Lenguajes de Marcas y Sistemas de Información

UT6. XML: ESQUEMAS Y VOCABULARIOS

Mª JESÚS BRAVO 2024-2025

XML POR FUERA …

- XML se emplea para organizar información de cualquier tipo.
- Muy popular, está omnipresente en multitud de aplicaciones para cualquier uso.
- Es un formato de texto, estructurado usando etiquetas.
- Aunque tiene orígenes comunes y es también muy popular en Internet, **no es HTML**.

XML POR DENTRO …

```
<Cliente ID="HVet950283">
  <Nombre>Hospital Veterinario
  Kermit</Nombre>
  <Direccion verificada="si">
      <Calle>Padre Mier 1528</Calle>
      <Ciudad>Monterrey</Ciudad>
      <Estado>NL</Estado>
      <CodigoPostal>64000</CodigoPostal>
  </Direccion>
</Cliente>
```

- Elementos.
- Atributos
- Entidades
- Texto de contenido

Sintaxis simple

Legible por personas

Muy parecido al HTML

¿METALENGUAJE?

XML es solo un metalenguaje

- Formato general, básico, común, obligatorio.
- Base para lenguajes con usos reales, concretos.

Diversos modelos para desarrollar cada lenguaje

  – Gramáticas, vocabularios, etc.

- Document Type Definition (DTD).
- Esquemas (XMLSchema).

Necesidad de corrección a los dos niveles:

- Documentos **bien formados**.
- Documentos **válidos**.

ESQUEMAS Y VOCABULARIOS EN XML

Existen otros tipos, por ejemplo

- RelaxNG

```
<element name="addressBook" xmlns="http://relaxng.org/ns/structure/1.0">
        <zeroOrMore>
                <element name="card">
                 <element name="name">
                        <text/>
                 </element>
                <element name="email">
                        <text/>
                </element>
        </element>
    </zeroOrMore>
  </element>
```

- Schematron

```
<schema xmlns="http://purl.oclc.org/dsdl/schematron">
  <pattern>
        <title>Date rules</title>
        <rule context="Contract">
              <assert test="ContractDate &lt; current-date()">.....</assert>
        </rule>
  </pattern>
</schema>
```

QUÉ ES DTD

- **DTD** (***D****ocument* ***T****ype* ***D****efinition*, Definición de Tipo de Documento) sirve para definir la estructura de un documento SGML o XML, permitiendo su validación.
  - **SGML** (***S****tandard* ***G****eneralized* ***M****arkup* ***L****anguage*, Lenguaje de Marcado Generalizado Estándar).
  - **XML** (*e****X****tensible* ***M****arkup* ***L****anguage*, Lenguaje de Marcado eXtensible) es un lenguaje desarrollado por **W3C** (*World Wide Web Consortium*) que está basado en SGML.
- Un documento XML es válido (*valid*) cuando, además de estar bien formado, no incumple ninguna de las normas establecidas en su estructura.

DECLARACIÓN DE TIPO DE DOCUMENTO

- Un DTD se puede escribir tanto interna como externamente a un archivo XML. ¿Cuándo utilizar una u otra?

- En ambos casos hay que escribir una definición **`DOCTYPE`** (*Document Type Declaration*, Declaración de Tipo de Documento) para asociar el documento XML al DTD.

- Asimismo, un archivo XML se puede asociar simultáneamente a un DTD interno y externo.

DOCUMENTO XML ASOCIADO A UN DTD INTERNO

- Sintaxis:

```
<!DOCTYPE elemento-raíz [ declaraciones ]>
```

- **EJEMPLO**

```
<?xml version="1.0" encoding="UTF-8"?>
<! DOCTYPE etiqueta[
<!ELEMENT etiqueta (nombre, calle, ciudad, pais,
codigo)>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT calle (#PCDATA)>
<!ELEMENT ciudad (#PCDATA)>
<!ELEMENT pais (#PCDATA)>
<!ELEMENT codigo (#PCDATA)>
]>
```

DOCUMENTO XML ASOCIADO A UN DTD INTERNO

```
<! DOCTYPE etiqueta[
<!ELEMENT etiqueta (nombre, calle, ciudad, pais, codigo)>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT calle (#PCDATA)>
<!ELEMENT ciudad (#PCDATA)>
<!ELEMENT pais (#PCDATA)>
<!ELEMENT codigo (#PCDATA)>
]>

<etiqueta>
<nombre>Fulano Mengánez</nombre>
<calle>c/ Mayor, 27</calle>
<ciudad>Valderredible</ciudad>
<pais>España</pais>
<codigo>39343</codigo>
</etiqueta>
```

DOCUMENTO XML ASOCIADO A UN DTD INTERNO

- ¿La DTD se muestra en el navegador?
- Validar un documento XML asociado a una DTD con XML Copy Editor
- Validar un documento XML asociado a una DTD con VS Code y XML Notepad++ (con XML Tools)

DOCUMENTO XML ASOCIADO A UN DTD EXTERNO

Existen dos tipos de DTD externa: privada y pública.

