

UT04. 03_XML-DTD

Curso: 2DAW

Lenguajes de Marcas y sistemas de gestión de información

Índice

1. Introducción
2. Documentos XML
3. Estructura jerárquica de un documento XML
4. Modelo de datos de un documento XML. Nodos
5. Corrección sintáctica: documento XML bien formado
6. Validación de documentos XML con DTD
7. Validación de documentos XML con esquemas XML
8. Otros mecanismos para validar XML
9. Otros lenguajes basados en XML
10. Otras formas de almacenar información

6. Documentos XML válidos

Documentos XML válidos

- Documento XML **válido**
 - Si cumple unas **reglas de validación** (por ejemplo en forma de definición de tipo de documento (DTD)).
 - **Reglas de validación** especifican la estructura gramatical y sintaxis que debe tener el documento XML.
- Todo documento XML válido está bien formado, pero no a la inversa.

7. Validación de documentos XML con DTD

Validación de documentos XML con DTD

- **DTD** (Document Type Definition-Definición de Tipo de Documento)
- Es una descripción de la **estructura de un documento XML**.
- Se especifica
 - Qué elementos tienen que aparecer
 - En qué orden
 - Cuáles son optativos/obligatorios
 - Qué atributos tienen los elementos, etc.
- Mecanismo de **validación** de documentos.
- Otras técnicas de validación:
 - **XSD** (XML Schema Document)

Generación automática de DTDs

- Existen herramientas de **generación automática de DTDs** a partir de un documento XML → **inferencia**.
 - Generadores on-line: http://www.hitsw.com/xml_utilities/default.html
 - Generadores instalables:
 - Trang <http://www.thaiopensource.com/>
 - Es un programa desarrollado en Java, ejecutable desde la línea de comandos, que genera un DTD a partir de un XML dado.
 - Editores XML que generan un DTD a partir de un documento XML.
Prácticamente todos los citados anteriormente:
 - XMLSpy de Altova
 - <oXygen/>, de Syncro Soft
 - XMLPad Pro Edition, de WMHelp

Estructura de un DTD. Elementos

- Tipos de DTD

- Ubicación: Interna, Externa o Mixta

- **Interna.** Incluida en el propio documento XML

<!DOCTYPE elementoRaiz[DTDInterna]>

- **Externa.** En forma de fichero independiente

<!DOCTYPE elementoRaiz SYSTEM DTDExterna>

→ **DTDExterna** . URI (Uniform Resource Identifier) de archivo con la DTD externa

- **Mixta.** Combinación de las anteriores

→ **<!DOCTYPE elementoRaiz SYSTEM DTDExterna [DTDInterna]>**

- Las declaraciones de la **DTD interna** prevalecen sobre las de la **DTD externa**.

- Carácter: público o privado

- **Privado:** con la palabra **SYSTEM**

- **Público:** **PUBLIC** + **FPI** (Formal Public Identifier – Identificador Público Formal)
identifica al DTD de manera universal

Estructura de un DTD. Elementos

- DOCTYPE + elemento_raíz
- PUBLIC/SYSTEM → DTD de uso público / DTD de uso interno de la organización que lo desarrolla.
- PUBLIC → FPI (por el que se conoce el DTD)

Sintaxis	Tipo de DTD
<!DOCTYPE elemento_raíz [reglas]>	Interno (y privado)
<!DOCTYPE elemento_raíz SYSTEM URL>	Externo y privado
<!DOCTYPE elemento_raíz SYSTEM URL [reglas]>	Mixto y privado
<!DOCTYPE elemento_raíz PUBLIC FPI URL>	Externo y público
<!DOCTYPE elemento_raíz PUBLIC FPI URL [reglas]>	Mixto y público

Estructura del FPI

- Campos separados por //
- **Campo 1:** norma formal/norma no formal
 - (-) no ha sido aprobado por una norma formal
 - (+) aprobado por un organismo no oficial.
 - () aprobado por un organismo oficial
- **Campo 2:** nombre del organismo responsable
- **Campo 3:** tipo del documento que se describe, con el número de versión
- **Campo 4:** idioma del DTD

- url existe cuando el DTD se encuentra declarado en un archivo externo, del que se da su ubicación.
- Este es el DOCTYPE para una página web escrita en XHTML 1.0 estricto, utilizada para validar miles de páginas web.

