Lecture 15: Querying XML

Friday, October 27, 2000

An Example of XML Data

```
<bib>
    <book>  <publisher> Addison-Wesley </publisher>
            <author> Serge Abiteboul </author>
            <author> <first-name> Rick </first-name>
                     <last-name> Hull </last-name>
            <author> Victor Vianu </author>
            <title> Foundations of Databases </title>
             <year> 1995 </year>
    </book>
    <book price="55">
            <publisher> Freeman </publisher>
            <author> Jeffrey D. Ullman </author>
            <title> Principles of Database and Knowledge Base Systems
            </title>
            <year> 1998 </year>
    </book>
</bib>
```

XPath

- Syntax for XML document navigation and node selection
- A *recommendation* of the W3C (i.e. a standard)
- Building block for other W3C standards:
  - XSL Transformations (XSLT)
  - XML Link (XLink)
  - XML Pointer (XPointer)

XPath

/bib/book/year

Result: <year> 1995 </year>
<year> 1998 </year>

/bib/paper/year

Result: empty (there were no papers)

XPath

//author

Result:<author> Serge Abiteboul </author>
<author> <first-name> Rick </first-name>
<last-name> Hull </last-name>
</author>
<author> Victor Vianu </author>
<author> Jeffrey D. Ullman </author>

/bib//first-name
Result: <first-name> Rick </first-name>

XPath

/bib/book/author/text()

Result: Serge Abiteboul
Jeffrey D. Ullman

Rick Hull doesn't appear because he has firstname, lastname

XPath

//author/*

Result: <first-name> Rick </first-name>
<last-name> Hull </last-name>

* Matches any element

XPath

/bib/book/@price

Result: “55”

@price means that price is has to be an attribute

XPath

/bib/book/author[firstname]

Result: <author> <first-name> Rick </first-name>
<last-name> Hull </last-name>
</author>

XPath

/bib/book[@price < “60”]

/bib/book[author/@age < “25”]

/bib/book[author/text()]

XPath Expressions

bib	matches a bib element
*	matches any element
/	matches the root element
/bib	matches a bib element under root
bib/paper	matches a paper in bib
bib//paper	matches a paper in bib, at any depth
//paper	matches a paper at any depth
paper|book	matches a paper or a book
@price	matches a price attribute
bib/book/@price	matches price attribute in book, in bib

bib/book/[@price<“55”]/author/lastname matches…

Query Language

- First research query language: XML-QL (1998)
- The W3C started a WG for a standard XML query language … still working
- We will see here Quilt that borrows from:
  - XML-QL
  - Xpath
  - SQL

Quilt

List all titles of books published by Morgan Kaufmann in 1998:

```
FOR      $b IN document("bib.xml")/book
WHERE $b/publisher = "Morgan Kaufmann"
    AND $b/year = "1998"
RETURN $b/title
```

Quilt

- Find all names with a firstname and lastname; group them in a <name>

```
FOR      $a IN document("bib.xml")//author,
         $f IN $a/firstName,
         $l IN $a/lastName
RETURN <name> <fn> $f </fn>
                  <ln> $l </ln>
          </name>
```

Quilt

- Retrieve the titles of the books written by Laing before 1967, together with their reviews.

```
FOR  $b in document("bib.xml")//book[@year<1967],
       $r in document("reviews.xml")//review
WHERE  $b/authors/lastname="Laing"  and $b/@ISBN=$r/@ISBN
RETURN
     <resultBook  ISBN=$b/@ISBN>
          <title> $b/title/text() </title>,
           $r
     </resultBook>
```

Quilt

- Retrieve the titles of the books written by Laing before 1967 together with their reviews.

```
FOR  $b in document("input.xml")//book[@year<1967]
  LET  $R = document("input.xml")//review[@isbn=$b/@isbn]
WHERE  $b/authors/lastname="Laing"
RETURN
      <resultBook  ISBN=$b/@ISBN>
           <resultTitle> $t </resultTitle>
            <bookReviews> $R </bookReviews>
     </resultBook>
```

QUILT

- List all authors that published both in 1998 and 1999

```
FOR      $a IN distinct(document("bib.xml")/book/author,
WHERE contains(document("bib.xml")/book[year=1998]/author, $a)
   AND contains(document("bib.xml")/book[year=1999]/author, $a)
RETURN $a
```

XSL

- Aka XSLT
- A recommendation of the W3C (standard)
- Initial goal: translate XML to HTML
- Became: translate XML to XML
  - HTML is just a particular case

XSL Templates and Rules

- query = collection of template rules
- template rule = match pattern + template

Retrieve all book titles:

```
<xsl:template> <xsl:apply-templates/> </xsl:template>

<xsl:template match = "/bib/*/title">
      <result>  <xsl:value-of/>  </result>
</xsl:template>
```

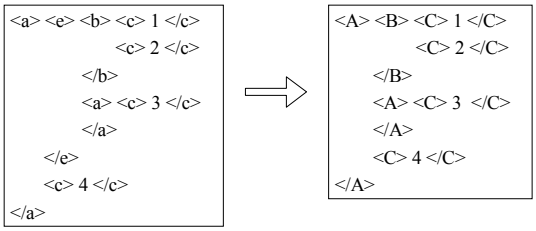
Flow Control in XSL

```
<xsl:template> <xsl:apply-templates/> </xsl:template>

<xsl:template match="a">  <A><xsl:apply-templates/></A>
</xsl:template>

<xsl:template match="b">  <B><xsl:apply-templates/></B>
</xsl:template>

<xsl:template match="c">  <C><xsl:value-of/></C>
</xsl:template>
```



XSLT

```
<xsl:template> <xsl:apply-templates/> </xsl:template>

<xsl:template match="a">  <A><xsl:apply-templates/></A>
                          <A><xsl:apply-templates/></A>
</xsl:template>
```

XSLT

- What is the output on:

<a> <a> <a>

?

- Answer: