XML Schema

Prof. B.D. Chaudhary

Table of contents

DTD Overview

- DTD's syntax inherited from SGML
- DTDs are not XML document
- DTDS can not be parsed and manipulated
- Simple and availability of tools

DTD Overview (Contd...)

- DTD describes the structure
- It has non-extensible content model
- Only content type is PCDATA
- Attributes have also non-extensible types
- Absence of user defined types

DTD Overview (Contd...)

- <quantity> 5 </quantity> and <quantity> HELLO </quantity> are valid
- One will like to restrict quantity to be numeric only and will expect the parser to detect the type violation
- Schema Valid: Document that conforms to schema

XML Schema

- **Schema** is an alternative modeling language
- Schema technology is still evolving
- Major schema models: XDR and XSD
- XDR: XML Data Reduced
- XSD: XML Schema Definition Language

XML Schema (Contd...)

- RELAX NG from OASIS Technical group
- Schematron: Rule based, XPATH base
- DTD and Schema both coexist

Some Observations

- Schema document uses XML syntax
- Schema's are XML documents
- Schema documents conform to DTDs
- Schemas are valid documents
- Schema processor provides additional information to application

Microsoft XDR Schema

- Schema developed by Microsoft
- XDR stands for XML Data Reduced
- For more details: http://msdn.microsoft.com/en-us/library/ms256208.aspx

Namespace Declarations

- An XML document may rely on more than one schema
- Same name may be used for different purpose
- Namespace resolves this ambiguity
- Declaration: Default and Explicit

Namespace Declarations (Contd...)

- Default: xmlns="Namespace"
- Explicit: xmlns:prefix="Namespace"
- prefix: Name that can be used to refer to the namespace
- Example: <addresslist xmlns="x-schema:addresslist.xdr" xmlns:old="x-schema:oldaddresslist.xdr">

Elements of XDR Schema

- XDR Schema DTD defines only eight elements
- Schema: Root element for XDR schema document
- datatype: Describes types of data for elements and attributes
- ElementType: Describes an individual element

Elements of XDR Schema (Contd...)

- group: Used to group elements for defining order
- AttributeType: Describes type of attribute
- attribute: Describes an attribute
- description: Provides descriptive information for an element or an attribute

Schema Element

- *Schema* element defines the root element of an XDR schema
- It has two attributes: name and xmlns
- name: Specifies the name of schema
- xmlns: Specifies schema name space
- Must be set to: urn:schemas-microsoft-com:xml-data
- xmlns defines default name space

Schema Element (Contd...)

```
<Schema name="schema-name" xmlns="name-space">
  <ElementType>
    <element>
      ........
    </element>
    <AttributeType>
    ..............
    </AttributeType>
      <attribute>
        ..........
      </attribute>
```

Schema Element (Contd...)

```
    <description>
      ............
    </description>
  </ElementType>
  <ElementType>
    ..........
    ..........
  </ElementType>
</Schema>
```

An Example Schema

```
<?xml version = "1.0"?>
<!-- Fig. 7.1: intro-schema.xml -->
<!-- Microsoft XML Schema showing the ElementType -->
<!-- element and element element -->
<Schema name=mySchema
xmlns="urn:schemas-microsoft-com:xml-data">
    <ElementType name="message" content="textOnly"
    model="closed" >
        <description>Text messages</description>
    </ElementType>
    <ElementType name="greeting" model="closed"
    content="mixed" order="many" >
        <element type="message"/>
    </ElementType>
```

An Example Schema (Contd...)

```
    <ElementType name = “myMessage” model=“closed”
    content=“eltOnly” order=“seq”>
        <element type=“greeting” minOccurs=“0”
        maxOccurs=“1”/>
        <element type=“message” minOccurs=“1”
        maxOccurs=“*”/>
    </ElementType>
</Schema>
```

Element Schema

- Element **Schema** can contain only:
  - ElementType: For defining elements
  - AttributeType: For defining attributes
  - description: For describing schema

Element Schema (Contd...)

- <ElementType name = “message” content = “textOnly” model = “closed” >
- Element being defined has name “message”
- It can contain only text
- closed: Only elements declared in the schema are permitted in the document
- Elements not defined in the schema will invalidate the document

Element Schema (Contd...)

- <description> Text message </description>
- Text that describes schema
- Schema author uses description to provide information about schema to a parser or application using schema
- Other value for content: “mixed”, “eltOnly”
- Can contain both element and CDATA

Element Schema (Contd...)

- Other attribute: order = “many”
- Other values for order: “seq”
- Any number of message element and text in any order may appear in greeting
- “eltOnly” value of content: Element only
- The child elements of myMessage must appear in the sequence defined in the schema

Element Schema (Contd...)

- minOccurs and maxOccurs: Specify number of occurences of element
- * indicates unlimited time
- 0 value for minOccurs indicate optional
- A sample document conforming to the schema is given in the next slide

XML Document Conforming to Schema

```
<?xml version = "1.0" >
<!-- File name intro.xml -->
<!-- File name of schema intro-schema.xml -->
<myMessage xmlns = "x-schema:intro-schema.xml">
    <greeting>Welcome to XML Schema !
        <message>This is firat message </message>
    </greeting>

    <message>This is second message.</message>
</myMessage>
```

Note: The schema in use, i.e., referenced by using xmlns

Attributes of ElementType

- There are five attributes
- name, model, content, order, dt:type
- name: Element name
- It is required attribute
- It must be unique within its scope
- content: empty, eltOnly, textOnly, and mixed
- Default value of content: mixed

Attributes of ElementType (Contd...)

- order: one, seq, many
- Default for order: many for no restriction
- seq if content is eltOnly
- model: open and close
- dt:type: Defines data type
- dt qualifies data types

Child Elements of ElementType

- AttributeType: defines a type of attribut local to the element
- attribute: defines attribute of element
- datatype: Specifies the data type
- element: Specifies the child element by name
- group: Groups related elements and denes their order and frequency
- description: Provides description of ElementType

Element AttributeType

- AttributeType defines attributes of an element
- AttributeType element has following attributes
- default: Specifies attributes default value
- dt:type: Defines element's data type
- Some of the data types are int, float, enumeration, date, time, string, entity, id, idref, ...

Element AttributeType (Contd...)

- Name space dt qualifies data types
- dt:values: Contains an enumeration data type values
- name: The attribute name.It is required
- required : Whether attribute is required
- The valid values are yes or no
- To indicate that an element has an attribute, element attribute is used
- The attributes of attribute are: default, type, required

element Element

- element is used to specify an occurrence of a particular type of element
- Defines the placement of element defined by ElementType
- Three attributes associated with element
- type, minoccurs, maxoccurs
- type: This must be an element type defined using ElementType
- Other two attributes are optional

group Element

- Used organize elements in groups
- One may specify frequency and order
- Attributes: order, minoccurs, maxoccurs
- Contained elements: element

Example of the group element

```
<ElementType name="stockitem">
  <group order="one">
    <element type="catalognumber" />
    <element type="partnumber" />
    <element type="itemnumber" />
  </group>
  <element type="description" />
  <element type="quantity" />
</ElementType>
```

Example AttributeType and attribute

```
<?xml version = "1.0"?>
<!-- Fig. 7.10 : contact-schema.xml -->
<!-- Defining attributes -->
<Schema name=mySchema
xmlns="urn:schemas-microsoft-com:xml-data">
    <ElementType name="contact" content="eltOnly"
    order="seq" mode ="closed">
        <AttributeType name="owner" required="yes"/>
        <attribute type="owner"/>
        <element type="name"/>
        <element type="address1"/>
        <element type="address2" minOccurs="0"
        maxOccurs="1"/>
        <element type="city"/>
```

Example AttributeType and attribute (Contd...)

```
    <element type="state"/>
    <element type="zip"/>
    <element type="phone" minOccurs="0"
    maxOccurs="*"/>
</ElementType>
<ElementType name="name" content="textOnly"
model="closed"/>
<ElementType name="address1" content="textOnly"
model="closed"/>
<ElementType name="address2" content="textOnly"
model="closed"/>
```

Example AttributeType and attribute (Contd...)

```
    <ElementType name="city" content="textOnly"
    model="closed"/>
    <ElementType name="state" content="textOnly"
    model="closed"/>
    <ElementType name="zip" content="textOnly"
    model="closed"/>
    <ElementType name="phone" content="textOnly"
    model="closed">
        <AttributeType name="location" default="home"/>
        <attribute type="location"/>
    </ElementType>
</Schema>
```

An Example Document Conforming to Schema

```
<?xml version = “1.0”?>
<!-- contact.xml -->
<!-- A contact list marked up as XML -->
<contact owner=“Bob Smith”
xmlns=“x-schema:contact-schema.xml”>
    <name>Jane Doe</name>
    <address1>123 Main St.</address1>
    <city>Sometown</city>
    <state>Somestate</state>
    <zip>12345¡/zip>
    <phone>617-555-1234</phone>
    <phone location = ”work”>978-555-4321</phone>
</contact>
```

Data Types

- Data type specifies the type of the content of an element or value of an attribute
- This can not be specified with DTD
- Name space prefix is dt
- URI of dt is urn:schema-microsoft-com:datatypes
- For more details: http://msdn.microsoft.com/en-us/library/ms256049.aspx

Data Types (Contd...)

- boolean
- char, string, int, oat
- date, time
- id, idref, and enumeration

An Example Schema with Data Types

```
<?xml version = "1.0"?>
<!-- id-schema.xml -->
<!-- Using datatype ID -->
<Schema name= mySchema
xmlns="urn:schemas-microsoft-com:xml-data"
xmlns:dt="urn:schemas-microsoft-com:datatypes">
    <ElementType name="bookstore" content="eltOnly"
    order="many" model="closed">
        <element type="shipping"/>
        <element type="book"/>
    </ElementType>
    <ElementType name="shipping" content="eltOnly"
    order="seq" model="closed">
```

An Example Schema with Data Types (Contd...)

```
    <AttributeType name="shipID" dt:type="id"
    required="yes"/>
    <attribute type="shipID"/>
    <element type="duration"/>
  </ElementType>
  <ElementType name="duration" content="textOnly"
  model="closed" dt:type="date"/>
  <ElementType name="book" content="textOnly"
  model="closed" dt:type="string">
    <AttributeType name="shippedBy" dt:type="idref"/>
    <attribute type="shippedBy"/>
  </ElementType>
</Schema>
```

A Document Conforming id-schema.xml

```
<?xml version = "1.0"?>
<!-- id.xml -->
<!-- Demonstrating ID and IDREF -->
<bookstore xmlns = "x-schema:id-schema.xml">
   <shipping shipID = "s1">
      <duration>2000-08-01</duration>
   </shipping>
   <shipping shipID = "s2">
      <duration>2000-08-20</duration>
   </shipping>
```

A Document Conforming id-schema.xml (Contd...)

```
    <book shippedBy = “s1”>
        Java How to Program 3rd edition.
    </book>
    <book shippedBy = “s2”>
        C How to Program 3rd edition.
    </book>
    <book shippedBy = “s2”>
        C++ How to Program 3rd edition.
    </book>
</bookstore>
```

Schema for Book Reatiler's Inventory

```
<?xml version = "1.0"?>
<!-- inventory-schema.xml -->
<!-- Data type example -->
<Schema name=anotherSchema
xmlns="urn:schemas-microsoft-com:xml-data"
xmlns:dt="urn:schemas-microsoft-com:datatypes">
  <ElementType name="inventory" content="eltOnly"
  model="closed">
    <element type="book" minOccurs="0" maxOccurs="*"/>
  </ElementType>
  <ElementType name="book" content="eltOnly" order="seq"
  model="closed">
```

Schema for Book Reatiler's Inventory (Contd...)

```
    <AttributeType name="isbn" dt:type="string"
    required="yes"/>
    <attribute type="isbn"/>
    <AttributeType name="inStock" dt:type="enumeration"
    dt:values="yes no" default="no"/>
    <attribute type="inStock"/>
    <element type="name"/>
    <element type="price"/>
    <group order="one">
        <element type="quantity"/>
        <element type="available"/>
    </group>
</ElementType>
```

Schema for Book Reatiler's Inventory (Contd...)

```
    <ElementType name="name" content="textOnly"
    model="closed" dt:type="string"/>
    <ElementType name="price" content="textOnly"
    model="closed" dt:type="float"/>
    <ElementType name="quantity" content="textOnly"
    dt:type="int" model="closed"/>
    <ElementType name="available" content="textOnly"
    dt:type="date" model="closed"/>
</Schema>
```

Inventory Document Conforming to Schema

```
<?xml version = "1.0"?>
<!-- inventory.xml -->
<!-- Data type example -->
<inventory xmlns = "x-schema:inventory-schema.xml">
  <book isbn = "0-13-012507-5" inStock = "yes">
    <name>Java How to Program 3/e</name>
    <price>68.00</price>
    <quantity>200</quantity>
  </book>
    <book isbn = "0-13-028418-1" inStock = "no">
    <name>Perl How to Program</name>
    <price>68.00</price>
    <available>2000-12-15</available>
  </book>
</inventory>
```

W3C XML Schema

- For more detail: www.w3.org/XML/Schema
- File extension used: .xsd
- Name Space used: xsd(prefix)
- URI: http://www.w3.org/2000/10/XMLSchema
- Root element: schema

XSD Schema Overview

- Hierarchy of data types
- Built-in simple data types: string, integer, ...
- User defined complex data types
- Use built-in or complex data types to define elements and attributes
- Defines data types for root element, for its children, and so on

Software Support

- From Microsoft: Tools in .NET Framework
- MSXML has upgraded to support XSD
- Appache parsers
- Most of the major software vendors

XSD Data Types

- Every element and attribute declared in schema must have a type
- Simple types and Complex types
- Complex types may contain elements and attributes
- Simple type may not contain element and attributes
- Built-in simple data types: Atomic and non-atomic
- string, byte, integer, int, long, short, anyType, anyURI

User Defined Simple Data Types

- You do not create a new atomic type
- Customizing the existing atomic types
- Customization by putting restrictions
- Restrictions are called: Constraining facets
- Some facets: enumeration, length, minLength, maxLength, minExclusive, ...
- You may customize already customized data type

Example 1: User Defined Simple Type

```
<xsd:simpleType name="registrationNumber">
    <xsd:restriction base="xsd:unsignedInt">
        <xsd:minInclusive value="2009001"/>
        <xsd:maxInclusive value="2009499"/>
    </xsd:restriction>
</xsd:simpleType>
```

Example 2: User Defined Simple Type

```
<xsd:simpleType name="holidays">
    <xsd:restriction base="xsd:date">
        <xsd:minInclusive value="2010-01-01"/>
        <xsd:maxInclusive value="2010-12-31"/>
    </xsd:restriction>
</xsd:simpleType>
```

Constraining Facets

Facet	Description	Applicable To
enumeration	List of specified values	All except boolean
length	Required no. of chars.	string, anyURI
minlength	Min. no. of chars.	string, anyURI
maxlength	Max. no. of chars.	string, anyURI
minExclusive	Must be $>$	Numeric, date & time
maxExclusive	Must be $<$	Numeric, date & time
minInclusive	Must be	Numeric, date & time
maxInclusive	Must be	Numeric, date & time
pattern	A regular expr	All types
totaldigits	Max. no. of digits	All integer types
whitespace	Set to preserve,..	All types

Enumerations and Lists

- Enumeration list: To customize the data types
- Enumeration can be based on any of the simple XSD types
- Exception: boolean
- A list type contains list of two or more individual data types
- Examples are given below

Examples: Enumerations and Lists

```
<xsd:simpleType name="colors">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="red"/>
        <xsd:enumeration value="green"/>
        <xsd:enumeration value="blue"/>
    </xsd:restriction>
</¡xsd:simpleType>

<xsd:simpleType name="listofdates">
    <xsd:list itemType="xsd:date"/>
</xsd:simpleType>
```

Examples: Enumerations and Lists (Contd...)

```
<xsd:simpleType name="holidaysThisyear">
    <xsd:restriction base="xsd:date">
        <xsd:enumeration value="2010-01-14"/>
        <xsd:enumeration value="2010-01-26"/>
        <xsd:enumeration value="2010-12-25"/>
    </xsd:restriction>
</¡xsd:simpleType>
```

Complex Data Types

- A complex data types can contain child elements and/or attributes
- A complex type is defined within a complexType element
- Syntax for definition:
  <xsd:complexType name="name">
  </xsd:complexType>
- The name attribute is optional
- Anonymous definition

Elements for Complex Type Definition

- *all*: Contains two or more elements each of which must appear once or not at all in any order
- *attribute*: Specifies an attribute that the complex type can contain
- *choice*: Contains two or more element and/or group elements and specifies that the complex type must contain one of the enclosed elements
- *element*: Specifies a child element
- *group*: Defines a group of two or more elements
- *sequence*: Specifies a group of child elements that must appear in specified order

Examples: Complex Types

```
<xsd:complexType name="fullname">
   <xsd:element name="firstname" type="xsd:string"/>
   <xsd:element name="lastname" type="xsd:string"/>
</xsd:complexType>
```

Alternative:

```
<xsd:element name="firstname" type="xsd:string"/>
<xsd:element name="lastname" type="xsd:string"/>
<xsd:complexType name="fullname">
   <xsd:element ref="firstname"/>
   <xsd:element ref="lastname" />
</xsd:complexType>
```

element Element

- element: Use to identify an element that a complex type may contain
- name: Attribute defines name of the element
- type: Defines type of the element
- It may be anonymous
- Other attributes: default, minOccurs, maxOccurs

group and sequence

- group: Defines a group of elements
- group element may contain one or more sequence, choice and/or all elements
- group element can occur within complexType, sequence, choice, and restriction
- sequence: A group of elements must appear in specified order
- Attributes: minOccurs, maxOccurs

Examples: group and sequence

```
<xsd:group name="personalinfo">
  <xsd:sequence>
    <xsd:element name="firstname" type="xsd:string"/>
    <xsd:element name="lastname" type="xsd:string"/>
  </xsd:sequence>
</xsd:group>
<xsd:complexType name="person">
  <xsd:group ref="personalinfo"/>
  <xsd:attribute name="citizenship" type="xsd:string"/>
  <!-- other elements -->
</xsd:complexType>
<xsd:sequence minOccurs="min" maxOccurs="max">
  - - -
</xsd:sequence>
```

An Example W3C Schema

```
<?xml version = “1.0”?>
<!-- schema.xsd -->
<!-- Example W3C XML Schema -->
<xsd:schema
xmlns:xsd=“http://www.w3.org/2000/10/XMLSchema”>
  <xsd:element name=“message” type=“xsd:string”/>
  <xsd:element name=“greeting” type=“greetingType”/>
  <xsd:complexType name=“greetingType” content=“mixed”>
    <xsd:element ref=“message”/>
  </xsd:complexType>
```

An Example W3C Schema (Contd...)

```
    <xsd:element name="myMessage" type="myMessageType"/>
    <xsd:complexType name="myMessageType">
        <xsd:element ref="greeting" minOccurs="0"
        maxOccurs="1"/>
        <xsd:element ref= 'message" minOccurs="1"
        maxOccurs="unbounded"/>
    </xsd:complexType>
</xsd:schema>
```