CS2Bh: Current Technologies

Introduction to XML and Relational Databases

Spring 2005

XML Basics

History: SGML, HTML, XML

SGML: Standard Generalized Markup Language

-- Charles Goldfarb, ISO 8879, 1986

- DTD (Document Type Definition)
- powerful and flexible tool for structuring information, but
  - complete, generic implementation of SGML proven extremely difficult
  - tools for working with SGML documents proven expensive
- two sub-languages that have outpaced SGML:
  - HTML: HyperText Markup Language (Tim Berners-Lee, 1991). Describing presentation.
  - XML: eXtensible Markup Language, W3C, 1998. Describing content.

From HTML to XML

HTML is good for presentation (human friendly), but does not help automatic data extraction by means of programs (not computer friendly).

Why? HTML tags:

- ✓ predefined and fixed
- ✓ describing display format, not the structure of the data.

```
<h3>  George Bush   </h3>
<b>    Taking Eng 055  </b> <br>
<em> GPA: 1.5  </em> <br>
<h3>  Eng 055   </h3>
<b>   Spelling    </b>
```


XML: a first glance

XML tags:

- ✓ user defined
- ✓ describing the structure of the data

```
<school>
    <student    id = “011”>
        <name>
            <firstName>George</firstName>   <lastName>Bush</lastName>
        </name>
        <taking> Eng 055 </taking>
        <GPA>   1.5  </GPA>
    </student>
    <course    cno = “Eng 055”>
          <title> Spelling </title>
   </course>
</school>
```

XML vs. HTML

- ✓ user-defined new tags, describing structure instead of display
- ✓ structures can be arbitrarily nested (even recursively defined)
- ✓ optional description of its grammar (DTD) and thus validation is possible

What is XML for?

- ✓ The prime standard for data exchange on the Web
- ✓ A uniform data model for data integration

XML presentation:

- ✓ XML standard does not define how data should be displayed
- ✓ Style sheet: provide browsers with a set of formatting rules to be applied to particular elements
  - CSS (Cascading Style Sheets), originally for HTML
  - XSL (eXtensible Style Language), for XML

Tags and Text

- ✓ XML consists of tags and text

```
<course  cno = "Eng 055">
    <title> Spelling </title>
</course>
```

- ✓ tags come in pairs: markups
  - start tag, e.g., <course>
  - end tag, e.g., </course>
- ✓ tags must be properly nested
  - <course> <title> … </title> </course> -- good
  - <course> <title> … </course> </title> -- bad
- ✓ XML has only one "basic" type: text, called PCDATA (Parsed Character DATA)

XML Elements

- Element: the segment between an start and its corresponding end tag
- subelement: the relation between an element and its component elements.

```
<person>
  <name> Wenfei Fan </name>
  <tel> (215) 204-6485 </tel>
  <email> wenfei@inf.ed.ac.uk </email>
  <email> wenfei@research.bell-labs.com </email>
  </person>
```


Nested Structure

- nested tags can be used to express various structures, e.g., “records”:

```
<person>
    <name> Wenfei Fan </name>
    <tel> (908) 5820424 </tel>
    <email> wenfei@inf.ac.ed.uk </email>
    <email> wenfei@research.bell-labs.com </email>
</person>
```

- a list: represented by using the same tags repeatedly:

```
<person> … </person>
<person> … </person>
...
```

Ordered Structure

XML elements are ordered!

- How to represent sets in XML?
- How to represent an unordered pair (a, b) in XML?
- Can one directly represent the following using a set of records?
  - <person> … </person>
    <person> … </person> …
  - <person>
    <name> Wenfei Fan </name>
    <tel> (908) 5820424 </tel>
    <email> wenfei@inf.ac.ed.uk </email>
    <email> wenfei@research.bell-labs.com </email>
    </person>

Exercise

Represent the following as XML elements

- A student record has name, student id, email and gpa; and name is in turn a record with attributes first-name and last-name
- A course record has attributes course number, name, and credit

```
<student>
        <name> <first-name> George </first-name>
                <last-name>  Bush      </last-name>
         </name>
        <sid> 09324343 </sid>
         <email> dump@Whitehouse.gov </email>
         <gpa> 1.2 </gpa>
</student>
```

Special elements

- root element: an XML document consists of a single element called the **root** element, e.g.,

```
<db>
    <person> … </person>
    <person> … </person> ...
</db>
```

- empty element: special element indicating non-textual content,
  - <giggle></giggle> or simply <giggle/>
  - an element may carry attributes

    <image img="picture.gif" />

    to be interpreted by applications

XML attributes

An start tag may contain attributes describing certain "properties" of the element (e.g., dimension or type)

```
<picture>
    <height dim="cm"> 2400</height>
    <width dim="in"> 96 </width>
    <data encoding="gif"> M05-+C$ … </data>
</picture>
```

References (meaningful only when a DTD is present):

```
<person id = "011"   pal="012">
    <name> George Bush</name>
</person>
<person id = "012"   pal="011">
    <name> Saddam Hussein </name>
</person>
```

The "structure" of XML attributes

- ✓ XML attributes cannot be nested -- flat
- ✓ the names of XML attributes of an element must be unique.

  one can't write <person pal="Blair" pal="Saddam"> ...
- ✓ XML attributes are not ordered

```
<person  id = "011"  pal="012">
  <name> George Bush</name>
</person>
```

is the same as

```
<person  pal="012"  id = "011">
  <name> George Bush</name>
</person>
```

- ✓ Attributes vs. subelements: unordered vs. ordered, and
  - attributes cannot be nested (flat structure)
  - subelements cannot represent references

Well-formed XML documents

a document is well-formed if it satisfies two constraints (when only elements and attributes are considered):

- ✓ tags have to nest properly
- ✓ attributes have to be unique

Very weak constraints: it does little more than ensure that XML data will parse into a labeled tree

Other XML constructs

- XML declaration: version information must be provided:

  <?xml version= '1.0'?>

- comments:

  <!-- This is a comment. Processors will ignore me -->

- CDATA: escape block containing characters that would otherwise be recognized as markup:

  <![CDATA[content]]>

  e.g., <![CDATA[<start> this is not an element </end>]]>

- PI (Processing Instruction): for applications, not parsers

  <?instruction?>

  Example: associate an XSL style sheet with XML document

  <?xml:stylesheet

  href="http://homepages.inf.ed.ac.uk/~wenfei/book.xsl" type="text/xsl" ?>

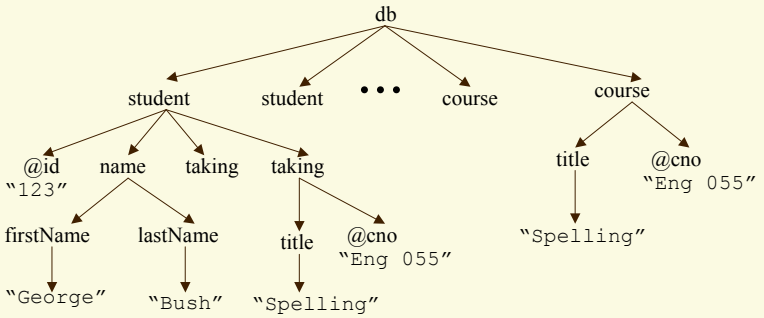
A complete XML document

```
<?xml version= '1.0'?>
<!DOCTYPE book PUBLIC "~wenfei/school.dtd">
<?xml:stylesheet href="school.xsl" type="text/xsl"?>
<school>              <!-- school database -->
   <student   id = "011">
      <name>
         <firstName>George</firstName>  <lastName>Bush</lastName>
      </name>
      <taking> Eng 055 </taking>
      <GPA>  1.5 </GPA>
   </student>
   <course   cno = "Eng 055">
      <title> Spelling </title>
  </course>
</school>
```

The XML tree model

An XML document is modeled as a node-labeled ordered tree.

- ✓ Element node: typically internal, with a name (tag) and children (subelements and attributes), e.g., student, name.
- ✓ Attribute node: leaf with a name (tag) and text, e.g., @id.
- ✓ Text node: leaf with text (string) but without a name.



Does an XML document always have a unique tree representation?

Summary and Review

XML basics – what to understand and remember:

- ✓ Elements
- ✓ Attributes
- ✓ Text nodes: PCDATA
- ✓ The tree model

Review questions:

- ✓ Why XML instead of HTML?
- ✓ When to use elements instead of attributes?
- ✓ How to represent a nested structure in XML?
- ✓ Can you represent a list in XML? A set of numbers? A set of records?

Tutorial: http://www.w3schools.com/xml/default.asp