XML, DTD y hojas de estilo

Introducción

XML existe porque HTML ha tenido mucho éxito. Pero con objeto de corresponder a este éxito, se le ha extendido introduciéndose muchas etiquetas nuevas (más de 100 sin contar las específicas de los exploradores), convirtiéndose en un lenguaje complicado. Pero aún así, para aplicaciones específicas como comercio electrónico, tiendas virtuales, intercambio de datos... se requieren definir más.

Otro problema relacionado con el HTML es que para formatear una página son necesarias muchas etiquetas, tardándose bastante tiempo en descargarlas y desplegarlas. Además hace que los exploradores sean bastante complejos.

Las aplicaciones de XML se pueden clasificar en Aplicaciones de Documento (publicar en Web) y Aplicaciones de Datos (intercambio de datos). Algunos ejemplos de uso pueden ser:

- Mantenimiento de sitios Web grandes.
- Intercambio de información entre organizaciones.
- Acceso a Bases de Datos.
- Contenido sindicado, donde el contenido se pone disponible para diferentes sitios web.
- Comercio electrónico.
- Libros electrónicos.

Características del lenguaje XML respecto HTML:

- No predefine etiquetas, el autor las crea. De ahí que sea extensible.
- Es estricto.

Estándares acompañantes del XML:

- XML Namespace: permite la extensibilidad y reutilización de un documento, asignando un dueño a los elementos del mismo.
- Hojas de Estilo: XSL (hojas de estilo extensible) y CSS (hojas de estilo en cascada). Especifican cómo deben presentarse en pantalla, papel o editor los documentos.
- DOM (modelo de objetos de documento) y SAX (API simple para XML) son APIs para acceder a documentos XML sin preocuparse de la sintaxis. DOM es mejor para formularios y editores, SAX es mejor para intercambio de datos.
- Xlink y Xpointer, en desarrollo, permite establecer relaciones entre documentos.

XML tiene sus raíces en la publicación electrónica (documentación técnica, páginas web,...) ligada generalmente a una hoja de estilo. Este lenguaje es una buena alternativa para mantener la documentación en un formato independiente de los medios y convertirla automáticamente en formatos de publicación como HTML, PDF, PostScript, RTF...

Sintaxis de XML

Un documento XML es texto puro y consiste en *datos de caracteres* (información) y *marcado* (estructura). El marcado se reconoce porque va encerrado entre paréntesis angulares (<>).

El pilar de XML es el **elemento**, que se definirá con su etiqueta de apertura y de cierre, siempre obligatorias.

Los nombres del elemento deben comenzar con letra o con underline y después cualquier carácter a excepción del espacio en blanco y los dos puntos. Los nombres no pueden comenzar con las letras xml. En XML se distinguen las mayúsculas de las minúsculas, generalmente se trabaja con minúsculas.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- documento de ejemplo inspirado en vCard 3.0 -->
<agenda>
    <entrada>
        <nombre> Marta </nombre>
        <direccion> Santander </direccion>
    </entrada>
    <entrada>
        <nombre> Pepe </nombre>
        <direccion> Bilbao </direccion>
    </entrada>
</agenda>
```

A través de los **atributos** se puede añadir información adicional a los elementos, estos tienen nombre y valor. El valor va entre comillas dobles o simples en función del contenido (si en la información hay comillas dobles, se enmarcará con simples y viceversa).

```
<tel preferente="true"> 942201363 </tel>
```

Los elementos que no tienen información se conocen como vacíos y se pueden escribir de estas dos formas:

<correo-e href=”malito:pepe@unican.es”> </correo-e>

<correo-e href=”malito:pepe@unican.es”/>

Un documento XML es un árbol de elementos, no hay límite de profundidad. Todos los elementos deben ser hijos de un solo elemento.

La declaración XML reconoce a un documento como tal y debe ir en la primera línea del documento. Un procesador XML puede rechazar un documento que tenga otra versión.

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

Los comentarios se insertan del siguiente modo:

<!-- documento de ejemplo inspirado en vCard 3.0 -->

XML organiza el documento en **entidades** y no en términos de archivos. Una entidad se representa por su nombre precedido de un & y seguido de un ;. Por ejemplo:

< inserta el carácter <

& inserta el carácter & (p.ej. compañía>Mark & Spencer </compañía>)

Atributos especiales:

- xml:space para controlar si la aplicación debe eliminar los espacios en blanco innecesarios (default) o no (preserve).
- xml:lang para indicar el idioma en el que está escrito.

  What colour is it?

Las instrucciones de procesamiento son un mecanismo para insertar instrucciones que no son XML, en forma de secuencia de comandos. Se representan por <? ?>. El primer nombre es el destino e indica la aplicación o dispositivo al que están dirigidas las instrucciones. El resto están en formato específico de la aplicación.

<?xml-hojaestilo href=”simple.xsl” type=”text/xsl”?>

Las secciones CDATA permiten que el procesador XML ignore el marcado, útil para la escritura de fórmulas. CDATA significa datos de caracteres, que difiere de PCDATA que son datos de caracteres analizados y que no pueden contener caracteres de marcado.

Sintaxis de DTD

La sintaxis de las DTDs es diferente a la sintaxis de los documentos XML.

El DTD es un mecanismo para describir cada objeto que puede aparecer en el documento, empezando con los elementos.

A continuación presentamos un ejemplo:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Elemento de alto nivel, la agenda de direcciones-->
<!ELEMENT agenda (entrada)+>

<!-- Una entrada es un nombre con sus direcciones, telefónos..Puede aparecer una o más veces.-->
<!ELEMENT entrada (nombre,(direccion)*,(tel)*,(fax)*,(correo-e)*)>

<!-- Un nombre se compone de una cadena de caracteres, nombre de pila y apellido-->
<!ELEMENT nombre (#PCDATA|nombre-pila|apellido)*>
<!ELEMENT nombre-pila (#PCDATA)*>
<!ELEMENT apellido (#PCDATA)*>

<!-- Dirección, si hay varias direcciones el atributo preferente indica la predeterminada-->
<!ELEMENT direccion (calle,(ciudad)?,codigo-postal,region,pais)>
<!ATTLIST direccion preferente  (true|false) "false">

<!ELEMENT calle (#PCDATA)*>
<!ELEMENT ciudad (#PCDATA)*>
<!ELEMENT codigo-postal (#PCDATA)*>
<!ELEMENT region (#PCDATA)*>
<!ELEMENT pais (#PCDATA)*>

<!-- telefono, fax y correo -->
<!ELEMENT tel (#PCDATA)*>
<!ATTLIST tel preferente  (true|false) "false">

<!ELEMENT fax (#PCDATA)*>
<!ATTLIST fax preferente  (true|false) "false">

<!ELEMENT correo-e EMPTY>
<!ATTLIST correo-e
        href CDATA #REQUIRED
        preferente  (true|false) "false">
```

Declaración de un elemento:

<!ELEMENT agenda (entrada)+>

donde agenda es el nombre del elemento y (entrada)+ es el modelo de contenido.

Palabras reservadas:

#PCDATA: se pone para analizar datos de caracteres y significa que el elemento puede contener texto. Generalmente se utiliza para los elementos hoja.

EMPTY: indica que el elemento es vacío. Siempre indica que el elemento es hoja.

ANY: indica que el elemento puede contener cualquier otro elemento declarado en el DTD. Apenas se utiliza.

Indicadores de ocurrencia:

+	1 o más veces
*	0 o más veces
?	0 ó 1 vez
	solo una vez

Conectores:

Carácter ",": indica que los elementos de la derecha e izquierda deben aparecer en el mismo orden dentro del documento.

Carácter "|": indica que sólo uno de los elementos de la derecha e izquierda deben aparecer dentro del documento.

Declaración de atributos (marcado, nombre del elemento, nombre del atributo, el tipo de atributo, valor predeterminado):

<!ATTLIST tel preferente (true|false) "false">

Se pueden agrupar la declaración de varios atributos al mismo elemento:

```
<!ATTLIST correo-e
        href CDATA #REQUIRED
        preferente (true|false) "false">
```

La DTD proporciona más control sobre el contenido de los atributos que sobre el contenido de los elementos. Los atributos se dividen en tres categorías:

- Atributos de cadena de caracteres que contienen texto

  <!ATTLIST correo-e href CDATA #REQUIRED>

- Atributos de validación que restringen el contenido del atributo

  <!ATTLIST entrada id ID #IMPLIED>

- Atributos de tipo enumerado que adeptan un valor dentro de una lista

  <!ATTLIST entrada preferente (true|false) "false">

Los tipos de atributos pueden tomar cualquiera de los siguientes valores:

- CDATA para atributos de cadena de caracteres
- ID para identificador. Su nombre es único para el documento.
- IDREF debe ser el valor de un ID usado en otra parte del mismo documento. Se usa para crear vínculos dentro de un documento.
- IDRES es una lista de IDREF separados por espacios.
- ENTITY debe ser el nombre de una entidad externa.
- ENTITIES es una lista de ENTITY separadas por espacios.
- NMTOKEN es una palabra sin espacios.
- NMTOKENS es una lista de NMTOKEN separados por espacios.
- Lista de tipos enumerados es una lista cerrada de nmtokens separados por |.

