes are just like built-in contradictions? Yes, but the key point is precisely that semantic paradoxes are built-in—they are present in all situations and thus they cause all situations to be ill-defined, resulting in a complete collapse of the logical formalism.

4.1 The Problem in General Terms

The problem with the same-syntax extension of RDF(S) to OWL is roughly the same as the problems that destroyed the initial formalization of set theory. This paradox, discovered by Russell, results from an attempt to make sets too powerful.

In the formalization of set theory that contains Russell's paradox, there is a (very large) collection of built-in sets. All models for sets include these built-in sets, and usually many more. Unfortunately, these built-in sets include the set consisting of those sets that do not contain themselves. Is this set a member of itself? If it is then it cannot be, because it contains exactly those sets that do not contain themselves, and it does contain itself. If it is not then it must be, via similar reasoning.

This violates the very principles of set theory, i.e., that set membership should be a well-defined relationship. Because this set has to be in every model for sets, there are no models for sets, resulting in a complete collapse of this formalization of set theory.

OWL layered on top of RDF Schema as a same-syntax extension has the same problem. We believe that to make the logical foundations of classes in the extension work correctly, there has to be a large collection of built-in classes in any model. Unfortunately, this collection includes the class that is defined as those resources that do not belong to the class. Membership in this class is ill-defined, via reasoning similar to the reasoning for Russell's paradoxical set above. This violates the semantic underpinnings of classes, resulting in no models for OWL defined in this way.

RDF(S) does not fall into this paradox because it does not need a large collection of built-in classes, not having any way to define classes.

4.2 The Problem in Detail

To understand the details of the problem with a same-syntax and extended semantics layering of OWL on top of RDF(S) it is first necessary to understand a bit about the syntax and semantics of RDF(S).

The surface syntax, i.e., the XML syntax, of RDF(S) is being modified by the RDF Core Working Group, but the basic ideas are unchanged. Further, the RDF Core Working Group is developing a syntax of RDF(S) [2] in terms of N-triples [1], a cut-down version of N3. So, for the purposes of this discussion, it suffices to view RDF syntax as a collection of triples in the form of $<$ subject property object $>$, where subject is a URI or an anonymous node ID, object is a URI or anonymous node ID or literal, and property is a URI.

Recently, a semantics for RDF(S) has been defined by the RDF Core WG [13] The model-theoretic semantics defines model-theoretic interpretations, how these interpretations provide meaning for RDF(S), and how one RDF(S) knowledge base can entail another. The details of interpretations cannot be given in a paper of this length, but the general ideas are fairly standard. Basically, interpretations are built on resources, i.e., referenceable objects, literal values, e.g., strings, and binary relationships between resources or from resources to literal values. Both URIs and anonymous node IDs denote resources, and literals denote literal values, e.g., strings. Triples denote relationships between resources or from resources to literal values that belong to a property. The only unusual part of this model theory is that properties are also resources. From interpretations the usual idea of models, i.e., interpretations that satisfy a KB, follow, as does entailment, i.e., all models of one KB are also models of another.

RDF itself has a simple built-in theory having to do with the types of resources. It provides a property between resources, rdf:type, that links a resource to the types that the resource belongs to. As all properties are resources in RDF, so rdf:type is a resource. The extra meaning of rdf:type accruing from its status as a relationship between resources and their types is formally defined in the model theory for RDF.

RDF Schema extends this theory by creating a theory of classes and properties. Classes in RDF Schema are those resources that can have members, i.e., RDF Schema classes are resources that can be the object of triples whose predicate is rdf:type. RDF Schema defines several built-in classes, including rdfs:Class, the class of all classes, and rdfs:Resource, the class of all resources. RDF Schema also defines several relationships between classes, including rdfs:subClassOf, the subclass relationship. All these resources are given meaning by the RDF(S) model theory.

