# XML for SGMLers

Deborah Lapeyre and B. Tommie Usdin

**Mulberry Technologies, Inc.**
**17 West Jefferson Street, Suite 207**
**Rockville, MD 20850**
**Phone: 301/315-9631**
**Fax: 301/315-8285**
**info@mulberrytech.com**
**http://www.mulberrytech.com**

Slide 1

## What Does *This Class* Mean by "XML"?

- Extensible Markup Langauge (XML) 1.0
- W3C Recommendation 10-February-1998

http://www.w3.org/TR/REC-xml-19980210.xxx

Where xxx = .xml .html .pdf .ps

- Only XML, not associated standards

Slide 2

## The Family of XML Standards

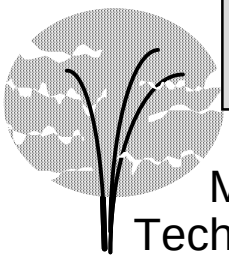
- "Written" to be part of XML
  - XLink
  - XPointer
- XSL
- "Using" or "for" XML
  - RDF (Resource Description Framework)
  - DOM (defines an API for XML)
- Replacements for XML syntax
  - DCD (Written in XML syntax!)
  - XML-Data (Written in XML syntax!)
  - Others

Slide 3

## XML Application "Standards"

- WIDL (Web Interface Definition Language)
- SMIL (Synchronized Multimedia Interface Definition Language)
- MathML
- CML (Chemical Markup Langague)
- XMI (XML Metadata Interchange Format)
- Many others

Slide 4



## What Does *This Class* Mean by "SGML"

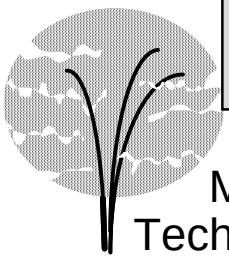
- SGML *before* XML
- Minus Web-SGML
- "SGML" means "Pre-TC SGML"

Slide 5

## Differences between SGML and XML for Your Application

- Document (Information) Analysis
- Documents
  - The logical document
  - Document instances
- DTDs
- The SGML Declaration
- The physical document

Slide 6



## Document (Information) Analysis

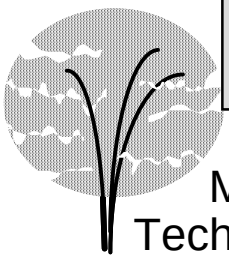
- Still necessary
- Still critical
- Still the basis for all applications
- Very little has changed
- Some structures slightly harder to model

Slide 7

## The Logical “Document”

- Still a prolog followed by an instance
- An SGML prolog contains (sometimes in separate files)
  - SGML Declaration
  - Document Type Declaration
  - Maybe some other obscure stuff
  - The tags and text that make a document (an instance)

Slide 8



## An XML Document Usually Contains (in one file)

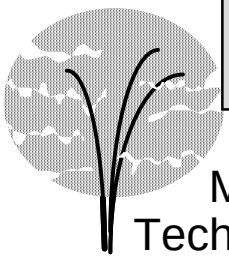
- XML Declaration
- Document Type Declaration
- The tags and text that make a document (an instance)

Slide 9

## Comparison of Logical Documents

| SGML | XML |
|---|---|
| • SGML Declaration | no equivalent |
| • no equivalent | XML Declaration |
| • DocType Declaration | same |
| • Instance | same |

Slide 10



## XML vs. SGML Declaration

- WARNING – These are very different things
- SGML Declaration
  - Sets all kinds of stuff
- XML Declaration
  - May set three variables

Slide 11

## The XML Declaration

<?xml version="1.0" encoding="UTF-8" standalone="no" >

Version of XML used?

Encoding Declaration:

*Name of the encoding: E.g.,
"UTF-8"
"UTF-16"
"EUC-JP"
"ISO-10646-UCS-2"

no = External DTD will affect parsing
yes = It won't

*Note: Encoding Declaration literal contains only Latin characters.

Slide 12

## A Very Basic XML Prolog(blue) and Document

```
<?xml version="1.0" encoding“UTF-8”?>
<address>
<str>17 Oak St.</str>
<city>Oakville</city>
<state>MD</state>
<ZIP>20895</ZIP>
</address>
```

Slide 13

## The Document Instance (the tags and text part)

- Unicode
- Names of element, attributes, etc.
- Entity and character references
- Empty elements
- Tag Minimization

Slide 14

## Unicode

- All XML is Unicode
- “UTF-8” is what you are probably using now
- XML Declaration names encoding
- You can mix encodings in a logical document

Slide 15

## Naming Rules

- Case Sensitive

  <tag> ≠ <TAG> ≠ <Tag> ≠ <TaG>
- MUST not start with
  - “xml”
  - “XML”
  - “Xml”
  - Or any combination

Slide 16

## Name start characters

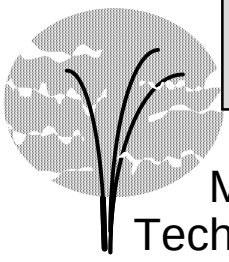
- Letters
- Underscore
- Colon

Slide 17

## What's in a Name

- Letters (full Unicode)
- Digits (full Unicode)
- Hyphen, underscore, period
- Colon (for namespaces)
- Combining Characters
- Extenders
- CANNOT EXPAND SET!

Slide 18



## Entity and Character References

- Must start with"&"
- MUST end with ";"
- Cannot reference external data entities
- Named character references not allowed (only numbered)
- "<" and "&" must be escaped to < and &

Slide 19

## Empty elements

- Typical SGML way:

```
<para>blah blah <graphic> blah
blah blah blah </para>
```

- XML way:

```
<graphic></graphic> OR
<graphic/>
```

Slide 20

## Tag Minimization

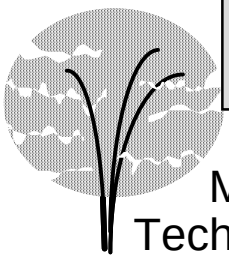
- All start and end tags MUST be there (NO omitted tags!)
- No SHORTREF
- No DataTag

Slide 21

## Attributes

- All attribute values that are specified
  - Must be quoted
  - Must have equal sign
- All “<” must be escaped to “<”
- Values may not contain external entity references

Slide 22



## Reserved Attributes

- xml:lang
  - In what language is this text
  - Values defined by IETF RFC 1766 (en, fr, de, i-yi, en-gb, x-klingon)
- xml:space
  - Controls whitespace handling within an element and its children
  - Preserve = keep all space

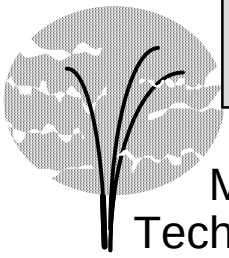
Slide 23

## Processing Instructions

- Delimiters are “<?” and “?>”
- First thing in a PI must be a name
- That name cannot be “xml”

```
YES <?lines, 3 extra, please    ?>
NO  <?3 extra lines, please      >
```

Slide 24



## Marked Sections (in Documents)

- No
  - INCLUDE or IGNORE
  - TEMP or RCDATA
- Only CDATA

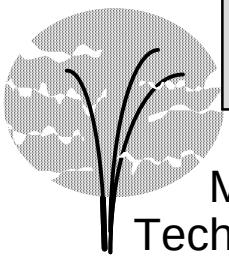
```
<[CDATA[ Tags and entities <such>
as this one are not <processed>]]>
```

Slide 25

## The DTD

- DTD optional in XML
- Well-formed versus Valid

Slide 26

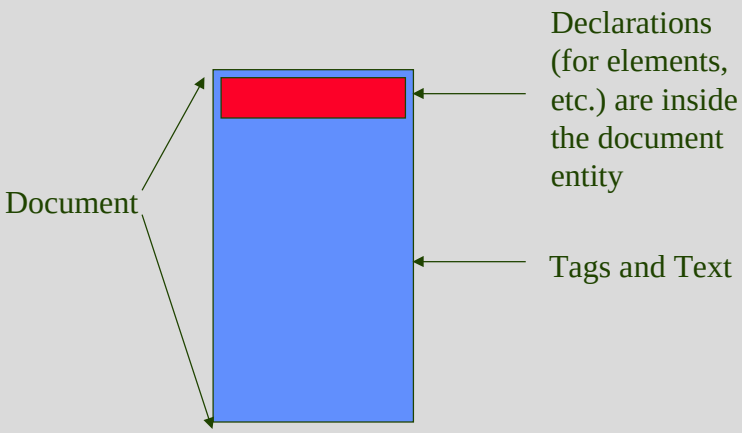


## Physical Location of the DTD

- Internal Subset
- External Subset

Slide 27

## Internal Subset



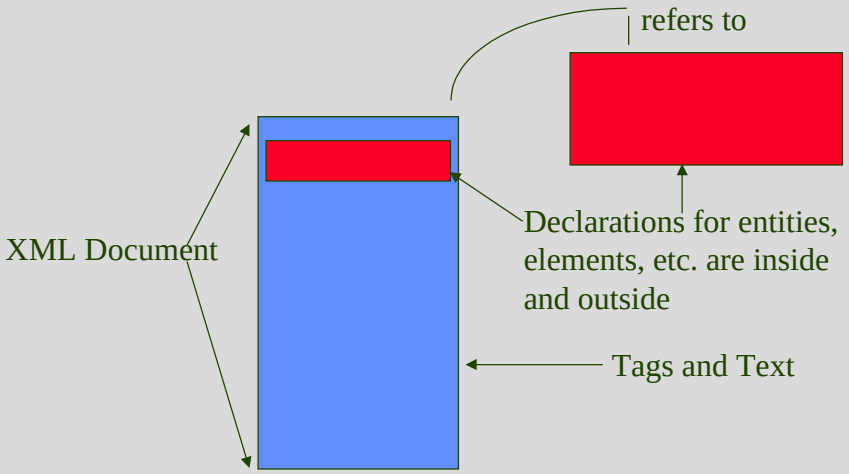

Slide 28

## XML Document with Internal Declarations

```
<?xml version="1.0" standalone="yes">
<!DOCTYPE memo [
<!ELEMENT memo (#PCDATA)           >
]>
<memo>This memo is all in the form of
character data.  This is legal XML,
but not usually useful XML.</memo>
```

Slide 29

## Internal and External Subset




Slide 30

## Document with Declarations Elsewhere

```
<?xml version="1.0" standalone="no">
<!DOCTYPE memo SYSTEM "sample.dtd">
<memo>This memo is all in the form
of character data.  This is legal
XML, but not usually useful XML.</memo>
```

- Blue italic is the External Identifier
- Says that there is an external DTD named "sample.dtd"

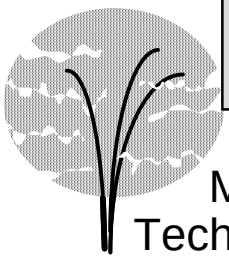
Slide 31

## Subset Parsing Order

- Internal Subset
- External Subset
- Instance

Therefore, the Internal Subset can override or add to the External Subset

Slide 32



## Differences in the written DTD

- Reserved words cannot be changed
- Reserved words in upper case only
  - DOCTYPE
  - ELEMENT
  - CDATA
  - #PCDATA
  - ATTLIST etc.

Slide 33

## Element Type Declarations

- No minimization indicators
- No inclusions
- No exclusions

```
<!ELEMENT memo  (to, body)      >
<!ELEMENT to    (#PCDATA)       >
<!ELEMENT body  (#PCDATA)       >
```

Slide 34

## Element Content Models

- No RCDATA or CDATA
- No “&” connector
- Cannot declare multiple elements in the same declaration

```
NO <!ELEMENT fred CDATA           >
NO <!ELEMENT fred RCDATA          >
NO <!ELEMENT fred (aa & bb)       >
NO <!ELEMENT (fred|sue) (#PCDATA) >
```