- Sintaxis DTD externa privada:

<!DOCTYPE elemento-raíz SYSTEM "URI">

- Sintaxis DTD externa pública:

<!DOCTYPE elemento-raíz PUBLIC "identificador-público" "URI">

- Ejemplo:

```
<!DOCTYPE raiz SYSTEM "fichero.dtd">
```

DECLARACIÓN DE ELEMENTOS EN UN DTD

- Sintaxis:

```
<!ELEMENT nombre-del-elemento tipo-de-contenido>
```

- En el **tipo-de-contenido** se especifica el contenido permitido en el elemento, pudiendo ser:
  - Texto, (#PCDATA).
  - Otros elementos (hijos).
  - Estar vacío, EMPTY.
  - Mixto (texto y otros elementos)
  - ANY (cualquier contenido).

EL CONTENIDO DE UN ELEMENTO PUEDE SER TEXTO -(#PCDATA)

- **EJEMPLO** En el siguiente documento XML, el elemento “ciudad” puede contener cualquier texto (cadena de caracteres):

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ciudad [
<!ELEMENT ciudad (#PCDATA)>
]>
<ciudad>Roma</ciudad>
```

- Escribiendo #PCDATA (Parsed Character Data) entre paréntesis “()”, se ha indicado que el elemento “ciudad” puede contener una cadena de caracteres.

UN ELEMENTO PUEDE CONTENER A OTROS ELEMENTOS

EJEMPLO En el siguiente ejemplo, el elemento “ciudad” contiene a los elementos “nombre” y “pais”:

```
<?xmlversion="1.0" encoding="UTF-8"?>
<!DOCTYPE ciudad [
  <!ELEMENT ciudad (nombre, pais)>
  <!ELEMENT nombre (#PCDATA)>
  <!ELEMENT pais(#PCDATA)>
]>
<ciudad>
    <nombre>Roma</nombre>
    <pais>Italia</pais>
</ciudad>
```

Nota: En un documento XML, los elementos (hijos) de un elemento (padre), deben escribirse en el mismo orden en el que han sido declarados en la DTD

ELEMENTOS VACÍOS - EMPTY

- Para declarar un elemento vacío en una DTD, hay que indicar que su contenido es **EMPTY**.

- Un ejemplo de ello podría ser el elemento como el "br" del HTML, el cual sirve para hacer un salto de línea y no tiene contenido:

```
<!ELEMENT br EMPTY>
```

- Dada la declaración anterior, en un documento XML el elemento "br" podría escribirse como:

  O también:
</br>

ELEMENTOS MIXTOS- MIXED

- **EJEMPLO**

```
<!ELEMENT note(#PCDATA|to|from|header|message)>
```

declara que el elemento "note" puede contener elementos alfanuméricos, "to", "from", "header", or "message"

ELEMENTOS CUALQUIER CONTENIDO - ANY

- **EJEMPLO** En la DTD interna del siguiente documento XML, se ha indicado que el elemento “persona” puede contener texto y otros elementos, es decir, contenido mixto, **ANY**:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE persona [
   <!ELEMENT persona ANY>
   <!ELEMENT nombre (#PCDATA)>
   <!ELEMENT ciudad (#PCDATA)>
]>
<persona>
 <nombre>Elsa</nombre> vive en <ciudad>
  Pamplona </ciudad>.
</persona>
```

SECUENCIAS DE ELEMENTOS

- En una DTD, un elemento (padre) puede ser declarado para contener a otro u otros elementos (hijos).
- En la sintaxis, los hijos –también llamados sucesores–tienen que escribirse entre paréntesis “()” y separados por comas “,”.

