



XML Databases

Introduction

Benjamin Piwowarski

XML Birth: TDL

Text Description Language

1970: “An Online System for Integrated Text Processing”

"to identify the **structure and purpose** of the parts of text. ... The composition program would identify the codes as calls to stored formats; the retrieval program would use them for classification."

“There would be many uses for an online integrated text processing system. A university press, for example, may engage in demand publishing. Its database, originally prepared for typesetting, could be searched for material relevant to the subject of a proposed publication.”

XML Birth: GML (1973)

Generalised Markup Language

■ C. Goldfarb, E. Mosher and R. Lorie

“This analysis of the markup process suggests that it should be possible to design a generalized markup language so that markup would be useful **for more than one application or computer system** (...)”

☞ Well-formed + valid

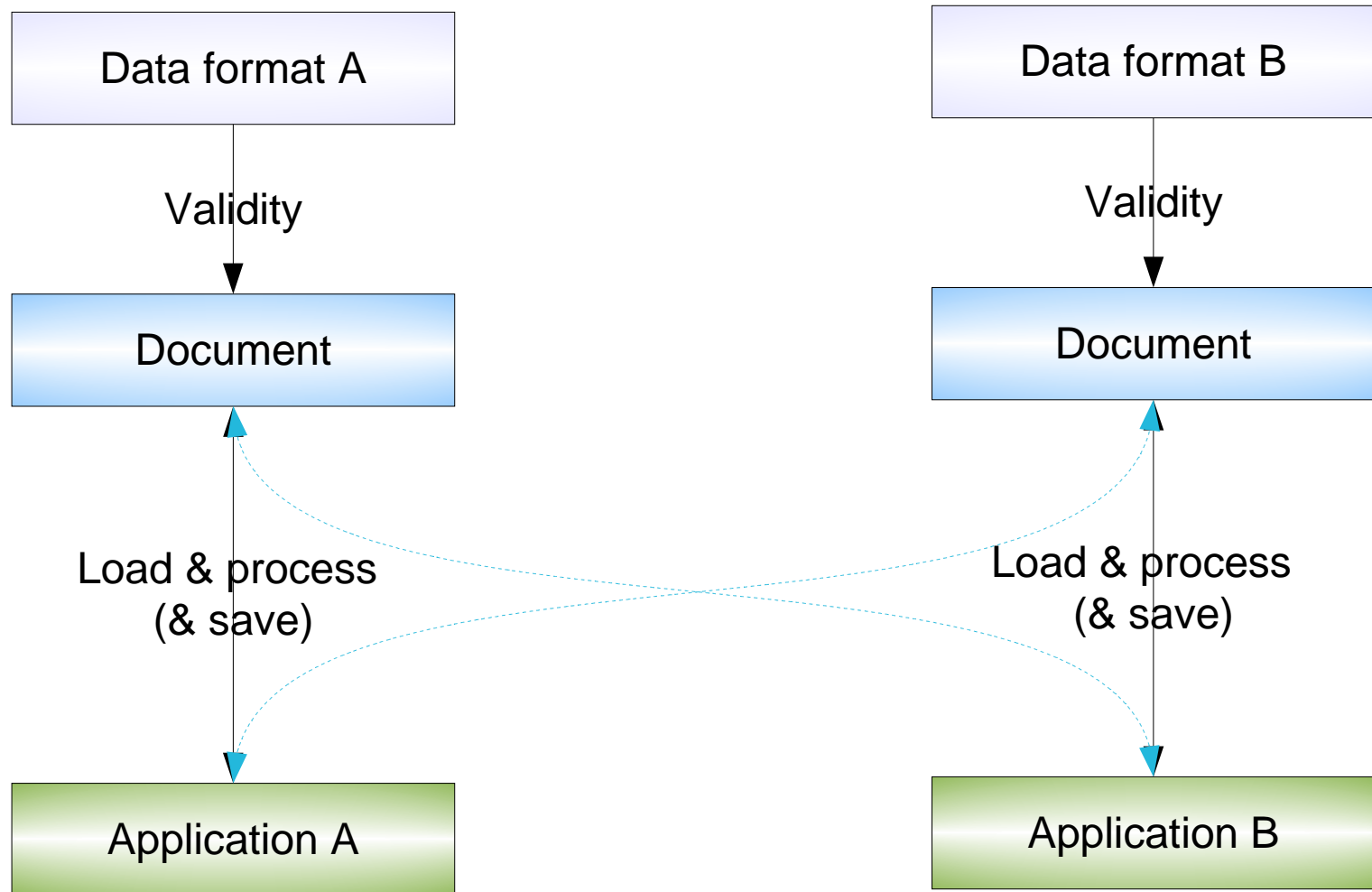
```
:h1.Chapter 1: Introduction
:p.GML supported hierarchical containers, such as
:ol
:li.Ordered lists (like this one),
:li.Unordered lists, and
:li.Definition lists
:eol.
```



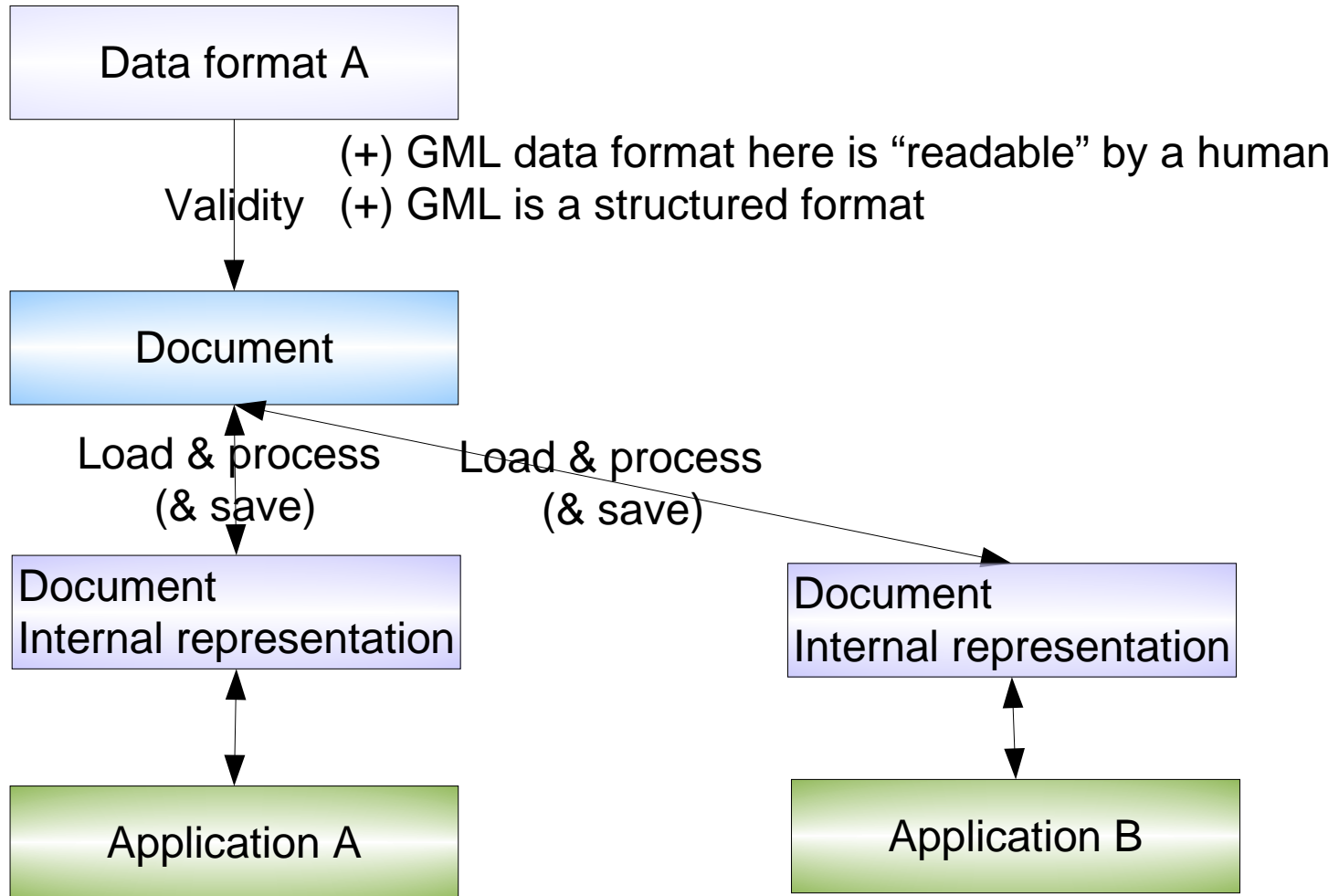
Markup Language =

The separation
of the
information content
of documents from their
format

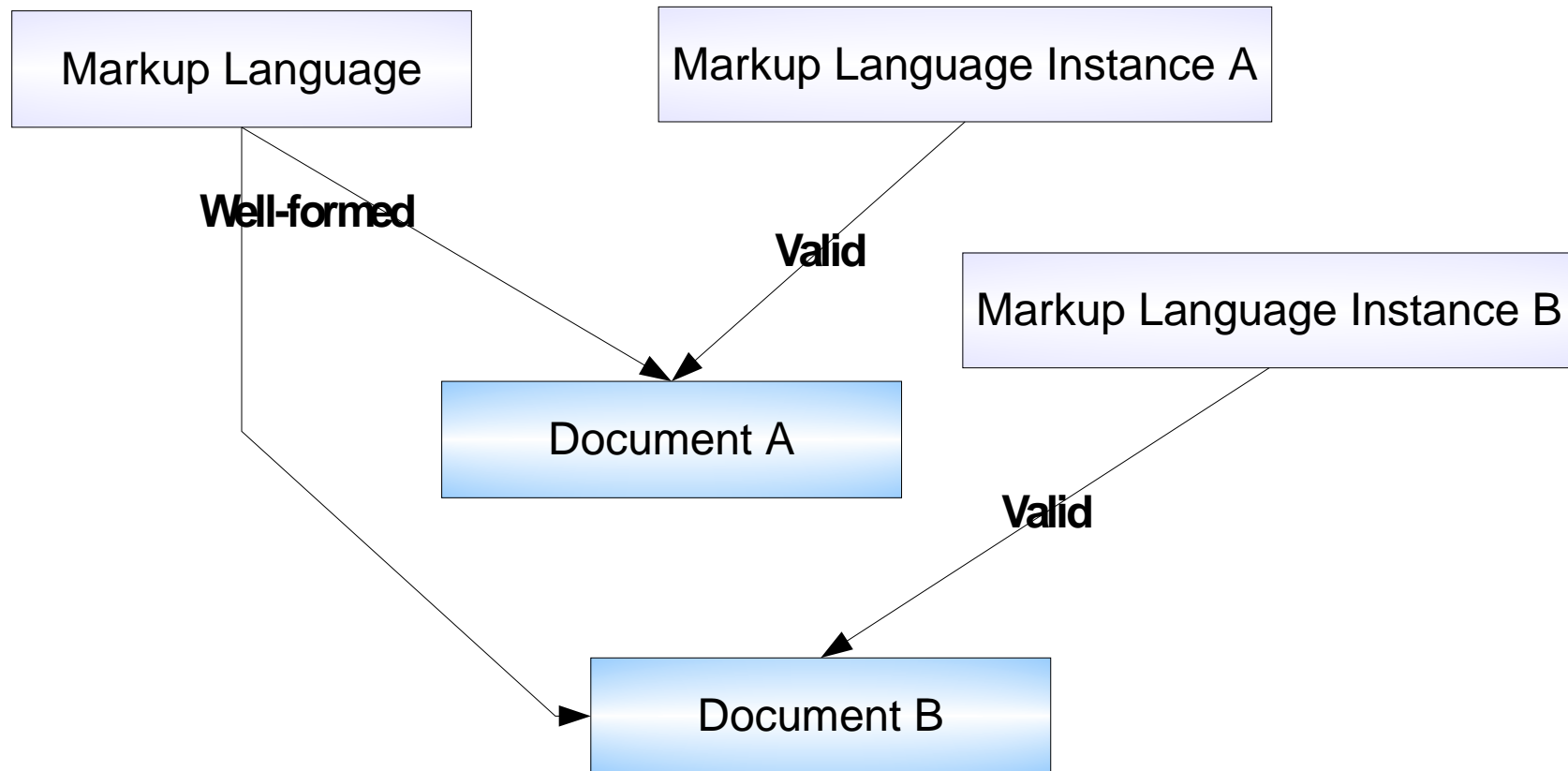
Proprietary/binary format



GML



Successor: SGML and XML



XML Birth: SGML (1980-86)

standard

- A Grammar:
 - Document Type Definition (DTD)
 - ☞ Validation
- <Tags> to delimit structure
- But... quite complex



HTML:

A famous SGML instance (1990-92)

```
<html>
```

```
<body>
```

```
  <p> Hello world.
```

```
</body>
```

```
</html>
```

```
<anthology>
<poem><title>The SICK ROSE</title>
<stanza>
<line>O Rose thou art sick.</line>
<line>The invisible worm,</line>
<line>That flies in the night</line>
<line>In the howling storm:</line>
</stanza>
</poem><poem> ... </poem>
</anthology>
```

```
<anthology>
<><>The SICK ROSE
<stanza>
<>O Rose thou art sick.
<>The invisible worm,
<>That flies in the night
<>In the howling storm:
```

```
<poem> ...
</anthology>
```



SGML is hard: Example

- 1) An anthology contains a number of poems and nothing else.
- 2) A poem always has a single title element which precedes the first stanza and contains no other elements.
- 3) Apart from the title, a poem consists only of stanzas.
- 4) Stanzas consist only of lines and every line is contained by a stanza.
- 5) Nothing can follow a stanza except another stanza or the end of a poem.
- 6) Nothing can follow a line except another line or the start of a new stanza.

```
<anthology>
<poem><title>The SICK ROSE</title>
<stanza>
<line>O Rose thou art sick.</line>
<line>The invisible worm,</line>
<line>That flies in the night</line>
<line>In the howling storm:</line>
</stanza>
</poem><poem> ... </poem>
</anthology>
```

```
<anthology>
<[1]><[2,3]>The SICK ROSE
[2,3]<stanza>
[4]<>O Rose thou art sick.
[4]<>The invisible worm,
[4]<>That flies in the night
[4]<>In the howling storm:
[3/5,1]
<poem> ...
</anthology>
```



SGML is hard:

Example

- 1) An anthology contains a number of poems and nothing else.
- 2) A poem always has a single title element which precedes the first stanza and contains no other elements.
- 3) Apart from the title, a poem consists only of stanzas.
- 4) Stanzas consist only of lines and every line is contained by a stanza.
- 5) Nothing can follow a stanza except another stanza or the end of a poem.
- 6) Nothing can follow a line except another line or the start of a new stanza.



XML Development Goals (W3C)

- Straightforward to use and easy to create
- Support of a variety of applications
- Compatibility with SGML
- Easy to write programs
- No optional features
- Human readable
- Terse



XML 1.0 (1998)

- W3C standard (version 1.1 in 2004)
- Simplification of SGML
 - (Tree structure)
 - Very regular language
- Simple yet powerful
- New grammars (XML Schema, ...)
- Separation of structure and formatting (CSS)

Changes: SGML to XML

Simplifications

1. Differences Between XML and SGML

XML allows only documents that use the SGML declaration in this note. This declares all the following SGML features as NO:

DATATAG
OMITTAG
OMITTAG
RANK
LINK (SIMPLE, IMPLICIT and EXPLICIT)
CONCUR
SUBDOC
FORMAL

Note that it differs from the reference concrete syntax in a number of ways:

It also declares no short reference delimiters. It follows that SHORTREF and USEMAP declarations cannot occur in XML.

The PIC (processing instruction close) delimiter is >.

Quantities and capacities are effectively unlimited.

Names are case sensitive (NAMECASE GENERAL is NO).

Underscore and colon are allowed in names.

Names can use Unicode characters and are not restricted to ASCII.

The following constructs which are permitted in SGML when SHORTTAG is YES are not allowed in XML:

Unclosed start-tags

Unclosed end-tags

Empty start-tags

Empty end-tags

Attribute values in attribute specifications entered directly rather than as literals

Attribute specifications that omit the attribute name

NET delimiters can be used only to close an empty element. In SGML without the Web SGML Adaptations Annex, the NET delimiter is declared as >. With this approach, XML is not allowing null end-tags and is allowing net-enabling start-tags only for elements with no end-tag. In SGML with the Web SGML Adaptations Annex, there is a separate NESTC (net-enabling start tag close) delimiter. This allows the XML <?> syntax to be handled as a combination of a net-enabling start-tag <? and a null end-tag >. With this approach, XML is allowing a net-enabling start-tag only when immediately followed by a null end-tag.

XML imposes the following restrictions not in SGML:

Entity references

Entity references must be closed with a REFC delimiter

References to external data entities in content are not allowed

General entity references in content are required to be synchronous

External entity references in attribute values are not allowed

Parameter entity references are allowed in the internal subset only within a declaration separator (that is, at a point where a markup declaration could occur)

Character references

Character references must be closed with a REFC delimiter

Named character references are not allowed

Numeric character references to non-SGML characters are not allowed

Entity declarations

A <DEFAULT> entity cannot be declared

External <SDATA> entities are not allowed

External <CDATA> entities are not allowed

Internal <SDATA> entities are not allowed

Internal <CDATA> entities are not allowed

PI entities are not allowed

Bracketed text entities are not allowed

External identifiers must include a system identifier

Attributes cannot be specified for an entity

The replacement text of general text entities and external parameter entities is required to be well-formed

An ampersand in a parameter literal must be followed by a syntactically valid entity reference or numeric character reference

Attribute definition list declarations

Associated element type in attribute definition list declarations cannot be a name group

Attributes cannot be declared for a notation

CURRENT attributes are not allowed

Content reference attributes are not allowed

NOTOKEN(S) declared values are not allowed

NUMBER(S) declared values are not allowed

NAME(S) declared values are not allowed

A name token group must use the or connector

Attribute values specified as defaults in attribute definition list declarations must be literals (SGML allows them not to be even when SHORTTAG is NO)

Element type declarations

Associated element type in element type declaration cannot be a name group

In an element declaration, a generic identifier can be specified as a rank stem and rank suffix (SGML allows this even when the RANK feature is NO)

Minimization parameters in element declarations are not allowed

RCDATA declared content are not allowed

CDATA declared content are not allowed

Content models cannot use the and connector

Content models for mixed content have a restricted form

Inclusions are not allowed

Exclusions are not allowed

Comments

A parameter separator cannot contain comments; this means that markup declarations (other than comment declarations) cannot contain comments

Empty comment declarations (<!-- in the reference concrete syntax) are not allowed

A comment declaration cannot contain more than one comment

In a comment declaration, an S separator is not allowed before the final MDC

Processing instructions

Processing instructions must start with a name (the PI target)

A processing instruction whose PI target is xml can only occur at the beginning of an external entity and must be an XML declaration if it occurs in the document entity, and otherwise a text declaration

