



**BASI DI DATI II – 2 modulo**  
**Parte II: XML e namespaces**

Prof. Riccardo Torlone  
Università Roma Tre



# Outline

- What is **XML**, in particular in relation to HTML
- The XML **data model** and its **textual** representation
- The XML **Namespace** mechanism

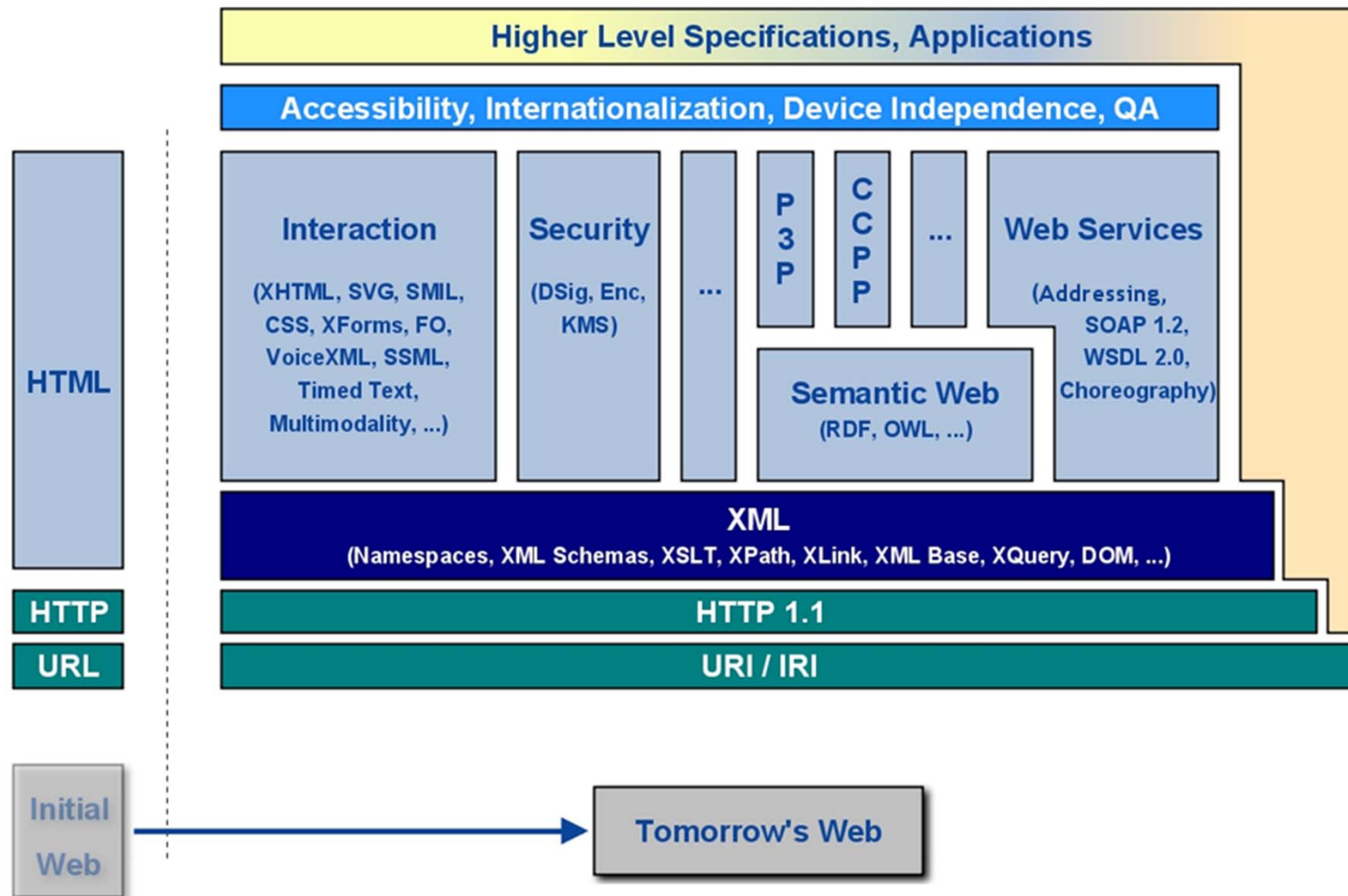
## What is XML?

- XML: *Extensible Markup Language*
- A **framework** for defining markup languages
- Each language is targeted at its own **application domain** with its own markup tags
- There is a common set of **generic tools** for processing XML documents
- **XHTML**: an XML variant of HTML
- Inherently **internationalized** and **platform independent** ([Unicode](#))
- Developed by W3C, standardized in 1998

# Evolution

- 1986: Standard Generalized Markup Language (SGML) ISO 8879-1986
- November 1995: HTML 2.0
- Jan 1997: HTML 3.2
- Aug 1997: XML W3C Working Draft
- Feb 10, 1998: XML 1.0 Recommendation
- Dec 13, 2001: XML 1.1 W3C Working Draft
- Oct 15, 2002 : XML 1.1 W3C Candidate Recommendation
- Aug 16, 2006: XML 1.1, Recommendation
- Nov 26, 2008: XML 1.0 (5th edition) Recommendation

# Role of XML



# Goal

- Nov 96: [initial XML draft](#) “The design goals for XML are:
  - XML shall be straightforwardly usable over the Internet
  - XML shall support a wide variety of applications
  - XML shall be compatible with SGML
  - It shall be easy to write programs which process XML documents
  - The number of optional features in XML is to be kept to the absolute minimum, ideally zero
  - XML documents should be human-legible and reasonably clear
  - The XML design should be prepared quickly
  - The design of XML shall be formal and concise
  - XML documents shall be easy to create
  - Terseness is of minimal importance

## Recipes in XML

- Define our own “*Recipe Markup Language*”
- Choose markup tags that correspond to concepts in this application domain
  - *recipe, ingredient, amount, ...*
- No canonical choices
  - granularity of markup?
  - structuring?
  - elements or attributes?
  - ...

## Example (1/2)

```
<col l e c t i o n>
  <descri p t i o n>Reci pes  suggested by Jane Dow</descri p t i o n>

  <reci pe  i d="r117">
    <ti t l e>Rhubarb Cobbl er</ti t l e>
    <date>Wed, 14 Jun 95</date>

    <i n g r e d i e n t  name="di ced rhubarb" amount="2.5" uni t="cup" />
    <i n g r e d i e n t  name="sugar" amount="2" uni t="tabl espoon" />
    <i n g r e d i e n t  name="fai r l y ri pe banana" amount="2" />
    <i n g r e d i e n t  name="ci n namon" amount="0.25" uni t="teaspoon" />
    <i n g r e d i e n t  name="nutmeg" amount="1" uni t="dash" />

    <preparati on>
      <step>
        Combi ne al l and use as cobbl er, pi e, or cri sp.
      </step>
    </preparati on>
```



## Example (2/2)

```
<comment>
```

```
  Rhubarb Cobl er made wi th bananas as the mai n sweetener.  
  It was del i ci ous.
```

```
</comment>
```

```
<nutri ti on cal ori es="170" fat="28%"
```

```
  carbohydrates="58%" protei n="14%" />
```

```
<rel ated ref="42">Garden Qui che is al so yummy</rel ated>
```

```
</reci pe>
```

```
</col l ecti on>
```

