XSDL: Making XML Semantics Explicit*

Shengping Liu, Jing Mei, Anbu Yue, and Zuoquan Lin

Department of Information Science, Peking University,
Beijing 100871, China
{lsp, mayyam, yueanbu, lz}@is.pku.edu.cn

Abstract. The problem that "XML formally governs syntax only - not semantics" has been a serious barrier for XML-based data integration and the extension of current Web to Semantic Web. To address this problem, we propose the XML Semantics Definition Language(XSDL) to express XML author's intended meaning and propose a model-theoretic semantics for XML. Consequently, XML becomes a sub-language of RDF in expressiveness and XML data can be semantics-preserving transformed into RDF data. We further discuss the semantic entailment and validity of the XML documents.

1 Introduction

XML[1] has achieved great success as standard document format for writing and exchanging information on the Web. However, one of the limitations of XML has been well recognized: "XML formally governs syntax only - not semantics"[2]. The tags in an XML document are only meaningful to human, but meaningless to machine. For example, humans can predict the information underlying between the tags in the case of <price></price>, but for any generic XML processor, the tag <price> is equal to the HTML tag <H1>, because nowhere in an XML document, or DTD and XML Schema, does it say what these tags mean. Therefore, XML cannot express formal semantics by itself. Nonetheless, there are implicitly semantic information lied in the tags and structure of an XML document. For example[1],

Example 1. An XML fragment with implicit semantics

```
<wineMerchant name="Bristol Bottlers" >
    <wine id="w100">
        <name>Vielles Bottes</name>
        <color>black</color>
    </wine>
</wineMerchant>
```

* This work was supported by the National Natural Science Foundation of China under grant numbers 60373002 and 60496323.

[1] This XML fragment is modified from examples in SWAD-Europe Deliverable 5.1: http://www.w3.org/2001/sw/Europe/reports/xml_schema_tools_techniques_report.

The above XML fragment expresses rich semantic information: there is a wine merchant called “Bristol Bottlers” who sells a kind of wine whose name is “Vielles Bottes” and the color is black. The facts and relationship represented by the XML document is called XML Semantics[3].

In fact, the XML semantics is implicitly expressed in the XML documents. The semantics is conveyed on the basis of a shared understanding derived from human consensus. If there is an implicitly shared consensus about what the tags mean, then people can hardcode this implicit semantics into applications. The disadvantage of implicit semantics is that they are rife with ambiguity[4]. People often disagree with the meaning of a term. For example, prices come in different currencies and they may or may not include various taxes. The hardcoding of the XML semantics into applications make the interoperation and integration difficult.

Moreover, due to the implicit XML semantics, XML is not suitable to represent the content in the next generation of Web, Semantic Web[5]. The metadata language RDF[6] with a formal semantics was proposed as the standard to fulfil the task[7], and the techniques from knowledge representation field, such as the Web ontology language OWL[8], was introduced to represent the domain knowledge. Consequently, the semantic discontinuity between XML and RDF is formed; most XML data on the current Web cannot be smoothly transformed to the Semantic Web. This is a serious barrier for one of the statements by Semantic Web: being an extension of the current Web, and is also a barrier for the wide acceptance of Semantic Web in industry.

Therefore, for broad applications of XML and developments of Semantic Web, the XML semantics is required to be formally and explicitly specified. To this end, P. Patel-Schneider and J. Siméon proposed the XML model-theoretic semantics[9] and later the Yin/Yang Web[10], in which the data model of XML and RDF are unified and the XML document is given a direct RDF-compatible meaning. However, because there is no specification for expressing the XML semantics, hence the author of XML document can express the same meaning in almost arbitrary ways and the intended meaning of author is hidden in the tags and structure of the XML document without any formal description. Therefore, without the author’s intervention, the direct interpretation for XML is difficult to capture the author’s intended meaning and thus decrease the Yin/Yang Web’s value in practical applications. For example, in the interpretation of XML in the Yin/Yang Web, an element node is mapped to an individual of the class with the same name. But, in fact, an element node in XML may represent an individual, a property and even a literal. Sometimes even worse, a node can have different meaning under different conditions. Thus, it is nearly impossible to capture the author’s intended meaning by only syntactic analysis and a language to specify XML semantics by author is required. MDL(Meaning Definition Language,[11]) is such a language that defines the XML semantics in terms of UML class model and defines how to extract the meaning in terms of XPath[12]. The main disadvantage of MDL is that it has no formal semantics and thus provides no much help to bridge the gap between XML and Semantic Web.

Motivated by the Yin/Yang Web and MDL, we propose a novel approach. First, we propose the XML Semantics Definition Language (XSDL)[2], in which the XML semantics is defined in terms of OWL DL ontology and is extracted in terms of XPath (namely XPath 2.0) path expression. The XML authors can use XSDL to define the intended meaning of an XML document. Second, we propose the XML model-theoretic semantics that gives XML meaning by two steps: we firstly define the XML's simple interpretation that gives a rough meaning of an XML document, for example, an element node is interpreted as an individual in the universe; then we define the XML's XSDL-interpretation that gives the exact meaning of the XML document by taking the XML's XSDL definition into account, for example, the individual is further interpreted as an instance of some class according to the XSDL definition.

After XML document having XSDL to define the semantics, XML can express OWL DL's fact assertions, i.e., the statement about a particular individual in the form of classes that the individual belongs to plus properties and values of that individual[17]. Therefore, XML can be viewed as a knowledge representation language which is less expressive than RDF. Furthermore, we introduce the semantic validity of XML document to check whether the document satisfies the semantic integrity constraints and show that this problem is equivalent to the satisfiability of the knowledge base in the description logic language $\mathcal{SHOIN}(\mathcal{D})$. In addition, we discuss one important reasoning task about XML: the semantic entailment between XML documents, and show that the entailment problem can be reduced to the same satisfiability problem of $\mathcal{SHOIN}(\mathcal{D})$ [13].

The paper is organized as follows. We describe the syntax of XSDL with some examples in Section 2. The XML model-theoretic semantics that give XML meaning by simple interpretation and XSDL-interpretation is presented in Section 3. In Section 4, we discuss some related works. Finally, we conclude this paper in the concluding section.

2 XML Semantics Definition Language(XSDL)

The XML semantics is implicitly expressed in almost arbitrary ways, so it is difficult to extract the semantic information in a purely syntactic way. A language is required to define the semantics by human. The language should at least include two parts: a formal language to represent the semantic information in XML and a mapping language to specify the mapping from XML constructs to the formal language. In XSDL, OWL DL is selected as the formal language because it is the standard Web ontology language and have a formal logic foundation; the mapping language is based on Schema Adjuncts Framework(SAF, [14]), which extends the XML's structural model given by XML schema with additional information about the meaning of XML instances. In SAF, information items are selected by means of XPath path expressions; the additional information is given

[2] In the XML Schema specification, a term "XML Schema definition language" is used, but "XSDL" is not proposed as the term's acronym by W3C.

by reference to an external schema. XSDL is the SAF implementation with the external schema as OWL DL ontology's XML presentation syntax[15].

Now we briefly introduce the XML syntax and abstract syntax of XSDL. For more detailed information about XSDL, refer to XSDL specification[25].

The XSDL document structure is as follows:

```
<schema-adjunct target="http://foo.org/myschema.xsd"
    xmlns:owlx="http://www.w3.org/2003/05/owl-xml">
 <document>
   <!-- global ontology definition: any legal syntax of OWL DL-->
   <owlx:Ontology owlx:name="http://foo.org/wine">
   ...
   </owlx:Ontology>
 </document>
 <!-- mapping rules definitions: mapping XML constructs to
      the global ontology -->
 <element context ="/wineMerchant">
   <owlx:Class owlx:name="WineMerchant" />
 <!--or:DataValue,Individual,ObjectProperty,DatatypeProperty-->
 </element>
 ...
 <attribute context ="/wineMerchant/wine/name">
   <owlx:DatatypeProperty owlx:name="name" />
    <!--or:ObjectProperty-->
 </attribute>
 ...
</schema-adjunct>
```

where the "target" attribute value is the XML Schema for which XSDL defines the semantics. Semantic information are given at all levels: document, element and attribute: the "document" node includes a global ontology definition; in "element" and "attribute" nodes, the "context" attribute selects the instance data by XPath 2.0 path expression, the child elements can be references to the individuals, classes, datatype properties and object properties defined in the global ontology. The global ontology is sometimes called the *ontology in XSDL*.

XSDL is defined at schema level and the XSDL definition can be applied to all XML documents conforming to the schema. But to be intuitive, the following examples are XML fragments when introducing the XSDL definitions.

2.1 Class Definition

In XML, individual is always denoted by XML element node, and then class is denoted by a set of element nodes. To define class in XSDL, we need an XPath path expression to select the set of nodes, and a reference to the class name in the global ontology. In addition, because URI reference is used to identify resources in Semantic Web, so we need a URI constructor to assign URI references to the resources mapped from XML nodes.

In Example 1, the set of wine nodes represent a *Wine* Class, every instance in the class have an URIref like "http://foo.org/wine#w100", this can be defined in XSDL as:

```
<element context="/wineMerchant/wine">
    <URIFunction>concat("http://foo.org/wine#",
        string("/wineMerchant/wine[$i]/@id"))</URIFunction>
    <owlx:Class owlx:name="Wine"/>
</element>
```

where the "context" attribute select the "wine" nodes; the "URIFunction" element is an XPath 2.0 function for URI construction, the parameter $\$i$ denote the i^{th} node in the set selected by XPath expression, *string*, *concat* are both XPath built-in functions, other functions, such as *document-uri*, *namespace-uri*, can also be used to construct the URI; the "owlx:Class" node refers to a class that has been already defined in the global ontology.

Note that the use of URI function partially solves one of the limitations of OWL DL: datatype property cannot be inverseFunctional, so if ID-typed attribute is mapped to datatype property, it cannot identify the individual. Now we can construct URIref through the ID-typed attribute, and use URIref to identify individual. If the URI function is not given, the nodes will be interpreted as anonymous individuals.

The abstract syntax for class definition is:

<CtxPath^^element, urifn, cn^^Class>,

where "CtxPath" is the context path, "^^element" means the type of nodes return by context path are element nodes, "urifn" is the URI constructor function and "cn" is the class name. "^^Class" means that "cn" is a name of class.

2.2 Individual Definition

Sometimes we need to make assertions about individual or define enumerated class in the global ontology. Then we need to define individual in XSDL. In Example 1, one specific wine node represents an individual of class *Wine*, the syntax is:

```
<element context ="/wineMerchant/wine[@id='w100']">
    <owlx:Individual owlx:name="w100" />
</element>
```

Note that all nodes selected by context path are interpreted as the same individual, whereas every node is interpreted as different individual of the same class in above class definition.

The abstract syntax for individual definition is:

<CtxPath^^element, uri^^Individual>,

where "uri" is the individual's name or URIref.

2.3 Literal Definition

In XSDL, the values of attribute and text nodes are predefined as literal values. However, sometimes element node that has no attribute may also represent a literal. For example:

Example 2. An XML fragment about literal definition

```
<Mathematics>
   <student name="John" grade="87" id="100" />
</Mathematics>
```

where the "Mathematics" node can be viewed as a literal "maths" and be the value of "courseName" attribute of student, the equivalent XML fragment is:

```
<student name="John" grade="87" id="100" courseName="maths"/>
```

This can be defined in XSDL as:

```
<element context="/Mathematics">
   <owlx:DataValue  owlx:datatype="&xsd;string">
   maths</owlx:DataValue>
</element>
```

The asbtract syntax for literal definition is:
<CtxPath^^element, literal^^ddd>,
where "ddd" is literal's data type.

2.4 Datatype Property Definition

In XML, attribute nodes and some element nodes with PCDATA type always represent datatype properties. To define this in XSDL, we need to further define the path attribute of "domainContext" and "rangeContext", which are the relative paths related to context path and define the way to extract the node pairs of the property.

In Example 1, node "id" and "name" represent datatype properties of class $Wine$, the syntax in XSDL is

```
<attribute context ="/wineMerchant/wine/@id">
    <domainContext path=".." />
    <rangeContext path="." />
    <owlx:DatatypeProperty owlx:name="wineID" />
</attribute>
<element context ="/wineMerchant/wine/name">
    <domainContext path=".." />
    <rangeContext path="text()" />
    <owlx:DatatypeProperty owlx:name="wineName" />
</element>
```

Note that the property $wineID$ and $wineName$ should have been defined in the global ontology. Because the range of datatype property must be literal values, so the node in the path of range context should be attribute node, text node or element node defined as literal. In addition, the context path is just a convenient way to locate the nodes that interpreted as individuals or literal values in the domain and range of the property. For the $wineName$ definition, the context path can also be "/wineMerchant/wine", then the path of domain context should be "." and the path of range context should be "name/text()".

In the Yin/Yang Web, there must be a property between element node and its child attribute nodes, by contrast, XSDL provides a way to define the datatype property on any node pairs not limited by the document order. For example, in Example 2, we can relate the "student" node to its parent "Mathematics" node by a $courseName$ property, the syntax is:

```
<element context ="/Mathematics/student">
    <domainContext path="." />
    <rangeContext path=".." />
    <owlx:DatatypeProperty owlx:name="courseName" />
</element>
```

Sometimes the document order in XML has significant meaning, for example, in the individual normal form for XML representations of structured data[3],

Example 3. An XML fragment extracted from the individual normal form

```
<Address>
       <string>US</string>
       <string>Alice Smith</string>
       <string>123 Maple Street</string>
</Address>
```

The "string" nodes have different meaning at different positions. Fortunately, XPath expressions can select nodes by position, for example, the first "string" node represents a *country* property, this can be defined in XSDL as:

```
<element context ="/Address/string[position()=1]">
    <domainContext path=".." />
    <rangeContext path="text()" />
    <owlx:DatatypeProperty owlx:name="country" />
</element>
```

As can be seen from the above example, XPath expression bridges the gap between ordered XML document and unordered semantic representation.

The abstract syntax for datatype property definition is:

<CtxPath^^nodeType, DPath, RPath,dpn^^DatatypeProperty >,

where "nodeType" can be "element" and "attribute" ,"DPath","RPath" are the "path" attribute of "domainContext" and "rangeContext" respectively, "dpn" is the name of datatype property.

[3] Henry S. Thompson, http://www.ltg.ed.ac.uk/~ht/normalForms.html

2.5 Object Property Definition

In XML, the nesting of elements always represent object property. In Example 1, the nesting of "wineMerchant" and "wine" nodes represent a "sell" object property. The definition is similar to datatype property definition, the syntax is:

```
<element context ="/wineMerchant">
    <domainContext path="." />
    <rangeContext path="wine" />
    <owlx:ObjectProperty owlx:name="sell" />
</element>
```

In addition, object property may be represented by reference in XML. The explicit referencing mechanism uses the ID/IDREF attribute combination, for example,

Example 4. An XML fragment with explicit reference by ID/IDREF

```
<wine id="w1001" name="Vielles Bottes" color="black" />
<wineMerchant name="Bristol Bottlers"  wineID="w1001" />
```

where "wineID" node is an IDREF-typed attribute node and refers to an ID-typed "id" attribute in a wine node. This ID/IDREF combination establishes a relationship between the "wine" individual and the "wineMerchant" individual, although XML does not say what is the relationship.

However, there are also implicit references by shared value in XML, for example[4],

Example 5. An XML fragment with implicit reference by shared value

```
<student name = "James Smith">
  <course>101</course>
</student>
<department name = "Mathematical Sciences">
  <courses>
     <course code = "101" name = "basic algebra"/>
  </courses>
</department>
```

The "attend" relationship between student and course is represented by the sharing of a value (the course code 101) between node < course>101</course> and node <course code="101" />.

To define object property by reference, we need further define the "IDPath" that is always the path of ID-typed nodes or the nodes implicitly referred outside and the "IDREFPath" that is always the path of IDREF-typed nodes or the nodes implicitly referring to other nodes by shared value. For implementation convenience, the "IDREFPath" should be relative XPath expression with respect

[4] This example is modified from an example in SWAD-Europe Deliverable WP5.2 : http://www.w3.org/2001/sw/Europe/reports/xslt_schematron_tool/".

to the context path, the “IDPath” should be absolute XPath expression and the “path” attribute of “rangeContext” should be relative XPath expression with respect to the “IDPath”. The syntax for Example 4 is:

```
<attribute context="/wineMerchant/@wineID">
    <domainContext path=".." />
    <rangeContext path=".." IDPath="/wine/@id"  IDREFPath="."/>
    <owlx:ObjectProperty owlx:name="sell" />
</attribute>
```

where the path attribute in domain context selects the “wineMerchant” nodes and the path attribute in range context selects the corresponding “wine” nodes.

The syntax for Example 5 is similar:

```
<element context="/student/course/text()">
    <domainContext path="../.." />
    <rangeContext path=".."
     IDPath="/department/courses/course/@code" IDREFPath="."/>
    <owlx:ObjectProperty owlx:name="attend" />
</element>
```

The abstract syntax for object property definition is:
<CtxPath^^nodeType, DPath, RPath, IDPath, IDREFPath, opn^^ObjectProperty>,
where “opn” is the name of object property.

3 XML Model-Theoretic Semantics

After XML document having XSDL to specify the semantics, XML documents not only carry the data, but also the data semantics. To make the semantics machine understandable, we now define XML’s model-theoretic semantics. First, we define XML’s simple interpretation; second, we define XML’s XSDL-interpretation that is an extension on simple interpretation; third, we introduce the semantic validity of XML document; finally, we discuss entailment problem for XML.

3.1 Simple Interpretation of XML

After parsing, XML document is validated against DTD or XML Schema and an XML XQuery 1.0 and XPath 2.0 Data Model[16] can be constructed. The data model serves as the vocabulary for XML’s simple interpretation. Because the data model contains information about data type, we first introduce the interpretation of datatype.

Definition 1 (Datatype). *A datatype d is characterized by a lexical space, $L(d)$, which is a set of Unicode strings; a value space, $V(d)$; and a total mapping $L2V(d)$ from the lexical space to the value space. A datatype map D is a partial mapping from URI references to datatypes.*

Definition 2 (XML Vocabulary). *An XML vocabulary V is the data model of XML that consists of:*

1. *N: the node set of XML document, $N = N_e \cup N_a \cup N_t$, where N_e, N_a and N_t denote the set of element nodes, attribute nodes and text nodes, respectively, other kinds of nodes are ignored by the interpretation;*
2. *NP: the set of node pairs, $NP = N \times N$.*

Definition 3 (Simple Interpretation). *A simple interpretation I of an XML vocabulary V is defined by:*

1. *R: a non-empty set of resources, called the universe of I;*
2. *LV: the literal values of I, is a subset of R that contains the set of Unicode strings, the set of pairs of Unicode strings and language tags, and the value spaces for each datatype in D;*
3. *O: a subset of R, is disjoint with LV and contains the individuals of I;*
4. *$S : V_{I_0} \rightarrow O$: a total mapping from URIrefs in XML, denoted as V_{I_0}, into O;*
5. *$M_c : N_e \rightarrow O \cup LV$: a partial mapping from element nodes into individuals and literal values;*
6. *$M_c : N_a \cup N_t \rightarrow O \cup LV$: a total mapping from attribute nodes and text nodes into individuals or literal values. For each node $m \in N_a \cup N_t$:*
   - *(a) if the type of m is xsd:anyURI, then $M_c(m) \in O$ and $M_c(m) = S(dm:$ typed-value$(m))$, where the "dm"-prefixed functions are the XML Data Model's accessor functions and the set of all URIrefs here is V_{I_0};*
   - *(b) if the type of m is supported by OWL(except xsd:anyURI), then $M_c(m) \in LV$ and $M_c(m) = dm :$ typed-value(m);*
   - *(c) if the type of m is not supported by OWL, then $M_c(m) \in V(D(xsd : string))$ and $M_c(m) = dm :$ string-value(m);*
7. *$M_o : NP \rightarrow R \times R$: a partial mapping from node pairs into pairs of resources, i.e., every node pair $< m, n > \in NP$ is interpreted as a resource pair $< M_c(m), M_c(n) >$ within an unknown relationship.*

The simple interpretation of XML has given a primary meaning to nodes and document order in XML Document, for example, an element node may represent an individual, but it cannot tell which class the individual belongs to. This information is further provided by XSDL interpretation.

3.2 XSDL-Interpretation of XML

XSDL provides further information about the author's intended meaning of XML document. The interpretation of XSDL is considered as extensions on XML's simple interpretation.

Definition 4 (XSDL Vocabulary). *An XSDL vocabulary V_X consists of:*

1. *V_0: the vocabulary of global OWL DL ontology, $V_0 = (V_L, V_C, V_D, V_I, V_{DP}, V_{IP}, V_{AP}, V_O)$, for the detailed meaning of OWL vocabulary items, refer to OWL Direct Model-Theoretic Semantics[17];*

2. *XP: an XPath path expressions set, $XP = AXP \cup RXP$, where AXP and RXP denote the set of absolute path expressions and relative path expressions, respectively;*
3. *FN: the set of URI construction functions in XSDL class definitions.*

Definition 5 (XSDL-Interpretation). *An XSDL-Interpretation I of XML vocabulary V extends XML's simple interpretation with:*

1. *$M_{ap} : AXP \rightarrow 2^N$: a mapping from absolute XPath path expression into a node set[5] that must be obtained according to W3C specification[26];*
2. *$M_{rp} : N \times RXP \rightarrow 2^N$: a mapping from relative XPath path expression with respect to node n into a node set that must be obtained according to W3C specification[26];*
3. *$M_{fn} : N \times FN \rightarrow V_{IX}$: a mapping from URI function, with respect to node n, into an URIref. This set of URIrefs is denoted as V_{IX};*
4. *$S : V_{I_0} \cup V_0 \cup V_{IX} \rightarrow R$: S is extended to map all URIrefs in the global ontology and URIrefs constructed by URI functions into R, and $S(V_{IX} \cup V_{I_0}) \subseteq O$;*
5. *I_0: the global ontology's interpretation, $I_0 = (R_0, LV, O_0, S, L, EC, ER)$, where $O_0 = S(V_I)$, $R_0 = S(V_0)$, L, EC and ER are OWL's interpretations of typed literals, classes and properties, respectively;*

The meaning of XPath path expression is provided by mapping into a node set, and then the nodes are mapped into the universe of interpretation by M_c in the simple interpretation. For simplification, we avoid analyzing the detailed syntax of XPath expressions and providing a model-theoretic semantics for them.

The XSDL definitions, such as class definitions, are interpreted as semantic conditions on XML's XSDL-Interpretation:

Definition 6 (Semantic Conditions). *The semantic conditions on XML's XSDL-Interpretation are:*

1. *if there is literal definition: <CtxPath^^element, literal^^ddd>, then:*

$$literal \in V_L, M_{ap}(CtxPath) \subseteq N_e,$$

for each $n \in M_{ap}(CtxPath)$, such that

$$M_c(n) = L2V(D(ddd))(literal);$$

2. *if there is individual definition:<CtxPath^^element, uri^^Individual>, then:*

$$uri \in V_I, M_{ap}(CtxPath) \subseteq N_e,$$

for each $n \in M_{ap}(CtxPath)$, such that

$$M_c(n) = S(uri);$$

[5] In XPath 2.0, the value of an expression is always a sequence. For path expression, the value is a sequences of nodes by eliminating duplicate nodes and sorting in document order. so the node sequence can be viewed as a node set.

3. *if there is class definition:<CtxPath^^element, urifn, cn^^Class>, then:*

$$cn \in V_C, M_{ap}(CtxPath) \subseteq N_e,$$

for each $n \in M_{ap}(CtxPath)$, such that

$$M_c(n) = S(M_{fn}(n, urifn)) \text{ if urifn is given,}$$
$$M_c(n) \in EC(cn);$$

4. *if there is datatype property definition:<CtxPath^^nodeType, DPath, RPath, dpn^^DatatypeProperty>, then:*

$$dpn \in V_{DP}, M_{ap}(CtxPath) \subseteq N(nodeType),$$
$$\text{where } N(nodeType) = \begin{cases} N_e, & \text{if nodeType="element";} \\ N_a, & \text{if nodeType="attribute";} \end{cases}$$

for each $n \in M_{ap}(CtxPath)$,

$$|M_{rp}(n, DPath)| = 1 \text{ or } |M_{rp}(n, RPath)| = 1,$$

i.e., for any context node n, there cannot be both more than one node in domain and range path; let the property's range is datatype d, for each $m \in M_{rp}(n, DPath)$, and for each $t \in M_{rp}(n, RPath)$, such that

$$M_c(t) \in V(d), M_c(t) = L2V(d)(dm : string\text{-}value(t))$$
$$M_o(< m, t >) =< M_c(m), M_c(t) >\in ER(dpn);$$

5. *if there is object property definition without reference: <CtxPath^^nodeType, DPath, RPath, opn^^ObjectProperty>, then:*

$$opn \in V_{IP}, M_{ap}(CtxPath) \subseteq N(nodeType),$$

for each $n \in M_{ap}(CtxPath)$,

$$|M_{rp}(n, DPath)| = 1 \text{ or } |M_{rp}(n, RPath)| = 1,$$

for each $m \in M_{rp}(n, DPath)$, and for each $p \in M_{rp}(n, RPath)$, such that

$$M_o(< m, p >) =< M_c(m), M_c(p) >\in ER(opn);$$

6. *if there is object property definition with reference:<CtxPath^^nodeType, DPath, RPath, IDPath, IDREFPath), opn^^ObjectProperty>, then:*

$$opn \in V_{IP}, M_{ap}(CtxPath) \subseteq N(nodeType),$$

for each $n \in M_{ap}(CtxPath)$,

$$|M_{rp}(n, DPath)| = 1 \text{ or } |M_{rp}(n, IDREFPath)| = 1,$$

for each $m \in M_{rp}(n, DPath)$, and for each $p \in M_{rp}(n, IDREFPath)$, there is one $q \in M_{ap}(IDPath)$ with $M_c(p) = M_c(q)$, then there is one $k \in M_{rp}(q, RPath)$, such that

$$M_o(< m, k >) =< M_c(m), M_c(k) >\in ER(opn).$$

According to the semantic conditions, XSDL further make assertions in the form of: $M_c(n) \in EC(cn)$, $M_c(t) \in V(d)$ and $M_o(< m, n >) =< M_c(m), M_c(n) >\in ER(pn)$, then, XML document is interpreted as a set of fact assertions with respect to the ontology in XSDL. we call this assertion set *XML Facts*. From a view of description logic, the ontology in XSDL is a TBox[6], an XML document conforming to the XSDL definition is an ABox with respect to the TBox.

The XML Facts should be consistent with the ontology in XSDL, otherwise, from the viewpoint of logic, one could draw arbitrary conclusions from it. In terms of our model-theoretic semantics we can easily give a formal definition of consistency.

Definition 7 (Model). *An XML document's XSDL-Interpretation I is the XML's model, if I satisfies both the semantic conditions and the ontology in XSDL.*

Therefore, if an XML's XSDL-Interpretation I is the XML document's model, the XML Facts represented by the document is consistent with the ontology in XSDL. Note that XML model is only meaningful with respect to the XML's XSDL definition.

3.3 Semantic Validity of XML Documents

The XML's model satisfies both the XML Facts and ontology in XSDL. If the XML document has no model, there must be inconsistent between the XML document and the ontology, so having a model is an important property of XML document, this property is called semantic validity[7].

Definition 8 (Semantic Validity). *An XML document is semantically validated with respect to its XSDL definition, if there is a model of the XML document.*

As well-formness and syntactic validity enable XML's syntax checking, semantic validity further enable checking XML's semantic integrity constraints.

To check the semantic validity of XML documents, we first introduce the notion of "corresponding ontology". Because the XML Facts are fact assertions with respect to the global ontology, so we can merge the XML Facts and the global ontology in XSDL to form a new ontology.

Definition 9 (Corresponding Ontology). *For an XML document D, let O_0 denote the global ontology in D's XSDL definition, the corresponding ontology of D is the ontology merged by O_0 with the XML Facts represented by D.*

Second, we introduce the notion of extension and expansion[18]. The first-order language L' is an *extension* of language L if every nonlogical symbol of L

[6] Strictly speaking, the counterpoint of OWL DL ontology in description logic is knowledge base, because the ontology includes fact assertions.

[7] The "semantic validity" is comparable to the (syntactic) validity of XML document and the meaning of "validity" is not in logical sense.

is an nonlogical symbol of L'. A theory T' is an *extension* of theory T if $L(T')$ is an extension of $L(T)$ and every theorem of T is a theorem of T'. Let I' be the interpretation of L', by omitting some interpretation of nonlogical symbol we obtain a interpretation for I. We call I the *restriction* of I' on L and I' an *expansion* of I to L'. We have the following lemma:

Lemma 1 (J. Shoenfield[18]). *If theory T' is an extension of T, M' is a model of T', then the restriction of M' on language $L(T)$ is a model of T.*

Applied this notion to our problem, XML plus XSDL is our language and an XML document plus the XSDL definition is a theory of our language. Obviously, our language is an extension of OWL DL language and the theory of our language is also an extension of the theory of OWL DL language.

Third, we introduce a lemma about the relation between XML's model and the corresponding ontology's model.

Lemma 2. *The restriction of an XML's model on OWL DL language is a model of the XML's corresponding ontology; a model of XML's corresponding ontology can be expanded to the XML's model.*

Proof. 1)By Lemma 1,the restriction of an XML's model on OWL DL language is a model of the XML's corresponding ontology. 2) Denote the corresponding ontology's model as M, then M includes an OWL DL interpretation I_0 whilst the interpretation functions are M_c and M_o. Because M satisfies XML facts, there must be some M_{ap}, M_{rp}, M_{fn} and S satisfy the semantics condition (about M_c and M_o). Trivially $(M_c, M_o, M_{ap}, M_{rp}, M_{fn}, S, I_0)$ is a model of the XML Document. □

Finally, we have the following theorem to decide the semantic validity of XML document.

Theorem 1. *An XML document is semantically validated with respect to its XSDL definition iff the corresponding ontology of XML document is satisfiable.*

The theorem is an obvious consequence of Lemma 2.

According to Theorem 1, if the corresponding ontology of XML document is not satisfiable, then the XML document is semantically invalid. There are two problems resulting in semantic invalidity: first, the ontology in XSDL is not satisfiable; second, there are inconsistences between XML Facts and the ontology in XSDL. Below are some examples to illustrate the inconsistence.

Example 6. A semantically invalid XML fragment

```
<book>
     <author>Jerry</author>
     <author>Tom</author>
     <price>illegal price</price>
</book>
```

If the property *author* is defined as functional property in the global ontology and the XSDL definition for *author* is:

```
<element context = "/book/author" >
    <domainContext path=".." />
    <rangeContext path="text()" />
    <owlx:DatatypeProperty owlx:name="author"/>
</element>
```

then for each XSDL-interpretation I of the XML document, denote the book node as n, if I satisfy the semantic conditions:

$$< M_c(n), ”Jerry” >\in ER(author)$$
$$< M_c(n), ”Tom” >\in ER(author)$$

then I cannot satisfy the global ontology because this is contrary to the ontology definition of “author” property as functional property. Therefore, this XML document is semantically invalid.

In addition, the ill-typed literal will lead to semantic inconsistence too. In example 6, if the *price* node is defined as a datatype property with range as $float$ in XSDL, then for each XSDL-interpreation I, if I satisfy:

$$M_c(\text{“}illegal\ price\text{”}) \in V(D(xsd : float))$$

I cannot satisfy the global ontology because “illegal price” cannot in the value space of data type $float$, so I is not the model of XML document.

Sometimes, semantic invalidity can be avoided by enforcing syntactic checking, such as define the *price* node type as $float$ in XML Schema, then (syntactically) validated XML document will also be semantically validated. But semantic validity checking can further decide whether XML instance satisfy the integrity constraints that cannot expressed by DTD and XML Schema. For example,

Example 7. Another semantically invalid XML fragment about pedigree:

```
<person id="s1" gender="male" name="John"/>
<person id="s2" gender="female" name="Jane"/>
<person id="s3" gender="male" name="Tom" father="s2" mother="s1"/>
```

where “id” is an ID-typed attribute, “father” and “mother” are IDREF-typed attributes, this XML document is semantically invalid, because the value of “father” attribute in the third “person” node is referred to a female person. But both DTD and XML Schema cannot invalidate this kind of error reference, because they cannot assert that the “father” attribute must refer to an “id” attribute accompanied with a “gender” attribute whose value is “male”.

However, this constraint can be expressed in XSDL as follows:

```
 (1) <"/person[@gender='male']"^^element, urifn1, "Man"^^Class>
 (2) <"/person[@gender='female']"^^element, urifn2, "Woman"^^Class>
 (3) <"/person/@father"^^attribute, "..","..", "/person/@id", ".",
      "hasFather"^^ObjectProperty>,
```

The "person" nodes with gender's value as male are defined as instances of class Man and the "person" nodes with gender's value as female are defined as instances of class $Woman$, attribute node "father" represent an object property $hasFather$ with reference. In addition, in the global ontology, we define:

```
(4) <hasFather, rdfs:range, Man>,
(5) <Man, owl:disjointWith, Woman>,
```

Denote the person with "id" equals to "s2" as p_2, and the person with "id" equals to "s3" as p_3, then according to the semantic conditions of XSDL-Interpretation, one part of the XML Facts is:

by (2): $p_2 \in EC(Woman)$,

by (3): $< p_3, p_2 >\in ER(hasFather)$,

Combined with the global ontology, we can easily infer that:

$p_2 \in EC(Man)$,

$p_2 \in EC(Man) \cap EC(Woman)$.

This is contrary to the disjointness between class Man and $Woman$. By Theorem 1, the pedigree.xml is semantically invalid with respect to the above XSDL definition.

3.4 Reasoning About XML

After XML has a formal semantics, we can define entailment of XML documents. However, the definition is somehow different from classical logic. If XML document D_2 have different nodes from document D_1, then any D_1's model cannot be a model of D_2, because D_1's interpretation do not give meaning to the different nodes in D_2. So we need extend D_1's vocabulary to some one(eg. D') that contains D_2's vocabulary. D_1's model M is simultaneously expanded to an interpretation M' defined on D' by keeping the universe and the interpretation of individuals unchanged.

Definition 10 (Entailment). *Assumed that XML document D_1 and D_2 have the same XSDL definition, D_1 entails D_2, if for every D_1's model, there exists an expansion that is a model of D_2. Denoted as $D_1 \models D_2$.*

Definition 11 (Equivalence). *Assumed that XML document D_1 and D_2 have same XSDL definition, D_1 is equivalent with D_2, if $D_1 \models D_2$ and $D_2 \models D_1$. Denoted as $D_1 \equiv D_2$.*

To reduce the XML entailment problem to ontology entailment problem, we have the following theorem:

Theorem 2. *Let the corresponding ontology of XML document D_1 and D_2 are O_1 and O_2, respectively, then $D_1 \models D_2$ iff $O_1 \models O_2$.*

Proof. $\Rightarrow$Let M_1' be an arbitrary model of O_1, by Lemma 2, M_1' can be expanded to D_1's model, denoted as M_1. Since $D_1 \models D_2$, we have M_1's expansion M_2, which is a model of D_2. By Lemma 1, the restriction of M_2 to OWL language is a model of O_2. On the other hand, M_2 is an expansion of M_1 and then is an expansion of M_1', hence the restriction of M_2 to OWL language is M_1', so M_1' is also a model of O_2. That is, $O_1 \models O_2$.

$\Leftarrow$: Let M_1 be an arbitrary model of D_1, by Lemma 1, the restriction of M_1 to OWL language is a model of O_1, denoted as M_0', since $O_1 \models O_2$, so M_0' is also a model of O_2. By Lemma 2, M_0' can be expanded to a model of D_2, denoted as M_2, in addition, M_2 can be further expanded to model M_2' by adding the interpretation of nodes in D_1, obviously, M_2' is an expansion of M_1 and is the model of D_2, that is, $D_1 \models D_2$. □

Theorem 3 (I. Horrocks and P. Patel-Schneider[13]). *OWL DL ontology entailment problem can be reduced to knowledge base satisfiability in description logic language $\mathcal{SHOIN}(\mathcal{D})$ in polynomial time.*

Corollary 1. *The XML entailment problem can be reduced to knowledge base satisfiability in $\mathcal{SHOIN}(\mathcal{D})$ in polynomial time.*

Example 8. Assume XML document D_1 simply is:

```
<man id="p1234" />
```

and XML document D_2 is :

```
<person id="p1234" />
```

If the XSDL definition is:

```
<"/man"^^element,"concat('http://foo.org/person#',
        string('/man[$i]/@id'))", "Man"^^Class>
<"/person"^^element,"concat('http://foo.org/person#',
        string('/person[$i]/@id'))", "Person"^^Class>
<Man, rdfs:subClassOf, Person>
```

Proposition 1. $D_1 \models D_2$ *with respect to the above XSDL definition.*

Proof. The corresponding ontology O_1 of D_1 is:

```
<Man, rdfs:subClassOf, Person>
<"http://foo.org/person#p1234",rdf:type, Man>
```

The corresponding ontology O_2 of D_2 is:

```
<Man, rdfs:subClassOf, Person>
<"http://foo.org/person#p1234", rdf:type, Person>
```

Obviously, $O_1 \models O_2$, by Theorem 2, XML document D_1 entails D_2. □

Unfortunately, the complexity for the satisfiability problem is in NExpTime and there are yet no known optimized inference algorithms or implemented systems for $\mathcal{SHOIN}(\mathcal{D})$. However, if the ontology language is restricted to OWL Lite, then the problem can be reduced to knowledge base satisfiability in $\mathcal{SHIF}(\mathcal{D})$, whose complexity is in ExpTime[13]. The highly optimized reasoner RACER[19] can provide efficient reasoning services for $\mathcal{SHIF}(\mathcal{D})$.

4 Related Works

The “semantics” of XML have different understanding. The analogy between a document tagged by XML and a source string generated by a BNF grammar is noticed and thus enable adding semantic attributes and functions to XML[20]. From the SGML field, the BECHAMEL project[3] are trying to apply knowledge representation technologies to the modelling of meaning and relationship expressed by XML markup. The prototype formalization language and implementation environment is based on Prolog[21]. The formalization is complex and difficult to fulfill the requirement of Semantic Web.

Recently, P. Patel-Schneider and J. Siméon propose the idea of Yin/Yang Web [10], in which XML XQuery 1.0 and XPath 2.0 Data Model is regarded as a unified model for both XML and RDF, and a RDF-compatible semantics is developed based on this data model. However, because XML author can express semantics by almost arbitrary ways, the direct interpretation for XML in Yin/Yang Web is difficult to capture the author’s intended meaning. We introduce XSDL to specify XML’s semantics and gives XML meaning by two steps: the simple interpretation and the XSDL-interpretation. The two-step semantics is of more clarity and closer to XML author’s intended meaning.

XSDL is similar to MDL[11] in adoption of Schema Adjuncts Framework and definition of XML semantics by conceptual model. However, MDL has proprietary syntax and takes UML as modelling language, in contrast, XSDL’s syntax are mostly the standard XPath and OWL’s XML syntax, hence XSDL is simple, easy to learn and implement; XSDL takes OWL DL as modelling language, thus XSDL has formal semantics, enables reasoning about XML and helps to bridge the gap between XML and Semantic Web.

XSDL defines XML semantics by mapping XML to ontology. There are other efforts: M. Erdmann and R. Studer[22] present a tool to generate DTD from ontology, then the tags of XML instances conforming to this DTD can be mapped to concepts and properties in the ontology; B. Amann, et al. [23]propose a rule-based language to map XML fragments into general ontology and later I. Fundulaki1 and M. Marx[24] provide a formal semantics by interpreting XML sources into ER models. The rule language does not support literal, individual definition and object property definition by reference in XSDL, and the ontology path in mapping rule is not supported by OWL. Besides, their work is intended for the querying of heterogeneous XML resources using an ontology-based mediator. In contrast, our work is intended to bridge the gap between XML and Semantic Web and is believed to be more tightly integrated with the Semantic Web architecture.

5 Conclusion and Future Works

In this paper, to address the problem that XML have no formal semantics, we propose XML Semantics Definition Language(XSDL) and a model-theoretic semantics for XML. XSDL is a simple language with which syntax mainly come from XPath, OWL XML syntax and SAF. There are only three additional constructs in XSDL: URI constructor, domain context and range context. The more significant work is the formal semantics for XML, which gives XML meaning by simple interpretation and XSDL-interpretation and is close to XML author's intended meaning. The semantics is compatible with a subset of RDF supported by OWL DL, hence, XML becomes a sub-language of RDF in expressive power and XML data can be semantics-preserving transformed to RDF data.

The expressive power of XML is the same as ABox in description logic language, thus is limited compared to general formal language. Therefore, XSDL is suitable to represent the semantics of data-centric XML document.

One limitation of our work is that XSDL document for XML need to be defined manually and the authoring is a laborious, time-consuming task. Note that XML Schema also has rich implicit semantic information, such as datatypes, cardinality constraints. The solution is to generate XSDL definition from XML Schema for author's further reviews and to develop user-friendly XSDL editor.

As Yin/Yang Web, our work can also be applied to semantic query of XML data, XML data integration and Semantic Web Services. In addition, XSDL is more natural and powerful to represent XML data integrity constraints than in a syntactic way, such as XML Schema. We will explore these application areas in future works.

Acknowledgement. We are grateful to Wei Song and the anonymous reviewers for their helpful suggestions for the improvements of this paper.

References

1. Bray, T., Paoli, J., Sperberg-McQueen, C.M., Maler, E.: Extensible Markup Language (XML) 1.0 (second edition) W3C recommendation (2000)
2. Cover, R.: XML and semantic transparency (1998)
3. Allen, R., Dubin, D., Sperberg-McQueen, C.M., Huitfeldt, C.: Towards a semantics for XML markup. In: the 2002 ACM Symposium on Document Engineering, 119–126
4. Uschold, M.: Where are the semantics in the Semantic Web? AI Magazine **24** (2003) 25–36
5. Berners-Lee, T., Handler, J., Lassila, O.: The Semantic Web. Scientific American **184** (2001) 34–43
6. Klyne, G., Carroll, J.J.: Resource Description Framework (RDF):concepts and abstract syntax,W3C recommendation 10 february 2004 (2004)
7. Berners-Lee, T.: Why RDF is more than XML (1998)
8. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL Web ontology language reference, W3C recommendation 10 february 2004 (2004)
9. Patel-Schneider, P.F., Simeon, J.: Building the Semantic Web on XML. In: the Twelfth International World Wide Web Conference, ACM Press (2003)
10. Patel-Schneider, P.F., Simeon, J.: The Yin/Yang Web: A unified model for XML syntax and RDF semantics. IEEE Transactions on Knowledge and Data Engineering **15** (2003) 797–812
11. Worden, R.: MDL: A Meaning Definition Language, version 2.06 (2002)

12. Berglund, A., Boag, S., Chamberlin, D., et al.: XML Path Language (XPath) 2.0 W3C working draft 12 november 2003 (2003)
13. Horrocks, I., Patel-Schneider, P.F.: Reducing OWL entailment to description logic satisfiability. In: the 2003 International Semantic Web Conference. 17–29
14. Vorthmann, S., Buck, L.: Schema Adjunct Framework draft specification 24 february 2000 (2000)
15. Hori, M., Euzenat, J., Patel-Schneider, P.F.: OWL Web ontology language XML presentation syntax . W3C note 11 june 2003 (2003)
16. Fernandez, M., Malhotra, A., Marsh, J., Nagy, M., Walsh, N.: XQuery 1.0 and XPath 2.0 data model, W3C working draft. (2003)
17. Patel-Schneider, P., Hayes, P., Horrocks, I.: OWL Web ontology language semantics and abstract syntax, W3C recommendation 10 february 2004 (2004)
18. Shoenfield, J.R.: Mathematical Logic. Addison-Wesley Publisher (1967)
19. Haarslev, V., Moller, R.: Racer system description. In: International Joint Conference on Automated Reasoning (IJCAR'2001), Siena, Italy (2001) 18–23
20. Psaila, G., Crespi-Reghizzi, S.: Adding semantics to XML. In: Second Workshop on Attribute Grammars and their Applications, (1999) 113–132
21. Dubin, D., Sperberg-McQueen, C.M., Renear, A., Huitfeldt, C.: A logic programming environment for document semantics and inference. Literary and Linguistic Computing **18** (2003) 225–233
22. Erdmann, M., Studer, R.: How to structure and access XML documents with ontologies. Data and Knowledge Engineering **36** (2001) 317–335
23. Amann, B., Fundulaki, I., Scholl, M., Beeri, C., Vercoustre, A.: Ontology-Based Integration of XML Web Resources. In: International Semantic Web Conference 2002. (2002) 117–131
24. Fundulaki, I., Marx, M.: Mediation of XML Data through Entity Relationship Models. In: First International Workshop on Semantic Web and Databases. (2003) 357–380
25. Liu S.P., Mei J., Lin Z.Q.: XML Semantics Definition Language(XSDL) draft specification(In Chinese). PKU-TCL lab techonology report. (2004)
26. Draper D., Fankhauser P., Fernedez M., et al.: XQuery 1.0 and XPath 2.0 Formal Semantics, W3C Working Draft 20 February 2004 (2004)