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

using material from
- Alan Robinson, (recommended ! http://industry.ebi.ac.uk/~alan/XMLWorkshop/)
- Silverschatz and M. Sapossnek .

3.1 Introduction

- Formats: trivial but very important problem
  Example: sequence formats .
  Details: .... .
- Infinitely many ways to structure data
  ⇨ Basic issue: processing data
  not a real issue: human readibility
- Needed: standardized way of representation
- Retrieving data from MB databases: yet another example

http://www.ebi.ac.uk/cgi-bin/dbfetch r1 r2 r3

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

  but
  - 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

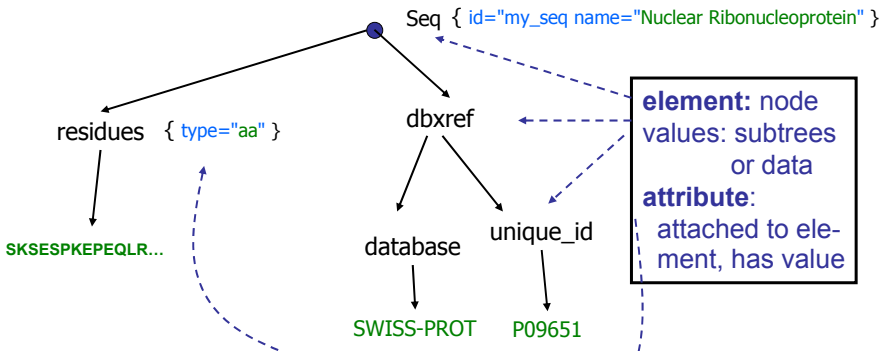
... it is already

XML markup: example

```
<?xml version="1.0"?>
<seq id="my_seq" name="NUCLEAR RIBONUCLEOPROTEIN">
  <dbxref>
    <database>SWISS-PROT</database>
    <unique_id>P09651</unique_id>
  </dbxref>
  <residues type="aa">
SKSESPKEPEQLRKLFIGGLSFETTDESLRSHFEQWGTLTDCVVMRDPNTKRS
RGFGFVTYATVEEVDAAMNARPHKVDGRVVEPKRAVSREDSQRPGAHLTVKKI
FVGGIKEDTEEHHLRDYFEQYGKIEVIEIMTDRGSGKKRGFAFVTFDDHDSVD
KIVIQKYHTVNGHNCEVRKALSKQEMASASSSQRGRSGSGNFGGGRGGGFGGN
DNFGRGGNFSGRGGFGGSRGGGGYGGSGDGYNGFGNDGGYGGGGPGYSGGSRG
YGSGGQGYGNQGSGYGGSGSYDSYNNGGGRGFGGGSGSNFGGGGSYNDFGNYN
NQSSNFGPMKGGNFGGRSSGPYGGGGQYFAKPRNQGGYGGSSSSSSYGSGRRF
  </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>
```

c.f Alan Robinson

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><br />
                 Stainless steel, one-piece construction,
                 lifetime guarantee.</p>
              </Description>
              <Price>9.95</Price>
           </Part>
           <Quantity>10</Quantity>
        </Line>
        .......
    </SalesOrder>
 </Orders>
```

relational schema?

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 centric XML documents: example

```
<Product>
  <Name>Variabler Maulschlüssel</Name>
   <Developer> Full Fabrication Labs, Inc. </Developer>
   <Summary> Großer, verstellbarer Schraubenschlüssel</Summary>
   <Description>
    <Para>Der Engländer besteht aus erstklassigem Stahl und
    besitzt einen gummierten Handgriff. Die Maulgröße liegt
    zwischen 0 und 32 mm.    </Para>
    <Para>Sie können.....    </Para>

   <List>
    <Item> <Link URL="Order.html"> Bestellen </Link></Item>
    <Item> <Link URL="Wrenches.htm"> Andere Werkzeuge ansehen
           </Link> </Item>
    <Item> <Link URL="catalog.zip"> Den Katalog herunterladen
    </Link> </Item>
   </List>
   <Para> Der Schraubenschlüssel kostet 15.33 Euro inkl. MWSt. Wenn
          Sie jetzt bestellen, erhalten Sie zusätzlich unsere
          wertlose Hobbybastler-Fibel.</Para>
   </Description>
</Product>
```

Typical:Long text elements

Note: Layout of logical elements can be defined independently!

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

  <TAG> <TAG2> ... </TAG2> </TAG>
- Attributes are optional
- Attribute values must quoted <TAG a1="val">...
  - No other data type than 'String '
- 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
    - Same information can be represented in two ways

