

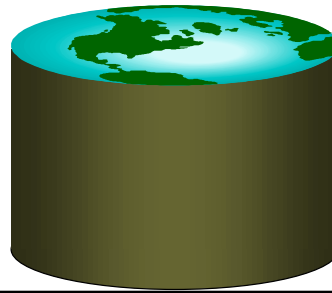
## On the Preservation of Memories

To maintain the condition of their memories, the famas proceed in the following manner: after having fastened the memory with webs and reminders, with every possible precaution, they wrap it from head to foot in a black sheet and stand it against the parlor wall with a little label which reads: "EXCURSION TO QUILMES" or "FRANK SINATRA".

Cronopios, on the other hand, disordered and tepid beings that they are, leave memories loose about the house. They set them down with happy shouts and walk carelessly among them, and when one passes through running they caress it mildly and tell it "Don't hurt yourself," and also "Be careful of the stairs." It is for this reason that the famas' houses are orderly and silent, while in those of the cronopios there is great uproar and doors slamming. Neighbors always complain about cronopios, and the famas shake their heads understandingly, and go and see if the tags are all in place.

Julio Cortázar, *Cronopios and Famas*, 1962.

XML



## XML History

- **XML was invented as a way to tag *documents* in an extensible way**
  - E.g. Shakespeare's plays could be tagged by speaker  
<Juliet>O Romeo, Romeo! wherefore art thou Romeo?</Juliet>
  - Goal vs. HTML: separate *content* from *presentation*
    - XSL is a companion language to specify screen presentation
    - Note that tags in HTML mix up these issues!
- **XML is used much more widely for *messages***
  - E.g. Ford's purchasing app generates a purchase order in XML format, e-mails it to a billing app at Firestone
  - Firestone's billing app ingests the email, generates a bill in XML format, and e-mails it to Ford's bank
  - A simplified picture, but XML for business messages is common
  - I.e. XML is a "wire format" for inter-program communication
- **XML is a way to represent data – compare to Relational? To Object-Relational?**



## XML by example

- Preamble includes XML declaration, root element, ref to DTD
- So far, looks like object-relational with nesting.
- Note that order matters (i.e. no sets, only lists)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Contract SYSTEM "http://xml.cXML.org/schemas/cXML/1.1.010/cXML.dtd">

<Contract>
  <SupplierID>123456</SupplierID>
  <Comments>
    Sample cXML contract
  </Comments>
  <ItemSegment>
    <ContractItem>
      <ItemID>
        <SupplierPartID>12345</SupplierPartID>
      </ItemID>
      <UnitPrice><Money>1.50</Money></UnitPrice>
      <Extrinsic> http://www.workchairs.com/CoolProducts </Extrinsic>
    </ContractItem>
    <ContractItem>
      <ItemID>
        <SupplierPartID>12347</SupplierPartID>
      </ItemID>
      <UnitPrice><Money>111.50</Money></UnitPrice>
    </ContractItem>
  </ItemSegment>
</Contract>
```



## Tags with Attributes

- Not clear why this was deemed important.
- Note that attributes cannot be sets, but nested tags can

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Contract SYSTEM "http://xml.cXML.org/schemas/cXML/1.1.010/cXML.dtd">

<Contract effectiveDate="1999-01-01T14:32:20-08:00"
  expirationDate="2000-06-01T12:00:00-08:00">
  <SupplierID domain="DUNS">123456</SupplierID>
  <Comments xml:lang="en-US">
    Sample cXML contract
  </Comments>
  <ItemSegment segmentKey="Detroit">
    <ContractItem>
      <ItemID>
        <SupplierPartID>12345</SupplierPartID>
      </ItemID>
      <UnitPrice><Money currency="USD">1.50</Money></UnitPrice>
      <Extrinsic name="URL"> http://www.workchairs.com/CoolProducts </Extrinsic>
    </ContractItem>
    <ContractItem>
      <ItemID>
        <SupplierPartID>12347</SupplierPartID>
      </ItemID>
      <UnitPrice><Money currency="USD">111.50</Money></UnitPrice>
    </ContractItem>
  </ItemSegment>
</Contract>
```



## Document Type Declarations (DTDs)

```
<?xml version="1.0" encoding="UTF-8"?>
<!ENTITY % datetime.tz "CDATA">
<!ENTITY % string "CDATA">
<!ENTITY % nmtoken "CDATA">      <!-- Any combination of XML name chars. -->
<!ENTITY % xmlLangCode "%nmtoken;"> <!-- Language code as defined by XML
                                     recommendation: Language and
                                     country. -->

<!ELEMENT SupplierID (#PCDATA)>
<!ATTLIST SupplierID
  domain %string; #REQUIRED
>
<!ELEMENT Comments ( #PCDATA | Attachment )* >
<!ATTLIST Comments
  xml:lang %xmlLangCode; #IMPLIED
>
<!ELEMENT ItemSegment (ContractItem+)>
<!ATTLIST ItemSegment
  segmentKey %string; #IMPLIED
>
<!ELEMENT Contract (SupplierID+, Comments?, ItemSegment+)>
<!ATTLIST Contract
  effectiveDate %datetime.tz; #REQUIRED
  expirationDate %datetime.tz; #REQUIRED
>
```

Here's a DTD for our Contract

Note: can be missing some fields, can have >1 of others.