However, the RDF Schema theory of classes and properties is very weak. For example, it is not possible in RDF Schema to provide defined classes—classes that give a formula that determines which resources belong to them. The intent of OWL is to provide an even richer theory of classes and properties, allowing for defined classes and more relationships between classes. Some of these defined classes are called restrictions in OWL. There are restrictions in OWL that provide local typing for properties and restrictions in OWL that ensure that there are several values (or at most several values) for a property that belong to a particular class. There is also a construct in OWL that creates a class from a set of resources.

It is this richer theory of classes that clashes with the underlying principles of RDF(S), resulting in paradoxes in a same-syntax and extended semantics layering of OWL on top of RDF(S). Let's now investigate how these semantic paradoxes arise.

As this layering of OWL on top of RDF(S) is a same-syntax layering, all OWL syntax is also RDF syntax, which is also the same as the RDF Schema syntax. Therefore every syntactic construct has to be either a URI, an anonymous node ID, a literal, or a triple. As the semantics of OWL in this layering is an extension of the semantics of RDF(S) the denotation of these constructs has to be the same as their denotation in RDF(S) and the semantic constraints on them have to include the semantic constraints on them in RDF Schema. Further, as OWL classes are an extension of RDF Schema classes, the OWL relationship from resources to their OWL classes must incorporate the RDF Schema relationship between resources and RDF Schema classes, namely rdf:type, and OWL classes must include RDF Schema classes. Let us call the property that is the OWL relationship from resources to their types, owl:type, which can either be rdf:type or some super-property of rdf:type.

Now consider entailment in this version of OWL. Suppose we have an OWL interpretation that contains an RDF Schema class, `Person`, a property `child`, and an object `John` that has no outgoing child relationships. OWL contains the notion of a restriction of a property to a type, i.e., given a property, say `child`, and a class, say `Person`, it is possible to create the restriction corresponding to those resources whose `children` all belong to `Person`. `John` belongs to this restriction in this interpretation because `John` has no `children` and thus all `John`'s `children` belong to `Person`. Therefore we need that any interpretation like the one above is a model for `John` belonging to this restriction. However, restrictions are resources and thus this can only happen if there is a resource that corresponds to the restriction, and that includes `John` as a member. So, this simple interpretation will not be correct, in our view, unless it includes such a resource and the appropriate `owl:type` relationships to it.

Thus OWL interpretations must include resources for many restrictions, essentially all the restrictions that can be built out of the classes and properties that are in the interpretation. As well, OWL interpretations must have the correct `owl:type` relationships to these resources. In this way, each OWL interpretation must have a theory of restrictions, including self-referential restrictions, and also other OWL constructs. Some of these restrictions can refer to themselves, as there are self-referential loops in RDF Schema classes and thus this cannot be ruled out in restrictions.

We are now in the same situation that the original formalization of set theory was, as the following shows.

Consider the restriction that states it is precisely those resources that have at most zero values for the property owl:type that belong to the class that consists of the restriction itself. In the N-triples syntax for RDF, this is

```
_:1 a owl:Restriction .
_:1 owl:onProperty owl:type .
_:1 owl:maxCardinalityQ 0 .
_:1 owl:hasClassQ _:2 .
_:2 oneOf _:3 .
_:3 owl:first _:1 .
_:3 owl:rest owl:nil .
```

This restriction, read slightly differently, is the restriction that consists of those resources that do not belong to it. This is not the paradoxical Russell set, but is paradoxical. Consider any resource. If it belongs to the restriction then it does not, but if it does not then it does. Just as with the Russell set, if this restriction is in an interpretation then the class membership relationship is ill-defined. However, this restriction is in all OWL interpretations, because it is constructed only from resources that must be in all OWL interpretations. Therefore all OWL interpretations have an ill-defined class membership relationship and thus this layering of OWL is paradoxical.

5 Solutions For Layering OWL On Top of RDF(S)

Given that the most attractive layering solution leads to semantic paradoxes, what can be done? One approach would be to change RDF or RDF Schema in some way, perhaps by removing some of the parts of RDF Schema that participate in the paradoxes. However, if we want to keep all of RDF and RDF Schema, it is necessary to pick one of the other solutions. The intent of this paper, however, is not to actually do the picking. Instead, this paper is concerned with laying out the benefits and drawbacks of the various solutions so that an informed decision can be made between them.

5.1 Limiting Entailment

One way of keeping the same-syntax extension relationship between RDF Schema and OWL would be to simply give up on the entailments that require the presence of the problematic restrictions in all interpretations. This would, however, result in some rather strange consequences.

For example, if `John` belonged to `Student` and `Employee` it would not follow that `John` belonged to the intersection of `Student` and `Employee`, because this intersection would not necessarily exist in all the required interpretations.

5.2 Syntactic and Semantic Extension

In the syntactic and semantic extension layering proposal, OWL defines new syntactic constructs that do not exist in RDF or RDF Schema. However, the syntactic constructs of RDF are all valid syntactic constructs of OWL, and have the same meaning in OWL as they had in RDF and RDF Schema, or an extension of that meaning. (Even the RDF constructs that are not addressed by OWL, such as reification, remain as valid OWL syntax.) The new syntactic constructs of OWL are given meanings that are compatible with the RDF and RDF Schema meaning of related RDF constructs.

The natural way of defining OWL in this layering proposal is to make the restrictions of OWL be new syntax. These restrictions would have a separate meaning defined by the OWL model theory. For example, a possible syntax for a maximum cardinality restriction like the one above could be:

```
<owl:cardinality maximum=''0'' property=''friend''>
Person
</owl:cardinality>
```

This syntactic construct would be given its own meaning by the OWL model theory, which would not include its presence as a resource in interpretations. In this way the model theory of OWL would not be subject to the semantic paradox above.

This relationship between RDF(S) and OWL would be the same as the relationship between propositional and modal logics. There are many other examples of this sort of layering between logical and knowledge representation formalisms.

In this proposal for OWL there would still be considerable overlap between RDF and RDF Schema, on one hand, and OWL on the other. OWL would still have classes and properties, and would have all the RDF Schema classes, like rdfs:Class and rdfs:-Resource. It is just that restrictions would not be classes. An OWL system would be able to process RDF(S) documents, and would give them a meaning that was compatible with the meaning given to them by an RDF(S) system. An OWL document would generally include three portions: an RDF(S) portion that set up base facts and typing relationships, such as John's friend is Mary and John is a Person; an RDF(S) portion that creates classes and gives some relationships between them, such as Student and Person are classes and Student is an rdfs:subClassOf Person; and an OWL-only portion that gives meaning to some of the classes, such as Student is defined to be those Persons who are enrolled in a School.

An OWL document would thus not be completely parseable by RDF(S) parsers. However, many OWL documents would include significant RDF(S) content and an RDF(S) system that was prepared to ignore the OWL constructs could still extract considerable information from an OWL document. For example, an RDF(S) system would have access to all base-level facts, the classes, and their subclass relationships, which would all still be in RDF(S) form.

In this proposed layering, the Semantic Web tower would be considered as a tower of more-and-more powerful logical languages, all sharing a common semantic core. Higher languages would be have more syntactic constructs and would extend the meaning for constructs from lower languages, but would be compatible with the meanings that come from the lower languages. Systems built for the lower languages could be considered sound but incomplete reasoners for the higher languages, only on the syntax of the lower language if they did not allow for unknown constructs, but if they allowed for unknown constructs they would be incomplete reasoners for the higher languages.

Nevertheless, this layering view has some problems. RDF(S) is not just about base facts, but also includes a theory of classes and properties. These classes and properties are resources and thus can participate in base-level facts. This is not a problem in RDF(S) because of its limited expressive power, but can cause problems in more-expressive languages, like OWL. For example, OWL could end up having to deal with transitive properties that are transitive only because they are the value of a property in some resource. Even worse, this transitivity could be conditional because the property might or might not be the value, depending on some other information.

For example, if John is a Person whose friends are all transitive properties, and John has a friend that is either bar or baz, then either bar or baz is a transitive property. This conditional transitivity can give rise to extraordinarily-complex and difficult patterns of reasoning.