```
<seq id="my_seq" name="NUCLEAR RIBONUCLEOPROTEIN">
        ....
</seq>          or...
<seq>
     <id>  my_seq      </id>
    <name> NUCLEAR RIBONUCLEOPROTEIN </name>
   …
</seq>
```

Suggestion: use attributes for identifiers of elements
use subelements for contents

Correctness?

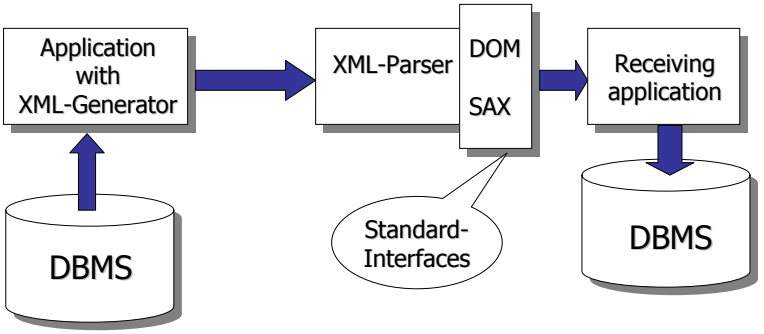
```
<?xml version="1.0" encoding="iso-2022-jp"?>
<!DOCTYPE 週報 SYSTEM "weekly-iso-2022-jp.dtd">
<!-- 週報サンプル -->
<週報>
  <業務報告リスト>
    <業務報告>
      <業務名>XMLエディターの作成</業務名>
      <業務コード>X3355-23</業務コード>
      <予定項目リスト>
        <予定項目>
          <P>XMLエディターの基本仕様の作成</P>
        </予定項目>
      </予定項目リスト>
      <実施事項リスト>
        <実施事項>
          <P>XMLエディターの基本仕様の作成</P>
        </実施事項>
        <実施事項>
          <P>競合他社製品の機能調査</P>
        </実施事項>
      </実施事項リスト>
      <問題点対策>
        <P>XMLとは何かわからない。</P>
      </問題点対策>
    </業務報告>
  </業務報告リスト>
</週報>
```

Correct or not correct ?

- Different encodings
- specified by encoding attribute

How to use XML data?

- Basic Idea



How does application know about
- syntactical correctness
- data semantics ?

DTD and XML schema

- Two ways to define the "schema" of an XML doc
  - Document Type Definition
  - XML Schema
- Document Type Definition (DTD)
  - Defines syntactic structure of a class of XML docs
  - Syntax
    - which elements? Attributes?

    <!ELEMENT elem (subelement-spec)>

    <!ATTLIST elem (attribute-specs) >
    - Nesting ⇨ tree structure
    - optional / mandatory elements

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

Entity:
simplest form abbreviation, here: enumeration

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

Attribute spec:
#REQUIRED, default val, #IMPLIED (= optional)
enumeration type
CDATA: not parsed, #PCDATA : parsed char data

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

Bioinformatic Sequence Markup Language

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

Internal DTD-declaration

```
<!DOCTYPE custDesc [ <!ELEMENT custDesc (#PCDATA)> ]>
<custDesc> consumer rights protagonist </custDesc>
```

Mixed usage

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


XLink / XPointer

- Link to external resources (documents, images...)
- XML link
  - URL ⇒ a resource
  - URL + XPointer ⇒ sub-resource of URL
  - XPointer ⇒ sub-resource of current URL

Example:

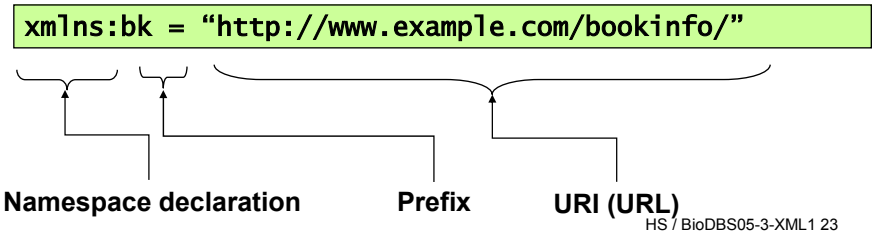
```
<mylink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="extended">
 <myresource xlink:type="locator"
   xlink:href="students.xml#Fred" xlink:label="student"/>
 <myresource xlink:type="locator"
      xlink:href="teachers.xml#Joe" xlink:label="teacher"/>
 <myarc xlink:type="arc" xlink:from="student" xlink:to="teacher"/>
</mylink>
```

??

cf the tutorial on XLink / XPointer http://www.brics.dk/~amoeller/XML/linking/

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



Namespace

- Examples

```
<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>
```

- No prefix: all elements belong to same namespace

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

XML Schema

- XML Schema (XSD): much more expressible 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
      Ability to map to RDB,…
- Significantly more complicated than DTD syntax
- Use of XSD recommended

XSD example

(from Silverschatz)

```
<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>
```

XML Schema

EML XML – Schema for sequences