A PI target must not match [a-zA-Z] unless it is xml

Marked sections

In marked section declarations, TEMP status keyword is not allowed

RCDATA marked sections are not allowed

INCLUDE/IGNORE marked sections are not allowed in the document instance

In a marked section declaration, a status keyword specification that contains no status keywords is not allowed

In a marked section declaration, a status keyword specification cannot contain more than one status keyword

Marked sections are not allowed in the internal subset

Parameter separators are not allowed in status keyword specifications in the document instance; in particular, parameter entity references are not allowed

Other

Names beginning with [a-zA-Z] are reserved

The SGML declaration must be implied and cannot be explicitly present in the document entity

When < and & occur as data, they must be entered as < and &

A parameter separator required by the formal syntax must always be present and cannot be omitted when it is adjacent to a delimiter

XML predefines the semantics of the attributes xml:space and xml:lang. It also reserves all attribute, element type and notation names beginning with [a-zA-Z].

XML requires that an SGML parser use an entity manager that behaves as follows:

Lines are terminated by newline (Unicode code #X000A) rather than being delimited by RS and RE as with a typical SGML entity manager

System identifiers are treated as URLs

The entity manager must support entities encoded in UTF-16 and UTF-8, and must be able automatically to detect which encoding an entity uses based on the presence of the byte order mark

The entity manager should be able to recognize the encoding declaration in the XML declaration and encoding PI and use it to determine the encoding of entity

XML imposes requirements on the information that a parser must make available to an application.

XML depends on the following changes to SGML, made by Web SGML Adaptations Annex:

HCORO delimiter (for hex numeric character references); for XML this is &#x

EMPTYNRM feature that allows elements declared EMPTY to have end-tags

NESTC delimiter

Duplicate enumerated attribute tokens are allowed

Relaxation of rules on use of parameter entity references inside groups

Multiple ATTLIST declarations for a single element type

ATTLIST declarations which don't declare any attributes

KEEPRESR feature that turns off SGML's rules for ignoring RSs and REs

Fully-tagged SGML documents: a document that is fully-tagged but not type-valid is a conforming SGML document; this makes all XML documents, including those that are well-formed but not valid, conforming SGML documents

Predefined data character entities in the SGML declaration (for it, amp and so on)

Unlimited capacities and quantities

The Web SGML Adaptations Annex also enables some XML restrictions to be enforced in SGML:

SHORTTAG is unfunded, so the SGML declaration can allow attribute defaulting and NET without allowing other SHORTTAG constructs

The SGML declaration can assert that a document is integrally stored, which disallows improperly nested entity references in content



XML Properties

- A Tree like structure = DOAG with one root
 - Root node: Document (virtual)
 - Inner node: Element
 - Leaf nodes:
 - Content
 - Comment
 - Processing instructions


Mr. the XML document

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE book SYSTEM "/etc/services/printers.dtd">
<driver id="driver/appledmp">
  <name>appledmp</name>
  <url>http://www.ghostscript.com/doc/gnu/7.05/Devices.htm#Apple<
/ur>
  <execution>
    <ghostscript/>
    <prototype>gs -q -dBATCH -dPARANOIDSAFER -dQUIET -dNOPAUSE
-sDEVICE=appledmp%A%Z -sOutputFile=- -</prototype>
  </execution>
  <printers>
    <printer>
      <id>printer/Apple-Dot_Matrix</id><!-- Apple Dot Matrix -->
    </printer>
  </printers>
</driver>
```




XML: Other changes

- Encoding, etc.
- The grammar is optional: separation of
 - XML Graph (validity)
 - Grammar-authorized subgraphs (well-formedness)
- Separation between
 - Format
 - Information Content
 - Content (eg. XHTML)
 - Presentation (eg. CSS)



Summary of (X)ML advantages

- Increased lifespan
- Communication between applications
- Documents can be modified with a simple text editor
- Information can be processed by external applications *easily* and *partially*
- Semantic

Summary: With(out) XML