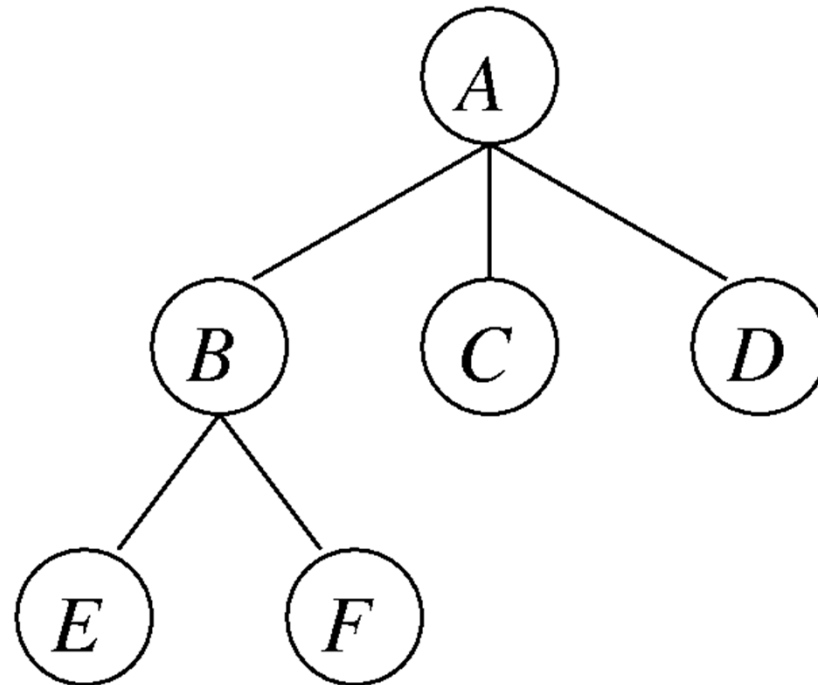
## Building on the XML Notation

- Defining the **syntax** of our recipe language
  - DTD, XML Schema, ...
- Showing recipe documents in **browsers**
  - XPath, XSLT
- Recipe collections as **databases**
  - XQuery
- Building a **Web-based** recipe editor
  - HTTP, Servlets, JSP, ...
- ...