```
<?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:sequence> .....
        </xs:complexType name="entryType">
```

XSD, not MB sequence!

XML Schema

- Based on an extensible type concept
- built-in types: **`double, float...`** (primitive), **`integer -> nonPositiveInteger,`**
  `-> nonNegativeInteger ..`
  `-> long, ...`
- simple types: defined by value type, representation and restrictions,
  do not have attributes or child elements

```
<xs:simpleType name = seqType>
      <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="single"/>
            <xs:enumeration value="join"/>
            <xs:enumeration value="order"/>
      </xs:restriction>
</xs:simpleType>
```

- Complex types:
  - used to define substructures

compare element structuring in DTD

```
<xs:complexType name="locationtype">
  <xs:annotation>
    <xs:documentation>Type can be either a single a join or an order
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
        <xs:element name="locationElement"
            type="locationElementType" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="type" use="required">
        <xs:simpleType>
              <xs:restriction base="xs:NMTOKEN">
                    <xs:enumeration value="single"/>
                    <xs:enumeration value="join"/>
                    <xs:enumeration value="order"/>
              </xs:restriction>
        </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="complement" type="xs:boolean" use="required"/>
</xs:complexType>
```

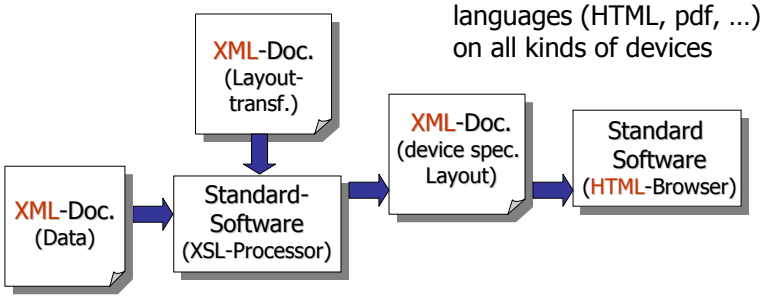
3.5 Using XML

- 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)
    .. and many, many more
  - Ressource Description in the web (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

Transformation into all kinds of languages (HTML, pdf, …) on all kinds of devices



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

```
<sales>
  <summary>
    <heading>Scootney Publishing</heading>
    <subhead>Regional Sales Report</subhead>
    <description>Sales Report</description>
  </summary>
  <data>
    <region>
      <name>West Coast</name>
      <quarter number="1" books_sold="24000" />
      <quarter number="2" books_sold="38600" />
      <quarter number="3" books_sold="44030" />
      <quarter number="4" books_sold="21000" />
    </region>
    ...
  </data>
</sales>
```


XML transformation: example (2)

- XSL style sheet - mapping to HTML

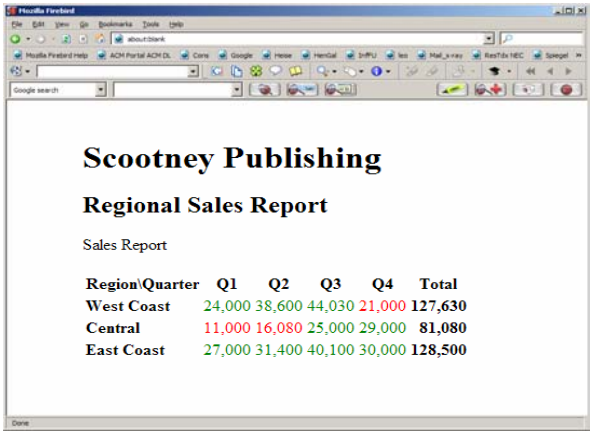
```
<xsl:param name="low_sales" select="21000"/>
<BODY>
 <h1><xsl:value-of select="//summary/heading"/> </h1>
 ...
 <table><tr><th>Region\Quarter</th>
   <xsl:for-each select="//data/region[1]/quarter">
     <th>Q<xsl:value-of select="@number"/></th>
   </xsl:for-each>
   ...
   <xsl:for-each select="//data/region">
   <tr><xsl:value-of select="name"/></th>
    <xsl:for-each select="quarter">
     <td><xsl:choose>
       <xsl:when test="number(@books_sold &lt;= $low_sales)">
         color:red;</xsl:when>
       <xsl:otherwise>color:green;</xsl:otherwise></xsl:choose>
     <xsl:value-of select="format-number (@books_sold,'###,###')"
       /> </td>
   ...
   <td><xsl:value-of
         select="format-number(sum(quarter/@books_sold),
                 '###,###')"/>
```

XPath expression

XPath: query language on doc trees

XML transformation: example (2)

- The result



Scootney Publishing

Regional Sales Report

Sales Report

Region\Quarter	Q1	Q2	Q3	Q4	Total
West Coast	24,000	38,600	44,030	21,000	**127,630**
Central	11,000	16,080	25,000	29,000	**81,080**
East Coast	27,000	31,400	40,100	30,000	**128,500**

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)