SECUENCIAS DE ELEMENTOS

EJEMPLOS

- Con un solo tipo de sub-elemento:

```
<!ELEMENT aviso (parrafo)>
```

- La coma denota una secuencia:

```
<!ELEMENT aviso (titulo, parrafo)>
```

- "|“ indica una opción:

```
<!ELEMENT aviso (parrafo | grafico)>
```

- El número de opciones no está limitado a dos, y se pueden agrupar usando paréntesis:

```
<!ELEMENT aviso (titulo, (parrafo | grafico))>
```

INDICADORES DE FRECUENCIA

- En un DTD, para definir el número de veces que pueden aparecer los elementos, se pueden utilizar los indicadores de frecuencia u operadores de cardinalidad, mostrados en la siguiente tabla:

Operador	*Cardinalidad*	*Significado*
? (*interrogación*)	0-1	El elemento es opcional, pudiendo aparecer una sola vez o ninguna.
***** (*asterisco*)	0-n	El elemento puede aparecer cero, una o más veces.
+ (*signo más*)	1-n	El elemento tiene que aparecer, obligatoriamente, una o más veces.

```
<!ELEMENT aviso (titulo?, (parrafo+, grafico)*)>
```

OPERADOR DE ELECCIÓN "|" Y OPERADOR "*"

EJEMPLO En la DTD del siguiente documento XML se indica que el elemento "articulos" puede contener varios elementos "codigo" e "id":

```
<?xmlversion="1.0" encoding="UTF-8"?>
<!DOCTYPE articulos[
    <!ELEMENT articulos(codigo|id)*>
    <!ELEMENT codigo(#PCDATA)>
    <!ELEMENT id(#PCDATA)>
]>
<articulos>
    <codigo>AF-32</codigo>
    <id>3891</id>
    <codigo>AF-50</codigo>
    <codigo>AF-89</codigo>
</articulos>
```

DECLARACIONES DE LISTA DE ATRIBUTOS

- La principal diferencia entre los elementos y los atributos es que los atributos no pueden contener sub-atributos

```
<mensaje prioridad="urgente">
    <de>Alfredo Reino</de>
    <a>Hans van Parijs</a>
    <texto idioma="holandés">
        Hallo Hans, hoe gaat het?
    ...
    </texto>
</mensaje>
```

- Otra diferencia es que cada uno de los atributos sólo se puede especificar una vez, y en cualquier orden.

DECLARACIONES DE LISTA DE ATRIBUTOS

- La sintaxis básica para declarar un atributo en una DTD es:

  `<!ATTLIST` etiqueta atributo (**tipo**) uso `>`

Tipo:

- NOTATION -> se ajusta a una notación determinada
- CDATA -> alfanumérico
- NMTOKEN/S -> solo los caracteres válidos
- ID -> nombre único, no puede empezar por número y no puede tener espacios
- IDREF/S -> hace referencia a un atributo ID de otro elemento

DECLARACIONES DE LISTA DE ATRIBUTOS

Tipo	*Descripción*
CDATA	(*Character DATA*) El valor son datos de tipo carácter, es decir, texto.
Enumerado	El valor puede ser uno de los pertenecientes a una lista de valores escritos entre *paréntesis* " **()** " y separados por el carácter " **|** ".
ID	El valor es un identificador único.
IDREF	El valor es un identificador que tiene que existir en otro atributo **ID** del documento XML.
IDREFS	El valor es una lista de valores que existan en otros atributos **ID** del documento XML, separados por espacios en blanco.
NMTOKEN	El valor es una cadena de caracteres, pudiendo contener letras minúsculas, letras mayúsculas, números, puntos " . ", guiones medios "**-**", guiones bajos "**_**" o el carácter dos puntos "**:**".
NMTOKENS	El valor puede contener uno o varios valores de tipo **NMTOKEN** separados por espacios en blanco.
NOTATION	El valor es el nombre de una notación.
ENTITY	El valor es el nombre de una entidad.
ENTITIES	El valor puede contener uno o varios valores de tipo **ENTITY** separados por espacios en blanco.
Especiales	Existen dos atributos especiales: **xml:lang** y **xml:space**.

DECLARACIONES DE LISTA DE ATRIBUTOS

<!ATTLIST etiqueta atributo (tipo) **uso** >

Uso:

- #REQUIRED-> no hay valor por defecto, hay que especificarlo
- #IMPLIED -> se puede omitir
- Valor por defecto (valor entre comillas)

DECLARACIONES DE LISTA DE ATRIBUTOS

Uso	*Significado*
`valor` entre comillas dobles (") o simples (').	El atributo tiene un valor por defecto.
`#REQUIRED`	El atributo es obligatorio escribirlo.
`#IMPLIED`	El atributo es opcional escribirlo.
`#FIXED valor` entre comillas dobles (") o simples (').	El valor del atributo es fijo.