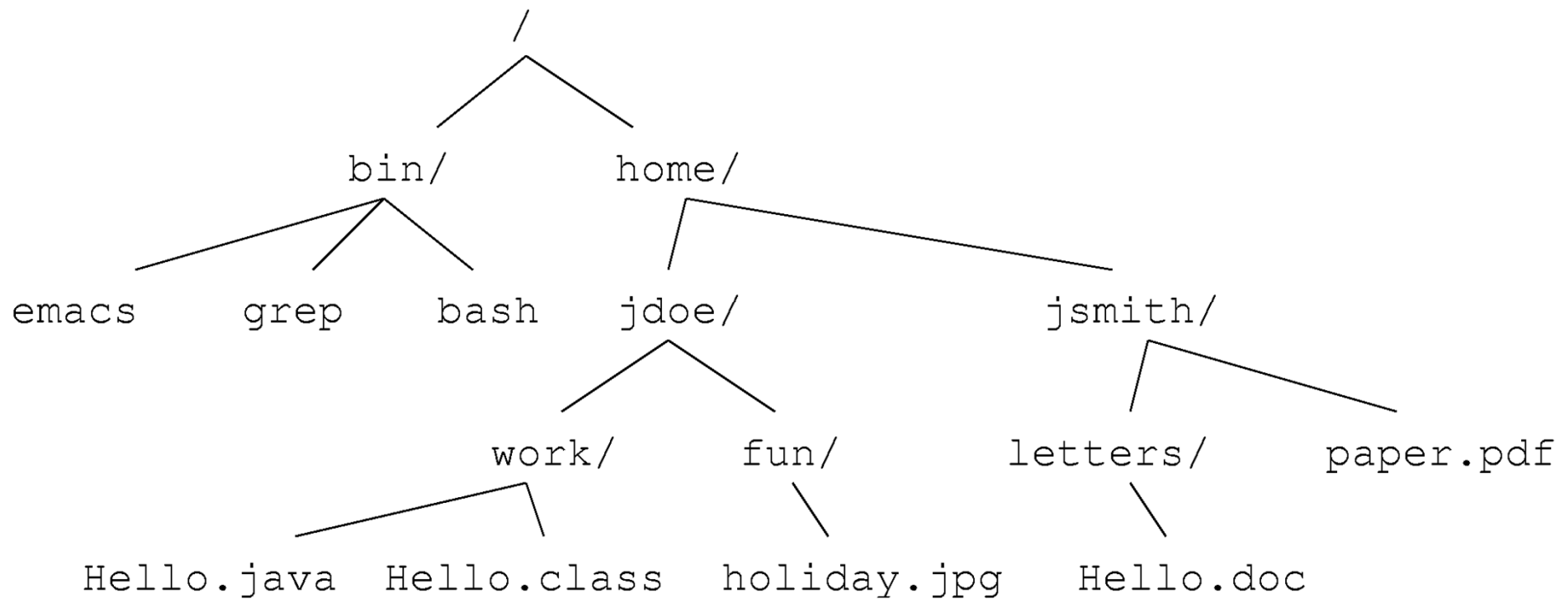
# XML Trees

- Conceptually, an XML document is a **tree structure**

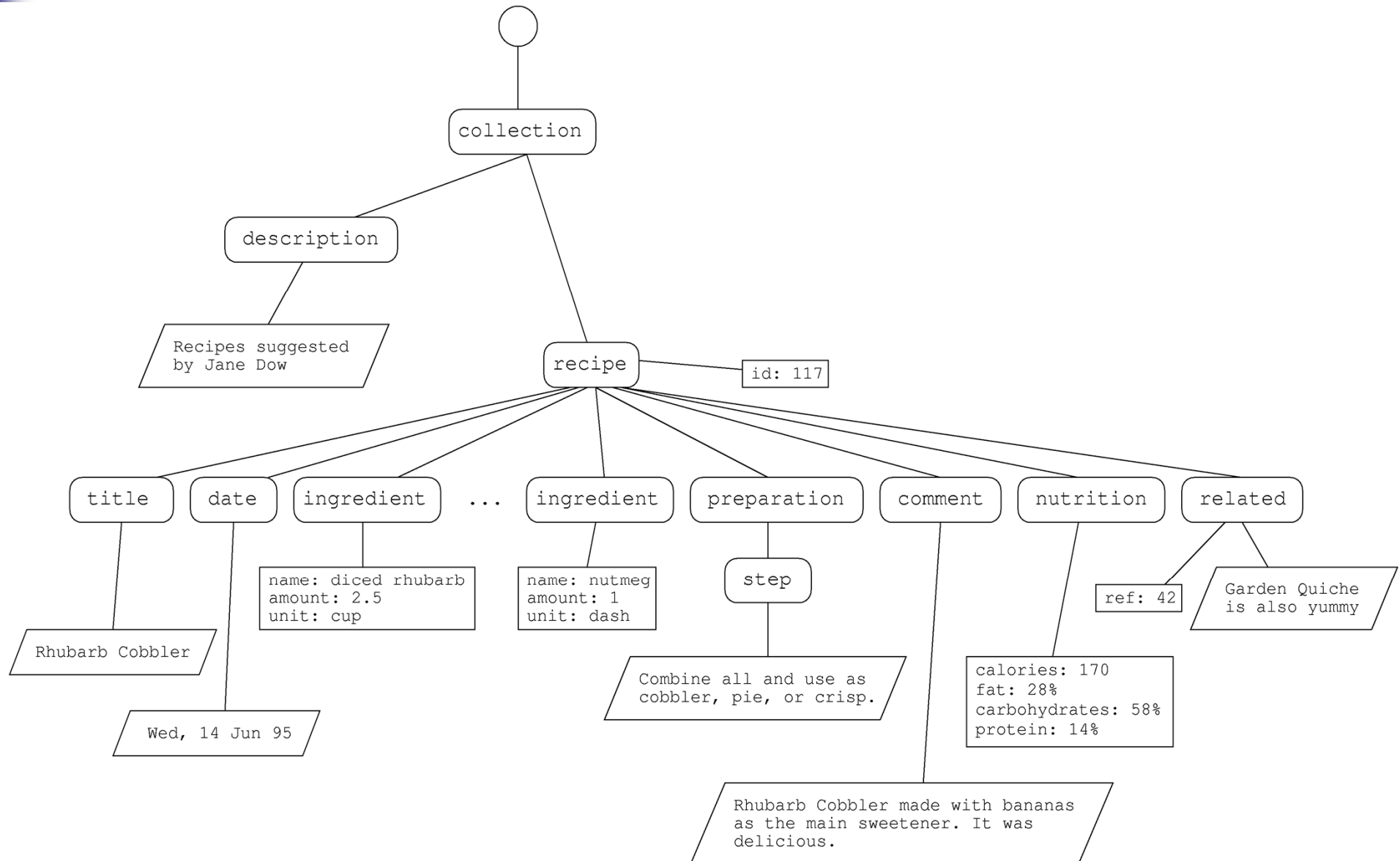
- node, edge
- root, leaf
- child, parent
- sibling (ordered), ancestor, descendant



