# Modelado de datos XML

Ofimática Avanzada

Profesor: Víctor Fresno Fernández

curso 2006/07

# Modelado de datos XML

- **Documento XML:** medio estructurado para almacenar información

- **Esquema:**
  - modelo utilizado para describir la estructura de los documentos
  - permiten modelar clases de documentos (vocabularios)

# Modelado de datos XML

- Un esquema establece restricciones en la estructura del contenido del documento:
  - El modelo de contenido determina el orden y anidación de los elementos
  - Es posible establecer los tipos de datos del documento (sólo en el caso del XML-Schema)
- Formas de modelar datos en XML: DTD y XML-Schema

# Modelado de datos XML

- Un documento XML debe adherirse a un esquema para ser **válido.**

- Sin un esquema un documento puede estar **bien construido** (bien formado) pero no será válido.

# Modelado de datos XML

- Diferencias principales entre las DTD y los Schema:
  - DTD tienen una sintaxis específica mientras que Schema utiliza sintaxis XML
  - Un Schema se puede manipular como cualquier otro documento XML
  - Hay muchas más herramientas para trabajar con DTD que con Schema

# Modelado de datos XML

- Diferencias principales entre las DTD y los Schema: (cont)
  - Un Schema soporta tipos de datos (int, float, boolean, date, ...) las DTD tratan todos los datos como cadenas
  - Un Schema soporta la integración de los espacios de nombres permitiendo asociar nodos de un documento con declaraciones de tipo de un esquema
  - La DTD sólo permite una asociación entre un documento y su DTD

# Document Type Definition (DTD)

- Es la formalización de la noción de esquema, tipo o clase de documento

- Una DTD consistirá en una serie de definiciones de tipos de elementos, atributos, entidades y notaciones

- Declara cuales de ellos son legales dentro de un documento y en qué lugar pueden ubicarse

# Document Type Definition (DTD)

- Las DTD son importantes para permitir un procesamiento robusto de los documentos

- Un documento se relaciona con su DTD en la declaración de tipo de documento (DOCTYPE)

  **<!DOCTYPE factura SYSTEM "http://www.sitio.com/dtd/factura.dtd">**

- Un XML no requiere obligatoriamente tener una declaración de tipo de documento (a diferencia de SGML)

# Document Type Definition (DTD)

Puede haber cuatro tipos de declaraciones en una DTD:

- Declaración de tipo de ***elemento***
- Declaración de ***atributos***
- Declaración de ***notación***
- Declaración de ***entidades***

# Document Type Definition (DTD)

Declaración de tipo de **elemento**: Identifican los nombres de los elementos y la naturaleza de su contenido

- Comienzan con "**<!ELEMENT**"
- Sigue el identificador genérico del elemento que se declara
- A continuación tienen una especificación de contenido

# Document Type Definition (DTD)

Declaración de tipo de **elemento**:

- Ejemplo:

```
<!ELEMENT factura (cliente, vendedor, pedido)>
```

.. el elemento `<factura>` podrá contener dentro los elementos `<cliente>`, `<vendedor>` y `<pedido>`

- Para que un documento pueda validarse frente a una DTD, todos sus elementos deberán estar definidos en la DTD.

# Document Type Definition (DTD)

Declaración de tipo de **elemento**:

Siguiendo la definición de elemento anterior:

```
<factura>
     <cliente>...</cliente>
     <vendedor>...</vendedor>
     <pedido>...</pedido>
  </factura>
```

.. sería un documento válido

# Document Type Definition (DTD)

Declaración de tipo de **elemento**:

```
<factura>
    <cabecera>...</cabecera>
    <cliente>...</cliente>
    <vendedor>...</vendedor>
    <pedido>...</pedido>
</factura>
```

.. no sería un documento válido

# Document Type Definition (DTD)

Declaración de tipo de **elemento**:

La especificación de contenido puede ser de cuatro tipos:

- **EMPTY**: no tiene contenido pero puede tener atributos

```
<!ELEMENT tienda EMPTY>
```

Se pueden representar en el documento de dos formas:

```
<tienda  ID= “01”> </tienda>

<tienda  ID= “01”/>
```

# Document Type Definition (DTD)

Declaración de tipo de **elemento**:

- **ANY**: Puede tener cualquier contenido (caracteres, otros elementos o mixto)
  - No se suele utilizar, ya que es conveniente estructurar y declarar adecuadamente nuestros documentos XML
  - Ejemplo: `<!ELEMENT cabeTodo ANY>`

# Document Type Definition (DTD)

Declaración de tipo de **elemento**:

- **Mixto**: Puede tener caracteres o una mezcla de caracteres y sub-elementos
  - Ejemplo de solo caracteres:

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

  - Ejemplo de caracteres y elementos:

    ```
    <!ELEMENT calle (#PCDATA|numero)*>
    ```

# Document Type Definition (DTD)

Declaración de tipo de **elemento**:

- **Mixto**: Puede tener caracteres  o una mezcla de caracteres y sub-elementos
    - Si aparece #PCDATA en el modelo de contenido debe hacerlo en primer lugar del grupo
    - El grupo debe ir seguido por un asterisco (*)

# Document Type Definition (DTD)

Declaración de tipo de **elemento**:

- **Solo elementos**: Sólo puede contener sub-elementos de la especificación de contenido.
  - Ejemplo:

```
<!ELEMENT direccionCliente (calle, numero, ciudad, pais)>
```

# Document Type Definition (DTD)

- Los símbolos que pueden aparecer en el modelo de contenido de los elementos son:

  - Paréntesis "**( )**": engloba una secuencia o grupo de sub-elementos

```
<!ELEMENT direccionCliente (calle, numero, ciudad, pais)>
```

  - Coma " **,** ": denota una secuencia o grupo de sub-elementos

```
<!ELEMENT direccionCliente (calle, numero, ciudad, pais)>
```

# Document Type Definition (DTD)

- Los símbolos que pueden aparecer en el modelo de contenido de los elementos son:
  - Canalización " **|** ": separa los elementos de un grupo de alternativas

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

  - Cuantificadores: ?, +, *

# Document Type Definition (DTD)

- Los **cuantificadores** que pueden aparecer en el modelo de contenido de los *elementos son*:
  - Interrogación “**?**“ 0 ó 1 apariciones
  - Asterisco “ * ” 0 ó *n* apariciones
  - Signo más “**+**” 1 ó *n* apariciones

  Ejemplo:

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

# Document Type Definition (DTD)

Declaración de tipo de **atributos**:

- Los atributos permiten añadir información adicional a los elementos de un documento
- Los atributos no pueden contener sub-atributos
- Se usan para añadir información corta, sencilla y no estructurada
- Un atributo sólo puede aparecer una vez en un elemento y el orden, en caso de existir varios, no es relevante

# Document Type Definition (DTD)

Declaración de tipo de **atributos**:

- Las declaraciones de los atributos empiezan con "**<!ATTLIST**“
- Sigue el identificador del elemento al que se aplica el atributo
- Sigue el nombre del atributo, su tipo y su valor predeterminado:

```
<!ATTLIST elemento atributo tipoAtributo valorDefecto>
```

# Document Type Definition (DTD)

Declaración de tipo de **atributos**.

.. tiposAtributo :

- **CDATA:** datos de caracteres
- Enumerados**:** una serie de valores de cadena .. **(x | y | z)**
- **NOTATION:** una notación declarada en la DTD
- **ENTITY:** una entidad binaria externa
- **ENTITIES:** múltiples entidades binarias externas separadas por espacios en blanco

# Document Type Definition (DTD)

Declaración de tipo de **atributos**.

.. tiposAtributo :

- **ID:** un identificador único
- **IDREF:** una referencia a un ID declarado en la DTD
- **IDREFS:** múltiples referencias a ID declarados en la DTD
- **NMTOKEN:** un nombre con sólo caracteres XML válidos (letras, números, puntos, guiones, subrayados y los dos puntos)

# Document Type Definition (DTD)

Declaración de tipo de **atributos**

.. tiposAtributo :

- **NMTOKENS:** múltiples nombres con sólo caracteres XML válidos (letras, números, puntos, guiones, subrayados y los dos puntos)

# Document Type Definition (DTD)

Declaración de tipo de **atributos**.

.. valorDefecto:

- **#REQUIRED:** el atributo es obligatorio
- **#IMPLIED:** el atributo es opcional
- **#FIXED *valor*:** el atributo tiene un valor fijo (constante)
- **Con *valorDefecto:*** el atributo tiene el valor predeterminado

# Document Type Definition (DTD)

Declaración de tipo de **atributos**.

