El lenguaje XML (eXtensible Markup Language) y algunos conceptos generales*

Susanna Allés Torrent

2019

I. Estándares web y XML

XML (Extensible Markup Language) es uno de los lenguajes más utilizados en el mundo de la informática, pues es uno de los más simples, flexibles y adecuados para asegurar la interoperabilidad con una gran cantidad de aplicaciones, plataformas y lenguajes informáticos.

En nuestras vidas lo utilizamos casi diariamente, incluso sin saberlo: ¿sabías que a la base de un documento Word está XML? Pues sí, si descomprimen un documento de Microsoft Word o OpenOffice, por ejemplo, descubrirán que el documento está compuesto por una serie de documentos XML.

En esta lección veremos los principios fundamentales que rigen el XML para así poder comprender los mecanismos de la Text Encoding Initiative.

Estándares web

Conviene en primer lugar detenernos muy brevemente en el concepto de estándar web. Un estándar web es un conjunto de prácticas consensuadas que permiten el intercambio de información entre todos los usuarios del World Wide Web, o lo que es lo mismo, la web o internet.

Al igual que sucede en las lenguas modernas, donde existe un "estándar" y un conjunto de reglas gramaticales, la web tiene también diferentes lenguajes regidos por una serie de reglas y recomendaciones.

El uso de estándares permite que todo el mundo tenga acceso a la totalidad de la información en la red y propicia un desarrollo más rápido y más sólido. Por ejemplo, el uso de estándares facilita la visualización de los sitios en todos los navegadores y permite una indexación de los metadatos de una manera más eficiente.

*Estos materiales fueron creados en el marco del certificado de Experto Universitario en Humanidades Digitales, ofrecido desde 2014 por el Laboratorio de Innovación en Humanidades Digitales de la Universidad Nacional de Educación a Distancia.

Una lectura útil y aclaradora sobre los estándares web la podéis encontrar aquí: ¿Qué son los estándares web y por qué debería usarlos?

Hay muchos tipos de estándares web pero los más conocidos y que nos interesan ahora son los del W3C, World Wide Web Consortium, que se ocupan de potenciar y optimizar la Web, creada por Tim Berners-Lee en 1994.

El W3C desarrolla especificaciones de acceso abierto, en realidad estándares, que permiten la interoperabilidad de todos los productos relacionados con la Web. Las recomendaciones del W3C son desarrolladas por grupos de trabajo formados tanto por miembros del consorcio como por expertos invitados. Estos grupos de trabajo, una vez obtenido el acuerdo general, tanto de las compañías como de las otras organizaciones involucradas en la creación de aplicaciones web, crean un borrador con las recomendaciones y lo entregan a los miembros del W3C y a su director para que las aprueben de manera oficial.

Algunos de los estándares más conocidos son:

- HTML (HyperText Markup Language): lenguaje utilizado para la creación de hipertexto (web), nacido en 1989 y derivado de SGML. Su última versión es el HTML 4.01 (1999), convertido en norma ISO/IEC 15445:2000.
- HTML5: la última versión de HTML.
- XML (Extensible Markup Language).
- XHTML (Extensible HyperText Markup Language): la reformulación de HTML 4 según los principios de compatibilidad con XML (2000, revisión del 2002).
- CSS (Cascading Style Sheets): Hojas de estilo en cascada para dar formato y diseño a HTML.
- DOM (Document Object Model): lenguaje para la creación de interfaces de programación.

Os recomiendo esta breve historia del World Wide Web por Tim Berners-Lee (The World Wide Web: A very short personal history, 1998). Una de las citas de este artículo me parece muy interesante en relación con este tipo de tecnologías:

> The web of human-readable document is being merged with a web of machine-understandable data. The potential of the mixture of humans and machines working together and communicating through the web could be immense. (Tim Berners-Lee 1998)

Existen otros tipos de estándares, no necesariamente relacionados o creados para la Web. En algún momento habrán oído hablar de los estándares ISO, aquellos creados por la Organización Internacional para la Estandarización (en inglés, International Ogranization for Standardization).

Esta organización se encarga de publicar estándares bajo forma de documentos oficiales que estipulan los requisitos, las especificaciones, guías directrices o características que pueden ser usadas de manera consistente para asegurar qué materiales, productos, procesos o servicios se adecuan a sus propósitos. XML no es una norma ISO pero sí puede utilizar algunos estándares aprobados por ISO, como es por ejemplo la codificación de los caracteres.

El estándar XML

XML es un estándar del W3C que responde a las siglas Extensible Markup Language. En origen, es un heredero, más sencillo y eficaz, de su antecesor SGML (Standard Generalized Markup Language), un estándar ISO 8879:1986 creado por IBM en los años setenta. Las primeras recomendaciones del W3C para el uso de XML se publicaron en febrero de 1998. Actualmente la última versión es XML 1.0 (5ª edición 2008) y toda su documentación puede encontrarse en la página web del W3C. Existe la versión 1.1., publicada en febrero de 2004 (2ª edición 2006) que integra algunas evoluciones relativas al Unicode, pero en general es poco utilizada. Nosotros nos centraremos en la versión 1.0.

La misión principal de XML consiste en la estructuración de cualquier tipo de datos, como puede ser el marcado electrónico de textos, de cualquier tipología, para que sean procesados por una máquina. El lenguaje utilizado para la marcación de datos es simple y legible por los humanos (lo que en inglés llaman "human-readable"). En el lado opuesto, un lenguaje que no puede ser leído por los humanos puede ser, por ejemplo, un código de barras o los mismos datos binarios, que pueden ser leídos por un ordenador, pero no por una persona.

Como señalamos en la unidad anterior, XML es un lenguaje de marcado descriptivo o semántico consistente en aislar y describir fragmentos de texto a través de etiquetas, sin especificar el tipo de formato o de presentación, incluso el orden, que tendrán en su aspecto final. Son otros lenguajes de marcado los que se ocupan de la presentación, como el HTML o las CSS. XML es, pues, un lenguaje de marcado, no es ni un lenguaje de programación, ni de presentación, ni mucho menos una base de datos.

XML constituye un método sencillo para representar datos de una manera estructurada digitalmente, como una línea de caracteres. Su funcionamiento se basa en el concepto de "marcado" que radica en aislar y marcar partes de esa cadena de datos (sean palabras, números, párrafos, etc.) con "etiquetas" o marcas con nombre propio para indicar una función estructural o semántica.

La misión principal, pues, es la de marcar los diferentes tipos de datos textuales que aparecen en un documento. El código o marca XML se incluye en el mismo documento XML como una cadena de texto y es delimitado por marcas de texto que describen su contenido. De esta manera, un fichero XML consiste en una serie de datos estructurados y representados como una secuencia de texto, por ello puede leerse, tanto los datos textuales como su código, con cualquier herramienta que pueda leer ficheros de texto.

La unidad básica del marcado en XML es la "marca" o "etiqueta", ("tag", en inglés), denominada técnicamente "elemento". XML tiene una serie de reglas que definen una sintaxis concreta, como por ejemplo, cómo deben delimitarse dichas etiquetas, cómo deben formarse, qué tipo de nombres son aceptables, dónde pueden situarse, etc. Si un documento cumple todos los requisitos de su gramática, estará bien formado.

Además, por norma general, un documento XML, como veremos, depende de un modelo abstracto que especifica la lista de las etiquetas que pueden utilizarse, sus nombres, dónde pueden situarse o cómo deben anidarse. Este modelo recibe el nombre de "esquema"; si el documento XML cumple los requisitos del esquema, será un documento válido.

XML juega hoy en día un papel central en el intercambio de una gran variedad de datos en la Web y otras aplicaciones y es el más usado para estructurar informaciones que pretendan durar en el tiempo y ser interoperables con otras aplicaciones o plataformas. Es compatible con la mayoría de lenguajes web (tipos de documento, de metadatos, de modelización, de presentación, de programación, de protocolos, etc.), característica que lo convierte en una de las mejores opciones para solucionar problemas de interoperabilidad y uno de los más idóneos para la preservación de datos en un formato de larga duración. Es utilizado a nivel mundial por una gran comunidad de usuarios y organizaciones que ofrecen soporte técnico y dispone de una gran variedad de herramientas con las que trabajar.

XML, en fin, soporta diferentes sistemas de escritura, gracias al uso del estándar Unicode por lo que puede contener textos en cualquier idioma, sea español, chino, hebreo o árabe.

En apariencia XML se parece mucho a HTML, pero es crucial entender algunas diferencias:

- XML es extensible, lo que significa que este lenguaje puede ser extendido y adaptado para responder a necesidades diferentes. XML es un metalenguaje de marcado sin una lista fija de etiquetas, pues estas se crean en función de las necesidades del proyecto. De esta manera, se permite a los usuarios o a una comunidad establecer una serie propia de etiquetas, como es el caso de TEI.
- XML tiene que estar bien formado ("well-formed"). Para que ello se cumpla, debe cumplir con una serie de normas bastante estrictas; esto permite a los procesadores XML parsear y poder validar cualquier documento. De hecho, si un documento no está bien formado, será rechazado por cualquier procesador y por tanto inservible.
- XML, como hemos dicho, se centra en la descripción semántica de los fragmentos de texto, mientras que HTML se ocupa solo de la presentación. Por ejemplo, imaginemos que tenemos un texto donde aparecen palabras en cursiva, unas indican palabras extranjeras y las otras indican títulos de obras. Si quisiéramos convertirlo en una página web y usar HTML, marcaríamos seguramente las palabras en cuestión con la etiqueta de cursiva, es decir, de la siguiente manera: `<i>palabra</i>`; en cambio, XML marcaría las palabra con etiquetas que indicaran "palabra extranjera" y "título": `<palabraExtranjera>palabra</palabraExtranjera>` o `<titulo>El título</titulo>`. XML se preocupa del qué son las cosas, y no del cómo lucen; separa, en definitiva, el contenido de la presentación.
- XML puede ser validado a través de un esquema específico, un aspecto que veremos más abajo. Los documentos XML, además de las reglas intrínsecas del mismo lenguaje, pueden depender de un esquema que determine qué elementos se pueden utilizar y de qué manera se pueden anidar, qué tipos de atributos pueden llevar, etc. En el caso de que el documento XML cumpla los requisitos del esquema, será "válido".

II. Reglas y conceptos generales

Veamos enseguida qué aspecto tiene un fichero XML e intentemos comprender la noción de marcado, así como algunas reglas generales:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<lista>
    <libro>
        <autor id="esquilo">Esquilo</autor>
        <titulo>Prometeo encadenado</titulo>
    </libro>
    <libro>
        <autor id="sofocles">Sófocles</autor>
        <titulo>Edipo Rey</titulo>
    </libro>
    <libro>
        <autor id="euripides">Eurípides</autor>
        <titulo>Medea</titulo>
    </libro>
</lista>
```

Ejemplo 1: un documento xml (GitHub).

Este documento es en realidad simple texto, podría crearse, editarse y guardarse en un fichero de texto y con uno de los múltiples editores de texto existentes, tales como Atom, Komodo, BBEdit, jEdit, Emacs, entre muchos otros. Entre la comunidad TEI, el editor más utilizado es oXygen porque, aunque es de pago, ofrece funcionalidades específicas para el marcado TEI.

A partir de este simple ejemplo, podemos constatar ya algunas ideas y reglas inherentes a todos los documentos XML:

- El marcado ("markup") consiste en aislar una porción de texto, grande o pequeña, con un significado semántico específico y señalarlo a través de una marca electrónica, que comúnmente llamamos etiqueta ("tag"). Dichas etiquetas son en realidad una secuencia lineal de caracteres con una estructura específica. Por ejemplo:

```xml
<autor>Esquilo</autor> <titulo>Prometeo encadenado</titulo>
```

- Como vemos, la marca indica una instrucción especial de procesamiento (`<...>`) que el ordenador interpreta como código informático, dónde empieza y dónde acaba. Estas marcas o etiquetas deben aparecer al inicio del segmento que queramos codificar a través de dos paréntesis angulares, y en el cierre del mismo, también con dos paréntesis angulares con la barra inclinada (`/`) al inicio. Es importante recordar que los caracteres `< >` son siempre interpretados como código por nuestro ordenador, de manera que si se quieren representar en el texto como tales deberemos utilizar caracteres diferentes (`&lt;` para representar `<` y `&gt;` para `>`).
- La estructura de cada una de estas marcas constituye lo que llamamos un "elemento". Su estructura es rígida y siempre tiene la misma estructura: el nombre del elemento, también llamado identificador, y, normalmente, uno o varios atributos con sus valores respectivos. La estructura de un elemento sería esta:

```xml
<elemento atributo="valor">contenido</elemento>
```

Y un ejemplo concreto podría ser este:

```xml
<autor id="esquilo">Esquilo</autor>
```

- Un documento XML tiene forma de árbol. La estructura consiste en la anidación sin límites de unos elementos al interior de otros, construyendo así una especie de árbol o estructura arbórea. Por eso, necesitamos siempre un solo elemento raíz del que cuelguen todos los otros. En el ejemplo, el elemento `<catalogo>`, engloba una serie de libros (`<libro>`), que a su vez, tienen dos elementos descendientes: `<autor>` y `<titulo>`.
- Los elementos y los atributos son sensibles a las mayúsculas y minúsculas, de manera que si se equivocan en una sola letra, poniéndola en mayúscula en lugar de minúscula o viceversa, se encontrarán con un error.

III. Estructura de un documento

Pasemos ahora a ver con más detalle cada una de las partes de un documento XML. Les aconsejo que para esta sección abran el documento Ejemplo 2: un documento XML asociado a una DTD, que es un pequeño fichero XML comentado. Lo pueden copiar, abrir con oXygen u cualquier programa de edición XML, y anotar libremente con sus comentarios.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!-- ¡esto es un comentario! -->
<!-- En primer lugar aparece la "Declaración XML" -->

<!DOCTYPE curso SYSTEM "ejemplo-2-DTD.dtd">
<!-- "Declaración de Tipo de Documento": DOCTYPE nos está indicando el tipo de
↪  documento con el que debe relacionarse este fichero XML, es decir, donde
↪  encontramos el modelo que debemos utilizar para marcar este texto. "curso"
↪  indica el elemento raíz, mientras que "SYSTEM" nos indica, en este caso, que
↪  el modelo se halla en nuestro sistema y que el modelo en cuestión se llama
↪  "ejemplo_DTD.dtd" que se encuentra en la misma carpeta. -->

<?xml-stylesheet href= "formato.css" type= "text/css"?>
<!-- Esto es una "Instrucción de procesamiento" que indica al procesador que este
↪  documento XML va asociado a una hoja de estilo CSS que se llama formato.css y
↪  que se encuentra en la misma carpeta.-->

<curso> <!-- "Elemento raíz" -->
    <metadatos>
        <titulo>Introducción a la edición digital de textos: TEI-XML</titulo>
        <fecha cuando="2018-01-22">22 enero 2018</fecha>
        <profesor>
            <nombre>Susanna</nombre>
            <apellido>Allés</apellido>
```

```
            <apellido>Torrent</apellido>
        </profesor>
        <universidad>UNED</universidad>
    </metadatos>
    <contenido>
        <tema n="1">
            <titulo>TEMA 1: ¿Qué es la Text Encoding Initiative y qué
            aplicaciones prácticas tiene?</titulo>
            <parrafo>En esta primera unidad ofreceremos...</parrafo>
            <parrafo>En el ámbito de las humanidades y las ciencias
            sociales...</parrafo>
        </tema>
        <tema n="2">
            <titulo>TEMA 2: Principios fundamentales de un
            documento XML </titulo>
            <parrafo>En este segundo tema se presentarán los
            principios fundamentales del lenguaje XML,</parrafo>
            <parrafo>...</parrafo>
        </tema>
    </contenido>
</curso>
```

Ejemplo 2: un documento XML asociado a una DTD (GitHub)

Declaración XML

Los documentos XML empiezan por una declaración XML:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
```

Este elemento no puede ir precedido por ningún comentario, ni espacio en blanco ni ningún otro elemento. Su función es simple pero crucial, y cubre al menos tres objetivos básicos:

- Declarar que se trata de un documento XML; el paréntesis angular seguido del interrogante de cierre es interpretado como código por nuestra máquina y le está indicando que se trata de una declaración o una instrucción de procesamiento, no de un elemento corriente.
- Declarar qué versión del estándar XML se sigue; No es lo mismo utilizar la versión 1.0 que la 1.1. Además, en el hipotético caso que en el futuro se publique una nueva versión (cosa que suele ocurrir con los lenguajes informáticos) es importante que dejemos para el futuro la versión específica utilizada para evitar problemas.
- Declarar qué caracteres de codificación utiliza el documento. Los documentos XML contienen texto cifrado en Unicode. Unicode es una combinación de caracteres que incluye prácticamente todas las lenguas del mundo y puede expresarse a través de diversos sistemas de codificación. Cada uno de estos sistemas establece un juego de caracteres ("character set") específico que representa los caracteres (alfanuméricos y otros símbolos) a través de una secuencia de números (por ejemplo, imaginemos que

el número 65 corresponde a la letra A). Así, cada combinación numérica corresponde a una letra o caracter, que difiere en las diferentes codificaciones. Cada una de estas combinaciones de puntos se llama "puntos de código" ("code points"). La codificación de los caracteres determina cómo estos puntos de código son representados en bytes. De esta manera, el conjunto de caracteres de Unicode puede tener diferentes codificaciones como UTF-8, UCS-2, UCS-4 o UTF-16. Por eso es importante señalar al inicio del documento qué codificación de caracteres hemos utilizado para que el parser sepa cómo descifrarlos y para que nuestra secuencia de texto se visualice de manera correcta. Aunque XML utiliza por defecto UTF-8, se podría también utilizar otro sistema según nuestras necesidades, como sería la combinación de caracteres ISO, por ejemplo, el ISO-8859-1 (Latin 1) que corresponde a las lenguas europeas occidentales, incluido el español.

- En algunos casos, en la declaración de documento también nos podemos encontrar el atributo "standalone":

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
```

- Los valores del atributo "standalone" pueden ser afirmativo "yes" o negativo "no"; en el primer caso significa que no depende de ningún esquema externo y que por tanto el documento se regirá solo según las reglas del lenguaje XML; en cambio, si no es "standalone" significa que depende de un modelo abstracto concreto, es decir, concretamente de un esquema o DTD externos.

Declaración de tipo de documento

En algunos casos, un documento XML puede depender de un modelo abstracto que puede tener diferentes formas. Un caso frecuente es el modelo de la DTD (Document Type Definition), ampliamente utilizado en XML, aunque en desuso en el campo de TEI (donde se prefieren los esquemas RelaxNG). De esta manera, podemos encontrarnos en el prólogo del documento con una declaración del tipo de documento, "Documento Type" que tiene esta forma:

```
<!DOCTYPE curso SYSTEM "ejemplo-2-DTD.dtd">
```

Esta declaración está compuesta del identificador DOCTYPE, seguido del nombre del elemento raíz del documento que estamos marcando, a su vez seguido de SYSTEM con la indicación de la URL o ubicación del fichero DTD en cuestión (en este ejemplo, la DTD estaría en el mismo fichero que el documento XML). A veces, en lugar de SYSTEM, podemos encontrar PUBLIC, seguido de una URL; esto significa que la DTD es accesible de forma pública en la Web (normalmente se trata de algún estándar o de algún modelo público creado por una comunidad específica). Es el mismo procedimiento que podemos encontrar al inicio de un documento HTML:

```
<!DOCTYPE html>
```

Instrucciones de procesamiento

Aunque ahora no nos detengamos en este aspecto, no está de más señalar que cuando nos encontramos con este tipo de sintaxis: al inicio `<?` y al final `?>`, en realidad se trata de Instrucciones de procesamiento (en inglés, "Processing Instructions"). En estos casos, XML está señalando que con ese fragmento de código el documento está ofreciendo un tipo determinado de información que debe ser procesada por otra aplicación. Por ejemplo, podríamos encontrar un script en PHP que iría destinada a ser procesada por una aplicación PHP; otro caso muy habitual es el de la asociación de una hoja de estilo CSS al mismo documento XML:

```
<?xml-stylesheet href= "formato.css" type= "text/css"?>
```

De momento es solo necesario saber diferenciar entre una instrucción de procesamiento y los elementos XML propiamente dichos.

Declaración del espacio de nombre

En los documentos TEI aparece normalmente una declaración del "espacio de nombre" (en inglés, "namespace"). Por ejemplo, en un documento TEI aparece de la siguiente manera:

```
<TEI xmlns="http://www.tei-c.org/ns/1.0">
```

Los documentos XML pueden incluir elementos declarados en diferentes espacios de nombre. Por eso, la función de una tal declaración es la de evitar la ambigüedad de los nombres de los elementos y los atributos que pertenecen a una aplicación concreta de XML. Por ejemplo, imaginemos que en un mismo documento se utilizan dos sistemas de marcado que poseen elementos con el mismo nombre (por ejemplo `<p>`). Señalando el "espacio de nombre" dentro del cual cada uno de estos elementos debe ser interpretado se evitan errores y ambigüedades.

Desde el punto de vista formal, el espacio de nombre se representa a través de su URI, que funciona, en realidad, como un identificador externo y al que se asocia un prefijo de espacio de nombre. Es importante que se declaren todos los espacios de nombre utilizados en un documento, bajo forma de atributos `@xmlns` en el elemento raíz del documento.

Retomando el ejemplo anterior, vemos como se ha añadido el espacio de nombre de TEI en el elemento raíz, y por consiguiente sabemos que todos los elementos en su interior responden a ese espacio de nombre por defecto:

```
<TEI xmlns="http://www.tei-c.org/ns/1.0">
  <teiHeader>
   ....
  </teiHeader>
  <body>
    <text>
```

```
    ...
  </text>
 </body>
```

De no haberse declarado de esta manera y si quisiéramos añadir otro espacio de nombre, tendríamos que especificar el prefijo de cada uno de los espacios de nombre de los elementos con su prefijo o espacio de nombre correspondiente. Imaginemos el siguiente caso en que utilizamos etiquetas relativas a TEI y a Math, pues necesitamos marcar fórmulas matemáticas:

```
<tei:TEI xmlns:tei="http://www.tei-c.org/ns/1.0"
xmlns:math="http://www.mathml.org">
 <tei:teiHeader>
    ...
  </tei:teiHeader>
  <tei:text>
    <tei:body>
      <tei:p>Aquí el texto del párrafo...
         <math:expr>...Aquí la fórmula matemática...</math:expr>
     </tei:p>
    </tei:body>
  </tei:text>
</tei:TEI>
```

Aunque el caso anterior sería correcto, lo más habitual es declarar un nombre de espacio por defecto y establecer los otros espacios de nombre con su prefijo correspondiente, de manera que el marcado no sea tan pesado:

```
<TEI xmlns="http://www.tei-c.org/ns/1.0" xmlns:math="http://www.mathml.org">
  <teiHeader>
      ...
  </teiHeader>
  <text>
    <body>
      <p>Aquí el texto del párrafo...
            <math:expr>...Aquí la fórmula matemática...</math:expr>
      </p>
   </body>
  </text>
</TEI>
```

Los dos objetivos principales del espacio de nombre son:

- Diferenciar entre elementos y atributos de diferentes vocabularios con significados diversos que puedan concurrir en el mismo documento con un nombre idéntico.
- Agrupar conjuntamente todos los elementos y atributos relacionados de una aplicación concreta de XML de manera que el software pueda reconocerlos de manera más fácil.

El elemento raíz

Todo documento XML tiene un solo elemento raíz (en inglés, "root"), el único que no depende de ningún otro elemento y que contiene todos los otros. En Ejemplo 2: un documento XML asociado a una DTD, el esquema sería el siguiente:

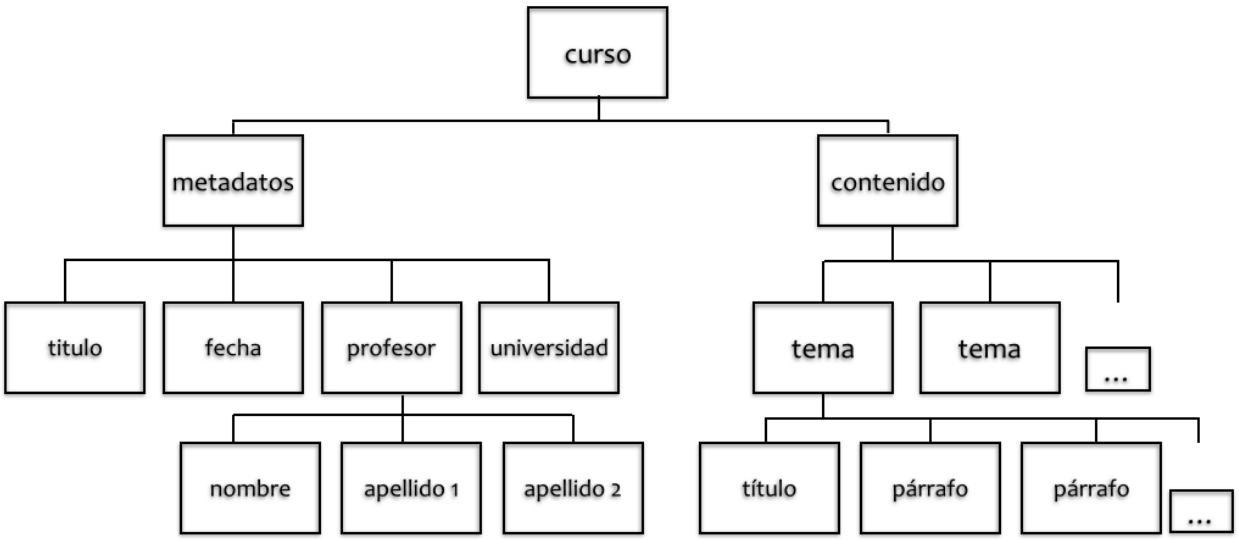

Figure 1: Ejemplo de esquema en árbol

Como vemos, la estructura arbórea consta de un solo elemento raíz ("curso"), del que descienden todos los otros elementos. Estos elementos pueden constituirse en lo que llamamos "nodos" (en inglés "nodes") que pueden ser: a) un subárbol: por ejemplo, metadatos con sus cuatro elementos descendientes titulo, fecha, profesor, universidad; b) un simple elemento, por ejemplo, el nodo titulo; c) la misma secuencia de caracteres que encontramos como contenido al interior de los elementos. Es importante conocer esta nomenclatura y utilizar este vocabulario para no dar lugar a equívocos.

Elementos y contenido

Los elementos son las unidades de información semántica compuestas de una marca de apertura < y de cierre >. El nombre del elemento constituye su identificador general y todo lo

que queda entre estas dos marcas se llama "contenido". Puede contener texto, otros elementos o estar vacío (`<elemento_vacio/>`).

En lo que concierne a los identificadores, es importante recordar que:

- El nombre del elemento puede solo contener letras y números (A-Z, a-z, 0-9), sin acentos ni espacios.
- Solo algunos signos pueden utilizarse: - (guión), _ (guión bajo), . (punto).
- El nombre no puede empezar por un número, ni por un signo, ni por xml.
- XML es sensible a las mayúsculas y las minúsculas.

Atributos y valores

Los elementos pueden poseer atributos que añadan una característica adicional al propio elemento; su orden y número no es relevante. El nombre del atributo va siempre al interior de la etiqueta de apertura y va precedido de un espacio, seguido del símbolo =, y de comillas, dentro de las cuales aparece el "valor". He aquí un ejemplo:

```
1 <fecha cuando="2017-11-09">09 noviembre 2017</fecha>
```

Los nombres de los atributos deben formularse igual que los nombres de los elementos. Su orden al interior del elemento es libre, siempre y cuando no se repita.

Comentarios y entidades

Los documentos XML pueden contener comentarios para que los autores puedan dejar sus notas, normalmente relativas al proceso de codificación del mismo documento. Pueden aparecer en cualquier lugar del documento mientras sea contenido; no podrán nunca situarse al interior de una marca o etiqueta. Por regla general, no están destinados a ser procesados por la máquina, y de hecho, son ignorados por defecto. Los comentarios están delimitados por `<!--` y `-->`. Por ejemplo:

```
1 <!-- ¡esto es un comentario! -->
```

Por otro lado, las entidades de referencia ("entity references" en inglés) pueden ser de diferentes tipos de mayor o menor complejidad. En esta ocasión citaremos solo las entidades internas de caracteres, pues las encontraremos con una cierta frecuencia. Se trata de caracteres especiales que no deben ser interpretados como código. Por ello, determinados signos deben ser codificados de una manera especial. Sucede así en los siguientes casos:

- `&lt;` para indicar el símbolo "menor que" o el paréntesis angular de la marca de apertura <
- `&gt;` para indicar el símbolo "mayor que" o el paréntesis angular de cierre de la marca >
- `&amp;` para indicar &
- `&quot;` para indicar las comillas dobles "
- `&apos;` para indicar el apóstrofe o la comilla simple '

IV. Esquemas

Los esquemas constituyen, en general, un fichero diferente, de esta manera podemos referirnos a él desde múltiples documentos XML. Su función principal es la de establecer y definir los requisitos que debe respetar el documento XML, donde la referencia al esquema aparece en el prólogo del documento XML (después de la declaración XML y antes del elemento raíz), tal y como vemos en el Ejemplo 2: un documento XML asociado a una DTD:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE curso SYSTEM "ejemplo-2-DTD.dtd">
<curso> .... </curso>
```

Un esquema establece la serie de elementos, sus nombres y los atributos que estos podrán contener; además, establece de qué manera se podrán anidar los unos con los otros. El parser que debe validar el documento compara el marcado XML con el esquema y recoge aquellos puntos en que el documento difiere de sus reglas. Algunos parsers si encuentran defectos, rechazarán el documento, otros no, depende de las reglas que establezca el mismo. Todas las reglas, a parte obviamente de aquellas que pertenecen a la buena formación de cualquier documento XML, deben declararse en el esquema, de no ser así el documento será "inválido".

Hay diferentes tipos de esquema:

- DTD (Document Type Definition): es el modelo más utilizado en el mundo de la informática pero no en cambio en el campo de la TEI, pues es menos flexible que otros modelos. En el fichero Ejemplo 2: una DTD encontraréis una muestra de una sencilla DTD comentada y que acompaña el fichero XML Ejemplo 2: un documento XML asociado a una DTD.
- Esquemas W3C: se trata de esquemas expresados en XML, publicados por el W3C en 2001 (2ª ed. 2004):
- Esquemas RelaxNG: este tipo de esquemas también se expresan en XML y son los más utilizados en el campo de la TEI. En 2003 se constituyó como una norma ISO/IEC 19757-2, actualizada en 2008.
- Existen otros esquemas también utilizados como Schematron, Examplotron.

En cualquier caso, una regla de oro es que el esquema esté bien documentado y comentado. Además, podríamos también acompañarlo de otra documentación escrita para complementarlo, como una especie de manual de marcado (no concebido como sustitución de la documentación formal del esquema, sino como un complemento).

V. Conceptos de "Válido" y "Bien formado"

Un documento XML debe estar bien formado y ser válido para poder ser procesado y visualizado.

La expresión "bien formado" significa que el documento XML debe estar expresado correctamente según la gramática de XML. Entre las reglas que debemos seguir escrupulosamente señalaremos las siguientes:

- Debe haber un único nodo raíz que contenga la totalidad del documento XML.
- Todos los elementos deben estar bien anidados ("nested") al interior de un nodo raíz y nunca deben solaparse los unos con los otros. Así, por ejemplo, esto sería correcto:

```
<metadatos>
<titulo>Introducción... </titulo>
<fecha>30 enero 2017</fecha>
</metadatos>
```

Pero no esto:

```
<metadatos>
<titulo>Introducción...
<fecha>30 enero 2017</titulo></fecha>
</metadatos>
```

- Los nombres de los elementos y atributos son siempre sensibles a las mayúsculas.
- Todas las etiquetas de apertura deben tener una etiqueta de cierre; excepto si se trata de elementos vacíos, en cuyo caso se expresa así `<nombre_elemento/>`.
- Los valores de los atributos van siempre entre comillas.
- Un elemento no puede tener dos atributos con el mismo nombre.
- Los comentarios y las instrucciones de procesamiento nunca pueden aparecer en el interior de una etiqueta.
- No pueden aparecer los caracteres < o & como ni en los nombres de los elementos ni en los atributos.
- Una recomendación: nunca utilicen acentos o espacios en los nombres de los elementos, ni en los atributos como tampoco en los valores. Su uso puede acarrear muchos problemas a la hora de procesar los documentos.

Una manera muy sencilla de comprobar si un documento XML está bien formado es abrirlo con un navegador web que sepa interpretar documentos XML, como por ejemplo Mozilla. Si el documento aparece en el navegador, significa que está bien formado, en caso contrario aparecerá un error.

Si utilizamos el programa oXygen que posee ya un parser que permite la comprobación de su correcta formación. La opción "Check Well-Formedness" (Menú superior, Document > Validate > Check Well–Formedness) nos permite llevar a cabo esta operación sin ninguna complicación; en caso de que se cumplan los requisitos nos aparecerá en la parte inferior el mensaje "Document is well formed". En caso contrario aparecerá: "Wellformed test – failed. Errors: 1" o el número de errores que localice en el documento. Además nos aparecerá el mensaje del tipo de parser utilizado, en el caso de oXygen, Xerces, con los detalles del error.

Por otro lado, el documento debe ser también "válido". Para ello, necesita un modelo que fije la estructura básica y las posibilidades de anidación del documento. Las DTD o esquemas (RelaxNG, W3CSchema) establecen la sintaxis que debe tener el documento XML. Si este está formado según sus reglas, será válido. A diferencia de cuando está bien formado, los navegadores web no validan los documentos XML, solo comprueban que esté bien formado.

Volviendo a nuestro ejemplo (Ejemplo 2) y al programa oXygen, podemos comprobar que nuestro fichero sea válido. En este caso, el fichero depende de una DTD, llamada Ejemplo 2: una DTD. Vamos al Menú superior > Document > Validate > Validate. El mensaje que nos aparecerá en la parte inferior del programa será "Document is valid".

Mientras que un documento XML debe estar siempre bien formado, el hecho de ser válido no es obligatorio, pero altamente recomendable pues establece qué tipo de marcado se ha utilizado. En el caso de TEI, siempre se usa un esquema a partir del cual validamos nuestro documento.

VI. Recapitulación

XML es uno de los lenguajes más sólidos y extendidos en toda la historia de la informática. A partir de XML se han creado otras tecnologías aliadas o de soporte que son utilizadas en muchas aplicaciones basadas en XML, como son: XLink, XSLT, XPointer, XPath, Namespaces, Schemas, DOM, XHTML 1.0, entre otros.

XML al ser extensible, simple y flexible es ampliamente utilizado por muchas comunidades cuyo objetivo es establecer una lista cerrada de etiquetas o marcas que, en realidad, corresponden a la fenomenología con la que trabajan. Así, por ejemplo, XML es utilizado por los siguientes modelos públicos:

- DocBook: para la creación de documentación técnica.
- SVG (Scalable Vector Graphics): para representar en forma de texto imágenes vectoriales.
- MathML: para expresar fórmulas matemáticas
- MEI (Music Encoding Initiative): para representar las partituras musicales.
- KML (Keyhole Markup Language): para localización geoespacial.

Y sobre todo para crear modelos de metadatos estructurados en ámbitos diferentes:

- Dublin Core Metadata Initiative: uno de los lenguajes metadatos más utilizados en campos muy diversos.
- EAD (Encoding Archival Description): para estructurar los documentos archivísticos.
- EAC-CPF (Encoded Archival Context – Corporate Bodies, Persons, and Families): utilizado para estructurar metadatos relativos a noticias de autoridad en archivos.
- MODS (Metadata Object Description Schema): para las referencias bibliográficas
- MARCXML: para el uso de MARC en XML en bibliotecas.
- MADS (Metadata Authorithy Description Schema): para la estructuración de los metadatos en noticias de autoridad.
- METS (Metadata Encoding and Transmission Standard): para la gestión de objetos digitales en bibliotecas.
- RDF (Resource Description Framework): un modelo de metadatos derivado de la web semántica.

VII. Bibliografía

Berners-Lee, Tim (1998). "The World Wide Web: A very short personal history". https://www.w3.org/People/Berners-Lee/ShortHistory.html

Birnbaum, David. «What is XML and why should humanists care? An even gentler introduction to XML", http://dh.obdurodon.org/, 2015

Muller, Martin. "A very gentle introduction to the TEI markup language." http://www.tei-c.org/Vault/Tutorials/mueller-index.htm

Rusty Harold, E. – Scott Means, S. «Part I. XML Concepts». En *XML in a Nutshell*. Second Edition, O'Reilly, 2002. https://telecommnet.com/files/cases/IPR2015-00716/Harold-XML-in-a-Nutshell.pdf.

TEI Consortium, *A Gentle Introduction to XML*. http://www.tei-c.org/release/doc/tei-p5-doc/en/html/SG.html.

Tutoriales:

Existe una enorme cantidad de tutoriales en línea, entre los cuales destacamos:

- w3schools.com https://www.w3schools.com/xml/ (Ofrece una gran variedad de tutoriales sencillos que realmente clarifican muchas dudas)
- L. Burnard, What is textual markup and why is it useful? What is XML?, digital.humanities@oxford, 2008, http://digital.humanities.ox.ac.uk/dhoxss/2008.
- Cours de XML http://www.gchagnon.fr/cours/xml/ (en francés)
- Tutorial de XML http://zvon.org/xxl/XMLTutorial/General_spa/book.html (en español)
- Manual de XML https://desarrolloweb.com/manuales/manual-introduccion-xml.html (en español)

Cita propuesta:

Allés Torrent, Susanna (2019). "El lenguaje XML (eXtensible Markup Language) y algunos conceptos generales". *TTHUB. Text Technologies Hub: Recursos sobre tecnologías del texto y edición digital*. https://tthub.io/ DOI: