# Unidad 4. Introducción a *XML.*



## 1 Características de XML

XML es un estándar internacional desarrollado por el Grupo de Trabajo de XML (conocido como el Comité de Revisión Editorial de SGML) formado bajo el auspicio del World Wide Web Consortium (W3C) en 1996. La Recomendación XML dice textualmente: "*El Lenguaje Extensible de Marcas, abreviado XML, describe una clase de objetos de datos llamados documentos XML y parcialmente describe el comportamiento de programas de computador que pueden procesarlos.*" Vamos a ver en detalle algunas de las características de este lenguaje.

- XML (eXtensible Markup Language) surge de la **revisión de SGML por el W3C** (World Wide Web Consortium), y que resultó en el estándar internacional recogido en la llamada Recomendación de XML [1]. XML es una simplificación y adaptación del SGML, y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML).Actualmente coexisten las siguientes versiones de XML:
  - XML 1.0 (Fith Edition), W3C Recommendation 26 November 2008 ([1])
  - XML 1.1 (Second Edition), W3C Recommendation 16 August 2006 ([2])

  Los principales cambios en XML 1.1 con respecto a XML 1.0 están en el uso de los caracteres Unicode (hablaremos de esto más adelante). Entre otras cosas, con XML 1.1 los documentos ya no dependen de una versión Unicode específica, sino que pueden siempre usar la última.

El W3C recomienda usar XML 1.0 si no se necesitan ninguna de las prestaciones nuevas en XML 1.1. Los analizadores deberían funcionar con ambas versiones.

- Como en todos los lenguajes de marcas, los documentos XML se componen de **datos carácter** (la propia información) y **marcado** (marcas o etiquetas XML). El marcado añade información adicional que posibilita una nueva manera de tratar la información, ya que permite realizar sobre los documentos tareas informáticas tales como búsquedas más precisas, filtrados, generación automática de informes, etc.
- En un documento XML toda la información se representa como **texto**. No hay tipos de datos como numéricos, binarios, lógicos, etc.
- Las marcas en un documento XML van entre los símbolos “<” y “>”, o bien, en el caso de las referencias de entidad, empiezan por "&" y acaban con ";".

**Ejemplo 1:** el siguiente ejemplo muestra una nota de Laura para Pedro en formato XML:

```
<nota>
<para>Pedro</para>
<de>Laura</de>
<titulo>Recordatorio</titulo>
<contenido>A las 7:00 pm en la puerta del teatro</contenido>
</nota>
```

En este ejemplo encontramos los siguientes elementos de marcado, cada uno con su marca inicial y final: “nota”, “para”, “de”, “titulo” y “contenido”. El resto son los datos carácter, es decir, la información de la nota.

- XML es **extensible**: en XML las marcas no están predefinidas, sino que podemos definir nuestras propias marcas, con tal de que cumplan los requisitos establecidos en el lenguaje, y que veremos más adelante. Gracias a esta característica XML se adapta a cualquier tipo de situación, necesidades del autor y software de procesamiento, ya que en función de los requerimientos se puede usar un software más sencillo o más complejo.

  Así, en el texto del ejemplo 1, las marcas utilizadas no están definidas en ningún estándar, sino que han sido “inventadas” por el autor del documento, y son específicas del tipo de documento en cuestión, es decir, una nota que una persona envía a otra.
- XML se ha concebido en un contexto **de separación entre el contenido** (información del documento), la **estructura** (tipo y organización de los elementos que componen el documento) y la **presentación** (manera en que la información es presentada al lector).
- XML en sí mismo “no hace nada”. XML está diseñado para facilitar la estructuración, almacenamiento y transporte de información. Pero un documento XML no hace nada en sí mismo, se necesita software adicional para procesarlo.
- XML es un **metalenguaje**, es decir, un lenguaje para la definición de lenguajes de marcado. XML define la sintaxis y los requisitos que deben cumplir los lenguajes de marcado que especifica. Así, XML ha servido para definir gran número de lenguajes de marcado tales como:
  - XHTML: revisión de HTML para adaptarlo a XML
  - MathML: descipción de fórmulas matemáticas
  - CML: Chemical markup language, en ciencias químicas
  - … y muchos otros
- Como se ha dicho anteriormente, XML separa totalmente el contenido de la

presentación de los documentos. XML tiene un lenguaje de estilo asociado para la **presentación** de la información, llamado **XSL** (eXtensible Stylesheet Language), basado en lanotación genérica XML, y que es una combinación de varios estándares: XPath, XSLT (XSL Transformations), FO (Formatting Objects). XML también admite hojas de estilo en cascada **CSS** como alternativa a XSL.

## 2 Estructura de un documento XML

La estructura general de un documento XML está formada por dos partes:

- **Prólogo** (opcional): contiene una secuencia de instrucciones de procesamiento y/o declaración del tipo de documento.
- **Cuerpo**: es el contenido de información del documento, organizado como un árbol único de elementos marcados.

El estándar también permite la inclusión opcional de un **epílogo**, al final del documento, que puede contener instrucciones de procesamiento. Esta parte en general se omite dada su poca utilidad, ya que resulta poco intuitivo poner las instrucciones de procesamiento al final.

Las instrucciones de procesamiento se utilizan para enviar información a las aplicaciones que van a procesar el documento XML. Las instrucciones de procesamiento pueden aparecer en varios lugares del documento (por ejemplo, entre el prólogo y el cuerpo, dentro del cuerpo o en el epílogo. Las veremos en más detalle más adelante.

Para distinguir entre el cuerpo del documento y el documento completo se usan los siguientes términos:

- **Document entity** (o Document root): se refiere a todo el documento
- **Document element**: se refiere al cuerpo

### *2.1 El prólogo*

El **prólogo** añade información sobre el documento. En concreto, declara que el documento es un documento XML e incluye información sobre la versión de XML utilizada para escribirlo (actualmente 1.0 o 1.1). Además, puede incluir información sobre el tipo de codificación de caracteres utilizado en el documento, si es autónomo (contiene en sí mismo toda la información necesaria para procesarlo) o no y el tipo al que se ajusta el documento. Si se incluye, el prólogo debe preceder al ejemplar del documento.

Aunque el prólogo es opcional, su inclusión es muy recomendable ya que facilita un procesado fiable y robusto de la información contenida en el ejemplar. El prólogo puede a su vez dividirse en dos partes:

- La declaración XML
- La declaración del Tipo de Documento

#### 2.1.1 La declaración XML

La **declaración XML** es una instrucción de procesamiento especial y cumple varias funciones:

- Marca el documento como texto XML
- Incluye la declaración de la versión de XML utilizada en el documento
- Aporta información sobre la codificación empleada para representar los caracteres
- Indica si el documento es autónomo o no

Si está presente, la declaración XML debe ser la primera línea del documento.

Un ejemplo de declaración XML completa podría ser:

```
<?xml version= “1.0” encoding= “ISO-8859-1” standalone= “yes”?>
```

Los campos dentro de la declaración XML deben seguir el orden estricto que vemos en el ejemplo anterior. Además, si se especifica la codificación (“encoding”), o la declaración de documento autónomo (“standalone”), es necesario incluir también la información sobre la versión (“versión”).

La **versión (“version”)** permite indicar la versión para la que se elaboró el documento (actualmente puede ser 1.0 o 1.1) y permitir que los documentos se adapten a la evolución del estándar.

La **codificación (“encoding”)** nos permite indicar el juego de caracteres utilizado en el documento. El valor por defecto es UTF-8. Esta codificación no siempre funciona bien con los acentos ni otros caracteres comunes en español. Por este motivo, la declaración de codificación nos será de gran utilidad ya que de otra manera habría que incluir estos caracteres a través de referencias a carácter, lo que puede resultar cuando menos laborioso (salvo que sean pocos). Para incluir acentos o caracteres especiales del castellano, se puede utilizar la codificación de 8 bits ISO-8859-1, asociada a los lenguajes de Europa Occidental. Veremos todo esto en más detalle en la sección *Conjunto de caracteres*. Es conveniente incluir siempre el atributo “encoding” en la declaración XML, y asegurarnos que al guardar los documentos con el editor que hayamos utilizado, estos se guardan con el mismo tipo de codificación.

Por último, la **declaración de documento autónomo (“standalone”)**, puede valer “yes” o “no”. El valor “yes”, indica que el documento contiene en su interior toda la información relevante para su interpretación. Más adelante veremos esto en más detalle. Baste decir por ahora que pueden existir ciertos contenidos, fuera del documento actual, que modifiquen la forma en la que se procesará el documento, esta característica implica que el documento no es autónomo.

### 2.1.2 La declaración del tipo de documento

La **Declaración de Tipo del Documento** es opcional. Se escribe en el prólogo y tiene un formato especial, distinto de las marcas y de las instrucciones de procesamiento. Provee una serie de mecanismos que aportan funcionalidad a XML. Gracias a ella es posible definir una serie de restricciones adicionales que deben cumplir los documentos. También incorpora la posibilidad de utilizar ciertas herramientas que facilitarán al usuario XML algunas tareas. Todas estas propiedades adicionales se engloban bajo lo que se denomina un **tipo**. Los documentos que tienen un tipo asociado, y que cumplen con él, podrán distinguirse del resto y formarán lo que se denomina un tipo de documentos o una clase de documentos.

La declaración de tipo del documento no siempre es necesaria. Es perfectamente posible trabajar con XML sin emplearlas, sobre todo en entornos en los que los documentos XML se generan automáticamente por programas y no es necesario comprobar ciertas condiciones. En el siguiente tema veremos documentos XML con declaraciones de tipo de documento.

Un ejemplo de una declaración de tipo de documento es el siguiente:

```
<!DOCTYPE Casas_Rurales SYSTEM “http://www.casasrurales.com/casasrurales.dtd”>
```

### 2.2 El cuerpo

El **cuerpo** es la parte más importante y que contiene la información del documento, es decir, los datos a los que se les ha añadido el marcado.

El cuerpo de los documentos XML tiene una estructura de árbol, en la que siempre existe un elemento principal, o **elemento raíz**, dentro del cual se encuentran el resto de los elementos. Se dice que el elemento raíz es el "padre" de todos los demás elementos, y de él se derivan las ramas del árbol hasta el nivel más bajo.

Todos elementos de un documento XML pueden a su vez contener sub-elementos o elementos "hijos", según la siguiente estructura genérica:

```
<raíz>
   <hijo1>
      <subhijo1_1>
         <subhijo1_1_1> ... </subhijo1_1_1>
         <subhijo1_1_2> ... </subhijo1_1_2>
         ...
      </subhijo1_1>
      <subhijo1_2> ... </subhijo1_2>
...
   </hijo1>

   <hijo2> ... </hijo2>
   ...
   </raíz>
```

**Ejemplo 2:** veamos un sencillo documento XML (document entity) cuyo cuerpo es la información que vimos en el ejemplo1, a la que se ha añadido un prólogo:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<nota>
<para>Pedro</para>
<de>Laura</de>
<titulo>Recordatorio</titulo>
<contenido>A las 7:00 pm en la puerta del teatro</contenido>
</nota>
```

- La primera línea es la declaración XML. En esta línea se especifica la versión XML que se utiliza (1.0), y el tipo de codificación utilizada (en este caso ISO-8859-1, que corresponde al juego de caracteres Latin-1 / West European). Veremos más sobre los tipos de codificación más adelante.
- La siguiente línea contiene la marca de apertura del **elemento raíz** del documento: "nota"
- Las cuatro líneas siguientes describen 4 elementos hijos del elemento raíz ("para", "de", "título" y "contenido").
- La última línea contiene el cierre del elemento raíz, y el final del documento.

Como se puede observar en este ejemplo, si elegimos identificadores para las etiquetas que sean significativos y representativos, los documentos XML pueden ser muy descriptivos y fáciles de entender.

**Ejemplo 3:** el siguiente ejemplo muestra información sobre libros de una librería.

```
<libreria>
  <libro categoría="COOKING">
    <título lang="en">Everyday Italian</título>
    <autor>Giada De Laurentiis</autor>
    <año>2005</año>
    <precio>30.00</precio>
  </libro>
  <libro categoría="INFANTIL">
    <título lang="en">Harry Potter</título>
    <autor>J K. Rowling</autor>
    <año>2005</año>
    <precio>29.99</precio>
  </libro>
  <libro categoría="WEB">
    <título lang="en">Learning XML</título>
    <autor>Erik T. Ray</autor>
    <año>2003</año>
    <precio>39.95</precio>
  </libro>
</libreria>
```

En este ejemplo el elemento raíz es “librería”. Todos los elementos “libro” están dentro de “librería”. El elemento “libro” tiene 4 hijos: “título”, “autor”, “año” y “precio”.

## 3 Conjunto de caracteres

Ya se ha dicho que en los documentos XML toda la información se representa como texto. Específicamente, todos los caracteres de un documento XML proceden del conjunto de caracteres Unicode - ISO/IEC 10646, según regula la recomendación de XML.

- **Unicode**: es un estándar internacional elaborado por el consorcio Unicode que pretende dar cobertura a la mayoría de los sistemas de escritura actualmente en uso en el mundo, facilitando una representación digital para una gran cantidad de caracteres. Cada carácter tiene una codificación binaria de 16 bits.
- **ISO/IEC 10646:** es otro estándar internacional, también conocido con el nombre de UCS (Universal Carácter Set), que tiene dos versiones, una de 16 bits (UCS-2, donde el 2 se refiere a los 2 octetos utilizados), y otra de 32 bits (UCS-4, por los 4 octetos). Su objetivo es más amplio, ya que abarca un mayor número de caracteres, procedentes de sistemas de escritura antiguos pero también de sistemas actuales.

En la práctica, ambos estándares se pueden considerar equivalentes en muchos aspectos. Para más información sobre Unicode es posible visitar su sitio Web en http://www.unicode.org/.

Hay que distinguir entre dos conceptos diferentes:

- Un **conjunto de caracteres** establece una correspondencia entre un carácter concreto (por ejemplo la letra Z mayúscula) y un número (por ejemplo, 90)
- La **codificación** de caracteres establece cómo se representan estos códigos numéricos en bytes. Por ejemplo, el número 90 puede codificarse mediante un byte con signo, o en complemento a dos, entero corto, etc.

Así, los estándares Unicode - ISO/IEC 10646 incorporan una serie de codificaciones o

métodos de transformación, entre los que cabe destacar UTF-8 y UTF-16. Anterior a estos, se habían definido la serie de estándares de codificación de caracteres de 8 bits ISO/IEC 8859. Esta serie de estándares consiste en varias partes enumeradas (15 en total, sin contar la 12 que no se utiliza). Por ejemplo:

ISO-8859-1: asociada a los lenguajes de Europa Occidental.

ISO-8859-2: asociada a los lenguajes de Europa del Este.

ISO-8859-3: asociada a los lenguajes del sur de Europa.

ISO-8859-4: asociada a los lenguajes del norte de Europa.

ISO-8859-5: contiene los caracteres del Cirílico.

ISO-8859-15: como el ISO-8859-1 pero con el símbolo del Euro.

UTF-8: Unicode comprimido.

UTF-16: UCS (*Universal Charcter System*) comprimido.

En junio de 2004 el grupo de trabajo responsable de mantener estos estándares se disolvió y se abandonó el mantenimiento de la serie de estándares ISO/IEC 8859, para concentrarse en UCS (ISO/IEC 10646).

En las aplicaciones informáticas cada vez se utilizan más las codificaciones que proporcionan soporte completo de UCS (tales como UTF-8 y UTF-16), en preferencia a los códigos de 8 bits como ISO/IEC 8859-1. Para los documentos en español utilizaremos ISO-8859-1, ya que UTF-8 no siempre funciona bien con caracteres como la “ñ” o las letras con acentos.

Aunque todos los caracteres de un documento XML deben estar recogidos en los estándares Unicode - ISO/IEC 10646, no todos los caracteres de estos estándares se permiten en un documento XML. Así, los caracteres legales son: tabulador, retorno de carro, fin de línea y los caracteres gráficos de Unicode y ISO/IEC 10646. El uso de la mayoría de los caracteres no gráficos está prohibido, es decir, no se pueden incluir en un documento XML, ya sea directamente o mediante referencias a carácter.

Podemos incluir en nuestro documento los caracteres permitidos de estos conjuntos mediante una **referencia a carácter,** que se compone de la cadena &#, seguida del número del carácter que se desee y finalizando con un carácter “;”. El número del carácter puede estar expresado en decimal o en hexadecimal (precedidos de una "x" obligatoriamente minúscula):

- &#NNNNN; decimal (hasta 5 dígitos)
- &#xNNNN; hexadecimal (hasta 4 dígitos)

El código numérico (decimal o hexadecimal) corresponde al código Unicode.

Hay caracteres, como “<” y “&”, que no se pueden usar directamente: se introducen como referencias a entidades. Una **entidad** es un fragmento de información, definido como un valor constante, al que se puede hacer referencia mediante un nombre. La forma de hacer referencia a una entidad es: &nombre;

Sólo hay 5 entidades predefinidas: < (<), > (>), & (&), ' (‘) y " (“). Otras entidades deben ser definidas para poder usarlas (lo veremos en detalle más adelante).

**Ejemplo 4:** En el siguiente ejemplo, la primera línea del texto sería correcta, pero no la segunda, porque contiene el carácter “<”.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<ejemplo>
El simbolo &#x3c; también se puede poner como &lt;
El simbolo < también se puede poner como &lt;
</ejemplo>
```

## 4 Elementos

Las distintas piezas de información en las que podemos dividir un documento reciben, en XML, el nombre de **elementos**. Los elementos son los ladrillos, las piezas básicas de la estructura del documento. Generalizando, podemos decir que los elementos cumplen las siguientes funcionalidades:

- Especificación de contextos
- Delimitación de contenidos
- Estructuración de contenidos
- Jerarquización de elementos

Como mínimo, siempre encontraremos en un documento XML un elemento: el elemento raíz. Como ya sabemos, todo documento XML tiene obligatoriamente un elemento raíz (uno y sólo uno).

Los elementos están formados por tres componentes: una **etiqueta de inicio o apertura** (también llamada marca inicial), por ejemplo <Titulo>, una **etiqueta de finalización o cierre** (marca final), por ejemplo </Titulo>, y un **contenido**, situado entre ambas etiquetas.

Hay que notar que, al contrario de lo que sucede en HTML, todos los elementos de XML, están formados por estas tres partes, las etiquetas no son el elemento en sí, son una parte de él que lo delimita y que lo marca. El hecho de olvidar una etiqueta de finalización o de inicio implica un error de buena formación. Una buena costumbre para no olvidar una etiqueta de fin es escribir la de inicio y la de fin al mismo tiempo.

En XML pueden existir elementos sin contenido, que reciben el nombre de **elementos vacíos**, estos elementos pueden aparecer expresados de dos maneras, mediante dos etiquetas, una de inicio y otra de fin, seguidas, o mediante una única etiqueta especial, la etiqueta de elemento vacío.

El contenido de un elemento consta de datos carácter y/o otros subelementos (o elementos hijo). Pueden incluirse referencias a caracteres, referencias a entidades y secciones CDATA (más adelante veremos qué es esto).

**Ejemplo 5:** Veamos el siguiente poema marcado en XML

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Poema>
        El Reino Perdido
        Las huestes de don Rodrigo
        desmayan y huían
        cuando en la octava batalla
        sus enemigos vencían
        ...
</Poema>
```

Éste es un documento XML muy sencillo, está reducido al mínimo y tiene un único elemento, “Poema” (el elemento raíz), gracias al cual podríamos saber que el documento contiene un poema, y podríamos diferenciarle de otros documentos como cartas, pedidos, recetas, etc. Lo cierto es que la información que añade este elemento no es demasiada y seguramente no queramos conformarnos con eso. El documento de partida se puede refinar para añadirle otros elementos (en negrita).

**Ejemplo 6:**

```
<?xml version=“1.0” encoding="ISO-8859-1"?>
<Poema>
   <Titulo>El Reino Perdido</Titulo>
   <Cuerpo>
         Las huestes de don Rodrigo
         desmayan y huían
         cuando en la octava batalla
         sus enemigos vencían
         ...
   </Cuerpo>
</Poema>
```

En este ejemplo existen elementos que identifican el título (“Titulo”) y el cuerpo (“Cuerpo”) del poema. Gracias a estas marcas seríamos capaces, por ejemplo, de listar todos los títulos de los poemas de los que disponemos, o buscar un título determinado y mostrar el cuerpo del poema. Las piezas de texto que forman el título y el cuerpo están perfectamente identificadas y delimitadas. Este proceso de refinamiento y adición de elementos podría continuarse mientras fuera necesario.

**Ejemplo 7:** El siguiente ejemplo nos muestra el uso de etiquetas de elemento vacío:

```
<?xml version=“1.0” encoding="ISO-8859-1"?>
<Poema>
      <Titulo>El Reino Perdido</Titulo>
      <Datos/>
      <Cuerpo>
            Las huestes de don Rodrigo
            desmayan y huían
            cuando en la octava batalla
            sus enemigos vencían
            ...
      </Cuerpo>
</Poema>
```

Vemos que se ha incluido un nuevo elemento vacío “Datos”. Tal y como está ahora este nuevo elemento no aporta información, pero como veremos más adelante, los elementos vacíos normalmente se complementan con atributos.

## 5 Atributos

Ya se ha comentado que los elementos XML pueden incluir, dentro de la etiqueta de inicio, atributos opcionales. Los **atributos** actúan como modificadores, como adjetivos que incluyen información adicional aplicable a un elemento.

**Ejemplo 8:** Si recordamos el ejemplo anterior vemos como ahora el elemento vacío “Datos” tiene otro aspecto.

```
<?xml version=“1.0” encoding="ISO-8859-1"?>
<Poema>
        <Titulo>El Reino Perdido</Titulo>
        <Datos Autor="desconocido" Fecha="1700"/>
        <Cuerpo>
                Las huestes de don Rodrigo
                desmayan y huían
                cuando en la octava batalla
                sus enemigos vencían
                ...
        </Cuerpo>
</Poema>
```

En él aparecen dos nuevos componentes “Autor” y “Fecha”, que son atributos aplicados al elemento “Datos”, y que aportan información sobre el autor del poema y la posible fecha de creación del mismo. También podrían haberse incluido como atributos del elemento raíz Poema.

A veces a la hora de diseñar, puede ser complicado elegir entre un atributo o un elemento, si bien muchas veces el resultado es el mismo. Hay que tener en cuenta que los atributos no pueden contener subelementos, ni subatributos, ni se organizan en ninguna jerarquía, por lo que tienen una capacidad de representación mucho más reducida que los elementos y no pueden reflejar una estructura lógica. La información que añaden suele ser de poca entidad, sencilla, sin estructura y no subdivisible. Como norma general la información que se desea mostrar al usuario se incluye en el documento XML en forma de datos carácter como contenido de algún elemento, ya que puede ocurrir que el sistema de visualización imprima por defecto los elementos pero no los atributos.

## 6 Comentarios

Como se ha comentado anteriormente, en un documento XML se encuentran entremezclados los datos carácter y los caracteres de marcado. Además, pueden incluirse **comentarios** en cualquier parte del documento.

Los comentarios son una forma de añadir información sobre el documento, que en principio no va a ser utilizada por las aplicaciones informáticas, sino por los lectores humanos. Los comentarios no se consideran datos carácter, y pueden ser ignorados por los procesadores XML, pero son importantes ya pueden aportar información útil tanto para el propio autor como para otras personas que vayan a trabajar con el documento. Lo s comentarios en XML se representan igual que en HTML, es decir:

<!-- ... texto del comentario ... -->

El texto del comentario admite cualquier carácter salvo la cadena “--". Pueden aparecer antes o después de un elemento, pero no dentro de una etiqueta. Tampoco pueden aparecer antes de la declaración XML.

**Ejemplo 9:** veamos un ejemplo de la utilización de comentarios

```
<?xml version="1.0" ?>
<!-- Este documento contiene un poema -->
<Poema>
   <Titulo>El Reino Perdido</Titulo>
   <!-- título del poema -->
   <Datos Autor="desconocido" Fecha="1700"/>
   <Cuerpo>
   <!-- cuerpo del poema -->
         Las huestes de don Rodrigo
         desmayan y huían
         cuando en la octava batalla
         sus enemigos vencían
         ...
   </Cuerpo>
</Poema>
```

## 7 Documentos XML bien formados

Un documento XML bien formado (en inglés *'well formed'*), es aquel que cumple con todas las reglas sintácticas definidas para XML. Los procesadores XML pueden rechazar cualquier documento que no esté bien formado.

No hay que confundir un documento XML bien formado con un documento válido. Un documento XML válido es el que está bien formado, y además cumple con la definición de un lenguaje de marcado particular especificado para el documento. Es decir, el cuerpo del documento tiene una estructura de elementos compatible con el lenguaje concreto al que corresponde. Así, todo documento XML válido es un documento bien formado (todos los documentos XML tienen que estar bien formados), pero no ocurre al contrario. En los temas siguientes hablaremos de documentos XML válidos.

En este apartado vamos a ver las reglas sintácticas que deben seguir los documentos XML para estar bien formados. Un procesador XML conforme con la Especificación XML, comprobará que el documento cumple estas restricciones de buena formación. Cualquier restricción no cumplida será detectada y se tratará como un error fatal: el procesador informará a la aplicación y dejará de trabajar de una manera normal. Fundamentalmente, el objetivo de estas restricciones es asegurar que los documentos XML puedan ser interpretados por los procesadores XML sin ninguna ambigüedad, de manera que todos los elementos (y atributos) quedan perfectamente definidos.

### *7.1 Reglas de buena formación genéricas*

Veamos un resumen de las reglas genéricas de buena formación de documentos XML. Después veremos algunas de ellas en más detalle.

- Los **documentos han de seguir una estructura** estrictamente jerárquica en lo que respecta a las etiquetas que delimitan sus elementos. Una etiqueta debe estar correctamente incluida en otra, es decir, las etiquetas deben estar correctamente anidadas. Los elementos con contenido deben estar correctamente cerrados.

  En el siguiente ejemplo, la primera línea sería incorrecta en XML, no así la segunda:

```
<LI>HTML <B>permite <I>esto</B></I>.
<LI>En XML la <B>estructura <I>es</I> jerárquica</B>.</LI>
```

- Los documentos XML requieren un elemento raíz, y sólo uno, del que todos los demás sean parte, es decir, la jerarquía de elementos sólo puede tener un elemento inicial.

- Los elementos no vacíos tienen siempre una etiqueta inicial y una etiqueta final.

- Los elementos vacíos (sin contenido), en su notación abreviada, deben incluir el símbolo “/”, justo antes del cierre de la etiqueta (antes del “>”).

- Los valores de atributos en XML siempre deben estar encerrados entre comillas simples o dobles.

  En el siguiente ejemplo, la primera línea sería incorrecta en XML, no así la segunda:

```
<A HREF=http://www.disney.com/>
<A HREF="http://www.developer.com/">
```

- XML es sensible a mayúsculas y minúsculas. Si un elemento de XML está definido como "ELEMENTO", no podemos usar "elemento", ni "Elemento", ni "eleMENto" para referirnos a él.

- Existe un conjunto de caracteres llamados espacios en blanco (espacios, tabuladores, retornos de carro, saltos de línea) que los procesadores XML tratan de forma diferente en el marcado XML. La especificación XML 1.0 permite el uso de esos "espacios en blanco" para hacer más legible el código, y en general son ignorados por los procesadores XML.

- Es necesario asignar nombres a las estructuras, tipos de elementos, entidades, elementos particulares, etc. En XML los nombres deben cumplir unos requisitos que veremos en el siguiente apartado.

- Las construcciones como etiquetas, referencias de entidad y declaraciones se denominan marcas; son partes del documento que el procesador XML espera entender. El resto del documento entre marcas (los datos carácter) son los datos "entendibles" por las personas.

## *7.2 Reglas de formación para los elementos*

Los elementos, al igual que todos los componentes de un documento XML, deben seguir ciertas reglas de buena formación, aplicables tanto a las etiquetas de inicio y fin como al contenido que admiten.

### 7.2.1 Nombres

Las reglas de formación de los nombres (reglas [4] y [5] de la *Recomendación*) definen la forma general que debe tener un nombre y básicamente, estas reglas de producción dicen que:

```
[4] NameChar ::=  Letter | Digit | '.' | '-' | '_' | ':' | CombiningChar |
Extender
[5] Name ::=  (Letter | '_' | ':') (NameChar)*
```

- Un nombre (Name) se compone como mínimo de **una letra**, (desde la "a" a la "z" o desde la "A" a la "Z"), de **un guión bajo** o de **un carácter de dos puntos**.
- El carácter inicial (letra, guión bajo o carácter de dos puntos) puede estar seguido

de otros caracteres, definidos dentro del grupo NameChar (regla [4]), es decir, una o más letras, dígitos, puntos, guiones, guiones bajos, caracteres de dos puntos, caracteres CombiningChar y caracteres Extender, en cualquier combinación.

- La codificación de los nombres es sensitiva a mayúsculas y minúsculas
- los nombres no pueden empezar por la cadena xml, incluidas todas las versiones cambiando el minúsculas por mayúsculas, (XML, Xml, xMl, ...), estos nombres se reservan para posteriores estandarizaciones.

El resto de los caracteres, destacando los espacios en blanco y los siguientes caracteres ",", "!" o ";", no pueden formar parte de un nombre. Observar además que un nombre no puede comenzar ni por un dígito, ni por un punto, ni por un guión.

Dice la Recomendación XML que el carácter de dos puntos está reservado para la experimentación con espacios de nombres. Esto implica que no debe utilizarse en nombres, salvo con espacios de nombres (lo veremos más adelante). También especifica la Recomendación que los procesadores de XML deben aceptar este carácter como un carácter de nombre, a pesar de ello, algunos no lo hacen con lo que el carácter de dos puntos debe retirarse del nombre si se desea utilizar ese procesador en concreto.

Para más información sobre los caracteres incluidos dentro del grupo "CombiningChar" y "Extender", consultar la Recomendación.

### 7.2.2 Etiquetas de inicio, de fin y de elemento vacío

Como ya hemos visto, los elementos XML se componen de una etiqueta de inicio, un contenido y una etiqueta de fin. En esta sección revisamos estos conceptos para formalizar algunos detalles, y veremos también como los elementos sin contenido pueden reducirse a una única etiqueta de inicio y fin.

- **Etiqueta de inicio.** Formato: <*nombre* atributos_opcionales>

  Las etiquetas de inicio comienzan con un carácter "<", y a continuación, sin ninguna separación, un identificador genérico (un nombre XML). A lo anterior, le pueden seguir espacios en blanco opcionales y una lista de atributos. Las etiquetas finalizan con un carácter ">". Todas las etiquetas de inicio deben tener su correspondiente etiqueta de fin con el mismo identificador genérico.

- **Etiqueta de fin.** Formato: </*nombre*>

  Las etiquetas de fin comienzan con la cadena "</", seguida, sin separación, de un identificador genérico (un nombre XML) seguido a su vez de un carácter ">". Entre el nombre del elemento y el carácter ">" puede, opcionalmente, haber espacios en blanco. Para que una etiqueta de fin tenga sentido debe haber una etiqueta de inicio anterior, con el mismo identificador genérico (incluidas las mayúsculas y las minúsculas).

- **Etiqueta de elemento vacío**: los elementos vacíos pueden aparecer expresados de dos maneras:

  - mediante dos etiquetas, una de inicio y otra de fin, seguidas:

    ```
    <nombre atributos_opcionales></nombre>
    ```

  - o, la opción más utilizada, mediante una única etiqueta especial, la **etiqueta de elemento vacío**, formada por el carácter "<", seguido de un identificador genérico (un nombre XML), espacios en blanco y atributos opcionales, y la cadena "/>":

```
<nombre atributos_opcionales />
```

#### 7.2.3 Caracteres prohibidos dentro del contenido de los elementos

Dentro del contenido de los elementos no pueden aparecer los siguientes caracteres y cadenas, ya que tienen funciones especiales. Si aparecen deben escaparse para que no se interpreten erróneamente:

- el carácter "<" porque podría confundirse con el comienzo de una etiqueta.
- el caracter "&" porque podría confundirse como comienzo de una referencia a entidad (todavía no se ha hablado de ellas).
- la cadena "]]>" no debe aparecer dentro del contenido de un elemento. Esto es así para asegurar cierta compatibilidad hacia atrás con SGML.

Ya se comentó en el apartado 'Conjunto de caracteres' que la mejor manera para escapar caracteres es utilizar referencias a entidades predefinidas.

#### 7.2.4 Restricciones en cuanto al anidamiento de los elementos

Los elementos deben anidarse correctamente. Es decir, si la etiqueta de inicio de un elemento está dentro de otro elemento, la etiqueta de finalización del primer elemento debe estar dentro de la etiqueta de finalización del segundo elemento.

Siempre que estas reglas de anidamiento se cumplan, y a falta de una declaración que especifique el contenido del elemento (se estudiarán más adelante), los elementos admiten cualquier combinación de elementos y datos carácter.

### *7.3 Reglas de formación para los atributos*

#### 7.3.1 Nombres

Los nombres de los atributos deben cumplir las mismas restricciones que los nombres de los elementos vistas anteriormente.

#### 7.3.2 Dónde especificar los atributos

Los atributos y sus valores se especifican dentro de una etiqueta de inicio o de elemento vacío, a continuación del nombre del elemento. Entre ambos debe haber al menos un espacio en blanco. Si un elemento tiene varios atributos éstos se deben separar entre sí, como mínimo, por un espacio en blanco. Entre los atributos y el último carácter de la etiqueta, ">", puede haber opcionalmente espacios en blanco.

#### 7.3.3 Valores de los atributos

La asignación del valor al atributo se realiza a través de un carácter "=". A ambos lados del igual pueden aparecer tantos espacios en blanco como se desee (o ninguno).

En XML, a diferencia de HTML, para especificar el valor de un atributo éste **se tiene que escribir entre comillas simples o dobles**, el tipo que no se utilice se puede incluir dentro del valor para marcar literales. El mismo tipo que abre el valor debe cerrarlo, lo contrario representa un error de buena formación.

#### 7.3.4 Otros requisitos

A falta de una declaración que especifique qué valores puede tomar un atributo (se

estudiarán más adelante), los atributos contienen cualquier combinación de datos carácter que no contenga los caracteres “<” y “&” ya que se interpretarían de forma errónea

Dentro de la etiqueta de inicio de un elemento no pueden aparecer dos referencias al mismo atributo. Esta restricción es lógica, ya que el procesador XML no podría conocer cuál es el valor correcto.

### *7.4 Herramientas que comprueban la buena formación*

Hay una gran variedad de aplicaciones disponibles para comprobar que un documento XML está bien formado.

Quizás las aplicaciones más sencillas de utilizar sean los navegadores para la Red que utilizamos diariamente. Si el documento se almacenó en un fichero de texto plano, podemos intentar abrirlo con exploradores como Microsoft Internet Explorer o Firefox. Si está bien formado podremos ver el árbol asociado al documento.

En caso de querer utilizar otras aplicaciones, en las siguientes direcciones podemos utilizar analizadores en línea:

- http://www.w3.org/2001/03/webdata/xsv (Validador para XML Schema del W3C)
- http://validator.w3.org/ (Servicio de Validación de marcado del W3C)
- http://tools.decisionsoft.com/schemaValidate.html (Validador para XML Schema de DecisionSoft)
- http://www.stg.brown.edu/service/xmlvalid/ (Validador XML 1.0 de Brown University Library)

## 8 Diseño de documentos XML

Los documentos XML no se elaboran para permanecer aislados, forman parte de un sistema más complejo que es capaz de utilizar la información contenida en estos documentos con algún objetivo. En este apartado vamos a ver las distintas fases en la elaboración de documentos XML, ilustrándolas con un ejemplo.

Podemos identificar tres fases imprescindibles para crear documentos XML de cierta complejidad:

1) Especificación de los requisitos.

2) Diseño de las etiquetas

3) Marcado de los documentos.

### *8.1 Especificación de requisitos*

Al igual que ocurre con la mayoría de los objetos que fabricamos, o en la creación de programas informáticos, los documentos XML parten de una **especificación** de requisitos. En esta primera fase se identifican y se documentan aspectos tales como objetivos que se pretende conseguir, uso que se va a dar a los documentos, necesidades existentes, perspectivas de crecimiento, recursos disponibles, limitaciones existentes (de tiempo, de presupuesto, de personal, etc.), personas o empresas que realizarán las distintas partes del proceso, etc. Todo ello resulta en un documento de especificación de requisitos que es el punto de partida para la fase de diseño.

**Supuesto práctico**

Como lo que nos interesa aquí es centrarnos el los aspectos de diseño y creación de los documentos XML, consideremos un posible escenario con una especificación de requisitos muy simple. Vamos a seguir utilizando el ejemplo del poema visto hasta ahora. Supongamos que se necesita hacer una recopilación de la poesía española del siglo XX para alimentar a un "*buscador de poesía*" en el que se pueda solicitar información de diferentes formas: por título del poema (o una porción de él), por autor y por movimiento al que pertenece el autor. Además se quiere poder almacenar información de la temática y el estilo del poema junto con las diferentes estrofas y líneas que lo componen. Por último se desea guardar información biográfica básica del autor, al menos la fecha de nacimiento y de muerte en caso de que haya fallecido.

## *8.2 El Diseño*

El diseño es la siguiente fase, se encarga de definir cuáles son las marcas a incluir, sus nombres, la información que contienen, la relación entre las marcas, etc.

Normalmente para abordar un problema es conveniente dividirlo en fases más sencillas que permitan resolverlo, esta frase cobra un especial significado si la resolución del problema no es trivial o admite varias soluciones. En esta sección veremos algunos pasos y consejos útiles a la hora de realizar el diseño de documentos XML. Esta exposición se hará a través de nuestro ejemplo.

### 8.2.1 Primer paso: ¿qué hay que marcar?

A partir de la especificación de requisitos se puede deducir cuáles son las piezas de información que es necesario incluir en el documento XML para poder soportar los procesos que nos piden, y que son, en definitiva, el fin último de XML. En este estado del proceso de diseño no es posible todavía hablar ni de elementos ni de atributos, de momento podemos referirnos a ellos como átomos informativos.

En nuestro ejemplo, identificamos la necesidad de guardar cierta información:

- Debe existir una pieza de información que marque el título del poema y que puede llamarse "titulo".
- Debe existir una pieza de información que marque el estilo del poema y que puede llamarse "estilo".
- Debe existir una pieza de información que marque la temática del poema y que puede llamarse "tematica".
- Debe existir una pieza de información que marque el nombre del autor y que puede llamarse "autor".
- Debe existir una pieza de información que marque la fecha de nacimiento del autor y que puede llamarse "fecha_nacimiento".
- Debe existir una pieza de información que marque la fecha de muerte del autor y que puede llamarse "fecha_muerte".
- Debe existir una pieza de información que marque el movimiento literario al que pertenece el autor y que puede llamarse "movimiento".
- Debe existir una pieza de información que marque las diferentes estrofas del poema y que puede llamarse "estrofa".
- Debe existir una pieza de información que marque las diferentes líneas del poema y que puede llamarse "linea".

Conviene utilizar una norma consistente para la realización del marcado y mantenerla a lo largo de todo el documento. En el ejemplo que nos ocupa, todos los nombres se han escrito con minúsculas y sin acentuar, y todos son además palabras completas.

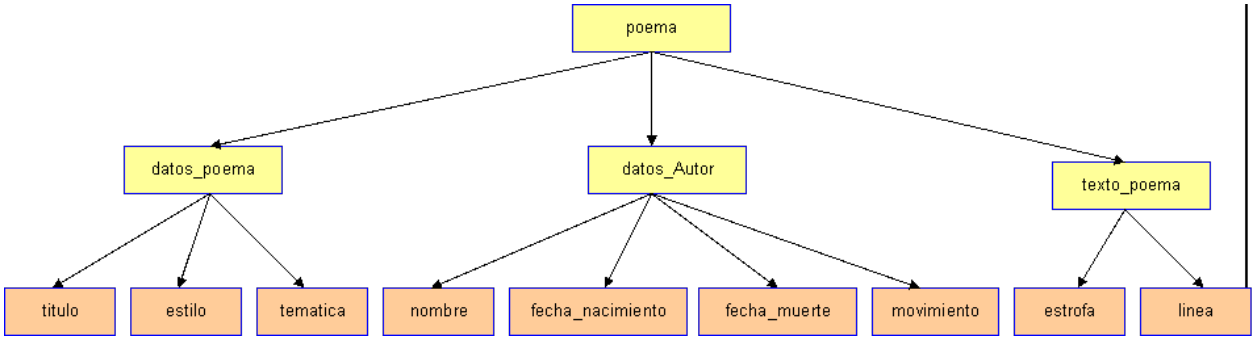
### 8.2.2 Segundo paso: añadir organización y estructura

Después de conocer qué marcas básicas hay que añadir, podemos intentar relacionarlas y agruparlas de la manera más coherente posible. Algunos de los tipos de datos anteriores se convertirán en aglutinantes de otros tipos, también es posible que aparezcan nuevos tipos de datos, que aparecen como necesarios para agrupar/separar componentes.

