Introduction to XML

Yanlei Diao
UMass Amherst
April 17, 2008
Structure in Data Representation

- Relational data is highly structured
  - structure is defined by the schema
  - good for system design
  - good for precise query semantics / answers

- Structure can be limiting
  - data exchange hard: integration of diff schema
  - authoring is constrained: schema-first
  - querying constrained: must know schema
  - changes to structure not easy
Need for A New Data Model

Loose (and rich) structure
- Integration of structured, but heterogeneous data sources
- Textual data with tags and links
- Combination of data models
- Evolving and irregular structures
XML: Universal Data Exchange Format

- XML is the confluence of many factors:
  - Databases needed a more flexible interchange format.
  - Data needed to be generated and consumed by applications.
  - The Web needed a more declarative format for data.
  - Documents needed a mechanism for extended tags.

- XML was originally proposed for online publishing, is becoming the wire format for data exchange.

- W3C Recommendation:
  http://www.w3.org/TR/REC-xml/
From HTML to XML

HTML describes the presentation.
<h1>Bibliography</h1>

<p><i>Foundations of Databases</i>  
Abiteboul, Hull, Vianu  
Addison Wesley, 1995</p>

<p><i>Data on the Web</i>  
Abiteboul, Buneman, Suciu  
Morgan Kaufmann, 1999</p>
XML

<book>
  <title> Foundations… </title>
  <author> Abiteboul </author>
  <author> Hull </author>
  <author> Vianu </author>
  <publisher> Addison Wesley </publisher>
  <year> 1995 </year>
</book>
...

XML describes the content!
XML: Syntax & Typing
XML Syntax

- Tags: book, title, author, ...
  - start tag: <book>
  - end tag: </book>
- Elements: <book>...</book>,<author>...</author>
  - elements are nested
  - empty element: <red></red>, abbrv. <red/>
- An XML document: single root element

An XML document is well formed if it has matching tags
XML Syntax

<book price="55" currency="USD">
  <title>Foundations of Databases</title>
  <author>Abiteboul</author>
  ...
  <year>1995</year>
</book>

Attributes are alternative ways to represent data.
XML Syntax

Oids and references in XML are just syntax.
XML Semantics: a Tree!

```
<data>
  <person id="0555">
    <name>Mary</name>
    <address>
      <street>Maple</street>
      <no>345</no>
      <city>Seattle</city>
    </address>
  </person>
  <person>
    <name>John</name>
    <address>Thailand</address>
    <phone>23456</phone>
  </person>
</data>
```

Order matters! IDREF will turn it to a graph.
XML Data

- XML is self-describing
- Schema elements become part of the data
  - Relational schema: persons(name, phone)
  - In XML <persons>, <name>, <phone> are part of the data, and are repeated many times
- Consequence: XML is much more flexible

Some real data:
Relational Data as XML

<table>
<thead>
<tr>
<th>name</th>
<th>phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>3634</td>
</tr>
<tr>
<td>Sue</td>
<td>6343</td>
</tr>
<tr>
<td>Dick</td>
<td>6363</td>
</tr>
</tbody>
</table>

**XML:**

```xml
<person>
  <row>
    <name>John</name>
    <phone>3634</phone>
  </row>
  <row>
    <name>Sue</name>
    <phone>6343</phone>
  </row>
  <row>
    <name>Dick</name>
    <phone>6363</phone>
  </row>
</person>
```
XML is Semi-structured Data

- Missing attributes:

```
<data>
<person>  <name>John</name>
         <phone>1234</phone>
</person>
<person>  <name>Joe</name>
</person>
</data>
```

- Could represent in a table with nulls

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>phone</td>
</tr>
<tr>
<td>John</td>
<td>1234</td>
</tr>
<tr>
<td>Joe</td>
<td>-</td>
</tr>
</tbody>
</table>

← no phone!
XML is Semi-structured Data

- Repeated attributes

```xml
<person>
  <name>Mary</name>
  <phone>2345</phone>
  <phone>3456</phone>
</person>
```

