UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Documentos XML: análisis sintáctico y validación con Xerces

Apellidos, nombre	**Agustí i Melchor, Manuel** (magusti@disca.upv.es)
Departamento	**Departamento de Informática de Sistemas y Computadores**
Centro	**Escola Tècnica Superior d'Enginyeria Informàtica** Universitat Politècnica de València

1 Resumen

Un documento *eXtensible Markup Language, en adelante* XML[1], es uno cuya estructura está definida por el usuario y se ha de respetar para facilitar procesos posteriores. Sobre este es posible aplicar transformaciones y eso lo hace especialmente indicado para, a partir de un único documento de partida, obtener otros tipos de documentos, por ejemplo a partir de XML se puede obtener una versión en texto ASCII, en otro vocabulario de XML (SVG, RSS,...), HTML, RTF, ODT, DOCX o PDF.

El primer paso de esta cadena de procesos aplicables a un documento XML es la comprobación de su **estructura sintáctica**. Aunque esto se puede hacer manualmente, el uso de herramientas informáticas que automaticen la labor hace posible que se pueda aplicar a un número grande de documentos y a documentos de gran extensión. Este es el contexto habitual en repositorios y bancos de documentación, donde las labores de mantenimiento sobre este tipo de documentos no se pueden realizar individualmente y por la posible introducción de errores debido a la manipulación directa, que es propicia a ellos.

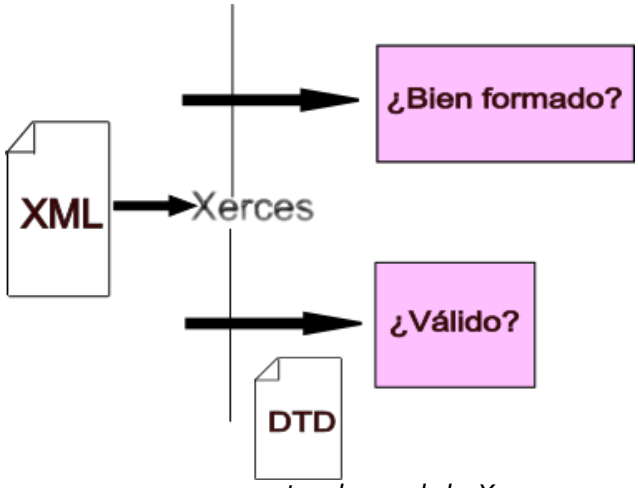

Figura 1: Diagrama que representa el papel de Xerces como analizador sintáctico.

En este trabajo vamos a comentar cómo se pueden **gestionar documentos XML desde la línea de órdenes** utilizando *Xerces* como analizador sintáctico. De su uso obtendremos la respuesta a las dos preguntas que muestra la Figura 1 de forma esquemática. Si está bien formado podremos plantearnos si es válido utilizando una declaración de tipo de documento (*Document Type Declaration* o, de forma abreviada, DTD[7]). Sí y solo sí, es válido, será posible aplicar una transformación a un documento XML, pero ese es otro tema. **En este documento no** abordamos la transformación del documento XML a otros posibles formatos de salida como los indicados.

Como requisitos para leer este documento, es interesante estar familiarizado con el uso de etiquetas, como p. ej. las que se usan en una página web.

En el texto aparecen órdenes a ejecutar en un terminal, se las reconoce porque van precedidas del signo '$', que no se ha de copiar al ejecutarlas.

---

1 Sobre su uso aplicado y los estándares derivados se pueden encontrar información en el sitio web de *xml.org* <http://www.xml.org/>.

2 Objetivos

La estructura sintáctica de un documento XML se define con dos términos: bien formado (o *well-formed* en la terminología anglosajona) y válido. Una vez que el lector haya explorado los contenidos de este documento con atención, será capaz de:

- Instalar las herramientas informáticas que se sugieren para automatizar las tareas indicadas.
- Determinar si un documento XML está bien formado.
- Aplicar la sintaxis de una DTD para la elaboración de un documento XML.
- Determinar si un documento XML es válido.

En lo que sigue, se asume que se está trabajando en una plataforma *GNU/Linux* donde ya existe una versión de *Java* instalada, pero la elección de las herramientas sugeridas permite estar trabajando en cualquier otra plataforma.

3 Introducción a XML y las herramientas para su análisis

Veamos primero cómo se escribe un documento XML, para ilustrar con un ejemplo el concepto de etiquetas y, después, hablamos de qué aplicaciones necesitamos para analizar la escritura de estos documentos XML.

XML es [1] un lenguaje de marcado que describe una clase de objetos de datos llamados "documentos XML" y, parcialmente, el comportamiento de los programas que los procesan. XML permite diseñar un lenguaje propio de etiquetas para describir esas clases de documentos. Puede encontrar en [2], [3] y [4] información ampliada sobre XML y su uso. XML proviene del estándar SGML[2], heredando de este el uso de las etiquetas para describir las partes de un documento, siguiendo unas mínimas reglas: que todo su contenido ha de estar escrito en forma de etiquetas con una serie de modificadores (llamados atributos); que las etiquetas pueden estar anidadas unas dentro de otras, pero parten todas de una (la raíz); y que toda etiqueta que se abra se tiene que cerrar y siempre en el mismo orden; y que en caso de que un elemento no tenga pareja (por no tener ningún contenido dentro), se le denomina elemento vacío y se indica con un / al final de la etiqueta inicial.

En este caso vamos a suponer que se ha de **construir un documento XML** que permita la descripción de un posible tipo "trabajo de asignatura". La Figura 2 muestra una posible organización de contenidos que podría servir para gestionar los trabajos que en una asignatura se estuvieran llevando a cabo en un determinado curso académico. Consta inicialmente del título del trabajo. El trabajo se define en base a las personas que lo realizan (los componentes) y las palabras clave que lo describen, la información necesaria para entenderlo se agrupa en la introducción. Y como es un trabajo práctico, tiene la información para ponerlo en marcha o ejecutarlo y la descripción de los archivos que componen el trabajo. Algunas de estas características del trabajo se deberán utilizar obligatoriamente, otras no. Y, algunas partes pueden repetirse porque, como los componentes o las palabras clave, pueden utilizarse en un número, en

2 Se puede leer sobre el estándar *Standard Generalized Markup Language (ISO 8879:1986)* en la Wikipedia <https://en.wikipedia.org/wiki/Standard_Generalized_Markup_Language).

principio, indeterminado. Que conste que es una decisión arbitraria, no lo tomes como una limitación.

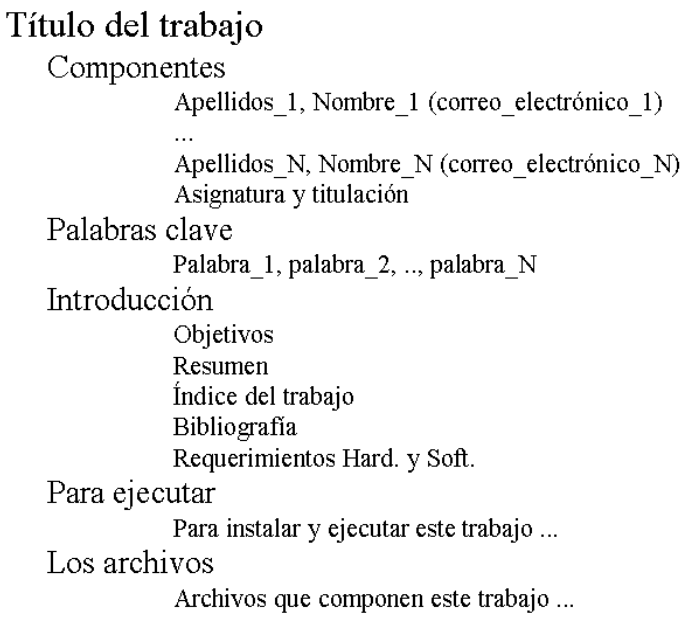

Figura 2: Esbozo de la estructura del documento que recogerá la propuesta de trabajo de la asignatura.

Siguiendo la estructura indicada en la Figura 2, el Listado 1 muestra un ejemplo de documento XML que llamaremos *trabajoV1.xml*. Los documentos XML se sugiere que empiecen con una línea que describe la versión de XML y el tipo de codificación del documento. Es una instrucción especial y se la reconoce por su nombre "?xml". Aquí utilizaremos el conjunto de caracteres ISO-8859-1 porque es el que incluye los caracteres propios del alfabeto español; facilitando así, entre otras cosas, la utilización de caracteres acentuados.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Trabajo>
      <Titulo> Un ejemplo básico de documento escrito en XML </Titulo>
      <Grupo>
            <Componente> <Nombre>Clark Kent</Nombre>  </Componente>
            <Componente> <Nombre> Bruce Wayne</Nombre> </Componente>
            <Componente> <Nombre>Yo Mismo También</Nombre> </Componente>
      </Grupo>
      <Resumen>
      <Parrafo>El trabajo consiste en explicar cómo comprobar la sintáxis
de un documento XML.</Parrafo>
      <Parrafo>Para ello, se describirá la edición de documentos XML.
</Parrafo>
      <Parrafo>A continuación se comprobará si está bien formado y,
después, si es válido el documento.</Parrafo>
      </Resumen>
</Trabajo>
```

Listado 1: Contenido del documento trabajoV1.xml.

En este *trabajoV1.xml*, el elemento raíz está representado por la etiqueta *Trabajo*. El resto de elementos representan información relativa al título, a los componentes del grupo y al resumen El elemento *Titulo* contiene una cadena de

caracteres Los elementos *Grupo* y *Resumen* constan de otro nivel de elementos (*Componente* y *Parrafo*, respectivamente) que a su vez contienen cadenas de caracteres.

Para maximizar las opciones de uso en diferentes plataformas de las acciones que aquí se proponen, la forma de aplicarlas será utilizando **herramientas o aplicaciones** en línea de órdenes, por lo que necesitaremos un terminal. También necesitamos un editor y un analizador de XML. En [5] encontrará un amplio catálogo bastante actualizado. Para la elección del editor ... cualquier editor de texto sirve, pero existen aplicaciones especializadas para ayudarte a escribir un archivo y minimizar los errores propios de la edición manual. Por ejemplo, *emacs* o *kate* (entre otros) y, también, se puede utilizar la mayoría de editores de código fuente para desarrollar aplicaciones.

Para realizar el análisis sintáctico se puede utilizar *Xerces* [6]. Vamos a emplear la versión *Xerces-J* que se ejecuta sobre *Java* por hacer lo más portable posible todo lo que se comenta en este documento. Si no tuvieras Java instalado, en la versión Ubuntu 18.04 de Linux es posible instalar varias alternativas, como por ejemplo:

$ apt install openjdk-11-jre

Para instalar *Xerces-J*, se ha de descargar el binario de nombre *Xerces-J-bin.1.4.4*[3] y descomprimirlo, p. ej. como muestra el Listado 2, cuya última instrucción sirve para comprobar la instalación: si se obtiene la ayuda que se muestra, todo está bien instalado. Como *Xerces* tiene muchas opciones y un largo formato para ejecutarlo, hemos preparado un *script*[4] (véase en el Listado 3 el contenido de *xerces.sh*) que permite invocarlo sin escribir demasiado.

```
$ mkdir libs
$ wget http://archive.apache.org/dist/xml/xerces-j/Xerces-J-bin.1.4.4.tar.gz
$ tar xvf  Xerces-J-bin.1.4.4.tar.gz -C libs
$ java -cp libs/xerces-1_4_4/xercesSamples.jar  dom.DOMWriter
usage: java dom.DOMWriter (options) uri ...
  [[  He suprimido parte de la salida por brevedad de la exposición]]
```

Listado 2: Instalación y comprobación de Xerces-J.

```
JAVA=java
JARS_HOME=libs/xerces.jar:bin/xercesSamples.jar
 if [ $# -lt 1 ];
  then
   echo "Verificar/Validar un documento XML mediante Xerces:"
   $JAVA -cp ${JARS_HOME} dom.DOMWriter
   exit -1
 fi
$JAVA -cp ${JARS_HOME} dom.DOMWriter $*
```

Listado 3: Contenido del script xerces.sh.

3 La puedes encontrar en *The Apache Xerces™ Project* , dentro del *Download* de <https://xerces.apache.org/xerces-j/install.htm>.

4 Si vas a utiliza *macOS*, puedes seguir los mismos pasos que aquí muestro. Si vas a utilizar la plataforma MS/Windows, pero solo en caso necesario ;-), a través del correo electrónico, te puedo hacer llegar la versión de *xerces.bat* para esa plataforma. El resto también te sirve.

Así que utilizaremos un terminal para escribir la orden correspondiente a la tarea que necesitemos realizar. Para que realice su tarea sobre el contenido de un fichero de nombre *fitxer.xml*, se puede invocar como muestran los ejemplos de uso de la **Tabla 1**.

Tarea	Orden a ejecutar
Verificar si *fitxer.xml* está bien formado	*$ xerces.sh -V fitxer.xml* *$ xerces.sh fitxer.xml*
Verificar si *fitxer.xml* está bien formado y codificar la salida en ISO-Latin-1 (por defecto se utiliza UTF-8)	*$ xerces.sh -e ISO-8859-1 fitxer.xml*
Validar *fitxer.xml* contra la DTD del DOCTYPE	*$ xerces.sh -v fitxer.xml*

Tabla 1: Ejemplos de uso del script xerces.sh.

4 Comprobaciones sobre un documento XML

Dado un documento XML, se pueden realizar un par de comprobaciones sobre él. Primero, se puede analizar si el documento está "bien formado" (*well-formed*), es decir, cumple las especificaciones de sintaxis de XML. Para ello se utilizará un analizador o *parser*. Si lo es, entonces se puede comprobar la validez del documento, esto es, si cumple con las reglas de definición de una gramática. Dicho de otra forma, el documento se puede considerar que está construido acorde con un vocabulario o instanciación de una DTD.

4.1 Análisis sintáctico (*parsing*)

Se define un documento XML bien formado [1], como aquel que cumple estas reglas:

- Solo se permite un elemento raíz.
- Es obligado Incluir etiquetas de inicio y final para todos los elementos.
- En caso de trabajar con etiquetas "vacías" (que no tienen contenido), hay que incluir la "barra" (/) antes del signo "mayor" (>) que acota la etiqueta.
- Hay que anidar las etiquetas correctamente.
- Es obligatorio que los valores de los atributos vayan entre comillas.
- XML distingue entre mayúsculas y minúsculas.

¿Cómo se comprueba? Ejecutando cualquiera de las versiones de la orden en la primera fila de la **Tabla 1** y veremos que, en pantalla, se nos muestra el contenido del fichero. Eso quiere decir que no se han detectado errores y, por tanto, se considera que es un fichero XML bien formado.

Veamos ahora cuál es el resultado del análisis sintáctico para el ejemplo de documento XML, que llamaremos *trabajoV2.xml* y que se muestra en el Listado 4. Si lo miramos atentamente, veremos el error que se ha producido y cómo se puede resolver. Una pista: es un detalle muy pequeño. Es el “precio” de usar XML, pero a cambio, el procesado posterior del documento es más sencillo.

El Listado 5 muestra el resultado del análisis sintáctico. Efectivamente, la etiqueta que cierra el elemento *Parrafo* no está bien escrita: “</Parrafo” debe escribirse como “</Parrafo>”

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Trabajo>
       <Titulo/>
         <Introduccion>
              <Resumen>
                    <Parrafo>El trabajo consiste en ...</Parrafo
              </Resumen>
         </Introduccion>
       <Grupo> <Componente><Nombre>Yo Mismo</Nombre> </Componente> </Grupo>
</Trabajo>
```

Listado 4: Contenido del fichero trabajoV2.xml.

```
$ xerces.sh trabajoV2.xml
trabajoV2.xml:
[Fatal Error] trabajoV2.xml:7:17: The end-tag for element type "Parrafo"
 must end with a '>' delimiter.
```

Listado 5: Resultado de la verificación de trabajoV2.xml.

4.2 Validación

Puedo comprobar si el documento es válido según la DTD que, en su caso, tenga asignado. Recordemos que: para que sea válido, primero hay que comprobar que está bien formado.

4.2.1 DTD

La comprobación de la estructura de un documento XML se puede realizar en base a la DTD. Esta proporciona la gramática para una clase de documentos XML. Esta gramática contiene la definición del conjunto de etiquetas que puede contener una clase de documentos XML.

Para nuestro ejemplo es la que muestra el Listado 6 y que recoge buena parte de la idea de la Figura 2. Esta declaración sirve para que otros puedan conocerla y utilizarla en la creación y en las transformaciones que se pudieran escribir para los documentos de este tipo. Aunque, recordemos que las transformaciones no están en los objetivos de este artículo.

Observando el contenido de la DTD del Listado 6 se puede ver que ¡no es un fichero XML! No cumple las reglas de “bien formado”. Es un fichero de texto que especifica la gramática: esto es, qué etiquetas pueden aparecer en un documento XML, su orden y número. La primera de sus líneas define la etiqueta raíz (que en este caso es “Trabajo”) y sus descendientes. A su vez, estas aparecen especificadas después, hasta llegar a un nivel terminal que se especifica con “#PCDATA”. Esto significa que esa etiqueta no tiene otras que descienden de ella. Las etiquetas vacías, las que no tienen contenido, ni descendientes, se especifican con “EMPTY”.

También se especifican los atributos de las etiquetas, si los hay, especificando si son obligatorios (*#REQUIRED*) o no. Si sus posibles valores están establecidos aparecerán en forma de lista separada con una ‘|’ (como, p. ej., “inf” ó “Doc”) o como *CDATA*, si es una cadena de caracteres cuyo contenido es libre.

```
<!ELEMENT Trabajo (Titulo, Grupo, PalabrasClave?, Introduccion,
            Desarrollo?, Archivos?)>
<!ELEMENT Titulo (#PCDATA)>
<!ELEMENT Grupo (Componente+)>
<!ELEMENT Componente (Nombre, Datos?, foto?)>
<!ELEMENT Nombre (#PCDATA)>
<!ELEMENT Datos EMPTY>
<!ATTLIST Datos
        Asignatura CDATA #REQUIRED
        Titulacion (Inf | Doc) #IMPLIED
        Correo CDATA #IMPLIED >
<!ELEMENT foto EMPTY>
<!ATTLIST foto
          fichero  CDATA #REQUIRED
          ancho    CDATA #REQUIRED
          alto     CDATA #REQUIRED >
<!ELEMENT PalabrasClave (#PCDATA)>
<!ELEMENT Introduccion (Resumen, Requerimientos? )>
<!ELEMENT Resumen (Parrafo+)>
<!ELEMENT Requerimientos (Parrafo+)>
<!ELEMENT Desarrollo (#PCDATA)>
<!ELEMENT Archivos (#PCDATA)>
<!ELEMENT Parrafo (#PCDATA)>
```

Listado 6: DTD para la descripción del tipo de documento trabajo_de_asignatura (trabajo.dtd).

Además, la DTD utiliza unos símbolos para indicar:

- '?' → Que el elemento es opcional.
- '*' → Que el elemento que lo antecede puede aparecer un número de veces mayor o igual que cero.
- '+' → Que el elemento que lo antecede puede aparecer un número de veces estrictamente mayor que cero.

4.2.2 ¿Cómo se puede validar un documento XML?

La validación se realiza mediante *Xerces*, para ello se empleará una orden del estilo de la tercera fila de la Tabla 1. Al aplicarla al ejemplo *trabajoV1.xml* (véase el *Listado 3*), se obtiene la salida que muestra el Listado 7 donde se puede observar que da error en todas las etiquetas porque aún no hemos definido cuáles pueden aparecer. Lo hacemos en un momento.

Se puede observar que entre los mensajes aparece un "[[...]]", eso lo hemos puesto para abreviar la salida: ahí va el contenido literal del fichero XML, puesto que está bien formado.

El resultado de validar el fichero *trabajoV2.xml* (véase el Listado 4) es similar al anterior, solo que si todavía no se ha arreglado el error del contenido del Listado 4, la validación indicará todavía el "`[Fatal Error]`", como se muestra en el Listado 5.

```
$ xerces.sh -v trabajoV1.xml
trabajoV1.xml:
[Error] trabajoV1.xml:2:10: Element type "Trabajo" must be declared.
[Error] trabajoV1.xml:3:10: Element type "Titulo" must be declared.
[Error] trabajoV1.xml:4:9: Element type "Grupo" must be declared.
[Error] trabajoV1.xml:5:15: Element type "Componente" must be declared.
[Error] trabajoV1.xml:5:24: Element type "Nombre" must be declared.
[[ … ]]
[
<?xml version="1.0" encoding="UTF-8"?>
<Trabajo>
[[...]]
</Trabajo>
```

Listado 7: Resultado de la validación de trabajoV1.xml.

El Listado 8 muestra un fichero XML que contiene la instrucción especial *DOCTYPE*. Dicha declaración, aunque no es obligatoria para que el documento sea conforme a las reglas de XML, sí que define cómo encontrar la información de la DTD: en este caso, indicando que está en nuestro propio ordenador (SYSTEM) y que se llama *trabajo.dtd* (la que hemos visto en el Listado 6). Y para comprobar la **validez** del documento XML contra esta DTD, véase Listado 9.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE Trabajo SYSTEM "trabajo.dtd">
<Trabajo>
      <Titulo> Un ejemplo básico de documento escrito en XML </Titulo>
      <Grupo>
        <Componente> <Nombre>Clark Kent</Nombre>  </Componente>
        <Componente> <Nombre> Bruce Wayne</Nombre> </Componente>
        <Componente>  <Nombre>Yo Mismo También</Nombre> <Datos
Correo="correu@servidor.es"/> <Foto Fichero="foto.jpg" Ancho="4"
Alto="4"/>         </Componente>
      </Grupo>
      <Resumen> <Parrafo>El trabajo consiste en explicar cómo
comprobar la sintáxis  de un documento XML.</Parrafo> <Parrafo>Para
ello, se describirá la edición de documentos XML.</Parrafo> <Parrafo>A
continuación se comprobará si está bien formado y, después, si es válido
el documento.</Parrafo> </Resumen>
</Trabajo>
```

Listado 8: Contenido del fichero trabajoV3.xml.

Si comprobamos el documento del Listado 8 obtendremos la salida que muestra el Listado 9. ¿Qué errores ha encontrado *Xerces*? Los tres primeros tienen que ver con los atributos, como es el caso de “Asignatura” que es obligatorio y no está. Recuerde que hemos dicho que XML diferencia entre mayúsculas y minúsculas? Lo mismo le ha pasado al atributo “Fichero” y al elemento “Foto” y a sus atributos “Ancho” y “Alto”. Y si miramos la DTD con calma, veremos que “Resumen” no puede ser un descendiente directo de “Trabajo”, falta ahí un elemento “Introduccion”.

```
$ xerces.sh -v artxius/trabajoV3.xml
artxius/trabajoV3.xml:
[Error] trabajoV3.xml:10:40: Attribute "Asignatura" is required and
must be specified for element type "Datos".
[Error] trabajoV3.xml:11:49: Element type "Foto" must be declared.
[Error] trabajoV3.xml:11:49: Attribute "Fichero" must be declared for
element type "Foto".
[Error] trabajoV3.xml:11:49: Attribute "Ancho" must be declared for
element type "Foto".
[Error] trabajoV3.xml:11:49: Attribute "Alto" must be declared for
element type "Foto".
[Error] trabajoV3.xml:12:16: The content of element type "Componente"
 must match "(Nombre,Datos?,foto?)".
[Error] trabajoV3.xml:22:11: The content of element type "Trabajo" must
 match
"(Titulo,Grupo,PalabrasClave?,Introduccion,Desarrollo?,Archivos?)".
<?xml version="1.0" encoding="UTF-8"?>
<Trabajo> [[…]]
</Trabajo>
```

Listado 9: Resultado del proceso de validación del fichero trabajoV3.xml.

5 Conclusión

A lo largo de este artículo se ha visto cómo se puede analizar la estructura de un documento XML y algunos de los errores que pueden aparecer en este proceso de revisión que incluye dos etapas: la comprobación de "bien formado" y la de validez del documento.

Para esta segunda etapa debe especificarse una DTD, cuya sintaxis también ha sido revisada para entender los errores que pueden aparecer dentro del proceso. Los ficheros empleados han sido deliberadamente de pequeño tamaño, para que puedan ser comprobados visualmente. Cuando el tamaño del fichero crece o hay que hacer esto mismo para varios archivos, se está en una situación en la que resulta muy interesante tener las herramientas expuestas a mano. Por ahora lo dejamos aquí, animando al lector a resolver los mensajes del Listado 9. Y, si nos decidimos a explorar, podemos probar a modificar el contenido del fichero XML: p. ej., viendo de poner un elemento (al menos) de cada tipo declarado en la DTD. ¡Ánimo y a divertirse!

6 Bibliografía

[1] Extensible Markup Language (XML) 1.0 (Fifth Edition). W3C Recommendation 26 November 2008. Disponible en <https://www.w3.org/TR/REC-xml/>.

[2] J. Merelo, Transformando documentos XML usando XSLT <http://geneura.ugr.es/~jmerelo/XSLT/XSLT-2001-1ed.htm>.

[3] Generación de páginas Web usando XSLT y XML <http://geneurael.ugr.es/~jmerelo/XSLT/>. Gupo GeNeura.

[4] Café con Leche XML. <http://www.ibiblio.org/xml/>.

[5] Catálogo de herramientas XML. Disponible en <http://www.garshol.priv.no/download/xmltools/cat_ix.html >.

[6] Xerces2 Java Parser. Project Apache XML. Disponible en <http://xerces.apache.org/xerces2-j/index.html >..

[7] Miloslav Nic (traducción de Iván García), "DTD Tutorial", Disponible en <http://www.zvon.org/xxl/DTDTutorial/General_spa/book.html>.