2.5 XML Schemas

- Short introduction to XML Schema
  - W3C Recommendation, 1st Ed. May, 2001; 2nd Ed. Oct, 2004:
    - XML Schema Part 0: Primer (readable non-normative introduction; Recommended)
    - XML Schema Part 1: Structures
    - XML Schema Part 2: Datatypes
  - Also started (but didn’t lead anywhere?):

Advantages of XSDL (1)

- XML syntax
  - schema documents easier to manipulate by programs (than the special DTD syntax)
- Compatibility with namespaces
  - can validate documents using declarations from multiple sources
- Content datatypes
  - 44 built-in datatypes (including primitive Java datatypes, datatypes of SQL, and XML attribute types)
  - mechanisms to derive user-defined datatypes.

Advantages of XSDL (2)

- Element names and content types independent;
  - Compare with
    - DTDs: 1-to-1 correspondence btw. element type names and their content models
    - CFGs: 1-to-1 correspondence btw. nonterminals and their productions
- For example, could define titles
  - of people as “Mr.”/“Mrs.”/“Ms.”, and
  - of chapters as strings

Advantages of XSDL (3)

- Ability to specify uniqueness and keys within selected parts of the document
  - for example, that titles of chapters should be unique; or key attributes of relations
  - uses XPath
- Support for schema documentation
  - element annotation with sub-elements
documentation (for human readers) and appInfo (for applications)
Disadvantages of XSDL

- Complexity of XSDL (esp. of Rec. Part 1) vs. added power
  - > a long learning curve
  - > possible rejection by users
- Possible immaturity of implementations (?)
  - W3C XML Schema Web site mentions ~40 tools or processors
  - Open-source Apache XML parsers (Xerces C++ 2.0 and Xerces 2 Java) claim full XML Schema support
- Alternative schema languages have been suggested, too (See later)

XSDL through Example

- Next: walk-through of an XML schema example
  - from Chapter 2 of the XML Schema Primer
  - Consider modelling purchase orders like below:

```xml
<purchaseOrder orderDate="1999-10-20">
  <shipTo country="US">
    <name>Alice Smith</name>
    <street>123 Maple Street</street>
    <city>Mill Valley</city>
    <state>CA</state>
    <zip>95802</zip>
  </shipTo>
  <items>
    <item partNum="872-9A">
      <productName>Lawnmower</productName>
      <quantity>1</quantity>
      <USPrice>39.98</USPrice>
      <shipDate>1999-05-21</shipDate>
    </item>
  </items>
  </purchaseOrder>
```

purchaseOrder instance continues

```xml
<billTo country="US">
  <name>Robert Smith</name>
  <street>8 Oak Avenue</street>
  <city>Old Town</city>
  <state>CA</state>
  <zip>95819</zip>
</billTo>
</comment>Hurry, my lawn is wild!</comment>
</item>
</items>
</purchaseOrder>

End of the example instance

```xml
</item>
</purchaseOrder>

Next: A schema for such purchase orders

The Purchase Order Schema (1/5)

```xml
<xs:schema xmlns:xsi="http://www.w3.org/2001/XMLSchema">
  <xs:element name="purchaseOrder" type="POrdType"/>
  <xs:complexType name="POrdType">
    <xs:sequence>
      <xs:element name="billTo" type="USAddr"/>
      <xs:element ref="comment" minOccurs="0"/>
      <xs:element name="items" type="Items"/>
    </xs:sequence>
    <xs:attribute name="ordDate" type="xs:date" fixed="US 10/20/99"/>
  </xs:complexType>
</xs:schema>
```

The Purchase Order Schema (2/5)

```xml
<xs:complexType name="USAddr">
  <xs:sequence>
    <xs:element name="name" type="xs:string"/>
    <xs:element name="street" type="xs:string"/>
    <xs:element name="city" type="xs:string"/>
    <xs:element name="state" type="xs:string"/>
    <xs:element name="zip" type="xs:decimal"/>
  </xs:sequence>
  <xs:attribute name="country" type="xs:NMTOKEN" fixed="US"/>
</xs:complexType>
```
The Purchase Order Schema (3/5)

```xml
<xs:complexType name="Item">
  <xs:sequence>
    <xs:element name="item" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

The Purchase Order Schema (5/5)

```xml
<xs:simpleType name="SKU">
  <xs:restriction base="xs:positiveInteger">
    <xs:pattern value="d(31-)(A-Z)2"/>
    <xs:restriction base="xs:decimal">
      <xs:maxExclusive value="100"/>
    </xs:restriction>
  </xs:restriction>
</xs:simpleType>
```

XSDL Content Models

- **Element content of complexType can be regulated using**
  - group elements sequence, choice and all, and
  - occurrence constraint attributes minOccurs and maxOccurs

- **Elements sequence and choice correspond to catenation and alternation (|) in regular expressions**

XSDL occurrence constraints

- **optionality (E?) can be expressed by**
  - minOccurs="0"
- **iteration (E*) can be expressed by**
  - minOccurs="0" and maxOccurs="unbounded"
- **Exactly five occurrences of element A:**
  - minOccurs="5" maxOccurs="5"
- **10 to 900 occurrences of element A:**
  - minOccurs="10" maxOccurs="900"

Regular expression vs an XSDL content model

- **A | B (C D)** could be expressed by

```xml
<xs:choice>
  <xs:element ref="A" maxOccurs="unbounded" />
  <xs:element ref="B" />
  <xs:sequence minOccurs="0" maxOccurs="unbounded">
    <xs:element ref="C" />
    <xs:element ref="D" />
  </xs:sequence>
</xs:choice>
```
Unordered content: the all group

- XSDL all group is a restricted version of the &-connector in SGML
  - E1 B ... & En allows sequences corresponding to any permutation of E1, ..., En
- XSDL restrictions to all:
  - cannot be combined with other content model groups
  - can have only (non-iterative) elements as children
  - can be optional (minOccurs=0) but not repeated (maxOccurs=1)

The all group: An example

- For example
  `<xs:element name="A" />`
  `<xs:element name="B" />`
  `<xs:element name="C" minOccurs="0" />`

  `</all>`

accepts the following element sequences:

- A B C; A C B; B A C; B C A; C A B; C B A;
- A B; and B A;

XML Schema: Summary

- XSDL: an XML-based grammar formalism
  - W3C Recommendation
  - Alternative to the DTD formalism
  - more powerful than DTDs (esp. content and attribute datatypes)
- Rather well accepted in XML industry
  - e.g., to describe messages btw Web services;
    e.g., WSDL (Web Services Description Language, Vers. 2.0, W3C Working Draft, 8/04) requires XSDL support of processors

XML Schema: Future?

- XSDL has also been criticized
  - for overwhelming complexity vs. relatively little added power
  - other XML-based schema languages proposed and implemented, too
  - For example, RELAX NG
    - Draft international ISO/IEC standard
    - Simpler and with a rigorous formal base, and more powerful at certain places (e.g. dependencies of attributes, unordered content)