- Impossible in tables: nested collections (non 1NF)

```
<table>
<thead>
<tr>
<th>name</th>
<th>phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mary</td>
<td>2345</td>
</tr>
<tr>
<td></td>
<td>3456</td>
</tr>
</tbody>
</table>
```

← two phones!
XML is Semi-structured Data

- Attributes with different types in different objects

```
<data>
  <person>
    <name>
      <first>John</first>
      <last>Smith</last>
    </name>
    <phone>1234</phone>
  </person>
  <person>
    <name>M. Carey</name>
    <phone>3456</phone>
  </person>
</data>
```

- Mixed content:
  - `<db>` contains both `<book>`s and `<publisher>`s

← structured name!

← unstructured name!
Data Typing in XML

- Data typing in the relational model: schema
- Data typing in XML
  - Much more complex
  - Typing restricts valid trees that can occur
    - theoretical foundation: tree languages
  - Practical methods:
    - DTD (Document Type Definition)
    - XML Schema
Document Type Definitions (DTD)

- Part of the original XML specification
- To be replaced by XML Schema
  - Much more complex
- An XML document may have a DTD
- XML document:
  - well-formed = if tags are correctly closed
  - Valid = if it has a DTD and conforms to it
- Validation is useful in data exchange
<!DOCTYPE company [ 
  <!ELEMENT company ((person|product)*)> 
  <!ELEMENT person (ssn, name, office, phone?)> 
  <!ELEMENT ssn (#PCDATA)> 
  <!ELEMENT name (#PCDATA)> 
  <!ELEMENT office (#PCDATA)> 
  <!ELEMENT phone (#PCDATA)> 
  <!ELEMENT product (pid, name, description?)> 
  <!ELEMENT pid (#PCDATA)> 
  <!ELEMENT description (#PCDATA)> 
]>
Example of valid XML document:

```xml
<company>
    <person>
        <ssn>123456789</ssn>
        <name>John</name>
        <office>B432</office>
        <phone>1234</phone>
    </person>
    <person>
        <ssn>987654321</ssn>
        <name>Jim</name>
        <office>B123</office>
    </person>
    <product>...</product>
    ...
</company>
```
DTD: The Content Model

- Complex = a regular expression over other elements
- Text-only = #PCDATA
- Empty = EMPTY
- Any = ANY
- Mixed content = (#PCDATA | A | B | C)*
DTD: Regular Expressions

sequence

<!ELEMENT name (firstName, lastName))>

optional

<!ELEMENT name (firstName?, lastName)

Kleene star

<!ELEMENT person (name, phone*)

alternation

<!ELEMENT person (name, (phone|email)))
Attributes in DTDs

<!ELEMENT person (ssn, name, office, phone?)>
<!ATTLIST person age CDATA #REQUIRED>

<person age="25">
  <name> ....</name>
  ...
</person>
Attributes in DTDs

```xml
<!ELEMENT person (ssn, name, office, phone?)>
<!ATTLIST person age CDATA #REQUIRED
    id ID #REQUIRED
    manager IDREF #REQUIRED
    manages IDREFS #REQUIRED>

<person age="25"
    id="p29432"
    manager="p48293" manages="p34982 p423234">
    <name> ....</name>
    ...
</person>
```
Attributes in DTDs

Types:
- CDATA = string
- ID = key
- IDREF = foreign key
- IDREFS = foreign keys separated by space
- (Monday | Wednesday | Friday) = enumeration
Attributes in DTDs

Kind:

- #REQUIRED
- #IMPLIED = optional
- value = default value
- value #FIXED = the only value allowed
Using DTDs

- Must include in the XML document
- Either include the entire DTD:
  - `<!DOCTYPE rootElement [ ....... ]>`
- Or include a reference to it:
  - `<!DOCTYPE rootElement SYSTEM “http://www.mydtd.org”>`
- Or mix the two... (e.g. to override the external definition)