```
[...]
00001470  00 00 00 00 00 00 00 00 00 00 02 00 d9 00 00 00 |.....|
00001480  48 00 65 00 6c 00 6c 00 6f 00 20 00 77 00 6f 00 |H.e.l.l.o. .w.o.|
00001490  72 00 6c 00 64 00 0d 00 00 00 00 00 00 00 00 00 |r.l.d.....|
000014a0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
00001680  00 04 00 00 18 04 00 00 fc 00 00 00 00 00 00 00 |.....|
[...]
```

```
<?xml version="1.0"?>
<w:wordDocument
  xmlns:w="http://schemas.microsoft.com/office/word/2003/wordml">
  <w:body>
    <w:p>
      <w:r>
        <w:rPr><w:b/></w:rPr><w:t>Hello World</w:t>
      </w:r>
    </w:p>
  </w:body>
</w:wordDocument>
```



Where to the XML come from?

- Web

- ☐ XHTML

- ☐ RDF

- ☐ ...

- More and more applications

- ☐ Open office

- ☐ Future Microsoft Office



Data or document?

■ Data-centric XML

- ☐ Machine consumption
- ☐ Very regular
- ☐ XML is “superfluous”

■ Document-centric XML

- ☐ Human consumption
- ☐ Less regular or irregular structure
- ☐ Mixed content



Data-centric

```
<SalesOrder SONumber="12345">  
  <Customer CustNumber="543">  
    <CustName>ABC Industries</CustName>  
    <Street>123 Main St.</Street>  
    <City>Chicago</City>  
    <State>IL</State>  
    <PostCode>60609</PostCode>  
  </Customer>  
  <OrderDate>981215</OrderDate>  
  <Item ItemNumber="1">  
    <Part PartNumber="123">
```

...

Document-centric

<Product>

<Intro>

The <ProductName>Turkey Wrench</ProductName> from

<Developer>Full

Fabrication Labs, Inc.</Developer> is <Summary>like a monkey wrench, but not as big.</Summary>

</Intro>

<Description><Para>The turkey wrench, which comes in
<i>both right- and left-

handed versions (skyhook optional)</i>, is made of the

...

XML usages

■ XML Databases

- Manage collection of XML documents
- Retrieve the information faster
 - ☞ Query languages, XML storage and indexing

■ XML Information Retrieval

- ☞ Constraints on Content and Structure
- ☞ Evaluation



XML Databases

Plan del curso



| XML Standards

- Plain XML
- Processing: SAX/DOM
- XML Grammars
 - DTD
 - Schema
 - Others
- Applications
 - XHTML + CSS, MathML, SVG, Semantic Web,
...



II Query Languages

- Fragment selection
 - XPath/XPointer (and others)
- Document transformation
 - XSL/XQuery
- Updating XML documents
- XML Views / Active XML



III XML Databases

- Relational vs Native
- XML documents storage
 - How to store?
 - How to update?
- Retrieving XML fragments
- XML databases properties
 - ACID and XML databases
 - Compression
 - Security



IV XML Information Retrieval

- Motivations
- What to retrieve?
 - Definition of relevance
- How to retrieve?
 - Query languages
- Evaluation
 - Metrics

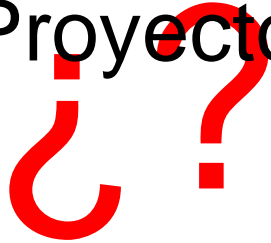
Evaluación

- Dos controles

- ☐ Hasta “XML Query Languages”
- ☐ Hasta “XML Databases”

- Examen

- Proyecto





Some references

■ GML

<http://www.sgmlsource.com/history/roots.html>

http://publibz.boulder.ibm.com/cgi-bin/bookmgr_05390/BOOKS/DSM05M00/CCONTENTS

■ XML en castellano

<http://www.programacion.net/html/xml/principal.htm>

■ O'Reilly

<http://www.xml.com>