Tema 1: Generación de documentos digitales: XML.

Índice.

1. Introducción
    1. Evolución de los lenguajes de marcas: SGML,HTML,XML
    2. Características de XML.
2. Documentos XML
    1. Sintaxis de XML
    2. Elementos, atributos.
    3. Documentos XML válidos y bien formados
3. DTD

1. Introducción

Internet

- Internet nació como una red de computadoras del ejercito de USA (ARPANET). Posteriormente la red se extendió a varias universidades.
- Actualmente es una enorme red de computadoras conectadas a nivel mundial.
- Una de las claves del crecimiento de Internet ha sido la aparición de la Web.
- Servicios estándar de Internet:
  - WWW.
  - Correo electrónico.
  - Chat.
  - FTP.
  - Telnet, etc..

WWW(I)

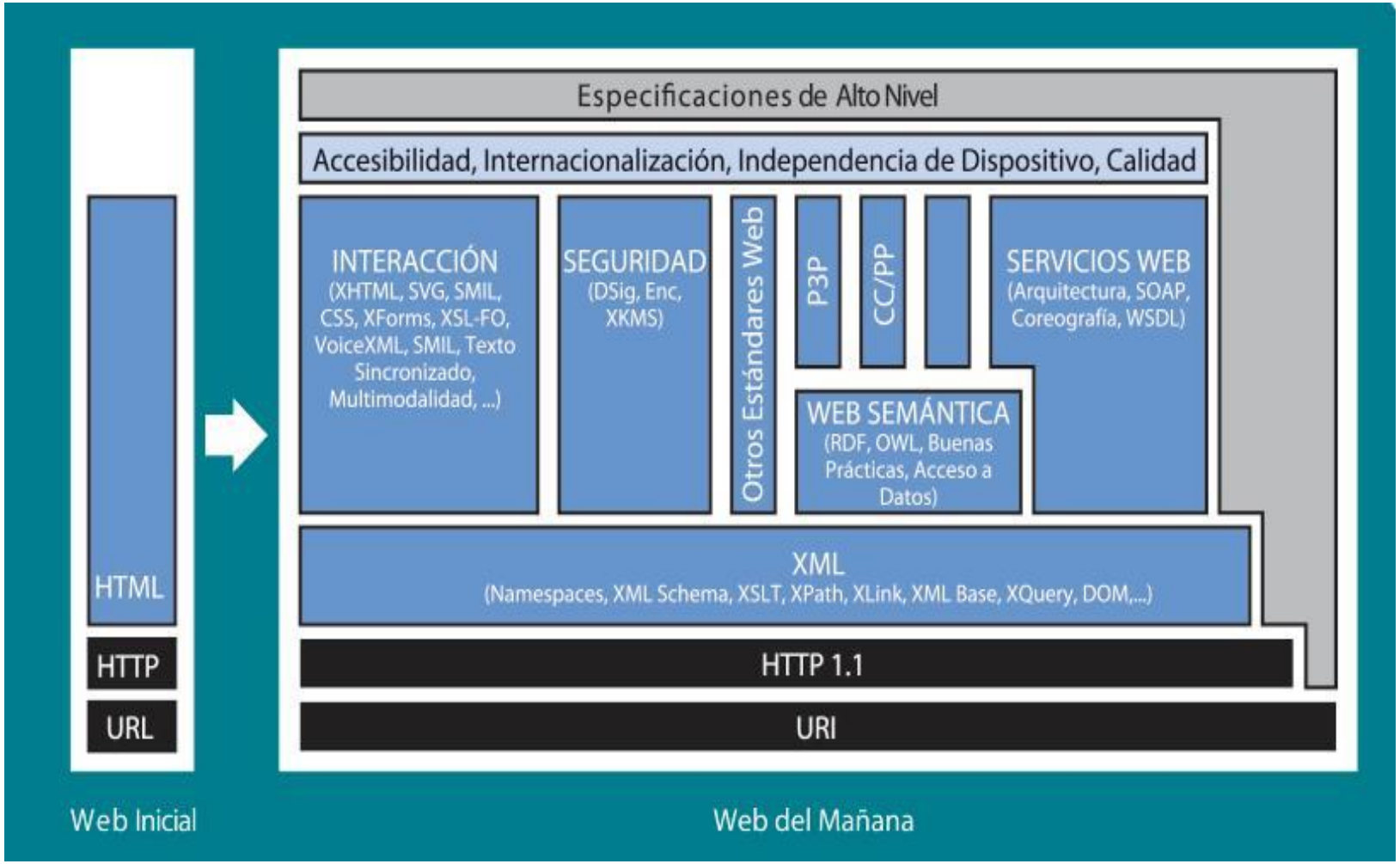
Algunas características :

- Las aplicaciones que intercambian la información están basadas en un esquema cliente/servidor.
- La información está inherentemente distribuida.
  - Cualquiera puede añadir más información.
- Se accede a la información de una manera estándar (tanto a la hora de solicitar como a la hora de ofrecer dicha información) independiente del sistema específico donde resida o a la cual se envía (URL y HTTP).
- La información es descrita en términos estándar (HTML).
- Integra fuentes de información externos (a los servidores), como BD y otras aplicaciones.

WWW(II)

La Web: basa su diseño y funcionamiento en:

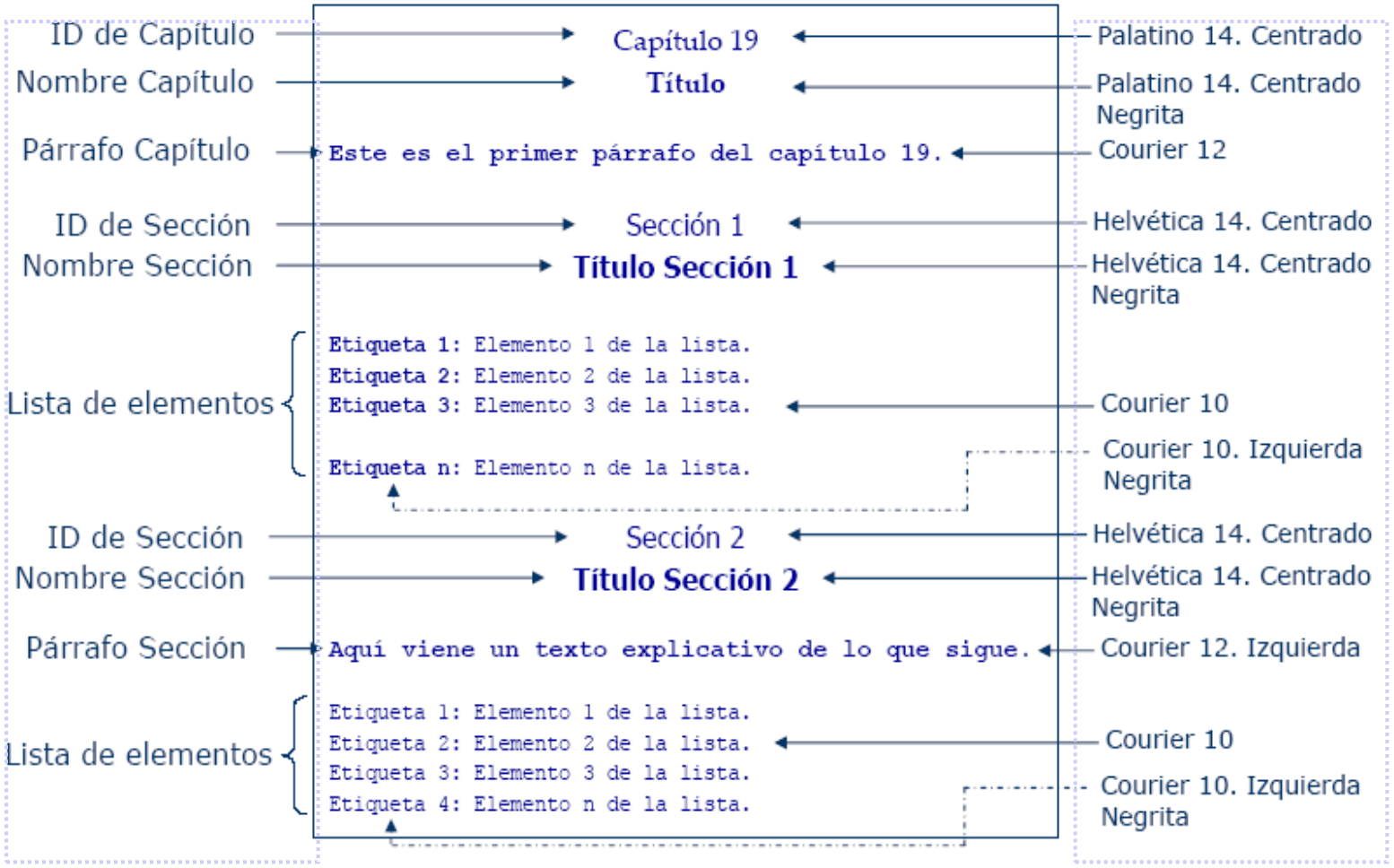
- Sistema de direcciones **URI** (Uniform Resource Identifier): identifica el recurso.
  - Protocolo://usuario:clave@maquina:puerto/directorio/fichero#seccion?parametros
  - Dentro del contenido de los recursos web se hace constantes referencias a otro recursos (links). (Maraña).
- **HTTP** (Hypertext Transfer Protocol): Protocolo estándar usado para la comunicación, y transferencia de los recursos, entre los servidores web y los clientes (navegadores).
- **HTML (XHTML**) (Hypertext Markup Language): Lenguaje estándar utilizado para describir el contenido de los documentos web.
  - Es una instancia del SGML (ISO, mediados de 1970).


Especificaciones de Alto Nivel
Accesibilidad, Internacionalización, Independencia de Dispositivo, Calidad
INTERACCIÓN
(XHTML, SVG, SMIL, CSS, XForms, XSL-FO, VoiceXML, SMIL, Texto Sincronizado, Multimodalidad, ...)
SEGURIDAD
(DSig, Enc, XKMS)
Otros Estándares Web
P3P
CC/PP
SERVICIOS WEB
(Arquitectura, SOAP, Coreografía, WSDL)
WEB SEMÁNTICA
(RDF, OWL, Buenas Prácticas, Acceso a Datos)
XML
(Namespaces, XML Schema, XSLT, XPath, XLink, XML Base, XQuery, DOM,...)
HTTP 1.1
URI
HTML
HTTP
URL
Web Inicial
Web del Mañana

Lenguajes de marcas

- Marcas: Termino empleado para describir los códigos, también llamados **etiquetas**, añadidos al texto electrónico y que definen la estructura y el formato en el que tienen que aparecer
- Puede ser utilizado con múltiples propósitos: escritura, impresión, intercambio de información, presentación de pantallas, etc
- Lenguajes de marcado: estructuras
  - **Estructura lógica**: formada por las partes que lo componen y sus relaciones
  - **Estructura física**: indica la apariencia del documento incluyendo sus componentes físicos, posicionamiento de elementos y tipografía empleada

Lenguajes de marcas



Estructura lógica

Estructura física

Lenguajes de marcas

- Lenguajes de marcas genéricos:
  - Son aquellos que sirven para especificar la estructura de cualquier documento *sin tener en cuenta* los aspectos relativos a *la presentación*.
  - Posteriormente, el mismo documento se podrá presentar de diversas formas, según las *normas de estilo* que se le apliquen

Lenguajes de marcas. SGML.

Standard Generalized Markup Language

- Norma ISO-8879 originada por la comunidad editorial para flexibilizar el diseño de documentos.
- Proporciona una forma *normalizada* de transmitir los documentos en formato adecuado para los procesos de edición e impresión
- Es apropiado para describir texto altamente estructurado.
- Es un **meta-lenguaje** para especificar lenguajes de marcado:
  - Contiene reglas para crear lenguajes de marcado, pero *no describe el formato* de los documentos marcados
- Mediante la utilización de una definición de tipo de documento (**DTD: Document Type Definition**) se puede especificar la estructura lógica de una clase de escrito.

Lenguajes de marcas. SGML.

```
<!ELEMENT antologia        - - (poema+) >
<!ELEMENT poema            - - (titulo?, estrofa+) >
<!ELEMENT titulo           - - (#PCDATA) >
<!ELEMENT estrofa          - - (linea+) >
<!ELEMENT linea            - - (#PCDATA) >
```

Ejemplo de DTD

```
<antologia>
  <poema>
     <titulo>A un panal de rica miel</titulo>
       <estrofa>
          <linea> A un panal de rica miel</linea>
          <linea>cien mil moscas acudieron,</linea>
          <linea>que por golosas murieron,</linea>
          <linea>presas de patas en él.</linea>
       </estrofa>
       <estrofa>
          <linea>Así, si bien se examina,</linea>
          <linea>los humanos corazones</linea>
          <linea>perecen en las prisiones</linea>
          <linea>del vicio que los domina. </linea>
       </estrofa>
   </poema>
   <!-- Aquí irían más poemas de la antología -->
</antologia>
```

Ejemplo de documento basado en el DTD anterior

Lenguajes de marcas.HTML

HyperTextual Markup Language

- HTML aparece como una aplicación de SGML para codificar documentos y publicarlos en la WWW
- HTML es una DTD, no un meta-lenguaje como SGML.
- Está orientado en gran medida a la presentación de los documentos más que a su estructura.
- Contiene un **número limitado** y no extensible **de marcas** que se intercalan en el documento y permiten describir el aspecto con el que ha de visualizarse.
- Permite publicar documentos en la red con:
  - Cabeceras, textos, tablas, listas, fotografías...
  - Encontrar información en línea mediante enlaces hipertextuales.
  - Diseñar formularios para realizar transacciones con servicios remotos, buscar información, hacer pedidos, etc.
  - Incluir elementos multimedia: vídeos, sonido...

HTML.Ejemplos.

```
<HTML>
  <HEAD>
    <TITLE>Mi primera página</TITLE>
    <META name="autor" content="una pruebecilla">
  </HEAD>
  <BODY>
    Este es un ejemplo de página web. Voy a añadir una imagen:<BR>
    <IMG src="prueba.jpg"> <BR>
  </BODY>
</HTML>
```

Ejemplo de HTML

Representación en el navegador

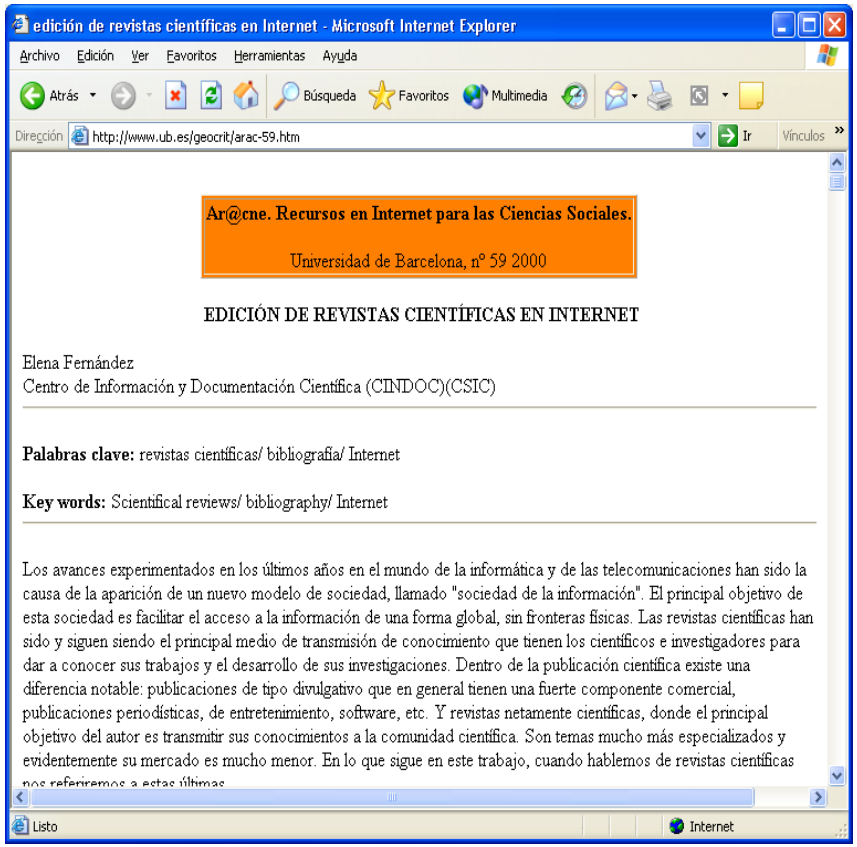
HTML.Ejemplos.

```
<p> <dt> <b> <a href="/exec/obidos/ASIN/0764531999/qid=919015337">
 Xml : Extensible Markup Language</a></b> ~
<font color=#990033>Usually ships in 24 hours</font>
<dd> Elliotte Rusty Harold / Paperback / Published 1998
<br> Our Price: $31.99 ~ <font color =#990033>You Save: $8.00 (20%)</font>
 <br>
<a href="/exec/obidos/ASIN/0764531999/qid=919015337">
 <i>Read more about this title...</i></a>
```

Ejemplo de HTML. Ficha de un libro:
Estructura de la página.
Visualización.

Xml : Extensible Markup Language ~ Usually ships in 24 hours
Elliotte Rusty Harold / Paperback / Published 1998
Our Price: $31.99 ~ You Save: $8.00 (20%)
Read more about this title...

HTML.Ejemplos.



Código fuente:

```
<!doctype html public "-//w3c//dtd html 4.0 transitional//en">
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-
1">
  <meta name="Author" content="Elena Fernandez">
  <meta name="GENERATOR" content="Mozilla/4.5 [es] (Win95; I)
[Netscape]">
  <title>edici&oacute;n de revistas cient&iacute;ficas en Internet</title>
<style type="text/css">
<!--
a:active {  text-decoration: none}
a:hover {  text-decoration: none}
a:link {  text-decoration: none}
a:visited {  text-decoration: none}
-->
</style>
</head>
<body text="#000000" bgcolor="#FFFFFF" link="#0000EE"
vlink="#551A8B" alink="#FF0000" style="text-decoration: none">
 
<center><table BORDER BGCOLOR="#FF8000" >
<tr>
<td>
<center><b>Ar@cne. Recursos en Internet para las Ciencias Sociales.</b>
<p>Universidad de Barcelona, n&ordm; 59 2000</center>
</td>
</tr>
</table></center>
```

Lenguajes de marcas: HTML

- HTML es fácil de aprender y está muy extendido, pero:
  - Es superficial: sólo describe cómo se deben mostrar los documentos, pero no describe su contenido. Esto hace que los servidores web sirvan muchas páginas que los usuarios piden para poder ver su contenido y averiguar si les son útiles o no.
  - Las reglas de HTML son muy relajadas, y además los navegadores suelen ser muy generosos con las páginas que no cumplen correctamente la sintaxis del lenguaje.
  - No es extensible: el conjunto de etiquetas está limitado y no se pueden definir otras nuevas, lo que supone que no se puede adaptar a las diferentes necesidades.

La solución pasa por buscar un lenguaje que permita describir los documentos desde un punto de vista de estructura y no solo de aspecto: XML

La solución: XML

XML: eXtensible Markup Language

- XML es un derivado del SGML que vendrá a simplificarlo, pero no será tan restringido como HTML.
- XML es un **meta-lenguaje** de marca: lenguaje genérico que podremos particularizar para definir lenguajes de marca para aplicaciones específicas.
- Permite describir la estructura del documento. Antes '**el qué**' que 'el cómo'.

Ejemplo XML:

```
<LIBRO>
   <TITULO> La Regenta </TITULO>
   <CAPITULO NUM="1" >
        <TITULOC>Uno </TITULOC>
        <PARRAFO NUM="1">
               La heroica ciudad dormía la siesta …
        </PARRAFO>
   </CAPITULO>
</LIBRO>
```

XML

- XML está orientado a la **estructura de los documentos**, no a su presentación.

Ejemplo. Tenemos un correo electrónico:

De: Antonia Lluesma [antonia.lluesma@uv.es]
Enviado: lunes, 03 de mayo de 2004 9:36
Para: SI@listserv.uv.es
Asunto: Parada Aplicación SIUV en MVS

Mañana martes, día 4-mayo-2004, la Aplicación SIUV (Matrícula, Registro, Nóminas, Actas...)de MVS no estará operativa desde las 15:00 hasta las 16:00 horas, por actualización y mantenimiento.
Rogamos disculpen las molestias.
Un saludo,

Antonia Lluesma (e-mail: antonia.lluesma@uv.es)
Teléfono 963544312 - Ext.44312
Fax 96-35-44200
Operación de Sistemas
Servicio de Informática
Universidad de Valencia

XML

Podríamos identificar sus partes de la siguiente forma:

```
<?xml version="1.0" ?>
<!DOCTYPE email SYSTEM "http://www.uv.es/DTDs/email.dtd">
<email id="E1X108">
     <head>
      <from><name>De: Antonia Lluesma</name><address> [antonia.lluesma@uv.es]</address>
      <date>Enviado: lunes, 03 de mayo de 2004 9:36</date>
      <to>Para: SI@listserv.uv.es</to>
      <subject>Asunto: Parada Aplicación SIUV en MVS</subject>
     </head>
    <body>
    Mañana martes, día 4-mayo-2004, la Aplicación SIUV (Matrícula, Registro, Nóminas, Actas...)de
    MVS no estará operativa desde las 15:00 hasta las 16:00 horas, por actualización y
    mantenimiento.
    Rogamos disculpen las molestias.

    Un saludo,
    ------------------------------------------------
    Antonia Lluesma (e-mail: antonia.lluesma@uv.es)
    Teléfono 963544312 - Ext.44312
    Fax 96-35-44200
    Operación de Sistemas
    Servicio de Informática
    Universidad de Valencia
    </body>
</email>
```

Características de XML

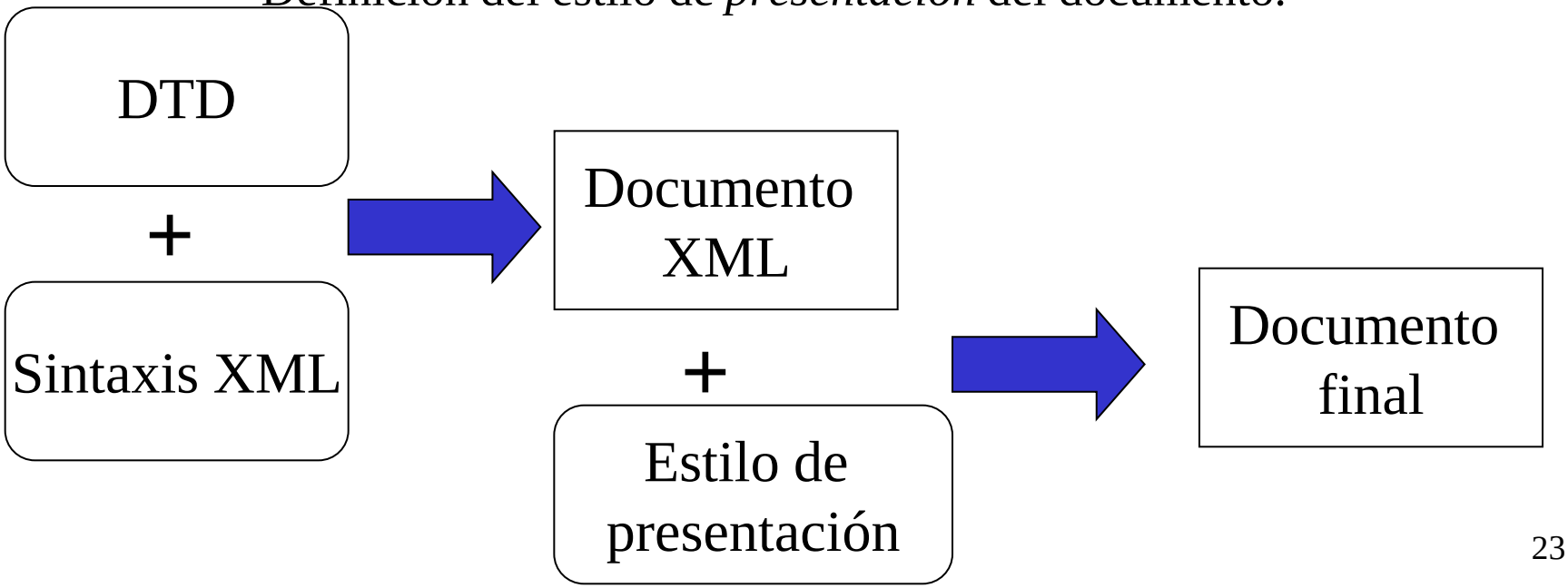
- Tiene una **sintaxis estricta**, lo que hace posible el desarrollo de aplicaciones que evalúen los documentos. XML tiene unas reglas sencillas pero muy estrictas, lo que lo hace fácilmente procesable. Por tanto, puede ser leído tanto por máquinas como por personas.
- Es **extensible**: XML permite definir las marcas que se van a utilizar para denotar la estructura de sus documentos.
- Es **estándar**: Es un lenguaje basado en un estándar público, de modo que no pertenece a ninguna organización comercial.

Características de XML

- XML es un subconjunto simplificado pero estricto de SGML (*Standard Generalized Markup Language*):
  - Extensible: se pueden definir nuevas etiquetas.
  - Estructurado: se puede modelar datos a cualquier nivel de complejidad, y su definición está en una DTD, *Document Type Definition*.
  - Validable: cada documento se puede validar frente a una DTD, o en su defecto, se puede declarar bien formado.
  - Independiente de medio: para publicar contenidos en múltiples formatos.
  - Independiente de fabricante y de plataforma: para poder utilizar cualquier herramienta estándar.
- XML es fácil de aprender y de usar.
- Los documentos XML son fácilmente procesables y compartidos en Internet.

Características de XML

- Al tratarse de un meta-lenguaje hemos de distinguir en él los siguientes componentes fundamentales:
  - *Sintaxis* XML (derivada de la de SGML)
  - Especificación de las definiciones de *tipo de documento* (DTD's) que son las que definen cada lenguaje particular.
  - Escritura de Documentos XML
  - Definición del estilo de *presentación* del documento.



2. Documentos XML

Sintaxis de XML

- Existen bastantes diferencias entre la sintaxis de XML y la de HTML.
- En general, XML es mucho más estricto con su sintaxis.
- A continuación se enumeran algunas reglas básicas de sintaxis:
  - En la cabecera del documento debe aparecer una declaración del tipo

    ```
    <?xml version= "1.0" encoding="ISO-8859-1"?>
    ```

    En la que se indica que es XML, la versión de XML que se está usando, de momento siempre la 1.0, y el conjunto de caracteres empleado, en este caso ISO-8859-1 (ISO Latin 1)
  - A continuación se podrá incluir un elemento del tipo

    ```
    <!DOCTYPE email system "http://www.uv.es/DTDs/email.dtd">
    ```

    En el cual se define cual es el elemento raíz del documento y según qué estructura se rige.

Sintaxis de XML

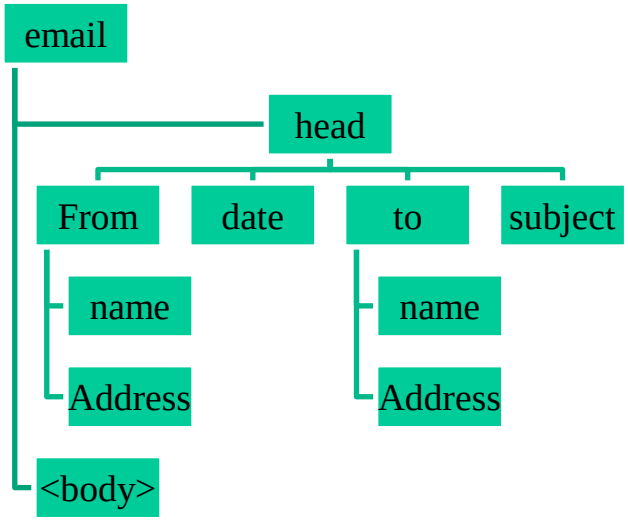
- XML **es sensible a mayúsculas y minúsculas**, de modo que las etiquetas deberán introducir exactamente como aparecen en la declaración.
- Cada etiqueta de apertura debe tener su etiqueta de cierre, o cerrarse ella misma (en caso de ser elemento vacío).
- Los documentos XML sólo tienen un elemento raíz.
- Los nombres de los elementos deben obedecer la convención de nombrado de XML.
- Las etiquetas no pueden superponerse, los elementos deben estar convenientemente estructurados.
- XML conservará el espacio en blanco en los contenidos de texto.

Nombrado de elementos

- Los nombres de elementos comienzan con letra o el subrayado (_), pero NO por números ni otros caracteres de puntuación.
- Tras este primer carácter, se permiten números, guiones, etc.
- Los nombres de los elementos no contienen espacios.
- Los nombres pueden contener el carácter (:). De modo estricto se permite, pero este carácter esta reservado para especificar los espacios de nombres.
- Ningún nombre de elemento comenzará con las letras “xml”, ya sea mayúsculas, minúsculas o combinación de ellas.

Sintaxis de XML

- Todo documento XML debe tener un **único elemento raíz**. En el caso de nuestro ejemplo, el elemento raíz sería `<email>` Podríamos representar nuestra estructura en forma de árbol, en cuya raíz estaría este elemento, de la siguiente forma:



Sintaxis de XML

- Las etiquetas se delimitan como en HTML con los **símbolos** <>. Todos los elementos deben incluir etiquetas de apertura y cierre.

  <etiqueta> </etiqueta>

- HTML permite elementos sin contenido, es decir, que no contienen elementos anidados ni texto. XML también, y en este caso la etiqueta puede seguir la siguiente sintaxis:

  <elemento-sin-contenido/>

  Esta sintaxis conlleva implícitas la apertura y cierre del elemento. El elemento podrá llevar atributos.

Sintaxis de XML

- Los elementos de XML, como en HTML, pueden tener **atributos**, con la sintaxis:

  ```
  <etiqueta atributo_1= "valor" .. atributo_n="valor"> <etiqueta>
  ```

  pero se deben cumplir las siguientes normas:

  - Los valores de los atributos deben estar siempre encerrados entre comillas.
  - Un mismo atributo no puede tomar dos valores distintos para el mismo elemento

- Se pueden incluir **comentarios** con la sintaxis:

  ```
  <!-- comentario -->
  ```

  Teniendo en cuenta que el doble guión no puede aparecer en ningún punto del texto del comentario

3. DTD

XML: información y estructura

- Los documentos XML contienen información estructurada mediante unos elementos que identifican los contenidos y los asocian a una estructura declarada explícitamente.
- La **estructura de un documento** XML se define en un documento especial denominado **DTD** (Document Type Definition) o bien mediante un **esquema** XML.
- La DTD es la encargada de establecer las **reglas de un vocabulario** XML.

- Diremos que un documento XML está **bien formado** cuando cumple las reglas sintácticas de XML

- Diremos que un documento XML es **válido** cuando además de estar bien formado cumple las reglas de la DTD asociada, sea externa o interna

Características de los DTD

- Un DTD indica:
  - Qué **elementos** pueden ser usados en los documentos asociados a ellos
  - El orden en que deben aparecer los elementos
  - Qué elementos son obligatorios y cuales opcionales
  - Cuales se pueden repetir y cuales no.
  - Cual es el anidado de los elementos, lo que denota la estructura en árbol
  - Qué **atributos** pueden tener esos elementos.
- Además un DTD puede contener otras referencias como:
  - Declaración de “textos patrón” que se van a utilizar frecuentemente.
  - Recursos externos a XML.
  - Instrucciones de procesado de los elementos no XML.

Declaración de un DTD

- Los DTD pueden ser *internos o externos*, según se declaren en el propio documento XML o sean documentos aparte referenciados en la declaración inicial:
- Internos:
  - El DTD puede definirse dentro de un documento XML de la siguiente forma:

    <!DOCTYPE raíz [definición de DTD]>
  - O bien definirse en un documento aparte (normalmente .dtd) y después enlazarse a los documentos XML asociados mediante la declaración:

    <!DOCTYPE raíz SYSTEM “fichero.dtd”>

- Externos: el fichero .dtd puede estar ubicado en una máquina diferente, por lo que se debe indicar su URL

<!DOCTYPE email SYSTEM "http://www.uv.es/DTDs/email.dtd">

En nuestro ejemplo, se está utilizando un DTD externo llamado email.dtd disponible en la URL que se indica.

Estructura del DTD

- El ejemplo anterior tenía declarado un DTD externo llamado *email.dtd*. Este DTD podría tener la siguiente forma:

```
<!ELEMENT email (head, body)>
<!ATTLIST email id ID #REQUIRED>
<!ELEMENT head (from, to+, cc*, subject)>
<!ELEMENT from (name?, address)>
<!ELEMENT to (name?, address)>
<!ELEMENT cc (name?, address)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT address (#PCDATA)>
<!ELEMENT subject (#PCDATA)>
<!ELEMENT body (p | attach)*>
<!ELEMENT p (#PCDATA)>
<!ELEMENT attach EMPTY>
<!ATTLIST attach encoding (mime|binhex) "mime"
                 name CDATA #REQUIRED>
```

Estructura del DTD

- El DTD constará de una *enumeración de declaraciones*
- Cada declaración se escribe marcada entre los “<!” y “>” indicando siempre el tipo de elemento de que se trata. Por ejemplo:

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

- Pueden existir cuatro tipos de declaraciones:
  - *Elementos* (ELEMENT): identifican los nombres de los elementos a usar en la estructura, su contenido y las relaciones entre los distintos elementos.
  - *Atributos* (ATTLIST): marcan los elementos que tienen atributos, cuáles serán y si tienen valores por defecto cuáles serán estos.
  - *Entidades* (ENTITY): permiten asociar nombres con contenidos constantes que se usan a menudo, como por ejemplo

    declaración: `<!ENTITY texto "En un lugar de la mancha">`

    uso: `&texto;`
  - *Notación* (NOTATION): identifican tipos específicos de datos binarios externos.

DTD: Elementos

- Son los contenedores que dividen un documento XML en partes definiendo **su estructura lógica**.
- En el ejemplo del Libro los elementos que podrían entresacarse son:

```
<LIBRO>..</LIBRO>
<TITULO>..</TITULO>
<CAPITULO>..</CAPITULO>
<TITULOC> ..</TITULOC>
<PARRAFO>..</PARRAFO>
```

- Los elementos pueden estar anidados unos dentro de otros.
- El elemento raíz engloba a todos los demás y se llamará como el tipo de documento.
- En HTML tenemos ejemplos de estos tipos de elementos: del primer tipo <HTML>..</HTML>, <HEAD>..</HEAD>, etc. y del segundo:
 <IMG />

DTD: Elementos

Declaraciones de Elementos en una DTD

- Tipos de elementos:
  - Contenedores: pueden albergar en su interior tanto texto como otros elementos (dependerá de su modelo de contenido como luego iremos viendo).
  - Vacíos: no puede contener ni texto ni ningún otro elemento. Se utilizan para transmitir información mediante los valores que toman sus atributos.
- Vamos a ir desarrollando una DTD de ejemplo para ver como se realiza la declaración de los elementos y demás componentes de la DTD.
  - Se trata de un sistema de catalogación de libros de una biblioteca, le denominaremos biblioteca.dtd.
  - Los elementos que vamos a considerar son:LIBRO, TITULO, AUTOR, EDITORIAL, CUBIERTA, CATEGORIA, ISBN, NOTA, COMENTARIOS.

DTD: Elementos

- La primera versión de nuestra dtd quedaría así:

```
<!ELEMENT libro>
<!ELEMENT titulo>
<!ELEMENT autor>
<!ELEMENT editorial>
<!ELEMENT cubierta EMPTY>
<!ELEMENT categoria EMPTY>
<!ELEMENT isbn>
<!ELEMENT nota EMPTY>
<!ELEMENT comentarios>
```

- De este ejemplo nos debe quedar claro lo siguiente:
  - Un elemento de una DTD se expresa a través de <!ELEMENT y un nombre:

    ```
    <!ELEMENT nombre_elemento>
    ```
  - Para declarar elementos vacíos se añade la palabra EMPTY después del nombre del elemento:

    ```
    <!ELEMENT nombre_elemento EMPTY>
    ```

DTD: Elementos

- Declaración de elementos:

```
<ELEMENT nombre_elemento categoría>
```

O bien

```
<ELEMENT nombre_elemento (contenido_del_elemento)>
```

- Un elemento puede tener contenido de varios tipos:
  - Datos: marcados como #PCDATA

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

  - Vacío: no contiene nada

```
<!ELEMENT nombre_elemento EMPTY>
```

  - Cualquiera: sin restricciones a su contenido

```
<!ELEMENT nombre_elemento ANY>
```

  - Otros elementos anidados dentro de él (elementos-hijo en el árbol)
  - Datos y otros elementos

DTD: Elementos con hijos

- Cuando un elemento tenga como contenido otros elementos se usará la secuencia:

```
<!ELEMENT nombre_elemento (elemento_hijo_1,…,elemento_hijo_n)>
```

- Cada uno de estos elementos podrá tener una cardinalidad:
  - El elemento puede aparecer 0 ó 1 veces

    ```
    elemento_hijo?
    ```

  - El elemento puede aparecer 0 ó más veces

    ```
    elemento_hijo*
    ```

  - El elemento debe aparecer 1 ó más veces

    ```
    elemento_hijo+
    ```

  - El elemento debe aparecer 1 vez

    ```
    elemento_hijo
    ```

  - Además, podrá definirse la aparición de un elemento u otro, pero no ambos usando el carácter "|"

    ```
    <!ELEMENT nombre_elemento (elemento_hijo_A | elemento_hijo_B)>
    ```

DTD: Elementos

- La primera versión de nuestra dtd quedaría así:

```
<!ELEMENT biblioteca (libro+)>
<!ELEMENT libro (titulo, autor?, editorial, cubierta,
   categoria, isbn, nota, comentarios?)>
<!ELEMENT titulo (#PCDATA)>
<!ELEMENT autor (#PCDATA)>
<!ELEMENT editorial (#PCDATA)>
<!ELEMENT cubierta EMPTY>
<!ELEMENT categoria EMPTY>
<!ELEMENT isbn (#PCDATA)>
<!ELEMENT nota EMPTY>
<!ELEMENT comentarios (#PCDATA)>
```

DTD: Atributos

Los Atributos

- Se trata de *modificadores* de los elementos. Suministran información adicional y más específica sobre un elemento y su contenido. Matizan su significado.
- Cada elemento puede tener una **lista de atributos**.
- En el ejemplo del libro, las etiquetas `CAPITULO y PARRAFO` poseían atributos que permitían numerarlos:

```
<CAPITULO NUM="1">
<PARRAFO NUM="1">
```

- El uso de los atributos en combinación con algunas etiquetas permite limitar el número de etiquetas necesarias para escribir un documento.
- En HTML también tenemos estos atributos.

```
<IMG SRC="milogo.gif" ALT="milogo">
```

- En HTML muchas veces se sobre explotan los atributos para temas de formato: BACKGROUND, FONT, COLOR, etc. En XML esto habrá que evitarlo y resolverlo con las hojas de estilo.

DTD: Atributos

- La sintaxis:

```
<!ATTLIST nombre_elemento nombre_atributo tipo valor_por_defecto>
```

- Según su contenido los atributos pueden ser de distinto tipo.
  - CDATA: Datos de tipo **carácter**. Los valores puede ser cualquiera cadena de texto.
  - (valor1|valor2|…|valorn): **enumeración** de literales de valores posibles. Los valores que puede asumir un atributo estan limitados a una colección de valores predefina
  - ID: Destino de enlaces. Se utilizan para dar una **identificación** única al elemento, que no se puede repetir a lo largo del documento.
  - IDREF: Origen de enlaces. Existe la posibilidad de utilizar los identificadores ID para hacer referencias cruzadas. En este caso donde queramos hacer la referencia deberemos poner un atributo de tipo IDREF.
  - ENTITY: Referencia a **entidad externa**. Se emplean para la inclusión de fichero gráficos, sonido,etc

DTD: Atributos

Tipos de atributos según su necesidad:

- La obligatoriedad o no de los atributos se define mediante las siguientes palabras reservadas:
  - Atributos **requeridos**: Se trata como su nombre indica de atributos que siempre que se ponga un elemento al que acompañan deben especificarse para que el documento sea válido.
    - La etiqueta asociada es #REQUIRED.
  - Atributos **fijos**:
    - Estos poseen un valor predeterminado que no puede ser alterado por el creador del documento.
    - Pueden no emplearse pero si se emplean deben tomar el valor establecido por el diseñador de la DTD.
    - La etiqueta asociada es #FIXED valor
  - Atributo opcionales:
    - #IMPLIED:

DTD: Atributos

- Si nos fijamos en el ejemplo de la biblioteca.dtd, en ella hemos añadido algunos atributos a los elementos:

```
<!ELEMENT biblioteca (libro+)>
<!ELEMENT libro (titulo, autor?, editorial, cubierta, categoria,
isbn, nota, comentarios?)>
<!ELEMENT titulo (#PCDATA)>
<!ELEMENT autor (#PCDATA)>
<!ELEMENT editorial (#PCDATA)>
<!ELEMENT cubierta EMPTY>
<!ATTLIST cubierta TIPO (BLANDA|DURA) #REQUIRED>
<!ELEMENT categoria EMPTY>
<!ATTLIST categoria CLASE (FICCION| FANTASIA| CFICCION| TERROR|
HISTORICO | NOFICCION) "FICCION">
<!ELEMENT isbn (#PCDATA)>
<!ELEMENT nota EMPTY>
<!ATTLIST nota CALIF (1 |2|3|4|5) #IMPLIED>
<!ELEMENT comentarios>
```

- Es posible añadir más de un atributo cada vez por ejemplo para una imagen HTML:

```
<!ATTLIST img SRC CDATA #IMPLIED
              ALIGN (left|rigth|center) "left" >
```

DTD: Comentarios

- Se pueden añadir comentarios para documentar el DTD.
- La sintaxis es la misma que en los documentos XML, marcándolos con los delimitadores “<!--” y “-->”, no pudiéndose usar el doble guión en ningún punto del comentario.

Entidades

Las Entidades

- Los documentos XML se definen en base a caracteres. Es posible definir **grupos de estos caracteres** como unidades de datos. Estas unidades de datos se denominan entidades.
- Una entidad tiene dos partes fundamentales:
  - Definición de la entidad.
  - Referencia a la entidad.
- Cuando se define la entidad se le asigna un nombre. La referencia a la entidad emplea dicho nombre para indicar al analizador que sustituya el texto correspondiente a la entidad en ese punto del documento.
- Podemos clasificar las entidades como:
  - Internas: Las que hacen referencia a elementos en el propio documento.
  - Externas: las que se refieren a archivos externos.
- O bien como:
  - Procesables: Las que son otros documentos XML
  - No procesables: Las que se refieren a otros documentos no XML.

XML: Entidades

- Las entidades cumplen diferentes funciones:
  - Inserción de **cadenas de texto**, sea porque éstas se usan habitualmente o porque están formadas por caracteres provenientes de juegos de caracteres específicos y diferentes de aquél en el cual está codificado el documento.
  - Inserción de **componentes de datos no textuales**, datos binarios como por ejemplo elementos multimedia (vídeo, imágenes, sonido).
  - **Modularización de los documentos** mediante referencias a otros documentos.
- En general, definiremos una entidad con la sintaxis:

```
<!ENTITY nombre_entidad "valor_entidad">
```

Entidades de carácter

- **Entidades de carácter**: se usan para predefinir caracteres especiales o que no están en el juego de caracteres en que está codificado el documento.
  - Ej: &aacute → á
- En el documento se podrá hacer referencia a entidades de carácter incluyendo &#N, donde N es el código ASCII del carácter que queremos incluir.
- XML incluye predefinidas varias entidades de carácter para los caracteres usados habitualmente: <, >, &, “ y ‘. Podremos usar el código decimal con el formato &#N; o el hexadecimal con el formato &#xN; donde N es el número hexadecimal del carácter a insertar.

Entidades de carácter

- Entidades habituales de XML
  - lt <
  - gt >
  - apos ‘
  - quot “
  - amp &
- Entidades útiles
  - aacute
  - eacute
  - Iacute
  - ….
  - Como en HTML

DTD: Entidades de texto

- **Entidades de texto**: Las entidades de texto se usan para sustituir cadenas de texto usadas habitualmente por otras cadenas más cortas o fáciles de recordar o teclear. Por ejemplo:

<! ENTITY uv “Universidad de Valencia”>

<!ENTITY iso “International Standards Organization”>

- En los documentos las referiremos escribiendo el identificador de la entidad entre los caracteres “&” y “;”. De este modo al poner en el documento XML

```
&uv;
```

estaríamos realmente denotando

```
“Universidad de Valencia”
```

DTD: Entidades de texto externas

- Sirven para referenciar otros **documentos XML externos** a la entidad documento, de forma que el conjunto puede verse como una única entidad resultante de *ensamblar* diferentes módulos.
- Permiten por tanto un trabajo de equipo al dividir en módulos o componentes ensamblables el trabajo de diseño en proyectos complejos.
- Se declaran mediante la sintaxis:

```
<!ENTITY módulo SYSTEM "módulo.xml">
```

- Cuando se crea un documento modular, sólo la entidad documento, la principal, contiene la declaración de tipo de documento. A partir de ahí, se asocia cada entidad externa, cada módulo, mediante un archivo XML que sólo contiene la declaración de la entidad. Por ejemplo, un documento XML podría tener la siguiente forma:

```
<!DOCTYPE libro system "libro.dtd"
[
   <!ENTITY indice system "indice.xml">
   <!ENTITY cap1 system "capitulo1.xml>
   …
   <!ENTITY capn system "capitulon.xml>
]>
```

DTD: Entidades externas no procesables

- Las Entidades externas no procesables hacen referencia a archivos externos no XML.
- Se declaran con el calificador SYSTEM o PUBLIC y van acompañadas de una notación:

```
<!ENTITY gráfico SYSTEM "fichero.gif" NDATA gif>
```

La notación debe ir al comienzo del DTD, antes de la declaración de entidad

```
<!NOTATION gif SYSTEM "\windows\gifviewer.exe"
```

La notación sirve para indicar el programa con el que procesar la entidad. En el ejemplo, la entidad es un gráfico que se debe visualizar con el programa *gifviewer.exe*

Diseño de los DTD

- Es más útil crear DTD externos, ya que se reutilizan para varios documentos y además los cambios se aplican directamente a todos ellos modificando una sola vez el DTD
- Si se declaran DTD externos e internos para un mismo documento, las declaraciones del conjunto interno tienen mayor precedencia que las del externo.

XML

- XML utilizando la DTD de la biblioteca

```
<?XML VERSION="1.0" ENCODING="UTF-8"?>
<!DOCTYPE biblioteca SYSTEM "biblioteca.dtd">
                                        <!-- HASTA AQUi ES PROLOGO-->
<BIBLIOTECA> <!-- ELEMENTO DE DOCUMENTO-->
    <LIBRO>
          <TITULO>LA REGENTA</TITULO>
          <AUTOR>Leopoldo Alas "Clarin"</AUTOR>
          <EDITORIAL>RBA Editores</EDITORIAL>
          <CUBIERTA TIPO="DURA" />
          <CATEGORIA CLASE="FICCION" />
          <ISBN>ISBN-33490</ISBN>
          <NOTA CALIF="4">
          <COMENTARIOS>
          Es un clasico muy bueno aunque se puede hacer dificil de leer
          </COMENTARIOS>
    </LIBRO>
    <!-- Aquí podrian ir tantos libros como quisiesemos con el formato
          de este, donde se pueden quitar los comentarios -->
</BIBLIOTECA>
```

Validación

- Si nuestro documento XML está asociado a un DTD, será necesario confirmar de alguna manera que el código XML se corresponde con lo que está definido en la DTD.
- Esto se conoce como validación del documento XML y se efectúa de manera automática mediante un programa llamado *parser*. El *parser* analiza la DTD y después el código del documento XML. Si el código del documento cumple los requisitos explícitos en la DTD se dice que es un documento XML *válido*.
- Los entornos de desarrollo XML incluyen *parsers* que permiten realizar esta labor. Además se pueden encontrar parsers en línea.

Para la semana que viene

- Uso de Unicode en XML
- ¿Puedo usar juegos de caracteres distintos en un mismo documento?
- Que es el tipo NMTOKEN!!
- Diferencia entre la versión 1.0 y la 1.1

UNICODE

- Norma ISO 10646. Ultima versión 6.0
  - http://www.unicode.org/versions/Unicode6.0.0/

    - UTF-8 — codificación orientada a byte con símbolos de longitud variable.
    - UTF-16 — codificación de 16 bits de longitud variable optimizada para la representación del plano básico multilingüe (BMP).
    - UTF-32 — codificación de 32 bits de longitud fija, y la más sencilla de las tres.

¿Puede definirse más de un código de caracteres en un documento XML?

- http://www.w3.org/TR/xml/
  - **2.12 Language Identification**
- http://www.rfc-editor.org/rfc/bcp/bcp47.txt
  - Tags for Identifying Languages
- http://es.wikipedia.org/wiki/ISO_3166-1
  - Listado de códigos.

Mejoras 1.1 a 1.0

- http://www.w3.org/TR/2006/REC-xml11-20060816/#sec-xml11 (apartado 1.3)

-Se permite usar caracteres unicode 2.0 en los nombres de elementos o atributos.

-Se permiten más caracteres para nombres.

-Es más "portable" que la versión 1.0

-Está normalizado respecto a la versión 1.0