5.3 Same Syntax, but Diverging Semantics

In the same-syntax but diverging semantics layering proposal OWL has the same syntax (or maybe a subset of RDF syntax) but does not abide by (all of) the RDF(S) meaning. This may seem rather strange at first glance. After all, shouldn't the Semantic Web retain meaning from lower languages. However, the Semantic Web does not do this at the lowest levels, as it ignores the meaning provided by XML in favor of a different RDF meaning for documents.

One way to rationalize this form is layering is to view RDF(S) as a means for reasoning about the syntax of an ontology. RDF would make distinctions based on the form of the ontology constructs, for example the order of conjuncts in a conjunction, or whether two classes are explicitly stated to be disjoint as opposed to having disjointness inferred from their properties. On the other hand, OWL would be solely interested with the logical consequences of ontology constructs and would not have any access to their form. In this view RDF would not be viewed as a basic ontology language but instead as a syntactic mechanism for defining ontology languages.

In support of this view, RDF(S) has many strange features considered as an ontology language, including reification, syntactic containers, and the heory of RDF Schema classes and properties. These features may only make sense if you use RDF(S) as a mechanism for defining ontology languages.

For example, it may make sense to distinguish at some level between the definition of

- a relation r as an attribute of a class c versus
- as a global property r with c as its domain.

Logically these two are the same but from a modeling point of view they are quite different. By having two types of entailments we can capture this without running into any problems. With syntactical RDF(S) reasoning we can ask for different syntactical styles of an ontology and with semantical OWL reasoning we infer logical consequences of an ontology.

This layering relationship allows us to deal with different modeling paradigms. We would define a frame syntax for OWL in RDF(S), making sure that it behaves the same as the non-frame version at the logical level but behaves different at the syntactical level, i.e., in the frame version you could ask whether something is explicitly defined as an attribute or as a property. This layer allows us to capture, infer, and query modeling information, as opposed to logical information.

So, although this layering proposal goes against the logical-extension view of the Semantic Web, it does have its own benefits.

5.4 Diverging Syntax and Diverging Semantics

One problem with the above approach is that RDF is a terrible syntax for complex constructs. As everything is RDF is a triple, if OWL has the same syntax as RDF, all OWL syntax has to be encoded as one or more triples.

This is not a severe problem if an OWL syntactical construct can indeed be encoded as one triple. However, most OWL syntactical constructs are more complicated. For example, the cardinality constructs used above have four components, and have to be encoded as four triples. Encoding a syntactical construct as more than one RDF(S) triple results in several severe problems.

First, the triples are not connected. An RDF(S) document could be missing some of the triples. If so, what syntactical construct is being encoded? For example, a cardinality restriction might be missing the property. Second, there is no way in RDF(S) to require that only the appropriate triples are present. For example, a cardinality restriction might have extra, random triples attached to it, such as saying that a cardinality restriction has a friend. Even worse, a cardinality restriction might have two properties or two numbers. What is being said here? So it might be useful to diverge from the RDF(S) semantics even when extended the syntax, because adhering to the semantics causes computational difficulties.

What then remains of the Semantic Web tower? Well, one could say that RDF(S) really should have been just about triples, and that the class and property theory embedded in RDF Schema is not useful. This would then result in an extension of this base-triple portion of RDF, and not a total breakdown of the Semantic Web tower.

In this layering view, OWL takes the useful portion of RDF syntax and semantics, and replaces the rest (both syntax and semantics) with a syntax and semantics that works for the ontology layer.

6 Conclusions

The biggest problem with the layering of OWL on top of RDF(S) comes not from any particular aspect of RDF(S) itself but instead from a vision of the developers and users of RDF(S). In this vision RDF encompasses the whole of the semantic web—all information in the semantic web is written in RDF syntax and the basic meaning of all this information is provided by RDF. The higher layers of the semantic web serve to provide extra meaning for the information that goes beyond the scope of RDF, but they are not free to either extend the syntax of RDF nor are they free to modify the basic meaning provided by RDF.

RDFS is an example of such an extension. RDFS uses the same syntax as RDF and extends the meaning of some constructs but retains their RDF meaning. However, RDFS is a very small extension of RDF. More significant extensions to RDF(S) require either an extension to the syntax of RDF or some deviation from the semantics of RDF(S). Perhaps the main lesson to be learned here is that a representation formalism should not claim the entire universe, always leaving open the possibility that extensions to it will need their own ways of stating information or their own ways of providing meaning for the information they state.

Figuring out the proper principles for building up the semantic web language tower will require more work and recalls earlier approaches in knowledge representation [5, 4] and knowledge modeling [14]. This work should prevent the language tower of the semantic web from ending up like the famous tower of Babel.

In this paper we took the first steps into the direction of a blue print for the semantic web language tower. We were focusing on how to build the fundament for OWL, which is the first floor of the ontology-enabled web. We explained some problems that occur when naïvely building OWL on top of RDF and we described three possible strategies to overcome the problem. Instead of prefering one solution we describe the solution space with various choices and their pros and cons. It will be up to the Web Ontology (WebOnt) Working Group of the Semantic Web Activity of the W3C, the recommendation organization of the World Wide Web, to determine the actual strategy to use in OWL.

Acknowledgement

This work was done in the context of the Web Ontology (WebOnt) Working Group (http://www.w3c.org/2001/sw/WebOnt) of the W3C.

References

1. Art Barstow and Dave Beckett. RDF test cases. W3C Working Draft, 15 November 2001, http://www.w3.org/TR/rdf-testcases/.
2. Dave Beckett. RDF/XML syntax specification (revised). W3C Working Draft, 18 December 2001, http://www.w3.org/TR/rdf-syntax-grammar/.
3. Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, May 2001.
4. R. J. Brachman. The myth of the one true logic. *Computer Intelligence*, 3(3), August 1987.
5. Ronald J. Brachman. What ISA is and isn't: An analysis of taxonomic links in semantic networks. *IEEE Computer*, 16(10):30–36, October 1983.
6. J. Broekstra, M. Klein, S. Decker, D. Fensel, F. van Harmelen, and I. Horrocks. Enabling knowledge representation on the web by extending RDF schema. *Electronic Transactions on Artificial Intelligence (ETAI)*, to appear.
7. Resource description framework (RDF) schema specification 1.0. W3C Candidate Recommendation, 27 March 2000, http://www.w3.org/TR/rdf-schema, March 2000.
8. Stefan Decker, Frank van Harmelen, J. Broekstra, M. Erdmann, Dieter Fensel, Ian Horrocks, M. Klein, and S. Melnik. The semantic web - on the respective roles of XML and RDF. *IEEE Internet Computing*, 2000.
9. D. Fensel and M. Musen. Special issue on semantic web technology. *IEEE Intelligent Systems (IEEE IS)*, 16(2), 2001.
10. Dieter Fensel, James Hendler, Henry Lieberman, and W. Wahlster, editors. *Semantic Web Technology*. MIT Press, Boston, 2002.
11. Dieter Fensel, Ian Horrocks, Frank van Harmelen, Stefan Decker, M. Erdmann, , and M. Klein. OIL in a nutshell. In R. Dieng et al., editor, *Knowledge Acquisition, Modeling, and Management, Proceedings of the European Knowledge Acquisition Conference (EKAW-2000)*, Lecture Notes in Artificial Intelligence, LNAI 1937. Springer-Verlag, October 2000.
12. Dieter Fensel, Ian Horrocks, Frank van Harmelen, Deborah L. McGuinness, and Peter F. Patel-Schneider. OIL: An ontology infrastructure for the semantic web. *IEEE Intellligent Systems*, 16(2), May 2001.
13. Patrick Hayes. RDF model theory. W3C Working Draft, http://www.w3.org/TR/rdf-mt/, 2001.
14. Allen Newell. The knowledge level. *AI Magazine*, 2(2):1–20, Summer 1981. The Presidential Address, AAAI-80, Stanford, California.
15. Resource description framework (RDF): Model and syntax specification. W3C Recommendation, 22 February 1999, http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/, February 1999.