No ha sido aprobado por una norma formal

Nombre responsable del DTD: W3C

Tipo de documento: XHTML Transicional en su versión 1.0

Idioma: inglés

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Estructura de un DTD. Elementos

- Es un fichero XML
 - Opcional
 - Declaración

```
<?xml version="1.0" standalone="no"?>  
<!DOCTYPE ...>  
<ElementoRaíz>  
....  
</ ElementoRaíz >
```

- Atributo *standalone*

- **no** (por defecto): el documento puede apoyarse en fuentes externas, incluida una DTD externa
- **yes**: el documento es autónomo

Estructura de un DTD. Elementos

- Qué tipo de DTD utilizar
 - Interna
 - Si no se requiere validación estricta
 - Se va a crear un solo documento xml
 - Minimizar el coste asociado con los documentos (un solo fichero en lugar de dos)
 - Externa
 - Validación de los documentos XML
 - Cuando existe o puedan existir múltiples documentos xml de la misma clase
 - Utilizar una DTD ya existente
 - Documentos xml más concisos

Ejemplo

```
<?xml version="1.0" encoding="iso-8859-1" standalone="no"?>
<!DOCTYPE charlas SYSTEM "charlas.dtd" [
  <!ELEMENT charlas (charla)+>
  <!ELEMENT charla (nombre, lugar, despedida)>
  <!ENTITY despedidaInglesa "Thank you, good bye!">
  <!ENTITY despedidaFrancesa "Merci, au revoir!">
]>
<charlas>
.....
</charlas>
```

charlas.xml

charlas.dtd

```
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT lugar (#PCDATA)>
<!ELEMENT despedida (#PCDATA)>
```

Ejemplo

- Archivo externo no lleva DOCTYPE
- **standalone="no"** → necesitamos documentos externos (**charlas.dtd**)

Componentes de DTD

- Tipos de componentes a declarar en un DTD
 - **Elemento** → **<!ELEMENT>**
 - **Atributo** → **<!ATTLIST>**
 - **Entidad** → **<!ENTITY>**
 - **Notación** → **<!NOTATION>**

1. Elemento

- **<!ELEMENT>**

< ! ELEMENT nombre_elemento modelo_contenido>
- **nombre_elemento** → elemento correspondiente del XML.
- **modelo_contenido (ANY/EMPTY/Datos/Elementos descendientes)**
 - **ANY** → Descripción de un elemento como válido en cualquier caso. Comodín que **no** debe aparecer en el DTD definitivo.
 - **EMPTY** (elemento vacío): describe un elemento sin descendientes.
 - **Datos** (caracteres), sean textuales, numéricos o cualquier otro formato que no contenga marcas (etiquetas) → **#PCDATA** y debe aparecer entre paréntesis.

<!ELEMENT titulo (#PCDATA)>

<titulo>Lenguajes de marcas</titulo>

1. Elemento

- **ANY, EMPTY, Datos**

- **Elementos descendientes:** entre paréntesis. Se pueden combinar.

→ **Cardinalidad** de los elementos: número de veces que puede aparecer un elemento o secuencia de elementos existen ciertos símbolos

Símbolo	Significado
?	El elemento (o secuencia de elementos) puede aparecer 0 o 1 vez
*	El elemento (o secuencia de elementos) puede aparecer de 0 a N veces
+	El elemento (o secuencia de elementos) puede aparecer de 1 a N veces
Por defecto	El elemento (o secuencia de elementos) aparecer 1 vez.

→ **Secuencias** de elementos: orden de los elementos, o bien si aparece como alternativa a otro

Símbolo	Significado
A, B	El elemento B aparecerá a continuación del elemento A
A B	Aparecerá el elemento A o el B, pero no ambos

1. Elemento

- Ejemplo
 - En un correo electrónico, se podría describir el elemento raíz <email> como una secuencia de elementos <para>, <cc> (optativo), <cco> (optativo), <asunto> y <cuerpo>.
 - Un contrato tiene una lista de cláusulas. Cada una de las cláusulas está compuesta de varios epígrafes y sus desarrollos asociados, y concluyen por un único epílogo.

1. Elemento

<!ELEMENT email (para, cc?, cco?, asunto, cuerpo)>

<!ELEMENT para (#PCDATA)>

<!ELEMENT cc (#PCDATA)>

<!ELEMENT clausulas (clausula)+>

<!ELEMENT clausula ((epígrafe, desarrollo)+, epilogo)>

1. Elemento

- **ANY, EMPTY, Datos, Elementos descendientes**
- **Contenido mixto**, mezcla de texto más elementos descendientes.
 - *Ejemplo*: Se quiere describir un elemento <párrafo> que simule el párrafo de un editor de textos, de forma que pueda contener texto sin formato, texto en <negrita> (rodeado de esta etiqueta) o texto en <cursiva> (rodeado de esta etiqueta). Estas dos últimas etiquetas podrán a su vez contener, bien texto sin formato, bien la otra etiqueta. Por tanto, <párrafo>, <negrita> y <cursiva> son elementos de contenido mixto.

```
<!ELEMENT párrafo (#PCDATA | negrita | cursiva)*>  
<!ELEMENT negrita (#PCDATA | cursiva)*>  
<!ELEMENT cursiva (#PCDATA | negrita)*>
```

```
<párrafo>  
  Aquí un <cursiva> tema <negrita>importante</negrita></cursiva>  
</párrafo>
```

Actividad 4.4

- Se quiere que el elemento `<grupoSanguineo>` tenga como único elemento descendiente a uno solo de los cuatro siguientes A, B, AB o O. Indica cuál de las siguientes es una declaración correcta del citado elemento.
 - a. `<!ELEMENT grupoSanguineo (A ? B ? AB ? O) >`
 - b. `<!ELEMENT grupoSanguineo (A, B, AB, O) >`
 - c. `<!ELEMENT grupoSanguineo (A | B | AB | O) >`
 - d. `<!ELEMENT grupoSanguineo (A + B + AB + O)>`

Ejemplos

<!ELEMENT línea EMPTY >

**<línea></línea>
<línea />**

**La etiqueta br de
las páginas web**

```
<!DOCTYPE persona [  
  <!ELEMENT persona (nombre, apellidos)>  
  <!ELEMENT nombre (#PCDATA)>  
  <!ELEMENT apellidos ANY>  
>  
<persona>  
  <nombre>Arturo</nombre>  
  <apellidos>Pérez  
    <nombre>Arturo</nombre>  
  </apellidos>  
</persona>
```

**Podrá contener atributos (si
se especifican en el DTD).**

**Al definir apellidos como elemento
ANY, permite incluso que dentro haya
una etiqueta nombre.**

Ejemplos

<!ELEMENT película (título, dirección+, argumento?, actor*)>

El elemento película consta de:

- **un título**
- **uno o más elementos dirección**
- **puede o no tener argumento**
- **varios o ningún actor**

Se tiene que respetar ese orden.

Componentes de DTD

- Tipos de componentes a declarar en un DTD
 - Elemento → <!ELEMENT>
 - **Atributo** → <!ATTLIST>
 - Entidad → <!ENTITY>
 - Notación → <!NOTATION>

2. Atributo

▪ **<!ATTLIST>**

- Declaración de tipo de atributo.
- Indica la existencia de atributos de un elemento en el documento XML.
- Un solo **attlist** para declarar todos los atributos de un elemento (aunque se podría usar un **attlist** para cada atributo).

```
< !ATTLIST nombre_elemento  
    nombre_atributo tipo_atributo carácter  
    nombre_atributo tipo_atributo carácter  
    ...  
>
```

2. Atributo

- El **carácter** del atributo puede ser:
 - un valor textual entre comillas → un valor por defecto
 - **#IMPLIED** → opcional **sin** valor por defecto
 - **#REQUIRED** → obligatorio **sin** valor por defecto.
 - **#FIXED** → obligatorio **con** valor por defecto y único valor que puede tener el atributo.

2. Atributo

- Los **tipos de atributo** son:
 - **CDATA**: caracteres que no contienen etiquetas
 - **ENTITY**: el nombre de una entidad (que debe declararse en el DTD).
 - **ENTITIES**: una lista de nombres de entidades (que deben declararse en el DTD), separadas por espacios. + **adelante** (ENTITIES, ENTITY)
 - **Enumerado**: una lista de valores de entre los cuales, el atributo debe tomar uno.

```
<!ELEMENT semáforo EMPTY>  
<!ATTLIST semáforo color (rojo | ámbar | verde) "verde">
```

- **ID**
 - Un identificador único (dos elementos no pueden tener el mismo valor)
 - Un elemento puede tener un único atributo de tipo ID.
 - El valor debe ser un nombre XML válido.
- **IDREF**: valor de un atributo ID de otro elemento que debe existir, con el mismo valor del primer elemento.

2. Atributo. Ejemplo

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Archivo directorio.dtd -->
<!ELEMENT directorio (persona)+ >
<!ELEMENT persona (#PCDATA) >
<!-- Atributos de persona -->
<!-- id: Obligatorio -->
<!-- madre: Opcional -->
<!-- padre: Opcional -->
<!-- Definición de los atributos -->
<!-- id: Obligatorio -->
<!-- madre: Opcional -->
<!-- padre: Opcional -->
<!-- Definición de los atributos -->
```

Obligatorio

Opcional

Valor de un atributo ID
de algún elemento

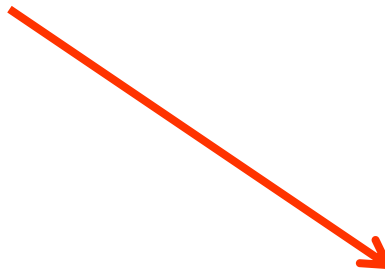
```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Archivo directorio1.xml-->
<!DOCTYPE directorio SYSTEM "directorio.dtd">
<directorio>
  <persona id="p1">Pedro</persona>
  <persona id="p2">Marisa</persona>
  <persona id="p3" madre="p2" padre="p1">Carmen</persona>
</directorio>
```

2. Atributo

- **IDREFS**: múltiples IDs de otros elementos, separados por espacios.
- `<persona id="p3" padres="p1 p2">Carmen</persona>`
- **NMTOKEN**: texto que cumple reglas más estrictas que CDATA: solo existen letras, números y el símbolo _ (no espacios en blanco).

Documento bien
formado pero **no** se
valida por el espacio del
valor de "pais"

Los espacios al principio y al final del
valor se ignoran (sí se valida). El resto
de los espacios no se permiten



```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE rio [
  <!ELEMENT rio (nombre)>
  <!ELEMENT nombre (#PCDATA)>
  <!ATTLIST rio
    pais NMTOKEN #REQUIRED>
]>
<rio pais="EE UU">
  <nombre>Misisipi</nombre>
</rio>
```

Componentes de DTD. Atributo

- **NMTOKENS**: una lista de nombres, sin espacios en blanco en su interior (los espacios en blanco anteriores o posteriores se ignorarán), separados por espacios.

Componentes de DTD

- Tipos de componentes a declarar en un DTD
 - Elemento → `<!ELEMENT>`
 - Atributo → `<!ATTLIST>`
 - **Entidad** → `<!ENTITY>`
 - Notación → `<!NOTATION>`

3. Entidad

- **<!ENTITY>**

- Es una declaración de tipo de entidad.
- Permite indicar abreviaturas de texto o utilizar caracteres que sería inválidos en el documento.
- Existen **entidades definidas** en XML (< > ...)
 - Referencia a entidades generales (internas o externas).
 - Referencia a entidades parámetro (internas o externas).
 - Entidades no procesadas (unparsed).

3. Entidad

- **<!ENTITY>.** Referencia a entidades generales (internas o externas).
 - Dentro del documento XML

```
<!ENTITY nombre_entidad definición_entidad>
```

```
<!ENTITY rsa "República Sudafricana">
```

```
....
```

```
<país><nombre>&rsa;</nombre></país>
```

3. Entidad

- **<!ENTITY>**. Referencia entidades generales externas, ubicadas en otros archivos.

<!ENTITY nombre_entidad tipo_uso url_archivo>

- **tipo_uso**: privado o público (SYSTEM/PUBLIC)

```
<?xml version="1.0"?>
<!DOCTYPE escritores [
    <!ELEMENT escritores (#PCDATA)>
    <!ENTITY autores SYSTEM "autores.txt">
]>
<escritores>&autores;</escritores>
```



Se inserta el contenido
de autores.txt

3. Entidad

- **<!ENTITY>.** Referencia a entidades parámetro, que se usarán en el propio DTD y funcionan cuando la definición de las reglas del DTD se realiza en un **archivo externo**. Pueden declararse en un DTD interno, pero la referencia tiene que ser en un DTD externo