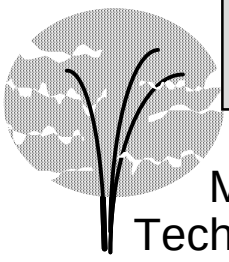
Slide 35

## Mixed Content Restriction

- #PCDATA must come first
- OR connector only
- Optional and repeatable

```
YES <!ELEMENT sue (#PCDATA | aaa)*>
NO <!ELEMENT sue  (aaa | #PCDATA)*>
NO <!ELEMENT sue (#PCDATA, aaa)*  >
NO <!ELEMENT sue (#PCDATA | aaa)+ >
```

Slide 36



## ATTLIST Syntax (1)

- Cannot declare attributes for multiple elements in the same declaration
- Default value must be quoted

```
NO <!ATTLIST (sue | fred)
        security CDATA #REQUIRED  >
NO <!ATTLIST sue
        rich (yes|no) no          >
```

Slide 37

## ATTLIST Syntax (2)

- Declared values types gone:
  - NAME and NAMES
  - NUMBER and NUMBERS
  - NUTOKEN and NUTOKENS
- Default values gone:
  - #CURRENT
  - #CONREF

Slide 38

## Declared Value Types Remaining

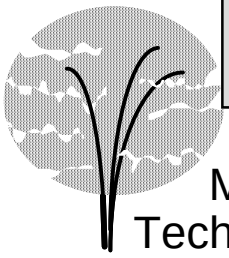
- ID
- CDATA
- ENTITY and ENTITIES
- IDREF and IDREFS
- NMTOKEN and NMTOKENS
- NOTATION

Slide 39

## Declared Value Types Remaining

- #REQUIRED
- #IMPLIED
- #FIXED
- A “value”

Slide 40



## What's Left

```
<!ELEMENT   graphic     EMPTY                          >
<!ATTLIST   graphic
   name           ID                      #IMPLIED
   status         (draft | final | dead)  "draft"
   lastupdate     CDATA                   #IMPLIED
   projectnos     NMTOKENS                #REQUIRED
   contractno     CDATA                   #FIXED "A432"
   real.file      ENTITY                  #REQUIRED
   pic.type       NOTATION (eps | tif)    "tif"       >
```

Slide 41

## Enumerated Attribute Values

- List of values must use OR connector
- Can use the same value for more than one attribute

```
<!ELEMENT fred (#PCDATA)            >
<!ATTLIST fred
        rich (yes|no) "no"
        cute (yes|no) "no"
        nice (yes|no) "yes"         >
```

Slide 42

## Multiple ATTLISTs

- More than one ATTLIST allowed per element
- Additive NOT replacing
- No two attributes of the same name

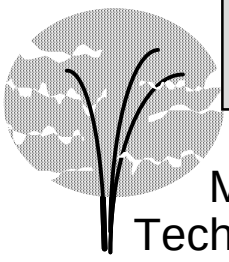
```
<!ELEMENT fruit (desc, recipe)>
<!ATTLIST fruit
        edible (yes|no)    "no"   >
<!ATTLIST fruit color CDATA "blue">
```

Slide 43

## Entity Declarations

- No SDATA entities
- No CDATA entities
- External ID MUST include SYSTEM (even if it includes PUBLIC)
- For General entities, replacement text must be well formed
  (can't start what you don't end!)

Slide 44



## SDATA Entities

- ISO 8879 sets of them (public identifiers)
  ISOlat1 ISOnum ISOpub etc.
- ISO sets contained SDATA entities
- Rewrittten as Unicode character entities

```
<!ENTITY reg SDATA "[reg   ]"    NO
  --/circledR =registered sign -->
<!-- Registered Trade Mark  -->
<!ENTITY reg "&#174;"          >
```

Slide 45

## Predefined General Entities

- Used to escape characters used as markup (“&”, “<“, Etc.)
- All XML processors must recognize
- All valid documents should declare them

| Entity | Displays As | Character Value |
|---|---|---|
| &amp; | & | & |
| &lt; | < | < |
| &gt; | > | &#62; |
| &apos; | ' | &#39; |
| &quot; | " | &#34; |

Slide 46

## Comments

- No inline comments
- No empty comments
- No space between "--" and ">"

```
NO <!ELEMENT sue (#PCDATA) --suzy-->
NO <!>
NO <!-- This is a comment   --     >
YES <!-- What a loverly comment. -->
```

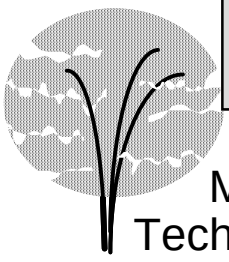
Slide 47

## Marked Sections (in DTDs)

- No
  - TEMP
  - RCDATA or CDATA
- Only INCLUDE and IGNORE

```
<[INCLUDE[
<!ELEMENT fred (#PCDATA)>     ]]>
<[IGNORE[
<!ELEMENT fred (name,title)>  ]]>
```

Slide 48



## The SGML Declaration

- There is NO equivalent
- Values set by SGML Declaration are:
  - Determined by XML Recommendation (standard)
  - No longer applicable
  - Set by XML Declaration (kind of)

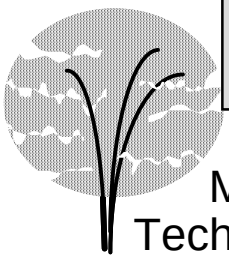
Slide 49

## Stuff from the SGML Declaration

- Unlimited Quantities and Capacities
- No Features

  RANK OMITTAG DATATAG
  LINK SUBDOC CONCUR
  FORMAL SHORTTAG
- Charsets are done with “Encoding”

Slide 50



## Other Miscellaneous Stuff

- Can't change angle brackets
- Can't rename the reserved words
- Whitespace handling has been simplified

Slide 51

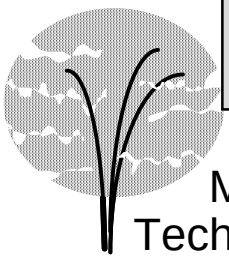
## Character References

- Used to be many different ways
- Now two:

  Ħ  hex

  ~  decimal

Slide 52



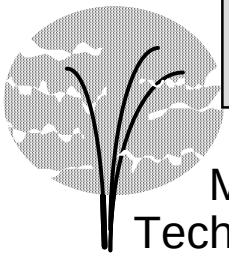
## Processing Documents in More than One Character Encoding

Slide 53

## The Physical Document

- Documents are still collections of physical entities
- XML document physically consists of
  - One “document entity”
  - (Optionally) Other named entities

Slide 54



## Parsed and Unparsed Entities

- Unparsed entity
  - Has associated Notation
  - Unconstrained (possibly non-XML) content
- Parsed entity
  - Contains text that is part of the document
  - Can be validated by an XML parser

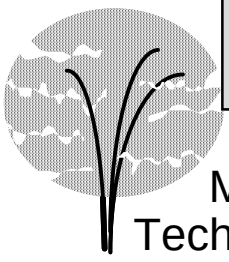
Slide 55

## Parsed Entity

- Each may use a different encoding
- May start with a Text Declaration
  - First line of a file (external parsed entity)
  - Names the version
  - Names the encoding

```
<?xml version="1.0"
        encoding="UTF-8" ?>
```

Slide 56



## SGML to XML: Getting Out of DTD Holes

- “&” connector
- Inclusions
- Exclusions
- Normalization

Slide 57

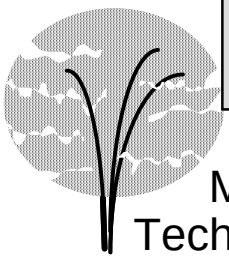
## The “&” Problem

```
<!ELEMENT memo
   (to+ & from & date & re?), p+) >

<!ELEMENT memo
   (to |from |date | re)+, p+)    >
```

- Loss of occurrence control
- Loss of requirement checking

Slide 58



## Exclusions

- Gone but not really missed
- Most uses no longer needed, since inclusions are gone
- Here's a loss though:

```
<!ELEMENT para (#PCDATA | ftnt)*>
<!ELEMENT footnote (para+) -(ftnt)>

<!ELEMENT footnote (para+)      >
```

Slide 59

## Inclusions: Here Be Problems

- An SGML-DTD using inclusions

```
<!ELEMENT doc (gee, wiz, zip) +(dot)  >
```

- Dots are possible
  - Before gee and after zip
  - Between gee and wiz, and wiz and zip
  - Anywhere inside gee, wiz, and zip
  - Anywhere inside all gee, wiz, and zip's children

Slide 60

## Ways to Solve Inclusions

- Add it to every content model
  - Obscures intent of model
  - May lead to ambiguous models if not done *very* carefully

```
<!ELEMENT doc
  (dot,gee,dot,wiz,dot,zip,dot)   >
BUT NOT:
<!ELEMENT doc
  (dot,gee?,dot,wiz,dot,zip?,dot) >
```

Slide 61

## Adding to Every Model (cont.)

Means ALL the way down

```
<!ELEMENT gee (PCDATA)            >
<!ELEMENT wiz (us|them)+          >
<!ELEMENT zip (#PCDATA | c)*      >

<!ELEMENT gee (PCDATA | dot)*     >
<!ELEMENT wiz (us | them | dot)+  >
<!ELEMENT zip (#PCDATA | c | dot)*>
```

Slide 62

## Inclusions: Other Ways

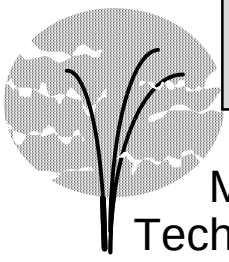
- Rethink the analysis
- Forget about the inclusion
- Forget the DTD, go for well-formed XML

Slide 63

## Normalization

- To add omitted tags
- To create correct empty tags
- Software

Slide 64



## Well-formed: Living without a DTD

XML adds the notion of “well-formed” data

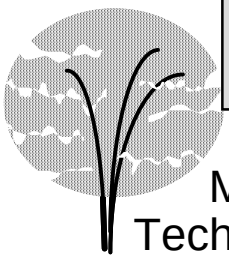
- Has no DTD
- A DTD could be written for the data
- Intended to be used by display tools

Slide 65

## Well-formed (1)

- Entity, element, attribute names are legal names
- One root element
- Every element that starts must end
- No element overlap
- Document at very least: (Prolog, element)

Slide 66



## Well-formed (2)

- Attributes:
  - No attribute name used more than once for one element
  - NO external entities in attribute values
  - All "<" are escaped to "<" in attribute values
  - NO external entities in attribute values
- Entities:
  - All "<"s in data are escaped to "<" or character reference
  - All ampersands in data are also escaped

Slide 67

## XML/SGMLTool Differences

- XML is easier:
  - For parser developers
  - For all software developers
  - For browsers to display
- XML is the same:
  - For creating valid documents
- XML adds complexity:
  - Well-formed can mean tags on the fly
  - For browsers to display

Slide 68

## So, No More DTDs, Right? Wrong!

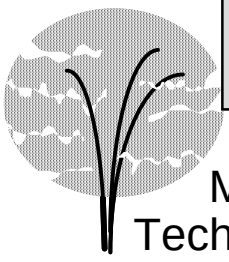
- DTDs (or equivalent) will be used for:
  - Consistency
  - Interchange
  - Most authoring
- DTDs might not be provided for:
  - Browsing
  - Simple searching

Slide 69

## We Left Out a Lot!

- XML applications will probably use/require bits of shared code, e.g. XLINK and XPOINTER
- XML applications may use alternate DTD syntax
- XML application are likely to be based on the DOM

Slide 70



## SGML is Changing, too!

- XML is SGML (now)
- New SGML tools may have cool new capabilities
  - Multiple ATTLISTS
  - Sole tag <TAG/>
  - Duplicate declared values
  - ...

Slide 71

## <? End of XML for SGMLers ?>

Slide 72