# An Analogy: File Systems



# Tree View of the XML Recipes



## Nodes in XML Trees

- **Text nodes:** carry the actual contents, leaf nodes
- **Element nodes:** define hierarchical logical groupings of contents, each have a name
- **Attribute nodes:** unordered, each associated with an element node, has a name and a value
- **Comment nodes:** ignorable meta-information
- **Processing instructions:** instructions to specific processors, each have a target and a value
- **Root nodes:** every XML tree has one root node that represents the entire tree

# Textual Representation

- **Text nodes:** written as the text they carry
- **Element nodes:** start-end tags
  - `<bl a ... > ... </bl a>`
  - short-hand notation for empty elements:  
`<bl a/>`
- **Attribute nodes:** `name="val ue"` in start tags
- **Comment nodes:** `<! -- bl a -->`
- **Processing instructions:** `<?target val ue?>`
- **Root nodes:** implicit

# Elements in XML

- Element only

- ex:      <course>  
                  <start> ... </start>  
                  <end> ... </end>  
                  <name> ... </name>  
                  </course>

- Content only

- “text”

- ex:      <coursename>Java</coursename>



# Elements in XML

## ■ Mixed

□ es:      <comments>  
                    An interesting course but ...  
                    <summary>the length...</summary>  
                    Therefore ...  
                    </comments>

## ■ Empty

□ es:      <coordinator prof="d01" />

# Elements in XML

- General rules

- Case sensitive

- Usually: lowercase for element names and attributes

- Names

- Usually: starts with a alphabetic char or with “\_” :
    - Example: exam, \_course

- Comments

- `<!-- this is a comment -->`

# Attributes in XML

- Syntax
  - pair name+value (between “ ” or ‘ ’)
- Use
  - Special values
    - ex: identifiers and references
  - Metadata
- Attributes or elements?
  - Free, but it is better to be consistent

# Browsing XML (without XSLT)

Il file XML specificato apparentemente non ha un foglio di stile. Il documento è mostrato di seguito.

```
- <collection>
  <description>Recipes suggested by Jane Dow</description>
  - <recipe id="r117">
    <title>Rhubarb Cobbler</title>
    <date>Wed, 14 Jun 95</date>
    <ingredient name="diced rhubarb" amount="2.5" unit="cup" />
    <ingredient name="sugar" amount="2" unit="tablespoon" />
    <ingredient name="fairly ripe banana" amount="2" />
    <ingredient name="cinnamon" amount="0.25" unit="teaspoon" />
    <ingredient name="nutmeg" amount="1" unit="dash" />
  - <preparation>
    - <step>
      Combine all and use as cobbler, pie, or crisp.
    </step>
  </preparation>
  - <comment>
    Rhubarb Cobbler made with bananas as the main sweetener. It was delicious.
  </comment>
  <nutrition calories="170" fat="28%" carbohydrates="58%" protein="14%" />
  <related ref="42">Garden Quiche is also yummy</related>
</recipe>
</collection>
```

```
- <collection>
  <description>Recipes suggested by Jane Dow</description>
  - <recipe id="r117">
    <title>Rhubarb Cobbler</title>
    <date>Wed, 14 Jun 95</date>
    <ingredient name="diced rhubarb" amount="2.5" unit="cup" />
    <ingredient name="sugar" amount="2" unit="tablespoon" />
    <ingredient name="fairly ripe banana" amount="2" />
    <ingredient name="cinnamon" amount="0.25" unit="teaspoon" />
    <ingredient name="nutmeg" amount="1" unit="dash" />
  - <preparation>
    <step>Combine all and use as cobbler, pie, or crisp.</step>
  </preparation>
  <comment>Rhubarb Cobbler made with bananas as the main sweetener. It was delicious.</comment>
  <nutrition calories="170" fat="28%" carbohydrates="58%" protein="14%" />
  <related ref="42">Garden Quiche is also yummy</related>
</recipe>
</collection>
```



## More Constructs

- XML declaration
- Character references
- CDATA sections
- Document type declarations and entity references explained later...
  