```
<!ENTITY % nombre_entidad definición_entidad >
```

```
<!ENTITY % dimensiones  
    "alto CDATA #IMPLIED  
    ancho CDATA #IMPLIED  
    profundo CDATA #IMPLIED">  
<!ELEMENT objeto (nombre)>  
<!ATTLIST objeto  
    codigo ID #REQUIRED  
    %dimensiones;>
```

```
<!--Equivalente al siguiente código-->  
<!ELEMENT objeto (nombre)>  
<!ATTLIST objeto  
    codigo ID #REQUIRED  
    alto CDATA #IMPLIED  
    ancho CDATA #IMPLIED  
    profundo CDATA #IMPLIED>
```

3. Entidad

- Otro ejemplo

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ejemplo [
  <!ENTITY % entidad-ejemplo "<!ELEMENT ejemplo (#PCDATA)">
  %entidad-ejemplo;
]>
<ejemplo>Aquí viene cualquier cosa</ejemplo>
```

3. Entidad

- **<!ENTITY>.** Referencia entidades parámetro externas, ubicadas en otros archivos
- Sintaxis `<!ENTITY % nombre_entidad tipo_uso fpi url_archivo >`
- **tipo_uso:** privado (**SYSTEM**) o público (**PUBLIC**).
 - **PUBLIC:** es necesario definir el FPI (nombre por el que se identifica públicamente)

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

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

3. Entidad

- **<!ENTITY>.** Entidades no procesadas, referencian a datos que no deben ser procesados por el analizador XML, sino por la aplicación que lo use (como una imagen).
- Sintaxis

```
<!ENTITY nombre_entidad tipo_uso fpi valor_entidad NDATA tipo>
```

```
<!ENTITY logo SYSTEM "http://server.com/logo.gif">
```

3. Entidad

- Ejemplo

```
<!NOTATION JPG SYSTEM "image/jpeg">  
<!ENTITY mediterraneo SYSTEM "mediterraneo.jpg" NDATA JPG>  
<!ELEMENT mares (mar)+>  
<!ELEMENT mar (nombre)>  
<!ELEMENT nombre (#PCDATA)>  
<!ATTLIST mar  
    imagen ENTITY #IMPLIED>
```

```
<mares>  
    <mar imagen="mediterraneo">  
        <nombre>Mediterráneo</nombre>  
    </mar>  
</mares>
```


Componentes de DTD. Entidad

- Ejemplo: múltiples imágenes como valor del atributo imagen del elemento <mar>

```
<!NOTATION JPG SYSTEM "image/jpeg">
<!ENTITY mediterraneo1 SYSTEM "mediterraneo1.jpg" NDATA JPG>
<!ENTITY mediterraneo2 SYSTEM "mediterraneo2.jpg" NDATA JPG>
<!ELEMENT mares (mar)*>
<!ELEMENT mar...>
<!ATTLIST mar
      imagen ENTITIES #IMPLIED>
```

```
<mares>
  <mar imagen="mediterraneo1 mediterraneo2">
    <nombre>Mediterráneo</nombre>
  </mar>
</mares>
```

Componentes de DTD

- Tipos de componentes a declarar en un DTD
 - Elemento → `<!ELEMENT>`
 - Atributo → `<!ATTLIST>`
 - Entidad → `<!ENTITY>`
 - **Notación** → **`<!NOTATION>`**

4. Notation

- Una notación se usa para especificar un formato de datos que no sea XML.
- Uso:
 - Para describir tipos MIME (image/gif o image/jpg).
- Se utiliza para indicar un tipo de atributo al que se le permite usar un valor que haya sido declarado como notación en el DTD.
- Sintaxis de la notación:

```
<!NOTATION nombre_notacion SYSTEM "identificador_externo">
```

- Sintaxis general del atributo que la usa:

```
<!ATTLIST nombre_elemento  
      nombre_atributo NOTATION valor_defecto>
```