DECLARACIONES DE LISTA DE ATRIBUTOS

EJEMPLO:

```
<?xmlversion="1.0" encoding="UTF-8"?>
<!DOCTYPE mensaje[
    <!ELEMENT mensaje (de, a, texto)>
    <!ATTLIST mensaje prioridad (normal | urgente) ‘normal’>
    ...
    <!ELEMENT texto (#PCDATA)>
    <!ATTLIST texto idioma CDATA #REQUIRED>
]>
<mensaje prioridad="urgente">
  <de>Alfredo Reino</de>
  <a>Hans van Parijs</a>
  <texto idioma="holandés">
        Hallo Hans, hoe gaat het?
  ...
  </texto>
</mensaje>
```

DECLARACIONES DE LISTA DE ATRIBUTOS

EJEMPLO:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE deportistas [
    <!ELEMENT deportistas (futbol | f1 | tenis)*>
    <!ELEMENT futbol (#PCDATA)>
    <!ELEMENT f1 (#PCDATA)>
    <!ATTLIST f1 pais (ESP | FRA | ITA | ALE) "ESP">
    <!ELEMENT tenis (#PCDATA)>
]>
<deportistas>
    <f1 pais="ALE">Sebastian Vettel</f1>
    <f1>Fernando Alonso</f1>
    <f1 pais="ESP">Carlos Sainz</f1>
    <tenis>Rafael Nadal</tenis>
</deportistas>
```

DECLARACIONES DE LISTA DE ATRIBUTOS

EJEMPLO:

```
<?xmlversion="1.0" encoding="UTF-8"?>
<!DOCTYPE deportistas [
   <!ELEMENT deportistas (futbol | f1)*>
   <!ELEMENT futbol (#PCDATA)>
   <!ELEMENT f1 (#PCDATA)>
   <!ATTLIST f1 código ID #REQUIRED>
]>
<deportistas>
   <f1 codigo="ALO">Fernando Alonso</f1>
   <f1 codigo="VET">Sebastian Vettel</f1>
</deportistas>
```

DECLARACIONES DE LISTA DE ATRIBUTOS

```
<?xmlversion="1.0" encoding="UTF-8"?>
<!DOCTYPE cine [
    <!ELEMENT cine (directores, peliculas)>
    <!ELEMENT directores (director)*>
    <!ELEMENT director (#PCDATA)>
    <!ATTLIST director coddir ID #REQUIRED>
    <!ELEMENT peliculas(pelicula)*>
    <!ELEMENT pelicula(#PCDATA)>
    <!ATTLIST película direccion IDREF #REQUIRED>
]>
<cine>
    <directores>
        <director coddir="CE">Clint Eastwood</director>
        <director coddir="JC">James Cameron</director>
    </directores>
    <peliculas>
        <película direccion="JC">Avatar</pelicula>
        <película direccion="CE">MysticRiver</pelicula>
        <película direccion="JC">Titanic</pelicula>
    </peliculas>
</cine>
```

DECLARACIONES DE LISTA DE ATRIBUTOS

```
<?xmlversion="1.0" encoding="UTF-8"?>
<!DOCTYPE cine [
    <!ELEMENT cine (peliculas, directores)>
    <!ELEMENT peliculas(pelicula)*>
    <!ELEMENT pelicula(#PCDATA)>
    <!ATTLIST película codpel ID #REQUIRED>
    <!ELEMENT directores (director)*>
    <!ELEMENT director (#PCDATA)>
    <!ATTLIST director filmografía IDREFS #REQUIRED>
]>
<cine>
<peliculas>
    <película codpel="P1">Avatar</pelicula>
    <película codpel="P2">MysticRiver</pelicula>
    <película codpel="P3">TheTerminator</pelicula>
    <película codpel="P4">Titanic</pelicula>
</peliculas>
<directores>
    <director filmografia="P2">Clint Eastwood</director>
    <director filmografia="P1 P3 P4">James Cameron</director>
</directores>
</cine>
```

DECLARACIONES DE LISTA DE ATRIBUTOS

EJEMPLO:

```
<?xmlversion="1.0" encoding="UTF-8"?>
<!DOCTYPE usuarios [
    <!ELEMENT usuarios (usuario)*>
    <!ELEMENT usuario (#PCDATA)>
    <!ATTLIST usuario clave NMTOKEN #REQUIRED>
]>
<usuarios>
    <usuario clave="123456789">Ana</usuario>
    <usuario clave="ab-c-d-fg">Iker</usuario>
    <usuario clave="A1_B2..C3">Elsa</usuario>
</usuarios>
```

En el valor de un atributo NMTOKEN **no** se pueden escribir espacios en blanco ni caracteres especiales, tales como: *, $, %, &, ?, @...

DECLARACIONES DE LISTA DE ATRIBUTOS

EJEMPLO:

```
<?xmlversion="1.0" encoding="UTF-8"?>
<!DOCTYPE usuarios [
   <!ELEMENT usuarios (usuario)*>
   <!ELEMENT usuario (#PCDATA)>
   <!ATTLIST usuario códigos NMTOKENS #REQUIRED>
]>
<usuarios>
   <usuario codigos="1234 567 89">Ana</usuario>
   <usuario codigos="ab c-d fg">Iker</usuario>
   <usuario codigos="A1:B2">Elsa</usuario>
</usuarios>
```

DECLARACIÓN DE ENTIDADES

- XML puede hacer referencia a objetos (ficheros, páginas web, imágenes, cualquier cosa) que no deben ser analizados sintácticamente según las reglas de XML, mediante el uso de entidades.
- Se declaran en la DTD mediante el uso de **"<!ENTITY"**
- Una entidad puede no ser más que una abreviatura que se utiliza como una forma corta de algunos textos.
- Al usar una referencia a esta entidad, el analizador sintáctico reemplaza la referencia con su contenido.
- Las entidades pueden ser:
  - Internas
  - Externas
  - Párametro

DECLARACIÓN DE ENTIDADES INTERNAS

- Las más sencillas. Son básicamente abreviaturas definidas en la sección de la DTD del documento XML.
- Son siempre entidades analizadas, es decir, una vez reemplazada la referencia a la entidad por su contenido, pasa a ser parte del documento XML y como tal, es analizada por el procesador XML.

```
<!ENTITY alf  "Alien Life From">
```

- Para utilizarlas en el xml se utiliza &

```
<texto><titulo> Un día en la vida de un &alf;
</titulo></texto>
```

DECLARACIÓN DE ENTIDADES INTERNAS

EJEMPLO

```
<?xmlversion="1.0" encoding="UTF-8"?>
<!DOCTYPE textos [
   <!ELEMENT textos (texto)+>
   <!ELEMENT texto (#PCDATA)>
   <!ENTITY escritor "Miguel de Cervantes">
   <!ENTITY obra "El Quijote">
   <!ENTITY fecha "29/09/1947">
]>
<textos>
   <texto>&obra;fue escrito por
   &escritor;.</texto>
   <texto>&escritor;nació el &fecha;.</texto>
</textos>
```

DECLARACIÓN DE ENTIDADES EXTERNAS

- Las entidades externas obtienen su contenido en cualquier otro sitio del sistema, ya sea otro archivo del disco duro, una página web o un objeto de una base de datos.
- Se hace referencia al contenido de una entidad así mediante la palabra SYSTEM seguida de un URI (Universal Resource Identifier)

```
<!ENTITY intro SYSTEM
"http://www.miservidor.com/intro.xml">
```

DECLARACIÓN DE ENTIDADES EXTERNAS

EJEMPLO

```
<?xmlversion="1.0" encoding="UTF-8"
standalone="no"?>
<!DOCTYPE textos [
   <!ELEMENT textos (texto)+>
   <!ELEMENT texto (#PCDATA)>
   <!ENTITY escritor SYSTEM "escritor.txt">
]>
<textos>
   <texto>El Quijote fue escrito por
   &escritor;.</texto>
</textos>
```

DECLARACIÓN DE ENTIDADES PARÁMETROS

- **Sólo pueden usarse en la DTD**, y no en el documento XML.
- Se suele utilizar para agrupar ciertos elementos del DTD que se repitan mucho.
- Se diferencian de las entidades generales, en que para hacer referencia a ellas, se usa el símbolo "**%**" en lugar de "&" tanto para declararlas como para usarlas.
- Tienen que declararse antes de ser referenciadas.

```
<!ENTITY % p "(#PCDATA)">
<!ELEMENT persona (nombre, mayor_de_edad?,
ciudad)>
<!ELEMENT nombre %p;>
<!ELEMENT mayor_de_edad EMPTY>
<!ELEMENT ciudad %p;>
```

DECLARACIÓN DE ENTIDADES PARÁMETROS

EJEMPLO

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE persona [
    <!ENTITY % persona SYSTEM "persona.dtd">
    %persona;
]>
<persona>
    <nombre>Iker</nombre>
    <mayor_de_edad/>
    <ciudad>Pamplona</ciudad>
</persona>
```

El archivo “**persona.dtd**”podría contener, por ejemplo:

```
<!ELEMENT persona (nombre, mayor_de_edad?, ciudad)>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT mayor_de_edad EMPTY>
<!ELEMENT ciudad (#PCDATA)>
```