The relation between ontologies and schema languages

Michel Klein, Dieter Fensel, Frank van Harmelen and Ian Horrocks
Vrije Universiteit Amsterdam

Goal of this talk

In this talk, we will

• briefly discuss the relation between ontologies and document schemas
• explain why it can be useful to derive schemas from ontologies,
• and show a translation procedure from an OIL ontology to an XML Schema
Outline

• Ontologies and document schemas
  - functions and characteristics
• Sketch of their relation
• Benefits of using an ontology on top of XML Schema
• Translation procedure
  - introduction to XML Schema
  - illustration of the translation
• Discussion
• Conclusions

Ontologies and document schemas

Ontologies
  - describe *conceptualization* of a domain
  - used for communication and reasoning

Structuring document schemas
  - prescribe *structure* of a document
  - provide integrity constraints
  - enable access to data

*Different things with different functions*
... do have a relation

**Document schemas**
- also represent a consensus between parties involved in the exchange of information

**Similarities**
- represent some domain knowledge
- are used to describe information for exchange
- use (sometimes advanced) modeling constructs

*In many cases, document schemas are preferably based on an ontology*

---

**Example: CIA World Fact Book**

- very structured information
- representation of real world objects
- should be based on a domain theory
Sketch of relation

Three layer model

1. an ontology modeling language (e.g. OIL)
2. a specific ontology describing a particular domain, using constructs from the modeling language
   - ontology written down in OIL
   - a document structure prescription (e.g. an XML Schema document) derived from the ontology
3. actual XML data conforming to the schema (instances of the ontology)

Prerequisite for translation

• the exact relation between the ontology language (e.g. OIL) constructs and XML Schema constructs

Derive XML Schema from ontology

... class-def Country_Comparison
  slot-constraints
  compared_to
  value-type Country
  proportion
  value-type STRING
...

(formatted)

Area -- comparative: slightly less than twice the size of New Jersey.

Capital: Amsterdam; The Hague is the seat of government.

OIL primitives:
- class, slot;
- value-type, etc;

used in

XML Schema primitives:
- complexType

used in

translate to

OIL ontology

prescribes structure

XML schema

XML documents
Benefits of using an ontology on top of XML Schema

- expressive modeling constructs (intentional types) can be used to model XML data
- document structure is grounded on a true semantic basis
- conceptual layer on top of XML data to facilitate higher level access, abstracted from document structure

Conceptual layer

Relational schema

```
SELECT Name FROM Country
```

Direct access:
```
(string)readNext(block);
```

OIL ontology

```
value this.country.name
```

XML Schema

```
<xsl:value-of select="parent::*/parent::*/@name"/>
```

Database

```
<country><name>TheNetherlands</name><capital>Amsterdam</capital></country>
```

XML documents
XML Schema introduction (1)

XML Schema has same function as DTD
- prescribes document structure
but has some advantages:
- XML Schema is XML itself (also namespaces)
- simple datatyping
- richer grammar
- type hierarchy with derivation

```xml
<complexType name="CountryType">
    <element name="Name" type="string"/>
    <element ref="Capital"/>
</complexType>
```

XML Schema (2): richer grammar

- content models
  - grouping, by choice, sequence or all
- cardinality
  - attributes: minOccurs, maxOccurs
- defaults and constants
  - attributes: default, fixed

```xml
<complexType name="WindowsType">
    <element name="version" type="string" minOccurs="0" maxOccurs="1" default="W98"/>
    <element name="includedBrowser" type="string" minOccurs="0" maxOccurs="1" fixed="Internet Explorer"/>
</complexType>
```
XML Schema (3): type hierarchy

- two types of derivation
  - derived by extension
  - derived by restriction
- no multiple inheritance
- derivation hierarchy is NOT a class hierarchy
  - inheritance is either restricting or extending
  - instances are not automatically instance of supertype

```
<complexType name="price" base="decimal" derivedBy="extension">
  <attribute name="currency" type="string"/>
</complexType>

<complexType name="USaddr" base="address" derivedBy="restriction">
  <element name="country" fixed="US"/>
</complexType>
```

Translation procedure

1. Materialize the OIL hierarchy
   e.g. deriving all subsumptions, applying slots to classes

2. Create an XML-type for each class and slot
   (exploiting the type-derivation mechanism)
   e.g. CountryType, derived from GeographicalLocationType

3. Add elements to the Type for each slot
   e.g. CountryType has an element <comparative_area>

4. Create an element definition for each Class
   (using the type that is defined in step 2)
   e.g. element <country> is of type CountryType
### Exact relation and mapping

<table>
<thead>
<tr>
<th>OIL construct</th>
<th>XMLS translation</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>class</td>
<td>complexType</td>
<td>all applicable slots appear as elements</td>
</tr>
<tr>
<td>slot</td>
<td>complexType</td>
<td>range appears as element</td>
</tr>
<tr>
<td>hasValue</td>
<td>minOccurs=&quot;1&quot;</td>
<td>a value is required</td>
</tr>
<tr>
<td>valueType</td>
<td>minOccurs=&quot;0&quot;</td>
<td>if there is a value, it should be of type type</td>
</tr>
<tr>
<td>cardinality</td>
<td>minOccurs=&quot;x&quot;</td>
<td>also for minCardinality and maxCardinality</td>
</tr>
</tbody>
</table>

### Example ontology for WFB

```
class-def Country
    subclass-of Geographical_Location
    slot-constraints
        name
            value-type STRING
        comparative_area
            value-type Country_Comparison

class-def City
    subclass-of Geographical_Location

class-def Country_Comparison
    slot-constraints
        compared_to
            value-type Country
        proportion
            value-type STRING
```

```python
slot-def capital
domain Country
range City

slot-def border_country
domain Country
range Country
properties symmetric
```
Resulting XML Schema and data

```xml
<complexType name="CountryType" base="Geo..Type" derivedBy="extension">
  <element name="name" type="string" minOccurs="0"/>
  <element ref="capital" minOccurs="0"/>
  <element ref="comparative_area" minOccurs="0"/>
</complexType>
<complexType name="Country_ComparisonType">
  <element name="proportion" type="string" minOccurs="0"/>
  <element ref="compared_to" type="CountryType" minOccurs="0"/>
</complexType>
<element name="country" type="CountryType"/>
<element name="comparative_area" type="Country_ComparisonType"/>
```

```
<country>
  <name>The Netherlands</name>
  <capital><city><name>Amsterdam</name></city></capital>
  <comparative-area>
    <proportion>slightly less than twice the size</proportion>
    <compared_to>
      <country><name>New Jersey</name></country>
    </compared_to>
  </comparative-area>
</country>
```

Discussion

- XML Schema is no KR language
  - *not meant to be*
  - but can capture some knowledge
- other translations also possible
  - weakness, because reverse procedure is ambiguous
- reverse direction is interesting
  - to reason over existing XML schemas
  - to find implicit inheritance relations
Conclusions

• Ontologies and document schemas serve different purposes
• Document schemas can be derived from domain theories
• Benefits of using an ontology to define a document schema are:
  - intentional type definition
  - semantic basis for document
  - conceptual level on top of XML data