Ejemplos:

```
<!ATTLIST mensaje fecha CDATA #REQUIRED>
```

.. el elemento siguiente sería correcto:

```
<mensaje fecha="15 de Julio de 1999">
```

```
<!ATTLIST mensaje fecha NMTOKEN #REQUIRED>
```

.. el elemento siguiente sería correcto:

```
<mensaje fecha="15-7-1999">
```

.. sin espacios

# Document Type Definition (DTD)

Declaración de tipo de **atributos**. Ejemplos:

```
<!ELEMENT mensaje (de, a, texto)>
<!ATTLIST mensaje prioridad (normal|urgente) #IMPLIED>
```

. . el atributo "prioridad" puede tener el valor "normal" o "urgente" pero es opcional

# Document Type Definition (DTD)

Declaración de tipo de **atributos**. Ejemplos:

```
<!ELEMENT texto(#PCDATA)>
<!ATTLIST texto idioma CDATA #REQUIRED
               numpalabras CDATA #IMPLIED>
```

.. el atributo "idioma" es obligatorio, mientras que "numpalabras" es opcional y, si se omite, no toma ningún valor por defecto)

# Document Type Definition (DTD)

Declaración de tipo de **atributos**. Ejemplos:

```
<!ELEMENT mensaje (de, a, texto)>
<!ATTLIST mensaje prioridad (normal | urgente) “normal”>
<!ELEMENT texto(#PCDATA)>
<!ATTLIST texto idioma CDATA #REQUIRED
          fuente CDATA “EFE”
          numpalabras CDATA #IMPLIED>
```

# Document Type Definition (DTD)

Declaración de tipo de **atributos**. En el ejemplo anterior..

- el atributo "prioridad" puede tener el valor "normal" o "urgente", siendo "normal" el valor por defecto si no especificamos el atributo
- el atributo “idioma" es obligatorio
- el atributo “fuente" tiene el valor predeterminado “EFE”
- el atributo “numpalabras" es opcional

# Document Type Definition (DTD)

Declaración de tipo de **atributos**.

```
<mensaje prioridad="urgente">
<de>Fulanito</de><a>Menganito</a>
<texto idioma="castellano“ fuente=“EFE”>Hola, ¿qué tal? ...</texto>
</mensaje>
```

```
<!ELEMENT mensaje (de, a, texto)>
 <!ATTLIST mensaje prioridad (normal | urgente) “normal”>
 <!ELEMENT texto(#PCDATA)>
 <!ATTLIST texto idioma CDATA #REQUIRED
                fuente CDATA “EFE”
                numpalabras CDATA #IMPLIED>
```

# Document Type Definition (DTD)

Declaración de tipo de **atributos**.

- El atributo NOTATION**:** permite al autor declarar que su valor se ajusta a una notación declarada en la DTD

```
<!ATTLIST mensaje fecha NOTATION (ISO-DATE |
        EUROPEAN-DATE) #REQUIRED>
<!ATTLIST imagen formato NOTATION (gif|jpeg)
                #REQUIRED>
```

# Document Type Definition (DTD)

Declaración de tipo de **atributos**. Ejemplo:

```
<!ATTLIST parrafo identificacion ID #REQUIRED
                 tematica IDREF #IMPLIED>
```

**identificacion** será un identificador único obligatorio

**tematica** deberá coincidir con un ID de otro elemento

- Los atributos ID e IDREF deben declararse como `#REQUIRED` o `#IMPLIED`

# Modelado de datos XML

Declaración de **notaciones**

- Describen información necesaria para procesar las entidades externas no analizadas

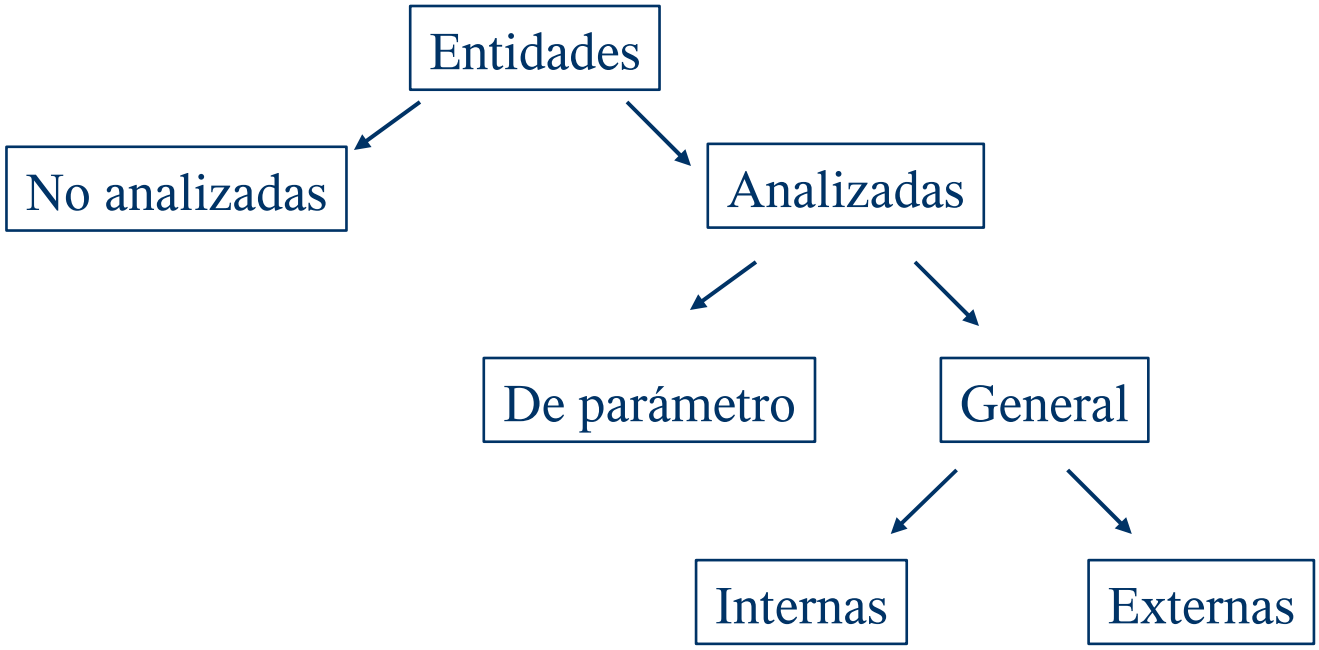
  Ejemplo: notación que describe el tipo de imagen GIF

```
<!NOTATION GIF SYSTEM “Iexplore.exe”>
```