- Whitespace?

# Example

```
<?xml version="1.1" encoding="UTF-8"?>
<!DOCTYPE features SYSTEM "example.dtd">
<features a="b">
  <?mytool here is some information specific to mytool?>
  El señor está bien, garçon!
  Copyright &#169; 2005
  <![CDATA[ <this is not a tag> ]]>
  <!-- always remember to specify the
        right character encoding -->
</features>
```

# Well-formedness

- Every XML document must be *well-formed*
  - start and end tags must **match** and **nest** properly
    - `<x><y></y></x>` ✓
    - ~~`</z><x><y></x></y>`~~
  - exactly one **root element**
  - ...
- in other words, it defines a proper tree structure
- **XML parser**: given the textual XML document, constructs its tree representation

# XML Namespaces

```
<widget type="gadget" >
  <head size="medium" />
  <big><subwidget ref="gizmo" /></big>
  <info>
    <head>
      <title>Description of gadget</title>
    </head>
    <body>
      <h1>Gadget</h1>
      A gadget contains a big gizmo
    </body>
  </info>
</widget>
```

- When combining languages, element names may become **ambiguous!**
- Common problems call for common solutions





## The Idea

- Assign a URI to every (sub-)language

e.g. for XHTML 1.0:

<http://www.w3.org/1999/xhtml>

- Qualify element names with URIs:

{<http://www.w3.org/1999/xhtml>}head

## The Actual Solution

- Namespace declarations bind URIs to prefixes

```
<... xml ns: foo="http://www.w3.org/TR/xhtml1">  
  ...  
  <foo: head>... </foo: head>  
  ...  
</...>
```

- Lexical scope
- Default namespace (no prefix) declared with `xml ns="..."`
- Attribute names can also be prefixed

# Widgets with Namespaces

```
<widget type="gadget" xml ns="http://www.widget.inc">
  <head size="medium"/>
  <big><subwidget ref="gizmo"/></big>
  <info xml ns:xhtml="http://www.w3.org/TR/xhtml1">
    <xhtml:head>
      <xhtml:title>Description of gadget</xhtml:title>
    </xhtml:head>
    <xhtml:body>
      <xhtml:h1>Gadget</xhtml:h1>
      A gadget contains a big gizmo
    </xhtml:body>
  </info>
</widget>
```

- **Namespace map:** for each element, maps prefixes to URIs



# Applications of XML

Rough classification:

- Data-oriented languages
- Document-oriented languages
- Protocols and programming languages
- Hybrids

## Example: XHTML

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head><title>Hello world! </title></head>
  <body>
    <h1>This is a heading</h1>
    This is some text.
  </body>
</html >
```

# SOAP

```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
  soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
  <soap:Body xmlns:m="http://www.example.org/stock">
    <m:GetStockPrice>
      <m:StockName>IBM</m:StockName>
    </m:GetStockPrice>
  </soap:Body>
</soap:Envelope>
```

**request**



```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
  soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
  <soap:Body xmlns:m="http://www.example.org/stock">
    <m:GetStockPriceResponse>
      <m:Price>34.5</m:Price>
    </m:GetStockPriceResponse>
  </soap:Body>
</soap:Envelope>
```

**response**



# Example: CML

```
<molecule id="METHANOL">
  <atomArray>
    <stringArray builtin="id">a1 a2 a3 a4 a5 a6</stringArray>
    <stringArray builtin="elementType">C O H H H H</stringArray>
    <floatArray builtin="x3" units="pm">
      -0.748 0.558 ...
    </floatArray>
    <floatArray builtin="y3" units="pm">
      -0.015 0.420 ...
    </floatArray>
    <floatArray builtin="z3" units="pm">
      0.024 -0.278 ...
    </floatArray>
  </atomArray>
</molecule>
```

## Example: ebXML

