3 XML Data Management and Bioinformatics applications

3.1 Introduction to XML
3.2 XML syntax
3.3 Document Type Definitions
3.4 Namespaces, schemas and more
3.5 Usage: Logical – physical Layout
3.6 XML in Bioinformatics: examples
3.7 Querying XML documents: XPath
3.8 XML Data Management: mapping documents to relations
3.9 Note on Information Integration using XML

What is XML?
- Acronym for eXtensible Markup Language
- Syntax for structuring data and documents in human-readable form
- THE "Syntax of the WEB"
- Meta language for defining data description languages called applications, e.g. GAME, BSML, ...
- Basis for many extensions
  - Namespaces
  - Stylesheets
  - Hyperlinks
  - Schemata
- Standardized by W3C
http://www.w3.org/TR/REC-xml

What XML is NOT..
- No protocol
  - Language for describing data
  - Used as data format in protocols
  - Protocols may be syntactically defined by XML
- No programming language
  - XML documents may contain code fragments
  - New languages allow for XML – code as part of the language (Xen, a MS extension of C#)
  - Some XML extensions with superimposed PL semantics, rule semantics in XSLT
- No magic semantics
  - Interpretation by humans, applications, standards derived from XML

Why XML?
- … not a question any more, since widely adopted
- Simple
- Extensible
- Easy to process
- Easy to generate
- Data interchange critical for networked applications

"XML will be the ASCII of the Web: basic, essential, unexciting" Tim Bray
... it is already

XML markup: example
<?xml version="1.0"?>
<seq id="my_seq" name="NUCLEAR RIBONUCLEOPROTEIN">
  <Share>
    <database>SWISS-PROT</database>
    <unique_id>P09651</unique_id>
  </Share>
  <residues type="aa">
    SKSESPKPEPEQLRKLFIGGLSFETTDESLHSHFQWNLQDGVVMQGIPKRES
    RGFQVYVTYVTEEGAAEMARKHVRGFDVXEEBINBNSGRRPAGASYHEKI
    FVGIGIKEMEEEDOYFTQOKXIEVENMTNGSSGHRMGAFVVTDGDSVDG
    KVIQKMSETLYNHHENKVSTQAKKQEMAKTNQGQDSSGDQEGGMSGSMG
    DRPFGQGNMGRGGSPGGGGGDGGYVGGGGYTMGGNMDGGGGGGS
    YSWSQQTQSGSSTGSGSGSGSGSGSSTGSGSGSGSGSGSGSGSGSSTGSG
    wsgswnfdsgsfnfgndsgsfnfgndsfndgfsdngfdnfdnfdn
  </residues>
</seq>
XML example

- Graphical representation

XML documents
- tree structured
- Data and metadata in the same document

(as opposed to RDBS / ODBS / ...)

Another Example

- Semantics??

<?xml version="1.0"?>
<pd>
  <b>
    <d>22</d>
    <m>5</m>
    <y>70</y>
  </b>
  <ec>green</ec>
  <pc>CB2 2EZ</pc>
</pd>

<?xml version="1.0"?>
<pd>
  <b>
    <d>22</d>
    <m>5</m>
    <y>70</y>
  </b>
  <ec>green</ec>
  <pc>CB2 2EZ</pc>
</pd>

Third example

<Orders>
  <SalesOrder SONumber="12345">
    <Customer CustNumber="543">
      <CustName>ABC Industries</CustName>
      <Street>123 Main St.</Street>
      <City>Chicago</City>
      ....
    </Customer>
    <Line LineNumber="1">
      <Part PartNumber="123">
        <Description>
          <p><b>Turkey wrench</b>: Stainless steel, one-piece construction, lifetime guarantee.</p>
        </Description>
        <Price>9.95</Price>
      </Part>
      <Quantity>10</Quantity>
    </Line>
    ....
  </SalesOrder>
</Orders>

XML Usage

- Basic types of XML usage
  - Document centric (document oriented)
    - structuring a digital document, including logical layout
    - primary focus of SGML - predecessor of XML
  - Data centric
    - Description of data in a self describing form for later processing
  - Distinction?
    - not always clear
    - data centric: database / query oriented
    - document: layout ... but on a logical level

Document or data centric?

Insulin receptor sequence
3.2 XML Syntax

- One, and only one, root element
- Sub-elements must be balanced and properly nested
- Attributes are optional
- Attribute values must be quoted
- Empty tag: `< Leer/ >`, comment `<-- .... -->`
- XML is case-sensitive
- `<tag>` and `<TAG>` are not the same type of element
- Special characters for `<`, `>`, `&`, `quot`
- Document always begins with XML version: `<?xml version="1.0"?>`

XML Attributes vs Elements

- Distinction between subelement and attribute
  - In the context of documents:
    - attributes are part of markup
    - subelement contents part of the basic document content
  - In the context of data representation:
    - difference not clear, but confusing
    - Some information can be represented in two ways
      - `<seq id="my_seq" name="NUCLEAR RIBONUCLEOPROTEIN">` ...
      - `<seq id="my_seq"> NUCLEAR RIBONUCLEOPROTEIN </seq>` ...

Correctness?

- Different encodings
- specified by encoding attribute

How to use XML data?

- Basic Idea
  - Application with XML-Generator
  - XML-Parser
  - Receiving application

Correctness of XML documents

- Syntactic correctness
  - Conformance to XML syntax
  - Document structured according to XML syntax is well-formed
  - Compare Syntax checker for program
- Semantic correctness
  - Given Meta level description of XML documents: Document Type Definition (DTD) or XML Schema
  - Document is valid with respect to DTD (Schema) if all definitions and restrictions have been fulfilled
  - No DTD ⇒ applications must know, what is meant

But: what is THE semantics of an XML doc?
**Example DTD**

```xml
<?xml version="1.0" encoding="US-ASCII"?>
<!DOCTYPE seq [ 
<!ENTITY % shape "(rect|circle|poly|default)">
<!ELEMENT seq (dbxref*, residues?) >
<!ELEMENT residues (#PCDATA)>
<!ATTLIST residues type (dna | rna | aa) #REQUIRED>
]> 
```

- **Nesting of elements:**
  - `|` : alternatives
  - `+` : 1 or more occurrences
  - `*` : 0 or more occurrences
  - `?` : 0 or one

- **Attribute spec:**
  - `#REQUIRED`, default val, `#IMPLIED` (= optional)
  - Enumeration type

**DTD attribute ID**

- At most one attribute of type `ID` per element
- `ID` attribute value of each element in an XML document must be distinct
  - `ID` attribute value is object identifier
- attribute of type `IDREF` must contain the `ID` value of an element in the same document
- attribute of type `IDREFS` contains a set of (0 or more) `ID` values. `ID` value must contain the `ID` value of an element in the same document
- `ID`, `IDREF`, `IDREFS` do not designate a particular domain (no type!)

**DTD declaration**

- **External DTD-declaration**
  ```xml
  <?xml version="1.0"?>
  <!DOCTYPE BSML PUBLIC "HTTP://LABBOOK.COM/DTD/BSML3_1.DTD" .../>
  ```

- **Internal DTD-declaration**
  ```xml
  <!DOCTYPE custDesc [ <!ELEMENT custDesc (#PCDATA)> ]>
  <custDesc> consumer rights protagonist </custDesc>
  ```

- **Mixed usage**
  ```xml
  <!DOCTYPE bank SYSTEM "http://www.x-ag.de/banks.dtd" [ 
  <!ATTLIST bank _Descr CDATA #REQUIRED>
  ]>
  <bank _Descr=" mostly private customers and ATM"> ... </bank>
  ```

**XML Namespaces**

- **Part of XML’s extensibility**
- Allow autonomous users to differentiate between tags with the same name (using a prefix)
  - Resolves naming conflicts
  - Allows multiple XML documents from multiple authors to be merged

```xml
xmlns:bk = "http://www.example.com/bookinfo/"
```

**Namespace**

- **Examples**
  ```xml
  <BOOK xmlns:bk="http://www.bookstuff.org/bookinfo">
  <bk:TITLE>All About XML</bk:TITLE>
  <bk:AUTHOR>Joe Developer</bk:AUTHOR>
  <bk:PRICE currency='US Dollar'>19.99</bk:PRICE>
  </BOOK>
  ```

- No prefix: all elements belong to same namespace

```xml
<BOOK xmlns="http://www.bookstuff.org/bookinfo" 
<TITLE>All About XML</TITLE>
<AUTHOR>Joe Developer</AUTHOR>
```
XML Schema

- XML Schema (XSD): much more expressive Schema language compared to DTD schemas
  - Typing of values
    - E.g. integer, string, etc
    - Constraints on min/max values
  - User-defined types
    - Specified in XML syntax, unlike DTDs
  - More standard representation, but verbose
    - Namespace support
    - Many more features
    - List types, uniqueness, and foreign key constraints, inheritance
  - Significantly more complicated than DTD syntax
  - Use of XSD recommended

XSD Example (from Silberschatz)

```xml
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="bank" type="BankType"/>
  <xsd:element name="account">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="account-number" type="xsd:string"/>
        <xsd:element name="branch-name" type="xsd:string"/>
        <xsd:element name="balance" type="xsd:decimal"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <!-- Definitions of customer and depositor should go in here-->
  <xsd:complexType name="BankType">
    <xsd:sequence>
      <xsd:element ref="account" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element ref="customer" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element ref="depositor" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

EML XML – Schema for sequences

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!--
************************************************************************
XML Schema for the components of an EMBL sequence record
Version 1.0, 15 March 2005
by Vincent Lombard
************************************************************************
The EMBL Nucleotide Sequence Database (also known as EMBL-Bank) constitutes Europe's primary nucleotide sequence resource.
This XML Schema describes the structure of entries in the EMBL database. These entries incorporate DNA or RNA sequences........
-->
<xs:schema xmlns:ebi="http://www.ebi.ac.uk/xml" xmlns:xs="http://www.w3.org">
  <xs:complexType name="entryType">
    .......
  </xs:complexType name="entryType">
</xs:schema>
```

3.5 Using XML

- Data exchange
- Data management:
  - Store, retrieve, query large document sets efficiently
    - Today’s solutions:
      - Mapping to RDB / ORDB / OODB
    - “Native” XML data management (not necessarily very different from storing in conventional DB)
  - Standardized data description: different extensions and applications
    - Bioinformatic Sequence Markup Language (BSML)
    - MathML, Scalable Vector Graphics (SVG)
    - RDF, Semantic Web (whatever that means...)
- Logical / Physical layout
Using XML: Logical – Physical Layout

• Layout of documents?
  – XML documents specify logical structure
  – Layout structure needed for output
• Use transformation language to describe device specific transformations

XML transformation

• XSLT: The language used for converting XML documents into other forms
• Describes how the document is transformed
• Expressed as an XML document (.xsl)
• Template rules
  – Patterns match nodes in source document
  – Templates instantiated to form part of result document
• XPath for querying, sorting, etc.
• XSL-FO language for describing layout

XSL = XSLT + XPATH + XSL-FO

XML transformation: example (1)

• Document

XML transformation: example (2)

• XSL style sheet - mapping to HTML

XML transformation: example (2)

• The result

Still to come....

• XML in bioinformatics (overview of activities)
• XPath, XQuery
• XML data management:
  Next generation data management system for molecular biology?
• (Similarity search in XML-DBS)