De forma opcional, la DTD puede especificar un valor predeterminado para el atributo. Puede tomar uno de los siguientes:

- #REQUIERED: el valor debe ser proporcionado por el documento
- #IMPLIED: si no se proporciona el valor, la aplicación debe usar su propio valor predeterminado.
- #FIXED seguido de un valor significa que el valor del atributo debe ser el valor declarado dentro de la DTD.
- Un valor literal significa que el atributo tomará este valor si no se proporciona ningún valor en el documento.

Declaración de tipo de documento en un documento XML(marcado, nombre del elemento de nivel superior,dtd):

<!DOCTYPE agenda SYSTEM “agenda.dtd”>

Ejemplo de utilización de agenda.dtd en el documento doc-agenda.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE agenda SYSTEM "agenda.dtd">
<agenda>
        <entrada>
                <nombre>Marta
                        <apellido>Zorrilla </apellido>
                </nombre>
                <direccion>
                        <calle> Avda Los Castros</calle>
                        <ciudad>Santander</ciudad>
                        <codigo-postal>39005</codigo-postal>
                        <region>Cantabria</region>
                        <pais>España</pais>
                </direccion>
                <tel preferente="true"> 942201363</tel>
```

```
            <correo-e href="zorrillm@unican.es"></correo-e>
        </entrada>
        <entrada>
            <nombre> Angel
                <nombre-pila> Dr. Angel</nombre-pila>
            </nombre>
            <tel preferente="true"> 942335353</tel>
            <tel>942232323</tel>
        </entrada>
</agenda>
```

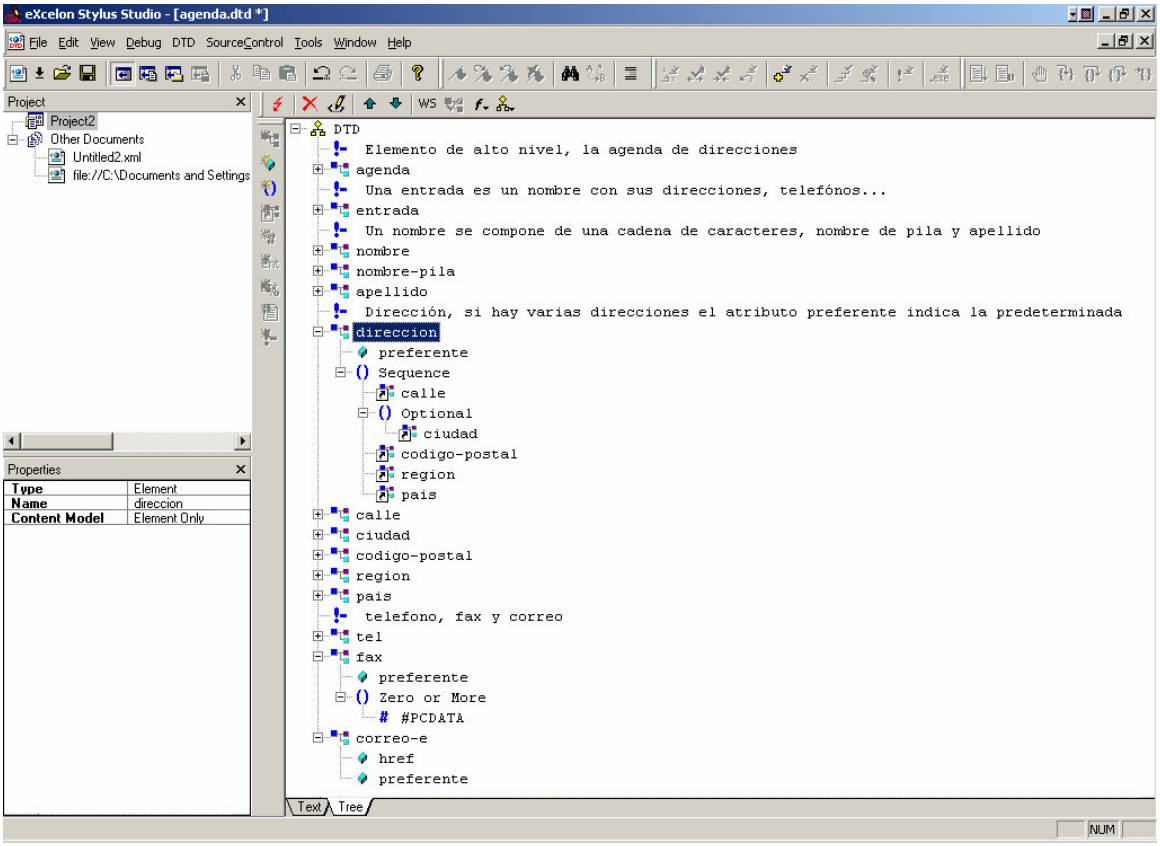


Fig. 1. Árbol de elementos de la DTD agenda creada con eXcelon Stylus Studio.

La DTD está dividida en subconjuntos externos e internos. Los internos están incluidos entre corchetes dentro de la declaración del tipo de documento. Los externos son referenciados o bien, por el URI (Identificador Universal de Recursos), o bien, por un identificador público que apunta a una DTD grabada con el ISO 9070. Estos pueden ser del sistema (SYSTEM) o públicos (PUBLIC). Estos últimos se utilizan para manejar copias locales de DTDs.

<!DOCTYPE agenda SYSTEM “http://www.aaa.com/agenda.dtd”>

<!DOCTYPE agenda PUBLIC “-//Pineappleasoft//libreta direcciones//SP”

“http://catwoman.pineapplesoft.com/dtd/libreta-direcciones.dtd”>

(“-//organización//propietario//idioma” URI)

Los identificadores de sistema deben ir después de los públicos.

Si todas las entradas que pueden influir en el documento están en subconjuntos internos de la DTD, se dice que el documento es independiente. XML tiene un atributo que indica si el documento es independiente, standalone.

```
<?xml versión=”1.0” standolene=”yes”?>
```

Las **entidades** son la representación física de los documentos XML. El documento XML, su DTD y los archivos a los que hace referencia son entidades. Las entidades se clasifican en :

- Generales y de parámetro.

  Las generales pueden aparecer en cualquier lugar del texto o de los marcadores (p.ej. &lt entidad predefinida).

  Las de parámetro solo pueden aparecer en la DTD. No tienen muchas aplicaciones.

  ```
  <!ENTITY %bolean “(true|false) ‘false’”>
  <!ELEMEN tel (#PCDATA)>
  <!ELEMEN tel prefered %bolean;>
  ```

- Internas y externas.

  Las internas son almacenadas en el documento y las externas apuntan a un identificador del sistema o público.

- Analizadas y no analizadas.

  Las externas pueden ser analizadas o no. Si son analizadas, la entidad debe contener texto y marcado XML válidos, se utilizan para compartir texto entre varios documentos.

  ```
  <?xml versión=”1.0” encoding=”ISO-8859-1”>
  <!DOCTYPE agenda [
  <!ENTITY marta SYSTEM ‘marta.ent’>
  <!ENTITY angel SYSTEM ‘angel.ent’>
  ]>
  <agenda>
      &marta;
      &angel;
  </agenda>
  ```

Las entidades no analizadas se utilizan para el contenido que no es XML como imágenes, sonidos, películas,.... Las entidades no analizadas proveen un mecanismo para cargar los datos y asociarlas con la herramienta adecuada.

```
<!NOTATION GIF89a SYSTEM "image/gif">
<!ENTITY warning SYSTEM "warning.gif" NDATA GIF89a>
```

Las entidades son muy útiles para crear módulos y ayudar a manejar grandes conjuntos de DTDs y documentos. Ejemplos de diseño en http://www.oasis-open.org/sml/xml.html

Espacios de nombre

Tratan de resolver los conflictos de extensión de las DTDs con los mismos elementos pero con distinto significado.

Para ello se deben declarar los prefijos de los elementos apoyándose en la URI (son únicos pues se basan en nombres de dominio). La declaración de un espacio de nombre se realiza mediante atributos con el identificador xmlns seguidos por el prefijo a utilizar.

```
<?xml versión=”1.0” encoding=”ISO-8859-1”?>
<agenda xmlns:cld=”http://jocker.playfield.com/star-categoria/1.0”
        xmlns:pdr= ”http://peguin.xmli.com/review/1.0”>
        <nombre>Marta</nombre>
        <cld:categoria>PF 2</cld:categoria>
        <pdr:categoria>Asociado</pdr:categoria>
</agenda>
```

Hojas de Estilo

XML se concentra en la estructura de la información y no en su apariencia. CSS es un conjunto de normas que le indican al explorador WEB cuál es la fuente, estilo y margen a utilizar para mostrar el texto. XSL es más ambicioso, permite transformar el documento antes de mostrarlo.

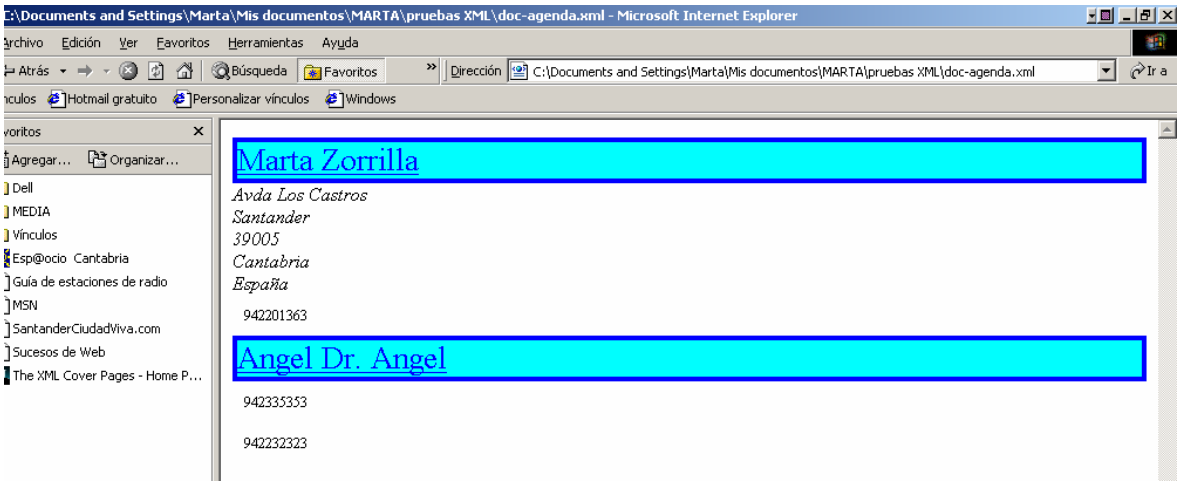
XSLT es un lenguaje para identificar la transformación de documentos XML. Permite agregar elementos específicamente para verlos como un logotipo, generar material a partir de otro, presentar información con un nivel adecuado de detalles, alternar entre diferentes DTDs o distintas versiones de las mismas y transformar documentos XML a HTML.

A continuación se presenta una hoja de estilos en CSS.

```
/*hoja de estilo CSS */
entrada
{
font-family: Palatino,Garoamond,"Times New Roman";
font-size: 12pt;
margin: 5px;
/*background-image: url(warning.gif);*/
}
nombre
{
display:block;
background: cyan;
margin-botton: 10px;
font-size: 20pt;
color: blue;
text-decoration: underline;
border-style: solid;
}
tel
{
display:block;
margin-botton: 10px;
font-size: 10pt;
padding: 10px;
}
direccion,calle,ciudad,codigo-postal,region
{
font-style: italic;
display:block
}
```

y su visualización en el explorador, al incorporarlo en el fichero de doc-agenda.xml con la siguiente instrucción:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet href="hoja1.css" type="text/css"?>
```



XSL describe la jerarquía del documento de origen, la del documento resultante y cómo transformar uno en el otro. El elemento principal es *stylesheet.* Dado que la hoja de estilo contiene elementos de diferentes documentos se utiliza el espacio de nombres para organizarlos. El espacio xsl se utiliza para el vocabulario XSL y el espacio de nombre del documento resultante es el predeterminado de HTML.

```
<xsl:stylesheet
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns="http://www.w3.org/TR/REC-html40">
<xls:output method="html"/>
```

La hoja de estilo mayormente es una hoja de plantillas. En una plantilla se distinguen: el parámetro match, que es la ruta de acceso al elemento en la jerarquía; y, el contenido de la plantilla, que lista los elementos por insertar en la jerarquía resultante. La ruta de acceso es similar a la de los archivos. Arranca de la raíz (“/”) .

```
<xsl:template match="seccion/epigrafe">
        <P><I>   <xsl:apply-templates/> </I> </P>
</xsl:template>
```

Esta plantilla sólo se aplica a las url con el atributo mailto.

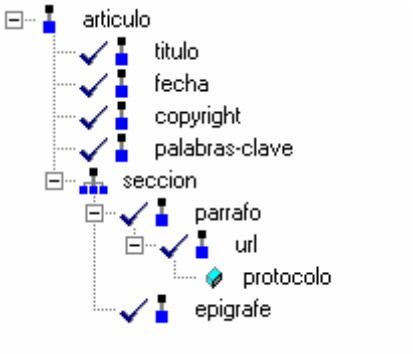
```
<xsl:template match="url[protocolo='mailto']">
        <A>    <xsl:attribute name="HREF">mailto:<xsl:apply-templates/>
          </xsl:attribute>
          <xsl:apply-templates/>
```

```
        </A>
    </xsl:template>
```

Esta plantilla hará que no se muestre ninguna de estas informaciones.

```
<xsl:template match="fecha|palabras-clave|copyright"/>
```

El procesador XML cargará al documento de origen haciendo coincidir al nodo en curso con la plantilla. Cuando el procesador se encuentra <xsl:apply-templates/> se mueve al nodo secundario del nodo en curso y hace corresponder una plantilla. En el ejemplo que a continuación se presenta el procesador realizará el siguiente camino: busca la plantilla que coincida con la raíz, luego se mueve al nodo articulo (como no hay ninguna definida, el procesador utiliza una predefinida). Sigue con el título, fecha, copyright y palabras-clave donde las tres últimas no generan datos en la jerarquía resultante. Después analizará la seccion que coincide con plantilla predeterminada y pasará a analizar el parrafo dando mayor prioridad a url con protocolo respecto a la url. Terminará con epígrafe.



En la siguiente tabla se listan otros elementos XSL que calculan nodos en la jerarquía:

xsl:element	Genera un elemento con un nombre calculado
xsl:attribute	Genera un atributo con un nombre calculado
xsl:attribute-set	Combina adecuadamente varios xsl:atributos
xsl:text	Genera un nodo text
xsl:processing-instruction	Genera una instrucción de procesamiento
xsl:comment	Genera un comentario
xsl:copy	Copia al nodo en curso
xsl:value-of	Calcula el texto al obtenerlo de la jerarquía origen o mediante la inserción de una variable
xsl:if	Instancia su contenido si la expresión es verdadera
xsl:choose	Selecciona los elementos por instanciar entre las alternativas posibles
xsl:number	Genera un número con formato

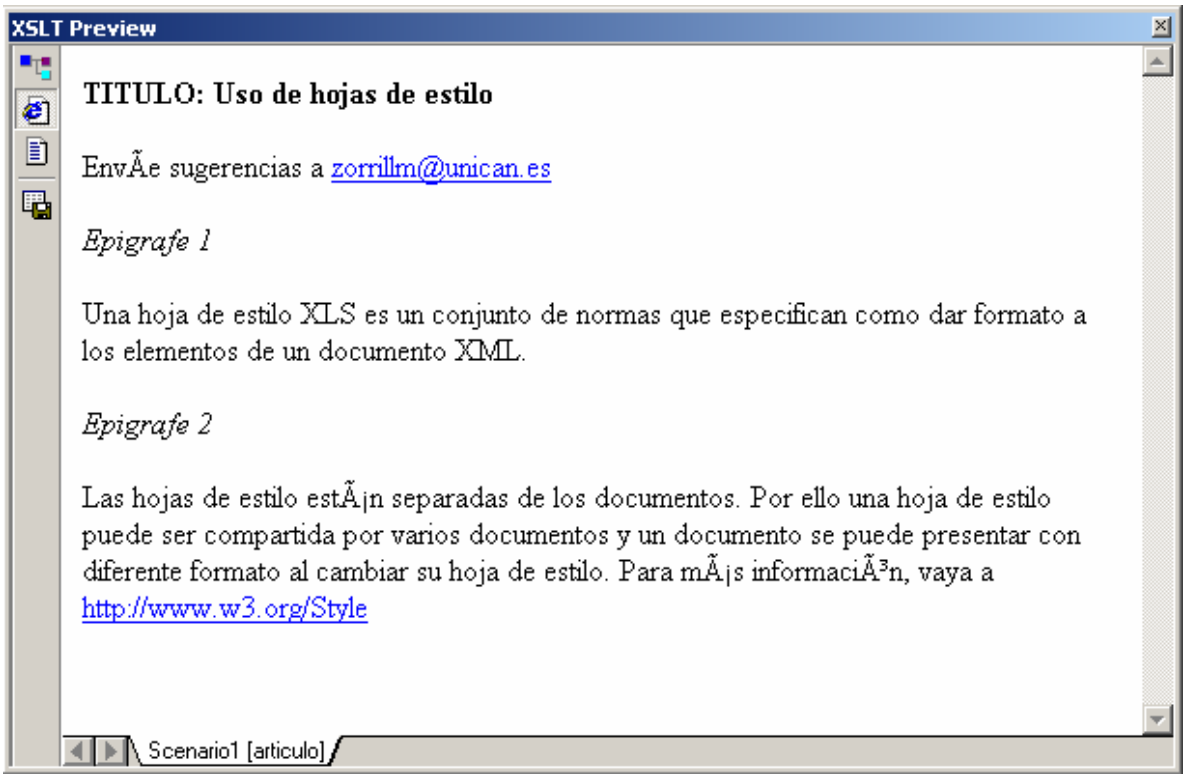
A continuación se presenta el ejemplo completo, primero la hoja de estilo, después el documento xml y por último se presenta el aspecto del documento en el explorador al utilizar la hoja de estilos.

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns="http://www.w3.org/TR/REC-html40">
<xls:output method="html"/>
<xsl:template match="/">
        <HTML>
        <head>
         <title> Articulo XML </title>
         </head>
         <body>
          <xsl:apply-templates/>
          </body>
         </HTML>
</xsl:template>
<xsl:template match="seccion/epigrafe">
        <P>    <I>       <xsl:apply-templates/> </I> </P>
</xsl:template>
<xsl:template match="articulo/titulo">
        <P>    <B>       <xsl:apply-templates/> </B> </P>
</xsl:template>

<xsl:template match="url">
        <A target="_blank"> <xsl:attribute name="HREF">
          <xsl:apply-templates/>
          </xsl:attribute>
          <xsl:apply-templates/>
   </A>
</xsl:template>
<xsl:template match="url[protocolo='mailto']">
        <A>    <xsl:attribute name="HREF">mailto:<xsl:apply-templates/>
```

```
            </xsl:attribute>
            <xsl:apply-templates/>
          </A>
</xsl:template>
<xsl:template match="parrafo">
          <P> <xsl:apply-templates/> </P>
</xsl:template>
<xsl:template match="fecha|palabras-clave|copyright"/>
</xsl:stylesheet>
```

```
<?xml version="1.0" encoding="ISO-8859-1"?><!DOCTYPE articulo [
<!ELEMENT titulo (#PCDATA)>
<!ELEMENT fecha (#PCDATA)>
<!ELEMENT copyright (#PCDATA)>
<!ELEMENT palabras-clave (#PCDATA)>
<!ELEMENT url (#PCDATA)>
<!ATTLIST url protocolo CDATA #REQUIRED>
<!ELEMENT parrafo (#PCDATA|url)*>
<!ELEMENT epigrafe (#PCDATA)>
<!ELEMENT seccion ((epigrafe)?,(parrafo)?)>
<!ELEMENT articulo (titulo,fecha,copyright,palabras-clave,(seccion)+)>]>
<articulo>
<titulo> TITULO: Uso de hojas de estilo </titulo>
<fecha> 1 de febrero 2001 </fecha>
<copyright> Marta Zorrilla </copyright>
<palabras-clave>XML, hojas de estilo, XLS</palabras-clave>
<seccion><parrafo>Envíe sugerencias a <url
protocolo="mailto">zorrillm@unican.es</url>
</parrafo></seccion>
<seccion><epigrafe>Epigrafe 1</epigrafe>
<parrafo>Una hoja de estilo XLS es un conjunto de normas que especifican como dar
formato a los elementos de un documento XML.</parrafo></seccion>
<seccion><epigrafe>Epigrafe 2 </epigrafe>
<parrafo>Las hojas de estilo están separadas de los documentos.
```

```
Por ello una hoja de estilo puede ser compartida por varios documentos
y un documento se puede presentar con diferente formato al cambiar su hoja de estilo.
Para más información, vaya a <url protocolo="_blank">http://www.w3.org/Style</url>
</parrafo></seccion>
</articulo>
```

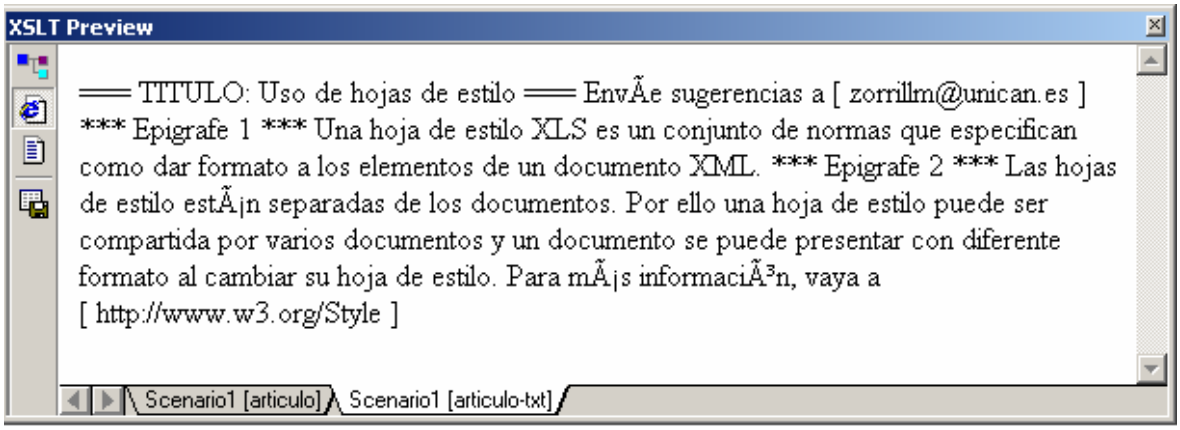


Si al mismo texto le convertimos en fichero de texto en vez de HTML, usaremos la plantilla siguiente:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xls:output method="text"/>


<xsl:template match="seccion/epigrafe">
        <xsl:text>*** </xsl:text>
                <xsl:apply-templates/>
        <xsl:text> ***</xsl:text>
</xsl:template>
```

```
<xsl:template match="articulo/titulo">
        <xsl:text>=== </xsl:text>
                <xsl:apply-templates/>
        <xsl:text> ===</xsl:text>
</xsl:template>

<xsl:template match="url">
        <xsl:text>[ </xsl:text>
          <xsl:apply-templates/>
        <xsl:text> ]</xsl:text>
</xsl:template>

<xsl:template match="parrafo">
        <xsl:text> </xsl:text>
          <xsl:apply-templates/>
        <xsl:text> </xsl:text>
</xsl:template>

<xsl:template match="fecha|palabras-clave|copyright"/>

</xsl:stylesheet>
```

y tendremos el resultado:



En Interner Explorer 5.0 se puede enviar el fichero xml junto con su hoja de estilos indicándoselo de la siguiente forma:

<?xml-stylsheet href=”articulo-explorer5.xsl type=”text/xsl”?>

Además habrá que modificar la hoja de estilos porque Explorer integra la versión inicial de XSL.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet
  xmlns:xsl="http://www.w3.org/TR/WD-xsl"
  xmlns="http://www.w3.org/TR/REC-html40">

<xsl:template match="*">
  <xsl:apply-templates/>
</xsl:template>

<xsl:template match="text()">
  <xsl:value-of select="."/>
</xsl:template>

<xsl:template match="/">
        <HTML>
        <head>
         <title> Articulo XML </title>
         </head>
         <body>
          <xsl:apply-templates/>
          </body>
         </HTML>
</xsl:template>

<xsl:template match="seccion/epigrafe">
        <P>    <I>       <xsl:apply-templates/> </I> </P>
</xsl:template>

<xsl:template match="articulo/titulo">
        <P>    <B>       <xsl:apply-templates/> </B> </P>
</xsl:template>
```

```
<xsl:template match="url">
        <A target="_blank"> <xsl:attribute name="HREF">
          <xsl:apply-templates/>
          </xsl:attribute>
          <xsl:apply-templates/>
   </A>
</xsl:template>
<xsl:template match="url[protocolo='mailto']">
        <A>    <xsl:attribute name="HREF">mailto:<xsl:apply-templates/>
          </xsl:attribute>
          <xsl:apply-templates/>
        </A>
</xsl:template>
<xsl:template match="parrafo">
        <P> <xsl:apply-templates/> </P>
</xsl:template>
<xsl:template match="fecha|palabras-clave|copyright"/>
</xsl:stylesheet>
```

Generar una tabla de contenido en el documento a partir de la información del DTD.

```
<?xml version="1.0" encoding="utf-8"?>

<!DOCTYPE xls:stylesheet [
  <!ENTITY sim-copy "&#0169;">]>

<xsl:stylesheet
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 xmlns="http://www.w3.org/TR/REC-html40">
<xls:output method="html"/>

<xsl:template match="/">
        <HTML>
        <head>
         <title> Artículo XML </title>
```

```
        <meta name ="palabras-clave">
          <xsl:attribute name="CONTENT">
            <xsl:value-of select="articulo/palabras-clave"/>,
                 </xsl:attribute>
        </meta>

        </head>
        <body>
         <P><B> Tabla de contenido </B></P>
         <UL>
                 <xsl:for-each select="articulo/seccion/epigrafe">
                  <LI><A>
                        <xsl:attribute name="HREF"> <xsl:value-of select="generate-
id()"/>
                          </xsl:attribute>
                        <xsl:value-of select="."/>
                        </A></LI>
                        </xsl:for-each>
                </UL>
           <xsl:apply-templates/>
                <P>Copyright &sim-copy; <xsl:value-of select="articulo/copyright"/>
</P>
         </body>
        </HTML>
</xsl:template>

<xsl:template match="seccion/epigrafe">
        <P>    <I>       <xsl:apply-templates/> </I> </P>
</xsl:template>

<xsl:template match="articulo/titulo">
        <P>    <B>       <xsl:apply-templates/> </B> </P>
</xsl:template>

<xsl:template match="url">
        <A target="_blank"> <xsl:attribute name="HREF">
```

```
            <xsl:apply-templates/>
            </xsl:attribute>
            <xsl:apply-templates/>
    </A>
</xsl:template>

<xsl:template match="url[protocolo='mailto']">
          <A>    <xsl:attribute name="HREF">mailto:<xsl:apply-templates/>
            </xsl:attribute>
            <xsl:apply-templates/>
          </A>
</xsl:template>

<xsl:template match="parrafo">
          <P> <xsl:apply-templates/> </P>
</xsl:template>

<xsl:template match="fecha|palabras-clave|copyright"/>

</xsl:stylesheet>
```