1.- Introducción

- XML ha sido propuesto como un estándar en el intercambio de información (emisor, receptor).
- No sólo es imprescindible que el documento esté bien formado sino que emisor y receptor se ciñan a un formato de fichero definido previamente ➔ Se ha de definir una estructura fija del documento.

2.- Modelado de datos XML

- Describe la organización de los datos de marcado y de los caracteres en un documento XML válido.
- Un documento XML debe adherirse a un esquema para ser válido, no así para estar bien formado.
- Un esquema permite tener múltiples documentos XML.

En términos de BDR un esquema XML podría compararse al esquema de Tabla o BD, y un documento XML al contenido de las tablas en un momento dado.

Esquema

- Los esquemas permiten describir vocabularios XML, de manera que estos permitan el entendimiento entre organizaciones.
- Posibilitan la creación de vocabularios XML específicos para crear y utilizar documentos XML acordes.
- Establecen restricciones en la estructura del contenido de un documento:
  - Modelo de contenido de los documentos: orden y anidamiento de elementos.
  - Tipos de datos del documento.

Alternativas

- **DTD (*Document Type Definition*)**
  - Origen en SGML.
  - Sintaxis especializada, no XML.
  - Contenidos limitados a cadenas de caracteres.
- **XML Schema**
  - Permite asociar tipos de datos: validación de contenidos.
  - Modelo de contenido abierto.
  - Utilización de espacios de nombres.
  - Esquema expresado en sintaxis XML (es un documento XML).

3.- DTD

- *Document Type Definition*.
- Determina la estructura de documentos XML.
  - Define los elementos y atributos que pueden aparecer en el documento XML.
- Puede ubicarse:
  - Dentro del propio documento XML (interna).
  - En un fichero aparte (externa).
  - De forma mixta (las declaraciones de la DTD interna prevalecen).

Declaración del DTD: <!DOCTYPE>

La sintaxis varía dependiendo de la ubicación (interna, externa, mixta) y del carácter (público, privado).

```
Interno (luego privado):
  <!DOCTYPE elemento raíz [reglas]>
Externo y privado:
  <!DOCTYPE elemento raíz SYSTEM URL>
Mixto y privado:
  <!DOCTYPE elemento raíz SYSTEM URL [reglas]>
Externo y público:
  <!DOCTYPE elemento raíz PUBLIC FPI URL>
Mixto y público:
  <!DOCTYPE elemento raíz PUBLIC FPI URL [reglas]>
```

Atributos DOCTYPE

- Nombre del elemento raíz del documento XML.
- Declaración de privacidad: `SYSTEM` o `PUBLIC`.
- Identificador: Sólo cuando el DTD es `PUBLIC` e indica el FPI (*Formal Public Identifier*) por el que se conoce el DTD.
- URL: Sólo cuando el DTD es externo (o al menos en parte, mixto), del que da su ubicación.

FPI

- *Formal Public Identifier*, etiqueta que identifica al DTD de manera "universal".
- 4 campos separados por //:
  - Norma formal (+ si organismo no oficial, nada si organismo oficial) o no formal (-).
  - Nombre del organismo responsable del estándar.
  - Tipo de documento que se describe.
  - Idioma.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
                      Transitional//EN"
http://www.w3.org/TR/xhtml11/DTD/shtml11-transitional.dtd>
```

¿Qué tipo de DTD utilizar?

- Interna:
  - si no se requiere validación externa.
  - se va a crear un solo documento XML.
  - se quiere minimizar el coste asociado con los documentos.
- Externa:
  - si se requiere validación externa.
  - cuando existan o puedan existir múltiples documentos XML de la misma clase.
  - cuando se quiera utilizar una DTD ya existente.
  - cuando se quieran documentos XML más concisos.

DTD interna

- Debe seguir la siguiente sintaxis:

```
<!DOCTYPE elemento raíz [declaración de elementos]>
```

- Ejemplo:

```
<?xml version="1.0"?>
<!DOCTYPE nota [<!ELEMENT nota (para,de,cabecera,cuerpo)>
  <!ELEMENT para (#PCDATA)>
  <!ELEMENT de (#PCDATA)>
  <!ELEMENT cabecera (#PCDATA)>
  <!ELEMENT cuerpo (#PCDATA)>
]>
<nota>
  <para>Jose</para>
  <de>Juani</de>
  <cabecera>Recordatorio</cabecera>
  <cuerpo>NO me olvides este finde</cuerpo>
</nota>
```

¿Para qué utilizar una DTD?

- Con una DTD, cada uno de los archivos XML se puede llevar a una descripción de su propio formato.
- Con una DTD, grupos independientes de personas se ponen de acuerdo para utilizar una DTD estándar para intercambiar datos.
- Su aplicación puede utilizar una norma DTD para verificar que los datos que recibimos del mundo exterior es válida.
- También se puede utilizar un DTD para verificar sus propios datos.

Limitaciones de los DTD

- Un DTD no es un documento XML, luego no se puede verificar si está bien formado.
- No se pueden fijar restricciones sobre los valores de elementos y atributos, como su tipo de datos, su tamaño, etc.
- No soporta espacios de nombres.
- Sólo se pueden enumerar los valores de atributos, no de elementos.
- Sólo se puede dar un valor por defecto para atributos, no para elementos.
- Existe un control limitado sobre las cardinalidades de los elementos, es decir, concretar el número de veces que pueden aparecer.

Componentes del DTD

- Elemento.
- Atributo.
- Entidad.
- Notación.

3.1.- Elementos

- Declaración:

```
<!ELEMENT nombre elemento modelo contenido>
```

- Nombre del elemento:
  - determina el nombre de la etiqueta de marcado.
  - debe ser único dentro de una DTD.
  - no debe contener & o empezar por XML, Xml, xml, …
- Modelo de contenido:
  - Regla.
    - ANY.
    - EMPTY.
  - Datos.
  - Elementos descendientes.
  - Contenido mixto.

Elementos ANY

```
<!ELEMENT nombreElemento ANY>
```

- "Cualquier cosa".
- Se puede utilizar al construir el DTD para dejar la descripción de un elemento como válida en cualquier caso, eliminando cualquier comprobación sintáctica.
- Es un "comodín" que no debería aparecer en el DTD definitivo.

Elementos EMPTY

```
<!ELEMENT nombreElemento EMPTY>
```

- Elemento vacío. Sin descendientes.
- Pueden contener atributos.
- Utilización en documento XML:
  - Etiquetas de inicio y cierre.

    ```
    <imagen fichero="Titulo.gif"></imagen>
    ```

  - Etiqueta vacía.

    ```
    <imagen fichero="Titulo.gif"/>
    ```

Elementos de datos

```
<!ELEMENT nombreElemento (#PCDATA)>
```

Elementos de elementos

- Contienen elementos secundarios.
- Se declaran especificando un modelo de contenido en el elemento, sólo con elementos secundarios.
- Declaración en DTD:

```
<!ELEMENT nombreElemento ModeloContenido>
```

- Modelo de contenido:
  - Patrón que establecen los elementos secundarios participantes y el orden de aparición.
  - Combinación de símbolos y elementos secundarios.

3.2.- Atributos

- Especifican información adicional de un elemento.
- Declaración en DTD:

```
<!ATTLIST elemento atributo tipo-atributo caracter>
```

- Ejemplo DTD:

```
<!ATTLIST pago tipo CDATA "cheque">
```

- Ejemplo XML:

```
<pago tipo="cheque" />
```

- En general se utiliza un solo ATTLIST para declarar todos los atributos de un elemento (aunque pueden usarse varios).
- El nombre del atributo debe ser un nombre XML válido.

Carácter

- Valor textual entre comillas, que representa un valor por defecto para el atributo.
- #IMPLIED, el atributo es de carácter opcional y no se le asigna ningún valor por defecto.
- #REQUIRED, el atributo es de carácter obligatorio, pero no se le asigna un valor por defecto.
- #FIXED, el atributo es de carácter obligatorio y se le asigna un valor por defecto que además es el único valor que puede tomar.

Tipos de atributos

- CDATA. Caracteres.
- Enumerados. Listas de valores de entre los cuáles el atributo debe tomar uno.
- ID (identificador único) / IDREF (valor de ID de algún elemento) / IDREFS (múltiples IDs de otros elementos)
- NMTOKEN (nombre sin espacios) / NMTOKENS (lista de nombres sin espacios).
- ENTITY (nombre de una entidad) / ENTITIES (lista de nombres de entidades).
- NOTATION. Nombre de notación.

Atributos CDATA

Datos de caracteres no analizados sintácticamente.

```
<!ATTLIST persona   nombre CDATA #REQUIRED
                    apellidos CDATA #REQUIRED
                    telefono CDATA #IMPLIED
                    ciudad CDATA "Madrid">
```

Atributos enumerados

Presentan un conjunto de valores permitidos para el atributo.

```
<!ATTLIST jugador nombre CDATA #REQUIRED
                  posicion (izquierda|centro|derecha) "centro">
```

Atributos ID

- Se usan para identificar elementos, es decir, caracterizarlos de manera única.
  - Un solo atributo ID por tipo de elemento.
  - Dos elementos no pueden tener el mismo valor en atributos de tipo ID.
- Un atributo de tipo ID se debe definir como #REQUIRED.
- Los valores deben ser nombres XML válidos.

```
<!ATTLIST receta id ID #REQUIRED>
```

Atributos IDREF

- El valor de un atributo IDREF debe corresponder con el valor de un atributo ID de algún elemento existente.

```
<!ELEMENT empleado (nombre, apellido)>
<!ATTLIST empleado  idEmpleado ID #REQUIRED
                    idEmpleadoJefe IDREF #IMPLIED>


<empleados>
  <empleado idEmpleado="e 111"> . . .
  </empleado>
  <empleado idEmpleado="e 222" idEmpleadoJefe="e 111"> . . .
  </empleado>
</empleados>
```

Atributos IDREFS

- Referencia a más de un atributo ID.

```
<!ELEMENT librococina (receta|ingrediente)*>
<!ELEMENT receta (#PCDATA)>
<!ATTLIST receta id ID #REQUIRED>
<!ELEMENT ingrediente (#PCDATA)>
<!ATTLIST ingrediente ref IDREFS #IMPLIED>

<librococina>
  <receta id="rec 1">Mousse de chocolate</receta>
  <receta id="rec 2">Fondue saboyana</receta>
  <receta id="rec 3">Fondue de chocolate</receta>
  <ingrediente ref="rec 1 rec 3">chocolate</ingrediente>
  <ingrediente ref="rec 2">Comte</ingrediente>
</librococina>
```

Atributos NMTOKEN / NMTOKENS

- Especifican atributos con caracteres permitidos por XML (letras, números, puntos, guiones, subrayados y los dos puntos).
  - En plural permite indicar una lista de NMTOKEN separados por espacios.

```
<!ELEMENT rio (nombre)>
<!ATTLIST rio pais NMTOKEN #REQUIRED>

<rio pais="EEUU">
  <nombre>Mississippi</nombre>
</rio>
```

3.3.- Entidades

- Son variables usadas para definir "atajos" a texto estándar o caracteres especiales.
- En una DTD son un mecanismo simple de reemplazo, el cual se aplica bien en el contexto del documento marcado de acuerdo a aquella, bien en el ámbito de la propia DTD.
- Tipos:
  - Referencia a entidades generales (internas o externas).
  - Referencia a entidades parámetro (internas o externas).
  - Entidades no procesadas (*unparsed*).

Entidades generales (internas)

- Sintaxis:

```
<!ENTITY nombre entidad definicion entidad>
```

- Ejemplo:

```
<!ENTITY writer "Donald Duck.">
<!ENTITY copyright "Copyright W3Schools.">

<author>&writer; &copyright;</author>
```

Entidades generales (externas)

- Sintaxis:

```
<!ENTITY nombre entidad tipo uso url archivo>
```

- Tipo de uso: `PUBLIC` o `SYSTEM`.
- Ejemplo:

```
<!DOCTYPE escritores [
  <!ELEMENT escritores (#PCDATA)>
  <!ENTITY autores SYSTEM "autores.txt">
]>
<escritores>&autores;</escritores>

Contenido de "autores.txt": Juan Manuel y José Ramón
```