```
<MultiPartyCollaboration name="DropShip">
  <BusinessPartnerRole name="Customer">
    <PerformsInitiatingRole='//binaryCollaboration[@name="Firm Order"]/
      InitiatingRole[@name="buyer"]' />
  </BusinessPartnerRole>
  <BusinessPartnerRole name="Retailer">
    <PerformsRespondingRole='//binaryCollaboration[@name="Firm Order"]/
      RespondingRole[@name="seller"]' />
    <PerformsInitiatingRole='//binaryCollaboration[...]/
      InitiatingRole[@name="buyer"]' />
  </BusinessPartnerRole>
  <BusinessPartnerRole name="DropShip Vendor">
    ...
  </BusinessPartnerRole>
</MultiPartyCollaboration>
```



## Example: ThML

```
<h3 class="s05" id="One. 2. p0. 2">Havi ng a Humbl e Opini on of Sel f</h3>
<p class="First" id="One. 2. p0. 3">EVERY man natural ly desi res knowl edge
  <note place="foot" id="One. 2. p0. 4">
    <p class="Footnote" id="One. 2. p0. 5"><added id="One. 2. p0. 6">
      <name id="One. 2. p0. 7">Ari stotl e</name>, Metaphysi cs, i. 1.
    </added></p>
  </note>;
  but what good is knowl edge wi thout fear of God? Indeed a humbl e
  rustic who serves God is better than a proud i ntellectual who
  neglects his soul to study the course of the stars.
  <added id="One. 2. p0. 8"><note place="foot" id="One. 2. p0. 9">
    <p class="Footnote" id="One. 2. p0. 10">
      Augusti ne, Confessi ons V. 4.
    </p>
  </note></added>
</p>
```

# Criticism

- Verbosity and complexity
- Relational mapping complex
- Human readability questionable
- Oriented to documents rather than data
- Competitors:
  - JSON
    - a lightweight text-based open standard designed for human-readable data interchange
    - derived from JavaScript
  - YAML
    - a human-readable data serialization
    - derived from C, Perl, and Python

# Wine list in XML

```
<?xml version="1.0" encoding="UTF-8"?>
<wine-list>
  <wine name="Domaine de l'Île Margaux"
        type="Bordeaux supérieur">
    <is-red>true</is-red>
    <origin>
      <country>France</country>
      <region>Bordeaux</region>
    </origin>
    <price>22.80</price>
    <year>2002</year>
  </wine>
  <wine name="Riesling Hugel" type="Alsace">
    <is-red>false</is-red>
    <origin>
      <country>France</country>
      <region>Alsace and East</region>
    </origin>
    <price>17.95</price>
    <year>2002</year>
  </wine>
</wine-list>
```

# Wine list in JSON

```
"wine-list": {
  "wine": [{
    "name": "Domaine de l'Île Margaux",
    "type": "Bordeaux supérieur",
    "is-red": true,
    "origin": {
      "country": "France",
      "region": "Bordeaux"
    },
    "price": 22.80,
    "year": 2002
  }, {
    "name": "Riesling Hugel",
    "type": "Alsace",
    "is-red": false,
    "origin": {
      "country": "France",
      "region": "Alsace and East"
    },
    "price": 17.95,
    "year": 2002
  }
]}
}
```

# Wine list in YAML

```
wine-list:  
  wine:  
    - name: "Domaine de l'Île Margaux"  
      type: Bordeaux supérieur  
      is-red: true  
      origin:  
        country: France  
        region: Bordeaux  
      price: 22.80  
      year: 2002  
  
    - name: Riesling Hugel  
      type: Alsace  
      is-red: false  
      origin:  
        region: Alsace and East  
        country: France  
      price: 17.95  
      year: 2002
```



## Summary

- XML: a notation for hierarchically structured text
- Conceptual tree model vs. concrete textual representation
- Well-formedness
- Namespaces
- Alternatives to XML



## Essential Online Resources

- <http://www.w3.org/TR/xml11/>
- <http://www.w3.org/TR/xml-names11>
- <http://www.unicode.org/>