2.5 XML Schemas

- A short introduction to XML Schema
  - W3C Recommendation, May 2, 2001:
    - XML Schema Part 0: Primer (readable non-normative introduction; Recommended)
    - XML Schema Part 1: Structures
    - XML Schema Part 2: Datatypes
  - Also under development(?):

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)

- Independence of element names and content
types; Compare with
  - DTDs: 1-to-1 correspondence b/w element type
  names and their content models
  - CFGs: 1-to-1 correspondence b/w. 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
  - uses XPath
- Support for schema documentation
  - element annotation with sub-elements
documentation (for human readers) and
appInfo (for applications)

Schema terminology

- Schema (kaavio): a formal description for the
  structure and allowed content of a set of data
  (esp. in databases)
- "XML Schema" is often used for each of ...
  1. XML Schema, the W3C Rec. that defines ...
  2. XML Schema Definition Language (XSDL),
  an XML-based markup language for expressing ...
  3. schema documents, each of which describes a
schema (= DTD) for a set of XML document instances
(This may cause some confusion!)

XSDL built-in types

(Part 2, Chap. 3)

*: XML attribute types

NB: values in documents
represented by strings
Disadvantages of XSDL

- Complexity of XSDL (esp. of Rec. Part II) vs. added power
  - a long learning curve
  - possible rejection by users
- Possible immaturity of implementations (?)
  - W3C XML Schema Web site mentions 40 of 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 Chap. 2 of the XML Schema Primer
  - Consider modelling purchase orders like below:

```
<xs:schema xmlns:xsi="http://www.w3.org/2001/XMLSchema"
  xsi:schemaLocation="http://www.example.com/PO.xsd">
  <xs:element name="purchaseOrder"><xs:complexType>
    <xs:sequence>
      <xs:element name="billTo" type="xs:USAddr"/>
      <xs:element name="item" type="Item"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

```
<billTo country="US">
  <name>Robert Smith</name>
  <street>8 Oak Avenue</street>
  <city>Old Town</city>
  <zip>95818</zip>
</billTo>

<item partNum="526-AA">
  <productName>Lawmower</productName>
  <quantity>1</quantity>
  <USPrice>39.98</USPrice>
  <shipDate>1999-05-21</shipDate>
</item>
```

The Purchase Order Schema (1/5)

```
<xs:schema xmlns="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <xs:element name="purchaseOrder" type="PurchaseOrder"/>
</xs:schema>
```

The Purchase Order Schema (2/5)

```
<xs:complexType name="USAddr">
  <xs:sequence>
    <xs:element name="name" type="xs:string"/>
    <xs:element name="state" type="xs:string"/>
    <xs:element name="city" type="xs:string"/>
    <xs:element name="zip" type="xs:decimal"/>
  </xs:sequence>
  <xs:attribute name="country" type="xs:token" fixed="US"/>
</xs:complexType>
```

purchaseOrder instance continues

```
<item partNum="872-AA">
  <productName>Baby Phone</productName>
  <quantity>1</quantity>
  <USPrice>148.95</USPrice>
  <comment>Only if electric</comment>
</item>
```

End of the example instance

```
<end>
</purchaseOrder>
```

The Purchase Order Schema (3/5)

```
<xs:complexType name="Item">
  <xs:sequence>
    <xs:element name="partNum" type="xs:IDRef"/>
    <xs:attribute name="comment" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

The Purchase Order Schema (4/5)

```
<xs:complexType name="PurchaseOrder">
  <xs:sequence>
    <xs:element name="billTo" type="xs:USAddr"/>
    <xs:element name="item" type="Item"/>
  </xs:sequence>
</xs:complexType>
```

The Purchase Order Schema (5/5)

```
<xs:element name="purchaseOrder" type="PurchaseOrder"/>
```

The Purchase Order Schema (6/5)

```
<xs:complexType name="Item">
  <xs:sequence>
    <xs:element name="partNum" type="xs:IDRef"/>
    <xs:attribute name="comment" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

End of the example instance

```
<end>
</purchaseOrder>
```

The Purchase Order Schema (7/5)

```
<xs:complexType name="PurchaseOrder">
  <xs:sequence>
    <xs:element name="billTo" type="xs:USAddr"/>
    <xs:element name="item" type="Item"/>
  </xs:sequence>
</xs:complexType>
```

The Purchase Order Schema (8/5)

```
<xs:element name="purchaseOrder" type="PurchaseOrder"/>
```

The Purchase Order Schema (9/5)

```
<xs:complexType name="Item">
  <xs:sequence>
    <xs:element name="partNum" type="xs:IDRef"/>
    <xs:attribute name="comment" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

End of the example instance

```
<end>
</purchaseOrder>
```

The Purchase Order Schema (10/5)

```
<xs:complexType name="PurchaseOrder">
  <xs:sequence>
    <xs:element name="billTo" type="xs:USAddr"/>
    <xs:element name="item" type="Item"/>
  </xs:sequence>
</xs:complexType>
```

The Purchase Order Schema (11/5)

```
<xs:element name="purchaseOrder" type="PurchaseOrder"/>
```

The Purchase Order Schema (12/5)

```
<xs:complexType name="Item">
  <xs:sequence>
    <xs:element name="partNum" type="xs:IDRef"/>
    <xs:attribute name="comment" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```
The Purchase Order Schema (3/5)

```xml
<xs:complexType name="Item">
  <xs:sequence>
    <xs:element name="item" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>  
        anonymous<del type="for quantity"></del>type for item
      </xs:complexType>
    </xs:element>
    <xs:element name="quantity">
      <xs:complexType>
        <xs:simpleType>
          <xs:restriction base="xs:integer" />
        </xs:simpleType>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
</xs:complexType>
```

The Purchase Order Schema (4/5)

```xml
<xs:complexType name="USPrice">
  <xs:attribute name="currency" type="xs:string" />
</xs:complexType>
```

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

Regular expression vs an XSDL content model

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

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

- XSDL all group is a restricted version of the &-connector in SGML
  - E1 & ... & 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
  ```xml
  <xs:all>
    <xs:element name="A" />
    <xs:element name="B" />
    <xs:element name="C" minOccurs="0" />
  </xs: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., WSDL (Web Services Description Language, Vers. 2.0, W3C Working Draft, 11/03) requires XSDL support for describing messages exchanged btw web services

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 more formally based than XSDL, yet more powerful at certain places [e.g., dependencies of attributes, unordered content]