Lecture 14: XML Schema

Wendy Liu

CSC309F – Fall 2007

Outline

- XML Name Space
- XML Schema
- Structure and Data Types

Name Space

- Document may include elements and attributes from different schemas
- Namespaces are used to disambiguate
- Defined with xmlns attribute
  - xmlns[:prefix]=URI
    - If no prefix is specified, it is referred to as the default name space
  - URI uniquely identifies namespace
    - Has no further meaning

Example 0: Name Space

```
<states
  xmlns="http://www.states-info.org/states"
  xmlns:cap="http://www.states-info.org/state-capitals" >
  <state>
      <name> South Dakota </name>
      <population> 754844 </population>
      <capital>
          <cap:name> Pierre </cap:name>
          <cap:population> 12429 </cap:population>
      </capital>
  </state>
  <!– more states -->
</states>
```

XML Schema

www.w3.org/XML/Schema

5

Deficiencies in DTDs

- Do not use XML syntax
- Do not support namespaces
- Data types cannot be strictly specified
  - Example
    - date vs. string

6

Schema

- Description of an XML content model
  - Define structure of instances
  - Define data types of elements and attributes
- Is an XML application
  - Follow XML syntax
- Support namespaces
  - The XML Schema language itself is a set of XML tags
  - The application being described is another set of tags
- Documents that conform to a schema's rules are considered *instances* of that schema

7

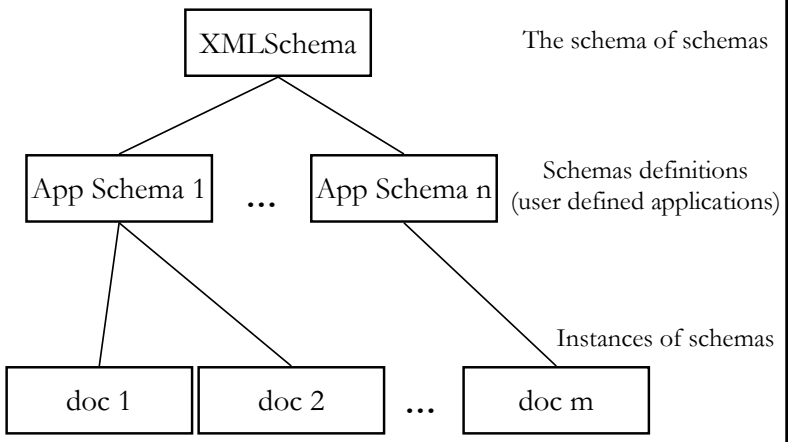
Sample Schema

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

    <xsd:element name="Book"> … </xsd:element>
    <xsd:element name="Library">
        <xsd:complexType>
            <xsd:sequence>
              <xsd:element ref="Book" minOccurs="0"
                    maxOccurs="unbounded"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
</xsd:schema>
```

8

Conceptual Model



9

XMLSchema

xmlns = "http://www.w3.org/2001/XMLSchema"

- Tags
  - <schema>
  - <element>
  - <attribute>
  - <simpleType>
  - <complexType>
  - etc.
- Primitive and derived data types
  - int, boolean, string, date, anyURI, etc.

10

Structure and Data Types

http://www.w3.org/TR/xmlschema-0
http://www.w3.org/TR/xmlschema-1
http://www.w3.org/TR/xmlschema-2
http://www.xmlschemareference.com

11

XML Schema Definition (XSD)

- Document model that conforms to the XML Schema standard
  - An XML application that can be used to describe other XML applications
    - Defined in terms of the XMLSchema tag set:
      - <schema>
      - <element>
      - <attribute>
      - <simpleType>
      - <complexType> etc.

12

<schema>

- Root element for schema documents
- Attributes
  - `xmlns`
    - For the schema namespace and for the namespace being defined
  - `targetNamespace`
    - Declare the namespace being defined
  - `elementFormDefault`
    - With the value `qualified`, all elements defined in the target namespace must be namespace qualified when used
      - I.e., either with a prefix or be the default

13

Example 1: <schema>

```
<schema
   xmlns = "http://www.w3.org/2001/XMLSchema"
   xmlns:c = "http://www.cs.toronto.edu/course"
   targetNamespace = "http://www.cs.toronto.edu/course"
   elementFormDefault = "qualified">
                  . . .
</schema>
```

14

Example 2 – More Commonly Used

```
<xsd:schema
   xmlns = "http://www.cs.toronto.edu/course"
   xmlns:xsd = "http://www.w3.org/2001/XMLSchema"
   targetNamespace = "http://www.cs.toronto.edu/course"
   elementFormDefault = "qualified">
                  . . .
</xsd:schema>
```

15

<element>

- Attributes
  - name
  - type
  - ref
    - Must reference a global element
  - maxOccurs
    - (nonNegativeInteger | unbounded) : 1
  - minOccurs
    - nonNegativeInteger : 1
- Content
  - (simpleType | complexType)

16

Example 3: <element>

- Definition

  <element name="score" type="integer"
  default="0" />
- Instances

  <score />

  <score>124</score>

17

<attribute>

- Attributes
  - name
  - type
  - default
  - use
    - (optional | prohibited | required) : optional
  - ref
    - Must reference a global element
- Content
  - simpleType

18

Example 4: <attribute>

- Create new attributes

  <attribute name="format" type="string" />

  <attribute name="stage" type="string"
  use="required"/>
- How to associate attributes to elements?

19

Example 5: Element with Attributes

- Definition

