Data-centric XML

XML Schema (Part 1)

Schema and DTD

- Disadvantages of DTD:
  - separate, non-XML syntax
  - very limited constraints on data types (just ID, IDREF, …)
  - no support for sets (i.e. each element type must occur precisely once)
  - all elements are global (i.e. their content model applies in the entire document)
  - no support for namespaces
- Goals of XML Schema
  - description of the structure of a document
  - description of the data of a document
  - *as a document*

References

http://www.informatik.hu-berlin.de/~obecker/Lehre/SS2002/XML/11a-schema.xml

- W3 Recommendations (XML Schema 1.0)
  - http://www.w3.org/TR/xmlschema-0/ (Primer)
  - http://www.w3.org/TR/xmlschema-1/ (Structures)
  - http://www.w3.org/TR/xmlschema-2/ (Datatypes)
  - XML Schema 1.1: Working Draft
- Other Resources
  - http://www.w3.org/XML/Schema
  - http://www.jeckle.de/xsd-de/
  - http://www.w3.org/2001/03/webdata/xsv

An Example

- Example document: catalog of a bookstore

```
<buchhandlung art="belletristik">
  <buch>
    <autor>...</autor>
    <titel>...</titel>
    <verlag>...</verlag>
    <isbn>...</isbn>
    <preis>...</preis>
  </buch>
  <buch>
    ...
  </buch>
</buchhandlung>
```

An Example (2)

- Define a schema for the document
  - Starting with boilerplate

```
<xs:schema
   xmlns:xs="http://www.w3.org/2001/XMLSchema">
  ...
</xs:schema>
```

An Example (3)

- Define the root element of the document (buchhandlung), which is of a complex type

```
<xs:schema
   xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="buchhandlung">
   <xs:complexType>
    ...
   </xs:complexType>
  </xs:element>
</xs:schema>
```

An Example (4)

- A buchhandlung contains multiple buch elements, and a art attribute

```
<xs:schema
   xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="buchhandlung">
   <xs:complexType>
     <xs:sequence>
       <xs:element name="buch"
              maxOccurs="unbounded">
       ...
      </xs:element>
    </xs:sequence>
    <xs:attribute name="art" type="xs:string"/>
   </xs:complexType>
  </xs:element>
</xs:schema>
```

An Example (5)

- Similarly, a buch is defined by listing its child elements

```
...
<xs:element name="buch" maxOccurs="unbounded">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="autor"  type="xs:string"/>
      <xs:element name="titel"  type="xs:string"/>
      <xs:element name="verlag" type="xs:string"/>
      <xs:element name="isbn"   type="xs:string"/>
      <xs:element name="preis"  type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
...
```

- Complete schema follows the Matryoshka design

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="buchhandlung">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="buch" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="autor"  type="xs:string"/>
              <xs:element name="titel"  type="xs:string"/>
              <xs:element name="verlag" type="xs:string"/>
              <xs:element name="isbn"   type="xs:string"/>
              <xs:element name="preis"  type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
      <xs:attribute name="art" type="xs:string"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Global Element Definitions

- An element defined in a schema can be used in multiple places

  e.g. defined through <xs:element name="author" type="xs:string/>

  can be referred as <xs:element ref="author"/> inside e.g. xs:sequence
- Global element definitions produce a flat schema, like DTDs

Reusable Element Types

- Structure of a type (attributes, sequence) can be given a name, and then used to define multiple elements

```
<xs:complexType name="buchlisteTyp">
  <xs:sequence>
    <xs:element ref="buch" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute ref="art"/>
</xs:complexType>

<xs:element name="buchhandlung"
            type="buchlisteTyp"/>
```

Content Models

- Like DTDs, XML Schema allows the definition of content models
- Empty content: Complex type with no content, possibly attributes

```
<xs:element name="leer">
  <xs:complexType />
</xs:element>

<xs:element name="leerMitAtt">
  <xs:complexType>
    <xs:attribute name="att" type="xs:string"/>
  </xs:complexType>
</xs:element>
```

Simple Content with Attributes

- Example:

<buchtipp isbn="3-251-00452-2">Ein Buch, das in keinem Bücherschrank fehlen darf.</buchtipp>

- Starting with xs:simpleContent, extended with attributes

```
<xs:element name="buchtipp">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="isbn" type="xs:string" />
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

Structured Types

- Sequences (xs:sequence): List child elements in order
- Choices (xs:choice): List alternative child elements
- Sets (xs:all): List child elements that must occur, but can occur in arbitrary order
  - Must not be part of other content models
  - Must contain only elements

Cardinalities

- Attributes minOccurs= and maxOccurs= on xs:element (default is 1)
  - Only allowed for nested elements, not in global elements
  - Value “unbounded” means arbitrary repetition

```
<xs:element name="x" type="..."
        minOccurs=“0" maxOccurs=“unbounded"/>
```

Cardinalities Example

The DTD content model

((arbeit, essen)*, (sport | spiel+)?, schlaf)+

translates to

```
<xs:complexType>
  <xs:sequence maxOccurs="unbounded">
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:element name="arbeit" type="..." />
      <xs:element name="essen" type="..." />
    </xs:sequence>
    <xs:choice minOccurs="0">
      <xs:element name="sport" type="..." />
      <xs:element name="spiel" type="..." maxOccurs="unbounded" />
    </xs:choice>
    <xs:element name="schlaf" type="..." />
  </xs:sequence>
</xs:complexType>
```

Mixed Content

- Attribute mixed="true" for xs:complexType

(#PCDATA | tier | ort)*

```
<xs:complexType mixed="true">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element name="tier" type="..." />
    <xs:element name="ort"  type="..." />
  </xs:choice>
</xs:complexType>
```

- Also allowed for xs:sequence and xs:all

Attributes

- xs:attribute
- Always occur after child elements
- name= specifies attribute name
- type= specifies simple type
- use="required"|"optional"
- default= specifies default value
- fixed= specifies fixed value

XML Schema So Far

- Additions with respect to DTDs:
  - More complex cardinalities (not just +, *, ?)
  - Set content models (xs:all)
  - Mixed Content for xs:sequence, xs:choice, xs:all