4. Notation

- Ejemplo

```
<!NOTATION GIF SYSTEM "image/gif">
<!NOTATION JPG SYSTEM "image/jpeg">
<!NOTATION PNG SYSTEM "image/png">
<!ELEMENT mares ... >
<!ATTLIST mar
    imagen ENTITY #IMPLIED
    formato_imagen NOTATION (GIF|JPG|PNG) #IMPLIED>
<!ENTITY mediterraneo SYSTEM "mediterraneo.jpg" >
```

```
<mares>
  <mar imagen="mediterraneo" formato_imagen="JPG">
    <nombre>Mediterráneo</nombre>
  </mar>
</mares>
```

Secciones condicionales

- Para incluir o excluir reglas en un DTD en función de condiciones.
- Sólo se pueden ubicar en DTDs externos.
- Combinación con referencias a entidades parámetro.
- Secciones condicionales:
 - **IGNORE** con precedencia sobre **INCLUDE**.
 - **INCLUDE**

Secciones condicionales. Ejemplo

- Se quiere diseñar un DTD que, en función del valor de una entidad parámetro, incluya una estructura de mensaje diferente. Por defecto se incluirá la estructura extendida.
- Dos estructuras diferentes para un mensaje:
 - Una que incluya emisor, receptor y contenido
 - Otra extendida que incluye los datos de la estructura sencilla más el título y el número de palabras.
- El DTD, que ha de ser externo.

Secciones condicionales. Ejemplo

- DTD externo

```
<!-- Mensaje corto -->
<![IGNORE [
  <! ELEMENT mensaje (emisor, receptor, contenido)>
]]>
<!-- Mensaje largo -->
<![INCLUDE [
  <!ELEMENT mensaje (titulo, emisor, receptor, contenido, palabras)>
  <!ELEMENT titulo (#PCDATA)>
  <!ELEMENT palabras (#PCDATA)>
]]>
<!-- Declaración de elementos y atributos comunes -->
<!ELEMENT emisor (#PCDATA)>
<!ELEMENT receptor (#PCDATA)>
<!ELEMENT contenido (#PCDATA)>
```

Secciones condicionales. Ejemplo

```
<!ENTITY %corto "IGNORE">
<!ENTITY %largo "INCLUDE">
<!--Mensaje corto-->
<![% corto[
    <! ELEMENT mensaje (emisor, receptor, contenido)>
]]>
<!-- Mensaje largo-->
<![% largo[
    <!ELEMENT mensaje (titulo, emisor, receptor, contenido, palabras)>
    <!ELEMENT titulo (#PCDATA)>
    <!ELEMENT palabras (#PCDATA)>
]]>
<!-- Declaración de elementos y atributos comunes -->
<! ELEMENT emisor (#PCDATA)>
<! ELEMENT receptor (#PCDATA)>
<!ELEMENT contenido (#PCDATA)>
```

DTD con las referencias a
entidad parámetro

Secciones condicionales. Ejemplo

Con declaración de las entidades en la DTD interna al documento XML

```
<?xml version="1.0"?>
<!DOCTYPE mensaje SYSTEM "mensaje.dtd" [
    <!ENTITY %corto "INCLUDE">
    <!ENTITY %largo "IGNORE">
```

Ejemplos de DTD

- Construir un DTD que valide el siguiente documento XML, persona.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE persona SYSTEM "persona.dtd">
<persona dni="12345678-L" estadoCivil="Casado">
  <nombre>María Pilar</nombre>
  <apellido>Sánchez</apellido>
  <edad>60</edad>
  <enActivo/>
</persona>
```

Ejemplo

- El esquema XML asociado se quiere que cumpla ciertas restricciones semánticas:
 - El atributo dni es un identificador obligatorio.
 - El estado civil puede ser: Soltero, Casado o Divorciado. Por defecto es Soltero.
 - El elemento <enActivo> es optativo.

Ejemplo

```
<!ELEMENT persona (nombre, apellido, edad, enActivo?)>  
<!ATTLIST persona dni ID #REQUIRED  
                estadoCivil (Soltero | Casado | Divorciado) "Soltero">  
<!ELEMENT nombre (#PCDATA)>  
<!ELEMENT apellido (#PCDATA)>  
<!ELEMENT edad (#PCDATA)>  
<!ELEMENT enActivo EMPTY>
```

Limitaciones de los DTD

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