Without a DTD, can have all kinds of tags! Some claim this as a benefit of XML.



## Other miscellaneous features

- **XML docs can have IDs and IDREFs, URIs**
  - reference to another document or document element
- **Document Object Model (DOM)**
  - A tree "object" API for traversing an XML or HTML doc
  - Typically for Java
- **XML Schema is a proposal to replace/augment DTDs**
  - Has a notion of types and typechecking
  - May introduce some notions of IC's
  - Quite complicated, controversial ... not really adopted yet
- **XML Namespaces**
  - Can import tag names from others
  - Disambiguate by prefixing the namespace name
    - I.e. berkeley-eecs:gpa is different from uphoenix:gpa
- **Lots of other details, tools, etc.**



## Advantages of XML vs. Relational

- **ASCII makes things easy**
  - Easy to parse
  - Easy to ship (e.g. across firewall, via email, etc.)
- **Self-documenting**
  - Metadata (tag names) come with the data
- **Nested**
  - Can bundle lots of related data into one message
  - (Note: object-relational allows this)
- **Can be sloppy**
  - don't *have* to define a schema in advance
- **Standard**
  - The most interesting thing about XML is that people are interested in it!
  - Lots of free Java tools for parsing and munging XML
    - Expect lots of Microsoft tools (C#) for same



## What XML does not solve

- **XML doesn't standardize metadata**
  - It only standardizes the metadata language
    - Not that much better than agreeing on an alphabet
  - E.g. my <price> tag vs. your <price> tag
    - Mine includes shipping and federal tax, and is in \$US
    - Yours is manufacturer's list price in ¥Japan
  - XML Schema is a proposal to help with some of this
- **XML doesn't help with data modeling**
  - No notions of IC's, FD's, etc.
  - In fact, *encourages* non-first-normal form!
- **You will probably have to translate to/from XML**
  - Relational vendors will help with this ASAP
  - XML "features" (nesting, ordering, etc.) make this a pain
  - Flatten the XML if you want data independence (?)



## What about XML queries?

- **Xpath**
  - A single-document language for “path expressions”
    - Not unlike regular expressions on tags
    - E.g. /Contract/\*/UnitPrice, /Contract//UnitPrice, etc.
- **XSLT**
  - XPath plus a language for formatting output
- **XML-Query (a.k.a. XQuery)**
  - An SQL-like proposal with extra XPath features
  - Data model is lists, not sets
    - Joins are explicit loops
    - Limits query optimization opportunities!
  - Proposals to change XQuery to have a “disorderly” mode
  - Still just a draft proposal, nobody using it



## Reminder: Benefits of Relational

- **Data independence buys you:**
  - Evolution of storage -- vs. XML?
  - Evolution of schema (via views) – vs. XML?
- **Database design theory**
  - IC's, dependency theory, lots of nice tools for ER
- **Remember, databases are long-lived and reused**
  - Today's “nesting” might need to be inverted tomorrow!
- **Upshot:**
  - XML is good for transient data (e.g. messages)
  - XML is fine for data that will not get reused in a different way (e.g. Shakespeare, database output like reports)
  - Relational is far cleaner for persistent data
- **Avoid the collective amnesia!**



## XML Databases?

- **Will XML require a new kind of database?**
- **Is there a market for new database vendors?**
- **A prediction:**
  - SQL-99 and XQuery will merge
  - Traditional DBMSs will be the serious XML repositories.
    - Doesn't extend their feature set
    - Oracle and IBM are paranoid and working hard on XML
    - Fighting Oracle and IBM is not a good business plan
    - Also a daunting engineering exercise:
      - Maybe you have a good XPath engine
      - Do you really want to reimplement storage, optimization, 2PL, ARIES, etc? How do you do row-level locking in XML?
  - There will be *lots* of user tools to help map to/from XML
    - Not unlike the "import" tools and "report writers" that already exist



## XML Summary

- **XML is an important new standard for representing data**
- **It is mostly being used for messages between business apps**
  - Good because it's easy to read and transmit
  - Can bundle a bunch of related data in one file
- **Also used for documents, separating content (XML) from presentation (XSL)**
- **Supports a number of non-first-normal-form "features"**
  - Hence troublesome as a "database"
- **Expect data to be translated to/from XML frequently**
  - And expect to be given nice tools to help with that.



## More on XML

- **100 books published in the last 6 months**
  - Each seems to be 1000 pages
  - How hard could this be??
    - A Berkeley EECS major can pick this up in no time
- **Try some websites**
  - [xml.org](http://xml.org) provides a business software view of XML
  - [xml.apache.org](http://xml.apache.org) has lots of useful shareware for XML
  - [www.ibm.com/developerworks/xml/](http://www.ibm.com/developerworks/xml/) has shareware, tutorials, reference info
  - [xml.com](http://xml.com) is the O'Reilly resource site
  - [www.w3.org/XML/](http://www.w3.org/XML/) is the official XML standard site
  - the most standardized XML dialects are:
    - Ariba's Commerce XML ("cxml", see [cxml.org](http://cxml.org))
    - RosettaNet (see [rosettanet.org](http://rosettanet.org))
    - Microsoft trying to enter this arena (BizTalk, now .NET)