```
<element name="picture">
  <complexType>
    <attribute name="format" type="string" use="optional" />
    <attribute name="uri" type="string" use="required"/>
  </complexType>
</element>
```

- Instance

  <picture format="GIF" uri="images/boat.gif" />

20

<complexType>

- Attributes
  - name
- Content
  - (group | all | choice | sequence)
    - group
      - A custom defined group
    - all
      - Each child appear (at most) once in any order
    - choice
      - Only one of its children is allowed
    - sequence
      - In the given order
  - attribute

21

Example 6: Nested Element

```
<xsd:element name="letterBody">
 <xsd:complexType mixed="true">
  <xsd:sequence>
   <xsd:element name="salutation">
    <xsd:complexType mixed="true">
     <xsd:sequence>
      <xsd:element name="name" type="xsd:string"/>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
   <xsd:element name="quantity"    type="xsd:positiveInteger"/>
   <xsd:element name="productName" type="xsd:string"/>
   <xsd:element name="shipDate"    type="xsd:date" minOccurs="0"/>
   <!-- etc. -->
  </xsd:sequence>
 </xsd:complexType>
</xsd:element>
```

22

Example 7: Repeatable Elements

```
<xsd:element name="Book">
        …
</xsd:element>
<xsd:element name="Library">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element ref="Book" minOccurs="0"
          maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

23

<simpleType>

- Attributes
  - name
- Content
  - (restriction | list | union)
    - restriction
      - Constrain the range of values
    - list
      - List of the defined type
    - union
      - Union of the included types

24

Constraining Facets (<restriction >)

- length
- minLength
- maxLength
- pattern
- enumeration
- whiteSpace
- maxInclusive
- maxExclusive
- minExclusive
- minInclusive
- totalDigits
- fractionDigits

25

Example 8: <simpleType>

```
<simpleType name='Sku'>
  <restriction base='string'>
    <pattern value='\d{3}-[A-Z]{2}' />
  </restriction>
</simpleType>
```

- Pattern matches
  - Three digits followed by a hyphen followed by two upper-case ASCII letters
- XML Schema Regular Expression References

  http://www.xmlschemareference.com/regularExpression.html
  http://www.w3.org/TR/xmlschema-2/#dt-regex

26

Example 9: <simpleType>

- Definitions

```
<xsd:element name="zips" type="zipUnion"/>
<xsd:simpleType name="zipUnion">
  <xsd:union memberTypes="USState listOfMyIntType"/>
</xsd:simpleType>
<xsd:simpleType name="USStateList">
  <xsd:list itemType="USState"/>
</xsd:simpleType>
<xsd:simpleType name="listOfMyIntType">
  <xsd:list itemType="myInteger"/>
</xsd:simpleType>
```

- Instances

```
<zips>CA</zips>
<zips>95630 95977 95945</zips>
<zips>AK</zips>
```

27

Scope of Type Definitions

- For <simpleType> and <complexType>
- Global declaration
  - Declaration in a child element of **<schema>**
  - Visible everywhere in the schema
- Local declaration
  - Declaration in a grandchild (or more distant descendent) element of **<schema>**
  - Visible only in the scope of its parent element

28

Example 10: Local Type

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

   <xsd:element name="quantity">
         <xsd:simpleType>
           <xsd:restriction base="xsd:positiveInteger">
             <xsd:maxExclusive value="100"/>
           </xsd:restriction>
         </xsd:simpleType>
   </xsd:element>
   ...
</xsd:schema>
```

29

Example 11: Global Types

```
<xsd:schema ... >
     <xsd:simpleType name="zipUnion">
       <xsd:union memberTypes="USState listOfMyIntType"/>
     </xsd:simpleType>
     <xsd:complexType name="USAddress" >
       <xsd:sequence>
         <xsd:element name="name"   type="xsd:string"/>
         <xsd:element name="street" type="xsd:string"/>
         <xsd:element name="city"   type="xsd:string"/>
         <xsd:element name="state"  type="xsd:string"/>
         <xsd:element name="zip"    type="xsd:decimal"/>
       </xsd:sequence>
       <xsd:attribute name="country" type="xsd:NMTOKEN" fixed="US"/>
     </xsd:complexType>
     <xsd:simpleType name="listOfMyIntType">
       <xsd:list itemType="myInteger"/>
     </xsd:simpleType>
     ...
</xsd:schema>
```

30

Example 12: A Sophisticated Type

```
<xsd:complexType name="Items">
 <xsd:sequence>
  <xsd:element name="item" minOccurs="0" maxOccurs="unbounded">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element name="productName" type="xsd:string"/>
     <xsd:element name="quantity">
      <xsd:simpleType>
       <xsd:restriction base="xsd:positiveInteger">
        <xsd:maxExclusive value="100"/>
       </xsd:restriction>
      </xsd:simpleType>
     </xsd:element>
     <xsd:element name="USPrice"  type="xsd:decimal"/>
     <xsd:element ref="comment"   minOccurs="0"/>
     <xsd:element name="shipDate" type="xsd:date" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="partNum" type="SKU" use="required"/>
   </xsd:complexType>
  </xsd:element>
 </xsd:sequence>
</xsd:complexType>
```

31

Built-in Data Types

- Built-in primitive types include
  - string, boolean, float, double, decimal, anyURI, time, date
- Built-in derived types include
  - int, integer, long, token

http://www.w3.org/TR/xmlschema-0/#CreatDt
http://www.w3.org/TR/xmlschema-2/#built-in-datatypes

32

Validation

Validation

- Online validation
  - http://www.w3.org/2001/03/webdata/xsv
- Apache Xerces
  - http://xerces.apache.org/xerces2-j/
    - java –cp xercesImpl.jar;. Validator –v library.xml
  - /u/csc309h/lib