En nuestro ejemplo podemos estructurar la información de un poema como un documento con tres partes principales: una parte en la que se recoge información general sobre el poema, una parte en la que se recoge información general sobre el autor, y el propio poema.

- Siguiendo con el esquema propuesto, podemos pensar que los componentes, título, estilo y temática podían agruparse en un componente tipo datos_poema.
- De igual forma, la información asociada al autor: nombre, fecha_nacimiento, fecha_muerte y movimiento podían agruparse en un componente del tipo datos_autor.
- Y por último los componentes que informan sobre la estructura del poema (estrofa y linea) podrían agruparse entorno a un componente del tipo texto_poema junto con el propio poema.

En este estado del proceso de creación resulta interesante la realización de un árbol que muestre la relación entre todos los tipos de datos, y que ayude a descubrir nuevas relaciones o una distribución distinta.




### 8.2.3 Tercer paso: Elementos o atributos

El siguiente paso es pensar si estos "tipos de datos" pueden ser elementos o atributos. Se descartan como atributos aquellas piezas que puedan contener subelementos (ahora o en el futuro), así como aquellas que contengan textos más largos de una frase o demasiada información como para guardarla con atributos elegantemente.

En nuestro ejemplo:

- Existe como era de esperar un elemento raíz que se llamará poema.
- El titulo podría ser un atributo o un elemento, la información que contiene no está sujeta a subdivisiones. Sin embargo, se ha decidido que sea un elemento, ya que se considera una parte fundamental de cualquier poema.
- Respecto al estilo y a la temática podemos pensar en ellos como atributos del titulo o incluso del propio poema.
- Lo mismo ocurre con la información biográfica del autor (fecha_nacimiento y fecha_muerte) que puede considerarse un atributo del autor.

- Respecto al movimiento de pertenencia del autor este dato tiene la suficiente importancia (podremos realizar búsquedas por él) como para considerarse un elemento.

### 8.2.4 Últimas consideraciones en el diseño

Este proceso de creación de etiquetas debe ser realizado con delicadeza y sabiduría, para no quedar defraudados por los resultados de un mal diseño. Deberemos pensar primero que es lo que esperamos de nuestro documento XML y diseñar un conjunto de marcas que se ajuste a nuestro objetivo y defina la estructura lógica y la semántica del documento convenientemente.

### 8.2.5 ¿Inclusión o no de una marca?

El marcado en XML se puede hacer tan rico o tan pobre como se desee, pero siempre debemos tener claro que lo que podamos hacer con el documento está limitado a la información aportada por los elementos. Las aplicaciones que usen el documento no pueden imaginar, sólo trabajar con la información incluida. Por ello, ante la duda de incluir una marca, debemos pensar en su utilidad, y si aun así no queda claro, es preferible incluirla, salvo si ello complica en exceso todo el proceso de marcado. Esta afirmación tiene su sentido ya que es más fácil ignorar información que tener que añadir marcas posteriormente, lo que hará necesario actualizar todos los documentos que se crearon anteriormente, con el consiguiente trabajo adicional. En algunos casos el número de ficheros que es necesario modificar puede ser bastante grande.

En nuestro ejemplo nos hemos dado cuenta de que al introducir los elementos estilo y temática como atributos del poema parece que la agrupación datos_poema se queda un poco vacía de contenido. Podemos pensar en eliminarla.

### 8.2.6 Necesidades futuras

Puede ser interesante intentar detectar necesidades futuras y anticiparse a ellas e incluir alguna marca "de más". O crear los documentos con una estructura que los haga fácilmente actualizables o ampliables. Esta afirmación es especialmente significativa con las etiquetas de más alto nivel, las más cercanas al objeto raíz, que deben ser permisivas en cuanto al contenido que pueden admitir y ser amplias en sus miras.

Normalmente, no se deberá utilizar un atributo para contener texto o información que pueda ser susceptible de una división futura mediante elementos.

En nuestro caso podemos pensar que sería interesante dada la estructura del documento que tenemos el poder diferenciar las diferentes estrofas y líneas del poema. Para ello podemos utilizar un atributo ne para las estrofas y nl para las líneas.

### 8.2.7 Revisión

Hay que comprobar que se cumplen los objetivos y los condicionantes especificados en la Especificación de Requisitos.

### 8.2.8 Nombres descriptivos

Los nombres de las marcas deben identificar la función o el contenido que delimitan, de esta manera los documentos XML se hacen autodescriptivos, facilitándose el trabajo que se realiza con ellos. Que sean descriptivos no implica que deban ser nombres largos, ni que decir tiene que los nombres deben ser lo más cortos posible, especialmente si el marcado es manual.

Es importante definir una política uniforme a seguir con los nombres que haga más fácil recordarlos y reduzca el número de equivocaciones: ¿se escribía con mayúscula o con minúscula? ¿Llevaba acento o no? ¿Cómo se separaba los nombres compuestos? ¿Con un guión? Hay que ser coherentes en este sentido para no trabajar más de lo necesario.

#### 8.2.9 Problemas con las búsquedas

Hay que tener en cuenta ciertos detalles que pueden afectar al resultado de las búsquedas. Por ejemplo, si los títulos de los poemas se escriben en minúsculas y con acentos se deben buscar con minúsculas y con acentos. Es importante en este sentido llegar a una convención que evite problemas en el futuro.

#### 8.2.10 Marcado automático o manual

Si el marcado es manual la única posibilidad que existe para simplificar el proceso es reducir el número de marcas a añadir. Las que se decida incluir deben ser las mínimas que aseguren la funcionalidad requerida y cumplan con las expectativas. Se debe ser especialmente cuidadoso para no incluir marcas gratuitas, ni complicar el proceso en exceso. Existen casos en los que se puede combinar una etapa de marcado mecánico y otra manual, siempre intentaremos reducir el marcado manual al mínimo, ya que, está sometido a errores y requiere más recursos. Hay veces en los que se pueden incluir ciertas marcas a mano que den pie a marcas automáticas.

#### 8.2.11 La estructura lógica y los programas

La estructura lógica y la organización de los elementos y los atributos debe reflejar el contenido del documento original, pero existen algunos pormenores a considerar. Las marcas y su estructura lógica deben tender a que su lectura sea fácilmente legible y asimilable por un lector humano, pero sobre todo deben tender a hacer que los programas que soportaran estas etiquetas sean más sencillos. Es decir, estructuras demasiado complejas (muchos niveles de anidamiento de elementos), o demasiado simples, complican los programas que se construyen sobre ellas. El primer caso porque es necesario navegar más por el documento para encontrar la información que se busca y en segundo caso porque hay que descartar información que no se desea y que está mezclada con la que se busca. Las marcas que reflejan la estructura lógica más que el contenido, es decir las marcas que engloban más marcas y que permiten dividir el documento en partes, facilitan las búsquedas de información o de marcas concretas ya que al "trocear" el documento sólo hay que buscar en una determinada zona semántica. Estas marcas facilitan además una rápida comprensión del documento. Pero un exceso, puede complicar las búsquedas y dificultar la lectura y compresión del documento.

## *8.3 El proceso de marcado*

El marcado de los documentos cierra las fases de creación de un documento XML y se fundamenta en los principios de diseño anteriores. En esta etapa el factor de automatización posible es decisivo:

- Si el procesado es manual puede introducir errores, ya sea por omisión o por marcado erróneo. El personal que realiza esta tarea no necesita conocer XML en profundidad, sobre todo si el marcado es rutinario, y puede ser diferente al que realizó las fases anteriores.
- Si el procesado puede automatizarse, será necesario un técnico que realice los programas.

Es conveniente concluir el marcado de los elementos comprobando si el documento está bien formado, de otra manera, cuando se procesen los documentos podríamos, encontrarnos con errores que impidieran acceder al documento

### Final del ejemplo

Para finalizar el ejemplo se añadirán las marcas diseñadas al poema “Poema de la guerra” de **Antonio Machado**. Esta operación se tiene que realizar inevitablemente a mano, sin posibilidad de automatizar ninguna parte de la misma.

```
Otra vez en la noche...Es el martillo
de la fiebre en las sienes bien venidas
del niño. -Madre, ¿el pájaro amarillo!
¡las mariposas negras y moradas!

Duerme, hijo mío. - Y la manita oprime
la madre, junto al lecho-. ¡Oh flor de fuego!
¿Quién ha de helarte, flor de sangre, dime:
Hay en la pobre alcoba olor de espliego;
...
```

Al añadir las marcas diseñadas en la fase anterior, el aspecto de este poema en XML sería el siguiente:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<poema estilo="modernismo" tematica="guerra">
   <titulo>Poema de la guerra</titulo>
   <datos_autor>
      <autor fecha_nacimiento="1875" fecha_muerte="1939">
          Antonio Machado
      </autor>
      <movimiento>Generación del 98</movimiento>
   </datos_autor>
   <texto_poema>
      <estrofa ne="1">
         <linea nl="1">Otra vez en la noche...Es el martillo</linea>
         <linea nl="2">de la fiebre en las sienes bien venidas</linea>
         <linea nl="3">del niño. -Madre, ¿el pájaro amarillo!</linea>
         <linea nl="4">¡las mariposas negras y moradas!</linea>
      </estrofa>
      <estrofa ne="2">
         <linea nl="5">Duerme, hijo mío. - Y la manita oprime</linea>
         <linea nl="6">la madre, junto al lecho-. ¡Oh flor de fuego!</linea>
         <linea nl="7">¿Quién ha de helarte, flor de sangre, dime:</linea>
         <linea nl="8">Hay en la pobre alcoba olor de espliego;</linea>
      </estrofa>
          ...
   </texto_poema>
</poema>
```

## 9 Referencias

[1] Extensible Markup Language (XML) 1.0 (Fifth Edition). W3C Recommendation 26 November 2008

http://www.w3.org/TR/2008/REC-xml-20081126/

[2] Extensible Markup Language (XML) 1.1 (Second Edition).W3C Recommendation, 16 August 2006.

http://www.w3.org/TR/2006/REC-xml11-20060816/

[3] XML a través de ejemplos. Abraham Gutiérrez y Raúl Martínez. Ed. Ra-Ma

[4] Material del curso de formación de profesores de la Comunidad de Madrid “Desarrollo de aplicaciones WEB con PHP y XML”, impartido por EUI Universidad Politécnica de Madrid.