.. en este caso, la aplicación XML se le pasa la información de que podría usar el programa “Iexplore.exe” para ver imágenes GIF

# Modelado de datos XML

- Tipos de **entidades**



# Modelado de datos XML

Declaración de **entidades** analizadas sintácticamente

- Entidades **generales**: se usan dentro del contenido del documento
- Entidades de **parámetro**: se usan sólo dentro de la DTD

# Modelado de datos XML

Declaración de **entidades generales**

- Son entidades analizadas sintácticamente y pueden ser **externas** o **internas**
- Representan y permiten sustituir a una cadena de texto
- Deben declararse:

```
<!ENTITY nombreEntidad definicionEntidad>
```

Ejemplo:

```
<!ENTITY urjc "Universidad Rey Juan Carlos" >
```

- Se usan en el contenido del documento: **&nombreEntidad;**

**(en el ejemplo: &urjc;)**

# Modelado de datos XML

**Entidades generales externas.**

- Obtienen su contenido en cualquier otro sitio del sistema (un archivo, una página web, ...)
- Analizadas sintácticamente: contenido XML
- No analizadas sintácticamente: no XML (texto, datos binarios, ...)
  - Llevan la palabra reservada NDATA en la declaración de entidad

# Modelado de datos XML

Declaración de **entidades generales externas** analizadas

- Se utiliza la palabra reservada **`SYSTEM`** para identificar un archivo del sistema local o de una red
- Se utiliza la palabra reservada **`PUBLIC`** para identificar un archivo de dominio público (con acceso público)
    - Es opcional y se usa siempre con SYSTEM

# Modelado de datos XML

Declaración de **entidades generales externas** analizadas

- La palabra `SYSTEM` seguida de un URI (*Universal Resource Identifier*):

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

# Modelado de datos XML

Declaración de **entidades generales externas** analizadas

- La palabra `SYSTEM` seguida de un URI (*Universal Resource Identifier*):

```
<!ENTITY intro SYSTEM “cap1.xml">
```

# Modelado de datos XML

Declaración de **entidades generales externas** analizadas (contendo XML).

- Permiten que un documento se forme con referencias a sub-documentos:

```
... <!ENTITY cap1 SYSTEM "cap1.xml">
      <!ENTITY cap2 SYSTEM "cap2.xml"> ...

<libro>
&cap1;
&cap2;
</libro>
```

# Modelado de datos XML

Declaración de **entidades generales externas** no analizadas (texto, datos, ..)

- Se referencian por medio de un atributo de tipo **`ENTITY`** o **`ENTITIES`**

# Modelado de datos XML

Declaración de **entidades generales externas** no analizadas.

Ejemplo:

```
<!ENTITY csfoto SYSTEM “csfoto.jpeg” NDATA JPEG>

    <!ELEMENT persona EMPTY>
    <!ATTLIST  persona  nombre  CDATA  #REQUIRED
      foto ENTITY #IMPLIED>

    <persona nombre=“Clara Sanz” foto=“&csfoto;”/>
```

# Modelado de datos XML

Declaración de **entidades generales internas**.

- Se definen dentro de la DTD en el prólogo del documento XML

Ejemplo:

```
<!DOCTYPE texto [...
  <!ENTITY urjc "Universidad Rey Juan Carlos"> ]>


<texto><titulo> La &urjc; ...</titulo></texto>
```

# Modelado de datos XML

Declaración de **entidades de parámetro**

- Solo se usan en la DTD
- Permiten modularizar una DTD

  Ejemplo: una entidad que almacene una lista de sub-elementos que se comparten entre varios elementos

- Sintaxis: `<!ENTITY % nombreEntidad definicionEntidad>`

# Modelado de datos XML

Declaración de **entidades de parámetro**

Ejemplo:

```
<!ENTITY % dimensiones “alto, ancho, largo”>
```

en la DTD se podrá hacer referencia a:

```
<!ELEMENT pared (%dimensiones;)>
<!ELEMENT techo (%dimensiones;)>
```

- Un uso inadecuado puede añadir complejidad innecesaria

# Modelado de datos XML

## DTD internas y externas

- La DTD de un documento puede ser:
  - Interna
  - Externa
  - Combinación de ambas

```
<!DOCTYPE elemRaiz SYSTEM DTDexterna [DTDinterna]>
```

# Modelado de datos XML

## DTD internas y externas

- Ejemplo:

```
<!DOCTYPE peliculas SYSTEM “peliculas.dtd” [
    <!ELEMENT actor (#PCDATA)> ]>
```

Elemento raíz: todos los demás deben estar dentro de él (`peliculas`)

DTD externa: “`peliculas.dtd`”

DTD interna: declaraciones adicionales (elemento `actor`)

- Las declaraciones internas prevalecen a las externas

# Modelado de datos XML

## Corrección de un documento XML

- Se pueden distinguir dos nociones de corrección en un documento XML:
  - documento **válido** (respecto a una DTD)
  - documento **bien construido**

# Modelado de datos XML

## Corrección de un documento XML

- Un **documento** es **valido** si:
  - Está bien construido
  - El nombre del elemento raíz coincide con el nombre de la declaración de tipo de documento
  - La DTD declara todos los elementos, atributos y entidades que se utilicen en el documento
  - El documento cumple con la gramática descrita en la DTD
- La validez la determina un analizador o parser

# Modelado de datos XML

## Corrección de un documento XML

- Un objeto de texto es **un documento XML bien construido** si:
  - Tomado como un todo, cumple la regla denominada **"document"**
  - **Respeta todas las restricciones de buena formación** dadas en la especificación
  - **Cada una de las entidades analizadas** que se referencia directa o indirectamente en el documento **está bien formada**

# Modelado de datos XML

## Corrección de un documento XML

- Cumplir **la regla "document"** significa:
  - Que contiene uno o más elementos
  - Hay un elemento llamado raíz (elemento documento) en el que se encuentran todos los demás
  - Todo elemento tiene una etiqueta de inicio y de final o una etiqueta de elemento vacío
  - Los elementos delimitados por etiquetas de principio y final se anidan adecuadamente
  - Los valores de los atributos van entre comillas (“ o ‘)

# Modelado de datos XML

## Corrección de un documento XML

- Ejemplo:

```
Mi documento XML
```

no es un documento XML bien construido: no contiene ningún elemento

# Modelado de datos XML

## Corrección de un documento XML

- Ejemplo:

```
<p> Mi  documento XML</p>
```

sí lo es: contiene al menos un elemento "p".

# Modelado de datos XML

## Corrección de un documento XML

- Ejemplo:

```
<p> Mi  documento XML</p>
<p> Mi  documento XML</p>
```

no es un documento XML bien formado: sólo puede existir un único elemento raíz

# Modelado de datos XML

## Corrección de un documento XML

- Ejemplo:

```
<documento>
    <p> Mi documento XML </p>
    <p> Mi documento XML</p>
</documento>
```

si está bien construido: `documento` es el elemento raíz, es único y no forma parte del contenido de ningún otro elemento

# Modelado de datos XML

## Corrección de un documento XML

- Ejemplo:

```
<documento>
    <p> Mi <elem> documento XML</p> </elem>
    <p> Mi documento XML</p>
</documento>
```

es incorrecto: la etiqueta inicio del elemento `<elem>` está dentro del contenido del elemento `<p>`, pero su etiqueta final está fuera

# Fundamentos de las DTD

## Corrección de un documento XML

- Ejemplo:

```
<documento>
<p> Mi primer <elem> documento XML</elem> </p>
<p> Mi primer documento XML </p>
</documento>
```

Documento bien construido