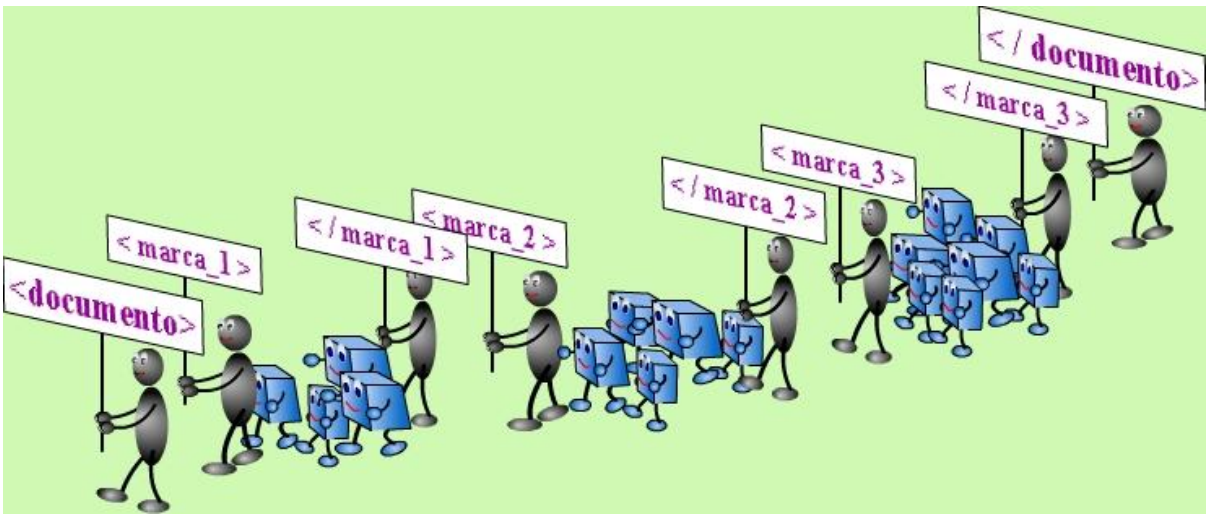
Reconocimiento de las características de los lenguajes de marcas

1. Lenguajes de marcas.



Un "lenguaje de marcas" es un modo de codificar un documento donde, junto con el texto, se incorporan etiquetas, marcas o anotaciones con información adicional relativa a la estructura del texto o a su formato de presentación.

Todo lenguaje de marcas está definido en un documento denominado DTD (Document Type Definition). En él se establecen las marcas, los elementos utilizados por dicho lenguaje y sus correspondientes etiquetas y atributos, su sintaxis y normas de uso.

Ejemplo:

Gobierno del Principado de Asturias
Consejería de Educación y Ciencia
Unión Europea
Fondo Social Europeo

```
<carta>
   <fecha>22/11/2006</fecha>
  <presentacion>Estimado cliente:</presentacion>
  <contenido>bla bla bla bla ...</contenido>
  <firma>Don Jose Gutiérrez González</firma>
</carta>
```

Algunos ejemplos de lenguajes de marcado agrupados por su ámbito de utilización son:

- **Documentación electrónica**
  - **RTF (Rich Text Format): Formato de Texto Enriquecido, fue desarrollado por Microsoft en 1987. Permite el intercambio de documentos de texto ente distintos procesadores de texto.**
  - **TeX: Su objetivo es la creación de ecuaciones matemáticas complejas.**
  - **Wikitexto: Permite la creación de páginas wiki en servidores preparados para soportar este lenguaje.**
  - **DocBook: Permite generar documentos separando la estructura lógica del documento de su formato. De este modo, dichos documentos, pueden publicarse en diferentes formatos sin necesidad de realizar modificaciones en el documento original.**

- **Tecnologías de internet**
  - **HTML, XHTML: (Hypertext Markup Language, eXtensible Hypertext Markup Language): su objetivo es la creación de páginas web.**
  - **RSS: permite la difusión de contenidos web (Desde el punto de vista Web es un estándar para la sindicación de contenidos, permite a un sitio utilizar los servicios o contenidos ofertados por otra web diferente**).

- **Otros lenguajes especializados**
  - **MathML (Mathematical Markup Language): su objetivo es expresar el formalismo matemático de tal modo que pueda ser entendido por distintos sistemas y aplicaciones.**

- **VoiceXML (Voice Extended Markup Language) tiene como objetivo el intercambio de información entre un usuario y una aplicación con capacidad de reconocimiento de habla.**
- **MusicXML: permite el intercambio de partituras entre distintos editores de partituras.**

2. Evolución de los lenguajes de marcas.

En los años 70 surgen unos lenguajes informáticos, distintos de los lenguajes de programación, orientados a la gestión de información. Con el desarrollo de los editores y procesadores de texto surgen los primeros lenguajes informáticos especializados en tareas de descripción y estructuración de información: los lenguajes de marcas. Paralelamente, también, surgen otros lenguajes informáticos orientados a la representación, almacenamiento y consulta eficiente de grandes cantidades de datos: lenguajes y sistemas de bases de datos.

Los lenguajes de marcas surgieron, inicialmente, como lenguajes formados por el conjunto de códigos de formato que los procesadores de texto introducen en los documentos para dirigir el proceso de presentación (impresión) mediante una impresora. Como en el caso de los lenguajes de programación, inicialmente estos códigos de formato estaban ligados a las características de una máquina, programa o procesador de textos concreto y, en ellos, inicialmente no había nada que permitiese al programador (formateador de documentos en este caso) abstraerse de las características del procesador de textos y expresar de forma independiente a éste la estructura y la lógica interna del documento.

Ejemplo:
Dado el siguiente documento:

<times 14><color verde><centrado> Este texto es un ejemplo para mostrar la utilización primitiva de las marcas</centrado></color></times 14>

<color granate><times 10><cursiva>Para realiza este ejemplo se utilizan etiquetas de nuestra invención. </cursiva> Las partes importantes del texto pueden resaltarse usando la <negrita>negrita</negrita>, o el <subrayar>subrayado</subrayar></times 10></color>

Al imprimirlo se obtendría:

Este texto es un ejemplo para mostrar la utilización primitiva de las marcas

Para realiza este ejemplo se utilizan etiquetas de nuestra invención. Las partes importantes del texto pueden resaltarse usando la **negrita**, o el subrayado

Posteriormente, se añadieron como medio de presentación a la pantalla. Los códigos de estilo de visualización anteriores ya no aparecen, y se emplean otros medios para marcarlos, distintos de la inclusión a mano de cadenas formateadoras, ahora ese proceso se automatiza y basta pulsar una combinación de teclas, o pulsar un botón, para lograr los resultados requeridos. Aunque esto es sólo una abstracción, para su uso interno las aplicaciones siguen utilizando marcas para delimitar aquellas partes del texto que tienen un formato especial.

Este marcado estaba exclusivamente orientado a la presentación de la información, aunque pronto se percataron de las posibilidades del marcado y le dieron nuevos usos que resolvían una gran variedad de necesidades, apareció el formato generalizado.

2.1 GML (Generalizaed Markup Language).

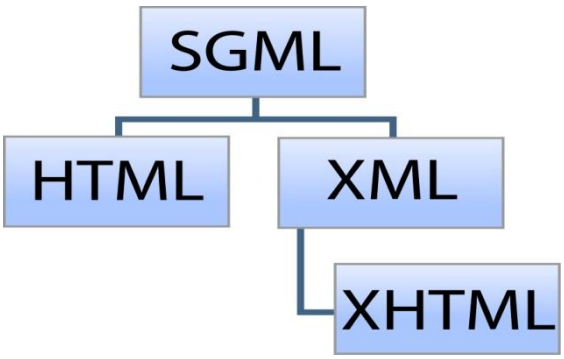
Uno de los problemas que se conocen desde hace décadas en la informática es la falta de estandarización en los formatos de información usados por los distintos programas.

Para resolver este problema, en los años sesenta IBM encargó a Charles F. Goldfab la construcción de un sistema de edición, almacenamiento y búsqueda de documentos legales. Tras analizar el funcionamiento de la empresa llegaron a la conclusión de que para realizar un buen procesado informático de los documentos había que establecer un formato estándar para todos los documentos que se manejaban en la empresa. Con ello se lograba gestionar cualquier documento en cualquier departamento y con cualquier aplicación, sin tener en cuenta dónde ni con qué se generó el documento. Dicho formato tenía que ser válido para los distintos tipos de documentos legales que utilizaba la empresa, por tanto, debía ser flexible para que se pudiera ajustar a las distintas situaciones.

El formato de documentos que se creó como resultado de este trabajo fue GML, cuyo objetivo era describir los documentos de tal modo que el resultado fuese independiente de la plataforma y la aplicación utilizada.

2.2 SGML (Standard Generalizaed Markup Language).

El formato GML evolucionó hasta que en 1986 dio lugar al estándar ISO 8879 que se denominó SGML. Éste era un lenguaje muy complejo y requería de unas herramientas de software caras. Por ello su uso ha quedado relegado a grandes aplicaciones industriales y ámbito universitario. SGML se creó con el objetivo de que los documentos electrónicos fuesen independientes de los formatos generados por los procesadores de texto y de los sistemas operativos.



2.3 HTML (HyperText Markup Language).

En 1989/90 Tim Berners-Lee creó el World Wide Web (WWW) (sistema hipermedia distribuido, accesible a través de Internet, que permite navegar con facilidad por una enorme cantidad de información) y se encontró con la necesidad de organizar, enlazar y compatibilizar gran cantidad de información procedente de diversos sistemas. Para resolverlo creó un lenguaje de descripción de documentos llamado HTML.
HTML es una versión simplificada de SGML, ya que sólo se utilizaban las instrucciones absolutamente imprescindibles. Era tan fácil de comprender que rápidamente tuvo gran aceptación logrando lo que no pudo SGML, HTML se convirtió en un estándar general para la creación de páginas web. Además, tanto las herramientas de software como los navegadores que permiten visualizar páginas HTML son cada vez mejores.

HTML (HyperText Markup Language) es el lenguaje de marcas (etiquetas) más conocido y utilizado para la creación de páginas web que permite la navegación tipo hipertexto (Hipertexto en informática, es el nombre que recibe el texto que en la pantalla de un dispositivo electrónico, permite conducir a otros textos relacionados, pulsando con el ratón en ciertas zonas sensibles y destacadas. La forma más habitual de hipertexto en informática es la de hipervínculos que van a otros documentos).

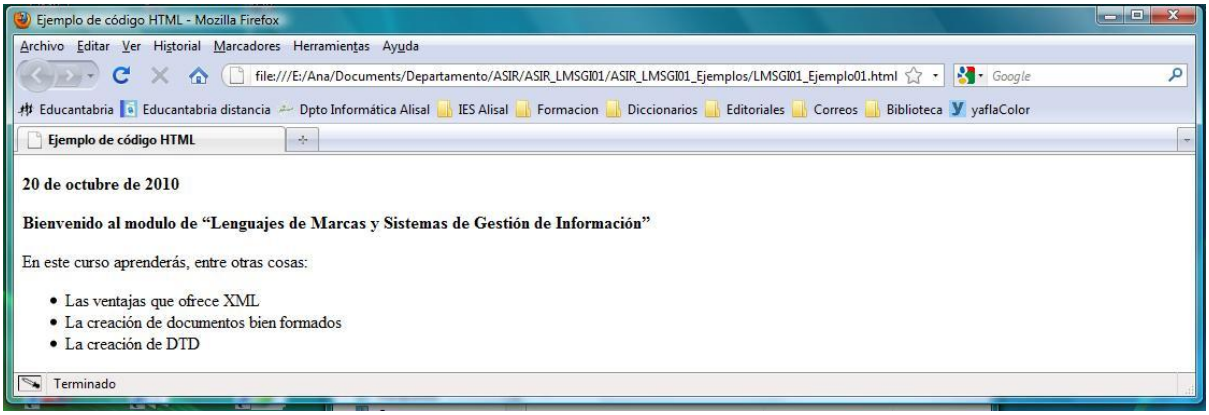
A pesar de todas estas ventajas HTML no es un lenguaje perfecto, sus principales desventajas son:

- **No soporta tareas de impresión.**
- **El lenguaje no es flexible, ya que las etiquetas son limitadas.**
- **No permite mostrar contenido dinámico.**
- **La estructura y el diseño están mezclados en el documento.**

Ejemplo de documento HTML:

```
<html>
  <head>
    <title> Ejemplo de código HTML</title>
  </head>
    <body bgcolor="#ffffff">
    <p></p>
    <p>
      <b>20 de octubre de 2010</b>
    </p>
    <p><b> Bienvenido al modulo de “Lenguajes de Marcas y Sistemas de Gestión de Información” </b></p>
    <p> En este curso aprender&aacute;s, entre otras cosas:<br/>
      <ul>
        <li>Las ventajas que ofrece XML </li>
        <li>La creaci&oacute;n de documentos bien formados </li>
        <li>La creaci&oacute;n de DTD</li>
      </ul>
    </p>
  </body>
</html>
```

Al publicarlo en un navegador, por ejemplo en el Firefox, tendríamos:



2.4 XML (eXtensible Markup Language).

El W3C (El World Wide Web Consortium es una comunidad internacional que desarrolla estándares para la web) establece, en 1998, el estándar internacional XML, un lenguaje de descripción de documentos estructural que no incluye ninguna información relativa al diseño. Está convirtiéndose con rapidez en estándar para el intercambio de datos en la Web. A diferencia de HTML las etiquetas indican el significado de los datos en lugar del formato con el que se van a visualizar los datos.

Pero XML es más que un lenguaje, es un metalenguaje que permite definir otros lenguajes de marcas con objetivos diferentes. Por ese motivo se le llama 'eXtensible'. Por lo tanto, XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes específicos. Un ejemplo de lenguaje que usa XML para su definición es XHTML (eXtensible, Hypertext Markup Language), nueva versión de HTML que cumple la especificación SGML y cuyo objetivo es sustituirlo como estándar de páginas web.

XML es un metalenguaje caracterizado por:

- **Permitir definir etiquetas propias.**
- **Permitir asignar atributos a las etiquetas.**
- **Utilizar un esquema para definir de forma exacta las etiquetas y los atributos.**
- **La estructura y el diseño son independientes.**

En realidad XML es un conjunto de estándares relacionados entre sí y que son:

- **XSL, eXtensible Style Language. Permite definir hojas de estilo para los documentos XML e incluye capacidad para la transformación de documentos.**
- **XML Linking Language, incluye Xpath, Xlink y Xpointer. Determinan aspectos sobre los enlaces entre documentos XML**
- **XML Namespaces. Proveen un contexto al que se aplican las marcas de un documento de XML y que sirve para diferenciarlas de otras con idéntico nombre válidas en otros contextos.**
- **XML Schemas. Permiten definir restricciones que se aplicarán a un documento XML. Actualmente los más usados son los DTD y los XSD.**

Ejemplo de código XML:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE biblioteca>
<biblioteca>
    <ejemplar tipo_ejem="libro" titulo="XML practico" editorial="Ediciones Eni">
    <tipo>
        <libro isbn="978-2-7460-4958-1" edicion="1" paginas="347"></libro>
    </tipo>
    <autor nombre="Sebastien Lecomte"></autor>
    <autor nombre="Thierry Boulanger"></autor>
    <autor nombre="Ángel Belinchon Calleja" funcion="traductor"></autor>
    <prestado lector="Pepito Grillo">
      <fecha_pres dia="13" mes="mar" año="2009"></fecha_pres>
      <fecha_devol dia="21" mes="jun" año="2009"></fecha_devol>
    </prestado>
  </ejemplar>

  <ejemplar tipo_ejem="revista" titulo="Todo Linux 101. Virtualización en GNU/Linux"
editorial="Studio Press">
    <tipo>
        <revista>
         <fecha_publicacion mes="abr" año="2009"></fecha_publicacion>
        </revista>
```

```
    </tipo>
    <autor nombre="Varios"></autor>
    <prestado lector="Pedro Picapiedra">
      <fecha_pres dia="12" mes="ene" año="2010"></fecha_pres>
    </prestado>
  </ejemplar>

</biblioteca>
```

2.5 Identificación de ámbito de aplicación.

- **El uso principal de XML es estructurar datos, recibirlos y/o enviarlos, pero también podemos usarlo para guardar datos. Si por ejemplo tenemos una pequeña web donde diariamente ponemos algún artículo o nota pero no queremos utilizar bases de datos MySQL, XML puede reemplazar a MySQL y además de esto, XML es mucho más simple que usar MySQL.**

- **Otro uso del XML es la sindicación de noticias. XML/RSS/RDF son términos bastante conocidos por los famosos "Bloggers", personas que tienen su propio Weblog. Estas tecnologías tienen como base, documentos XML. La sindicación de noticias es un nuevo servicio que permite obtener información de un documento XML generado automáticamente por un sistema de publicación de un sitio web, en mi Weblog , es decir, si un sitio X permite la sindicación puedo hacer que esas mismas noticias aparezcan en mi Weblog y esta exportación se realiza en XML.**

  Los FeedReaders o lectores de noticias RSS, son programas que permiten agregar la URL de un sitio y sin necesidad de visitar ese sitio, recibir las novedades y las noticias de este.

  http://www.rss.nom.es/ejemplos-contenido-rss/

- **Otro uso del XML puede ser ahorrar recursos. En informática es un tema supremamente importante el ahorro de recursos en servidores. Este ejemplo nos**

va a mostrar cómo es posible ahorrar consultas a bases de datos que se encuentran en un servidor. Supongamos que tenemos una gran base de datos con información de los clientes de la empresa, esta base de datos aloja información que es constantemente actualizada, lo cual genera un consumo alto de recursos. A la información de esos clientes (teléfono, dirección, E-mail, fax, etc.) acceden cerca de unas 400 maquinas, son los vendedores de nuestra empresa quienes necesitan información de cada cliente, pero hay un problema, no todas las aplicaciones que manejan esas maquinas están construidas en el mismo lenguaje, unas pueden estar en Visual Basic, otras en Java , otras usan PHP o JSP, así que lo lógico es que cada máquina haga una consulta a la base de datos haciendo una búsqueda entre los registros y sacando el cliente que necesitamos, pero eso nos da como resultado un consumo de recursos muy alto, además constantemente se agregan más y más clientes.

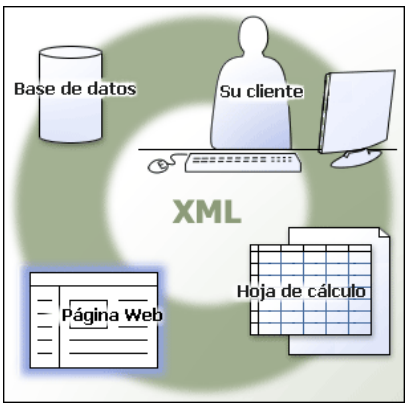
Entonces, aquí puede ser utilizado el XML, como XML es un estándar (W3C), cualquier lenguaje de programación puede trabajar fácilmente con él. Ahora, lo que podríamos hacer es una serie de archivos XML que se obtienen de solo una consulta a la base de datos (con una utilidad llamada XQ) y que alojen los datos de cada cliente en esos archivos (los cuales se actualizaran cada cierto tiempo) y así las maquinas acceden directamente a los archivos XML y no a la base de datos ahorrándonos multitud de consultas a la BD.

- **XML tiene hoy en día gran cantidad de aplicaciones, especialmente en entornos B2B, es decir, comercio electrónico orientado a empresas, donde se usa para describir catálogos y productos ofrecidos en un sitio web.**

- **También se usa en entornos WAP (terminales móviles), donde se usa para generar diferentes contenidos dependiendo de las características del dispositivo al que se sirven. En general, permite separar el contenido de la presentación, presentando la información transformada de diferentes formas dependiendo de quién esté esperándola.**

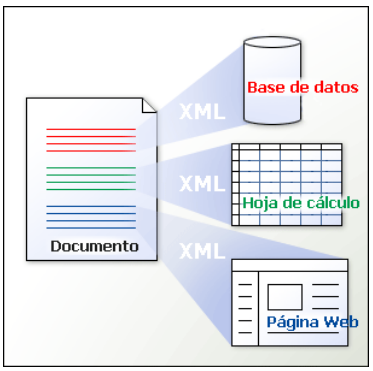
¿Por qué es conveniente usar XML? Porque los negocios de hoy en día dependen de la información y los datos pueden provenir de varios orígenes de información distintos:

bases de datos, páginas Web, hojas de cálculo, correo electrónico, por mencionar sólo algunos. XML le permite trabajar con más datos de más orígenes y hacer más cosas con esos datos. El lenguaje XML es tan popular porque resulta muy útil.



Con XML puede extraer datos de varios orígenes de distintos, almacenar la información en un lugar para saber exactamente dónde buscarla y volver a utilizar los datos cuando y donde los necesite.

Por ejemplo, ¿necesita presentar en un informe los datos presupuestarios de los últimos tres años? Independientemente de cómo guarde los datos, si éstos están en formato XML, puede seleccionar lo que necesita e importarlo en un documento, una hoja de cálculo o una base de datos, según lo requiera la situación.



Otra de las ventajas de XML es su capacidad para automatizar cualquier cantidad de procesos de la organización. Suponga que un cliente le envía un pedido de compra. Lo recibe y tiene el aspecto de cualquier documento, pero la información que contiene se

encuentra en formato XML. Su equipo puede transformar automáticamente ese pedido de compra en una hoja de cálculo para que lo pueda utilizar su departamento de envíos, así como importar automáticamente los datos del pedido en su base de datos de contabilidad. Dependiendo del diseño del sistema, lo único que debe hacer es presionar un par de botones.

XML es la abreviatura en inglés de Lenguaje de marcado extensible, y se denomina "extensible" por una buena razón: puede extenderlo adaptándolo para que se ajuste a casi cualquier necesidad: negocios, publicación, gobierno, ciencia, investigación académica... en cualquier campo en el que la información pase de un uso a otro.

Ejemplo de uso de XML.

Gobierno del Principado de Asturias
Consejería de Educación y Ciencia
Unión Europea
Fondo Social Europeo

Comparación de XML con HTML

Ejemplos de código:

Fichero Ejemplo02.xml

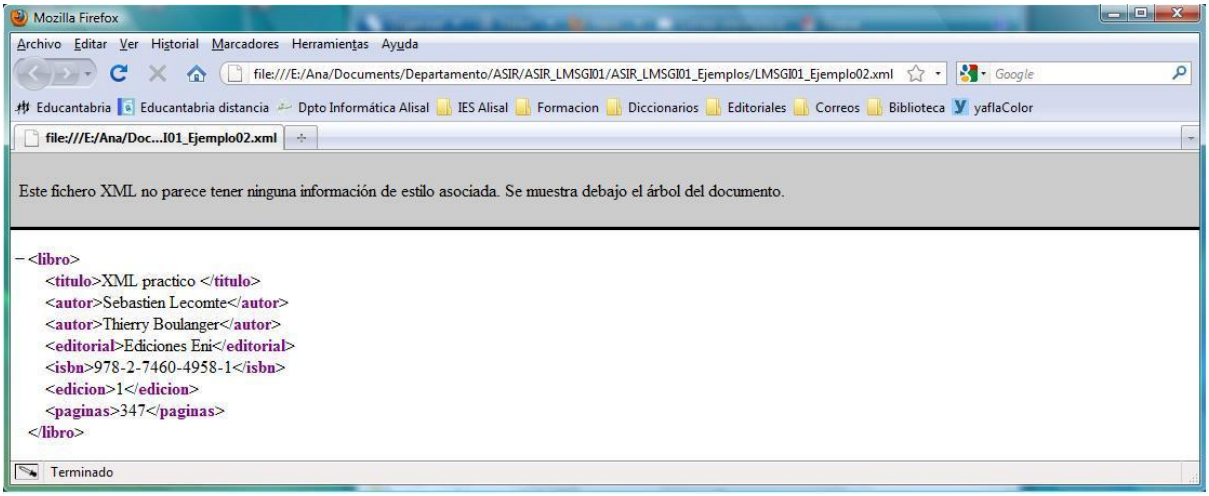
```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE libro>
<libro>
  <titulo>XML practico </titulo>
  <autor>Sebastien Lecomte</autor>
  <autor>Thierry Boulanger</autor>
  <editorial>Ediciones Eni</editorial>
  <isbn>978-2-7460-4958-1</isbn>
  <edicion>1</edicion>
  <paginas>347</paginas>
</libro>
```

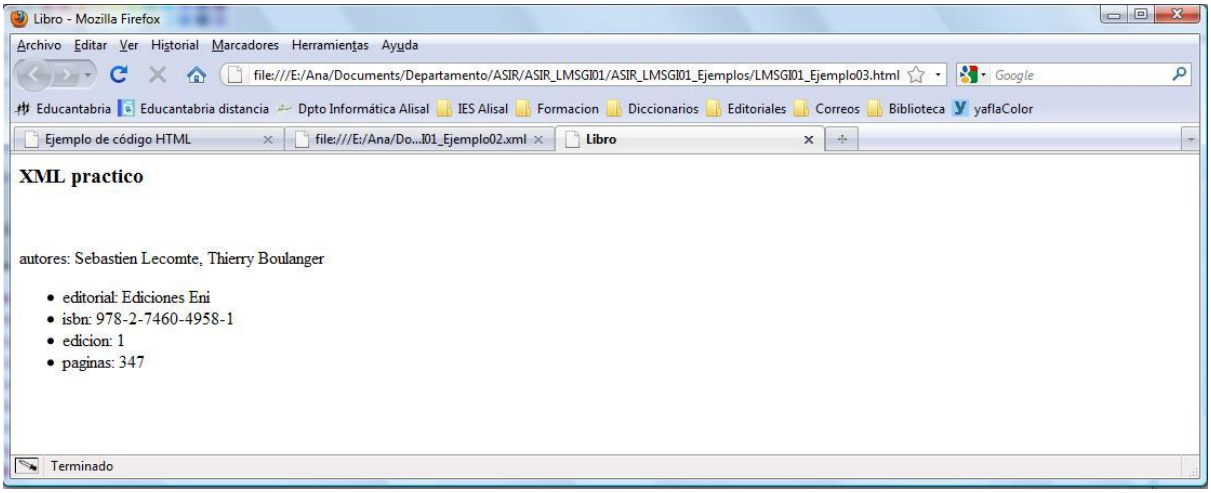
Fichero Ejemplo03.html

```
<html>
  <head>
    <title> Libro</title>
  </head>

  <body>
    <h3>XML practico</h3><br>
      <p> autores: Sebastien Lecomte,
        Thierry Boulanger</p>
      <ul>
        <li>editorial: Ediciones Eni</li>
        <li>isbn: 978-2-7460-49581</li>
        <li>edicion: 1 </li>
       <li>paginas: 347</li>
     </ul>
   </body>
 </html>
```

Al interpretar estos ficheros con un navegador, por ejemplo el Firefox, el resultado es:



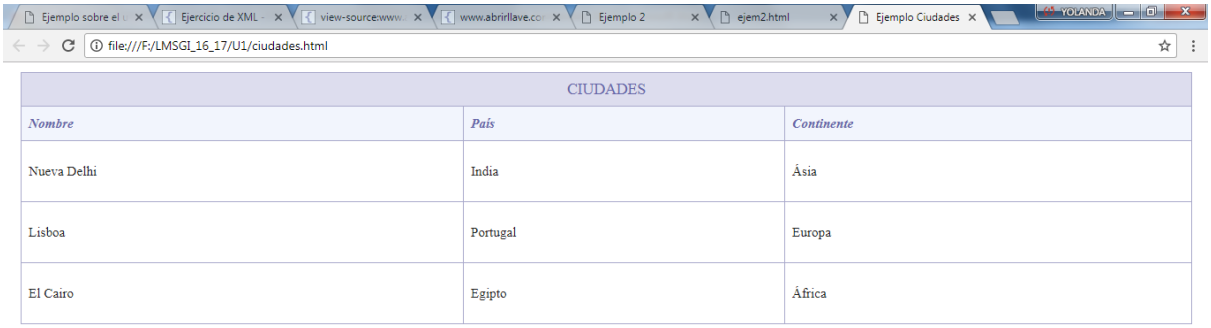


Ejemplo Ciudades:

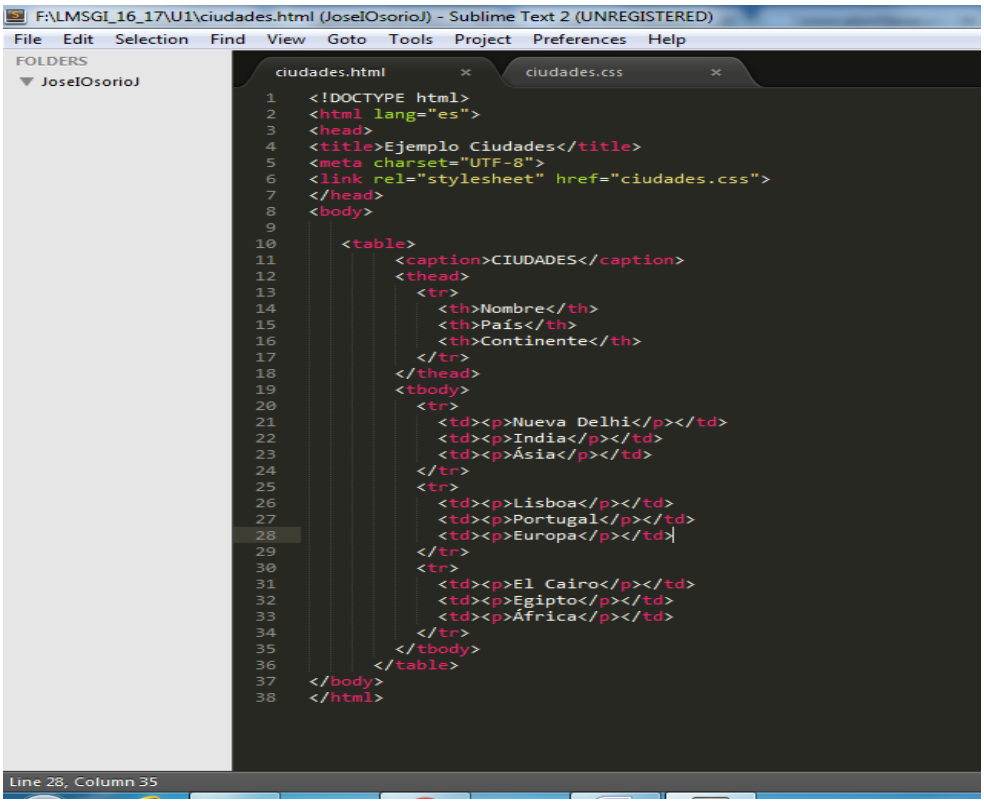
Ciudades.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<ciudades>
  <ciudad>
    <nombre>Nueva Delhi</nombre>
    <pais continente="Ásia">India</pais>
  </ciudad>
  <ciudad>
    <nombre>Lisboa</nombre>
    <pais continente="Europa">Portugal</pais>
  </ciudad>
```

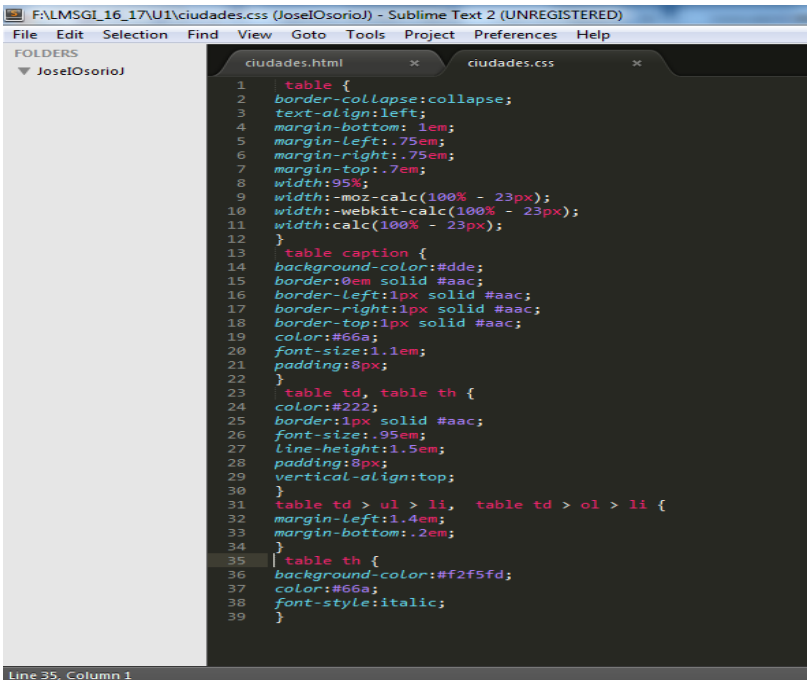
```
  <ciudad>
    <nombre>El Cairo</nombre>
    <pais continente="África">Egipto</pais>
  </ciudad>
</ciudades>
```



ciudades.html



Ciudades.css



3. Etiquetas.

Los lenguajes de marcas utilizan una serie de etiquetas especiales intercaladas en un documento de texto sin formato. Dichas etiquetas serán posteriormente interpretadas por los intérpretes del lenguaje y ayudan al procesado del documento.

Las etiquetas se escriben encerradas entre ángulos, es decir < y >. Normalmente, se utilizan dos etiquetas: una de inicio y otra de fin para indicar que ha terminado el efecto que queríamos presentar. La única diferencia entre ambas es que la de cierre lleva una barra inclinada "/" antes del código.

```
<etiqueta>texto que sufrirá las consecuencias de la etiqueta</etiqueta>
```

Por ejemplo en HTML:

```
<u>Esto está subrayado</u>
```

Al interpretarlo en un navegador se verá así:

<u>Esto está subrayado</u>

Las últimas especificaciones emitidas por el W3C indican la necesidad de que vayan escritas siempre en minúsculas para considerar que el documento está correctamente creado.

4. Herramientas básicas de XML

Para trabajar en XML es necesario editar los documentos y luego procesarlos, por tanto tenemos dos tipos de herramientas:

- **Editores XML**
  Una característica de los lenguajes de marcas es que se basan en la utilización de ficheros de texto plano por lo que basta utilizar un procesador de texto normal y corriente para construir un documento XML.

  Para crear documentos XML complejos e ir añadiendo datos es conveniente usar algún editor XML. Estos nos ayudan a crear estructuras y etiquetas de los elementos

usados en los documentos, además algunos incluyen ayuda para la creación de otros elementos como DTD, hojas de estilo CSS o XSL, ...

- **Procesadores XML**

  Para interpretar el código XML se puede utilizar cualquier navegador. Los procesadores de XML permiten leer los documentos XML y acceder a su contenido y estructura. Un procesador es un conjunto de módulos de software entre los que se encuentra un parser o analizador de XML que comprueba que el documento cumple las normas establecidas para que pueda abrirse. Estas normas pueden corresponderse con las necesarias para trabajar sólo con documentos válidos (documento XML que, además de estar bien formado, verifica las restricciones de otro elemento del que depende para su interpretación DTD o XSD) o sólo exigir que el documento esté bien formado (documento XML que verifica las reglas establecidas por la recomendación de W3C), los primeros se conocen como validadores y los segundos como no validadores. El modo en que los procesadores deben leer los datos XML está descrito en la recomendación de XML establecida por W3C.

  Para publicar un documento XML en Internet se utilizan los procesadores XSLT, que permiten generar archivos HTML a partir de documentos XML.

5. XML: estructura y sintaxis.

El XML, o Lenguaje de Etiquetas Extendido, es lenguaje de etiquetas creadas por el programador, que estructuran y guardan de forma ordenada la información. No representa datos por sí mismo, solamente organiza la estructura.

Un documento XML es un documento de texto, en este caso con extensión ".xml", compuesto de parejas de etiquetas, estructuradas en árbol, que describen la organización del documento, que puede editarse con cualquier editor de texto y que es interpretado por los navegadores Web.

Las características básicas de XML son:

- **Dado que XML se concibió para trabajar en la Web, es directamente compatible con protocolos que ya funcionan, como HTTP (Protocolo de transferencia de hipertexto) y los URL (Uniform Resource Locator, es decir, localizador uniforme de recurso y se refiere a la dirección única que identifica a una página web en Internet).**
- **No se requieren conocimientos de programación para realizar tareas sencillas en XML.**
- **Los documentos XML son fáciles de crear.**
- **La difusión de los documentos XML está asegurada ya que cualquier procesador de XML puede leer un documento de XML.**
- **El marcado de XML es legible para los humanos.**
- **El diseño XML es formal y conciso.**
- **XML es extensible, adaptable y aplicable a una gran variedad de situaciones.**
- **XML es orientado a objetos.**
- **Todo documento XML se compone exclusivamente de datos de marcado y datos carácter entremezclados.**

El proceso de creación de un documento XML pasa por varias etapas en las que el éxito de cada una de ellas se basa en la calidad de la anterior. Estas etapas son:

- **Especificación de requisitos.**
- **Diseño de etiquetas.**
- **Marcado de los documentos.**

El marcado en XML son etiquetas que se añaden a un texto para estructurar el contenido del documento. Esta información extra permite a los ordenadores "interpretar" los textos. El marcado es todo lo que se sitúa entre los caracteres "<" y ">" o "&" y ";"
Los datos carácter son los que forman la verdadera información del documento XML.

Los documentos XML pueden tener comentarios, que no son interpretados por el interprete XML. Estos se incluyen entre las cadenas "<!--" y "-->", pueden estar en cualquier posición en el documento salvo:

- **Antes del prólogo.**
- **Dentro de una etiqueta.**

Los documentos XML pueden estar formados por una parte opcional llamada prólogo y otra parte obligatoria llamada ejemplar.

5.1 El prólogo

Si se incluye, el prólogo debe preceder al ejemplar del documento. Su inclusión facilita el procesado de la información del ejemplar. El prólogo está dividido en dos partes:

- **La declaración XML: en el caso de incluirse ha de ser la primera línea del documento, de no ser así se genera un error que impide que el documento sea procesado.**

  El prólogo puede tener tres funciones:

  – **Declaración la versión de XML usada para elaborar el documento.**
  Para ello se utiliza la etiqueta:

  <?xml versión= "1.0">

  En este caso indica que el documento fue creado para la versión 1.0 de XML.

  – **Declaración de la codificación empleada para representar los caracteres.**
  Determina el conjunto de caracteres que se utiliza en el documento. Para ello se escribe:

<?xml versión= "1.0" encoding="iso-8859-1">

En este caso se usa el código iso-8859-1 (Latin-1) que permite el uso de acentos o caracteres como la ñ.

Los códigos más importantes son:

Estándar ISO	Código de país
UTF-8 (Unicode)	Conjunto de caracteres universal
ISO -8859-1 (Latin-1)	Europa occidental, Latinoamérica
ISO -8859-2 (Latin-2)	Europa central y oriental
ISO -8859-3 (Latin-3)	Sudoeste de Europa
ISO -8859-4 (Latin-4)	Países Escandinavos, Bálticos
ISO -8859-5	Cirílico
ISO -8859-6	Árabe
ISO -8859-7	Griego
ISO -8859-8	Hebreo
ISO -8859-9	Turco
ISO-8859-10	Lapón. Nórdico, esquimal
EUC-JP oder Shitf_JIS	Japonés

- **Declaración de la autonomía del documento.**

  Informa de si el documento necesita de otro para su interpretación. Para declararlo hay que definir el prólogo completo:

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

  El valor "yes" indica que no existen declaraciones de marcas externas al documento. En este caso, el documento es independiente.

<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>

<!DOCTYPE BIBLIOTECA SYSTEM "boe.dtd">

El valor (por defecto) "no" indica que existe o que puede haber dichas declaraciones de marcas.

5.2 El ejemplar. Los elementos

Es la parte más importante de un documento XML, ya que contiene los datos reales del documento. Está formado por elementos anidados.

Los elementos son los distintos bloques de información que permiten definir la estructura de un documento XML. Está, delimitados por una etiqueta de apertura y una etiqueta de cierre. A su vez los elementos pueden estar formados por otros elementos y/o por atributos.

En el ejemplo visto anteriormente:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE libro>
<libro>
  <titulo>XML practico </titulo>
  <autor>Sebastien Lecomte</autor>
  <autor>Thierry Boulanger</autor>
  <editorial>Ediciones Eni</editorial>
  <isbn>978-2-7460-4958-1</isbn>
  <edicion>1</edicion>
  <paginas>347</paginas>
</libro>
```

El ejemplar es el elemento <libro>, que a su vez está compuesto de los elementos <titulo>, <autor>, <editorial>, <isbn>, <edicion> y <paginas>.

En realidad, el ejemplar es el elemento raíz de un documento XML. Todos los datos de un documento XML han de pertenecer a un elemento del mismo.

Los nombres de las etiquetas han de ser autodescriptivos, lo que facilita el trabajo que se hace con ellas.

La formación de elementos ha de cumplir ciertas normas para que queden perfectamente definidos y que el documento XML al que pertenecen pueda ser interpretado por los procesadores XML sin generar ningún error fatal. Dichas reglas son:

- **En todo documento XML debe existir un elemento raíz, y sólo uno.**

- **Todos los elementos tienen una etiqueta de inicio y otra de cierre. En el caso de que en el documento existan elementos vacíos, se pueden sustituir las etiquetas de inicio y cierre por una de elemento vacío. Ésta se construye como la etiqueta de inicio, pero sustituyendo el carácter ">" por "/>. Es decir, <elemento></elemento> puede sustituirse por: <elemento/>**

- **Al anidar elementos hay que tener en cuenta que no puede cerrarse un elemento que contenga algún otro elemento que aún no se haya cerrado.**

- **Los nombres de las etiquetas de inicio y de cierre de un mismo elemento han de ser idénticos, respetando las mayúsculas y minúsculas. Pueden ser cualquier cadena alfanumérica que no contenga espacios y no comience ni por el carácter dos puntos, ":", ni por la cadena "xml" ni ninguna de sus versiones en que se cambien mayúsculas y minúsculas ("XML", "XmL", "xML",...).**

- **El contenido de los elementos no puede contener la cadena "]]>" por compatibilidad con SGML. Además no se pueden utilizar directamente los caracteres mayor que, >, menor que, <, ampersand, &, dobles comillas, ", y apostrofe, '. En el caso de tener que utilizar estos caracteres se sustituyen por las siguientes cadenas:**

Carácter	Cadena
>	>
<	<

Carácter	Cadena
&	&
"	"

Carácter	Cadena
'	'

- **Para utilizar caracteres especiales, como £, ©, ®,... hay que usar las expresiones &#D; o &#H; donde D y H se corresponden respectivamente con el número decimal o hexadecimal correspondiente al carácter que se quiere representar en el código UNICODE. Por ejemplo, para incluir el carácter de Euro, €, se usarían las cadenas € o €**

5.3 Atributos

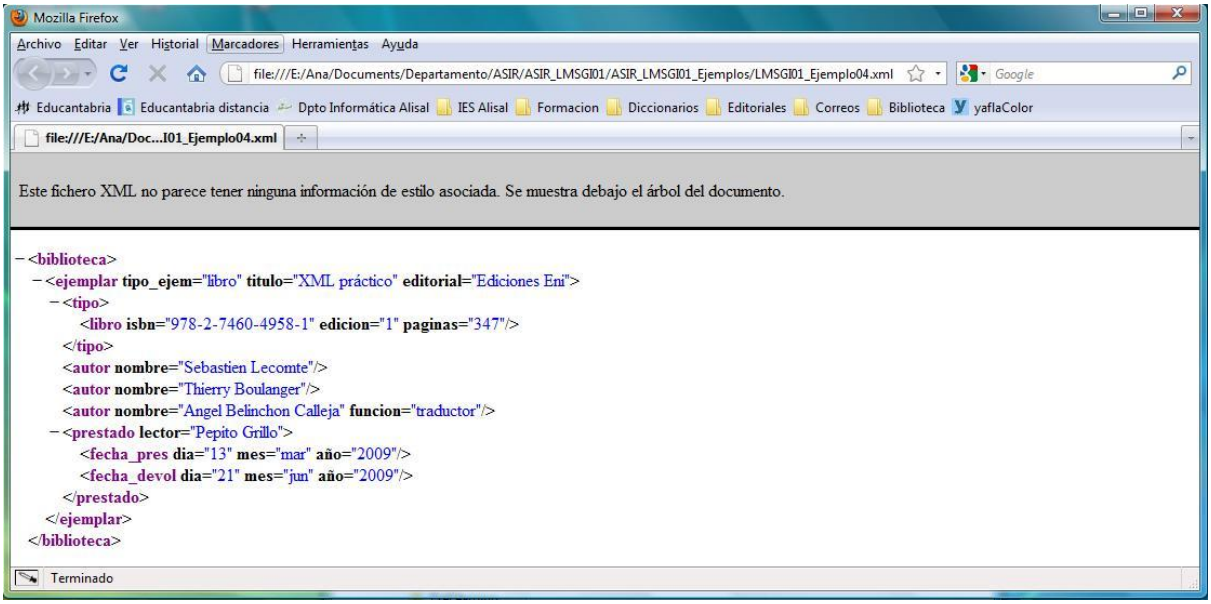
Son complementos de información que se pueden asignar a un elemento previamente declarado. Permiten añadir propiedades a los elementos de un documento. Los atributos no pueden organizarse en ninguna jerarquía, no pueden contener ningún otro elemento o atributo y no reflejan ninguna estructura lógica.
No se debe utilizar un atributo para contener información susceptible de ser dividido.

En el ejemplos siguiente

```
<?xml version="1.0" encoding="iso-8859-1" standalone="yes" ?>
<!DOCTYPE biblioteca >
<biblioteca>
  <ejemplar tipo_ejem="libro" titulo="XML práctico" editorial="Ediciones Eni">
    <tipo> <libro isbn="978-2-7460-4958-1" edicion="1" paginas="347"></libro>
</tipo>
    <autor nombre="Sebastien Lecomte"></autor>
    <autor nombre="Thierry Boulanger"></autor>
    <autor nombre="Angel Belinchon Calleja" funcion="traductor"></autor>
    <prestado lector="Pepito Grillo">
      <fecha_pres dia="13" mes="mar" año="2009"></fecha_pres>
      <fecha_devol dia="21" mes="jun" año="2009"></fecha_devol>
    </prestado>
```

```
    </ejemplar>
</biblioteca>
```

Al abrir el documento anterior con el navegador Firefox obtenemos:



Vemos que los elementos aparecen coloreados en ciruela, los nombres de los atributos en negro y sus valores en azul.

Como se observa en el ejemplo, los atributos se definen y dan valor dentro de una etiqueta de inicio o de elemento vacio, a continuación del nombre del elemento o de la definición de otro atributo siempre separado de ellos por un espacio. Los valores del atributo van precedidos de un igual que sigue al nombre del mismo y tienen que definirse entre comillas simples o dobles.

Los nombres de los atributos han de cumplir las mismas reglas que los de los elementos, y no pueden contener el carácter menor que, <.

6. Documentos XML bien formados

Todos los documentos XML deben verificar las reglas sintácticas que define la recomendación del W3C para el estándar XML. Esas normas básicas son:

- **El documento ha de tener definido un prólogo con la declaración xml completa.**
- **Existe un único elemento raíz para cada documento: es un solo elemento en el que todos los demás elementos y contenidos se encuentran anidados.**
- **Hay que cumplir las reglas sintácticas del lenguaje XML para definir los distintos elementos y atributos del documento**

7. Utilización de espacios de nombres en XML

Permiten definir la pertenencia de los elementos y los atributos de un documento XML al contexto de un vocabulario XML. De este modo se resuelven las ambigüedades que se pueden producir al juntar dos documentos distintos, de dos autores diferentes, que han utilizado el mismo nombre de etiqueta para representar cosas distintas.

Los espacios de nombres también conocidos como name spaces, permiten dar un nombre único a cada elemento, indexándolos según el nombre del vocabulario adecuado, además están asociados a un URI (Uniform Resource Identifier, cadena corta de caracteres que identifica inequívocamente un recurso (servicio, página, documento, dirección de correo electrónico ...),) que los identifica de forma única. Los espacios de nombres son un prefijo que ponemos a los elementos xml para indicar a qué contexto se refiere el elemento en cuestión.

La construccion de estos nombres extendidos se realiza uniendo el nombre del espacio (prefijo) y el nombre del elemento o atributo con el símbolo ":", esto es:

<prefijo:nombre_etiqueta></prefijo:nombre_etiqueta>

Esta etiqueta se denomina "nombre cualificado". Al definir el prefijo hay que tener en cuenta que no se pueden utilizar espacios ni caracteres espaciales y que no puede comenzar por un digito.

Para usar un espacio de nombres en un documento, debemos declararlo previamente. La declaración consta de unos atributos xmlns (XML namespace), donde proporcionamos el prefijo que usaremos para el espacio de nombres y una URI (Uniform Resource Identifier) que será un identificador único del espacio de nombres. Su sintaxis es la siguiente:

<conexion>://<direccionservidor>/<apartado1>/<apartado2>/...

Ejemplo:

Una empresa utiliza tres documentos xml distintos para recoger información. Dicha información deben de reunirse en un único documento llamado empresa.xml:

Clientes.xml

```
<?xml version="1.0" encoding="iso-8859-1"?>
<clientes>
 <cliente>
  <direccion>Calle Real</direccion>
          …
 </cliente>
 <cliente>
  <direccion> Calle Maria Alonso</direccion>
          …
 </clienter>
</clientes>
```

Atencion.xml

```
<?xml version="1.0" encoding="iso-8859-1"?>
<atencion>
 <correo>
  <direccion> ventas@cliente.com </direccion>
          …
 </correo>
 <correo>
  <direccion> compras@prov.com </direccion>
 </correo>
</clientes>
```

Redes.xml

```
<?xml version="1.0" encoding="iso-8859-1"?>
<redes>
 <red>
  <direccion> 192.168.168.192 </direccion>
          …
 </red>
 <red>
  <direccion>  191.169.168.105</direccion>
 </red>
</redes>
```

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

        <empresa
         <datos>
                        ...
                        <direccion>Calle Real</direccion>
                        ...
                        <direccion>ventas@cliente.com</direccion>
                        ...
                        <direccion>192.168.168.192</direccion>
                        ...
         </datos>
        </empresa>
```

Utilizando espacios de nombres:

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

        <empresa
        xmlns:comer="http://www.empresa.com/comercial"
        xmlns:aten="http://www.empresa.com/atencion"
        xmlns:red="http://www.empresa.com/red">
         <datos>
                        ...
                        <comer:direccion>Calle Real</comer:direccion>
                        ...
                        <aten:direccion>ventas@cliente.com</aten:direccion>
                        ...
                        <red:direccion>192.168.168.192</red:direccion>
                        ...
         </datos>
        </empresa>
```

Ejemplo:

Sean los documentos XML que organizan la información sobre los profesores y los alumnos del ciclo INF respectivamente:

```
<?xml version="1.0" encoding="iso-8859-1" standalone="yes" ?>
<!DOCTYPE alumnos>
<alumnos>
  <nombre>Fernando Fernández González</nombre>
  <nombre>Isabel González Fernández</nombre>
  <nombre>Ricardo Martínez López</nombre>
</alumnos>
```

```
<?xml version="1.0" encoding="iso-8859-1" standalone="yes" ?>
<!DOCTYPE profesores>
<profesores>
  <nombre>Pilar Ruiz Pérez</nombre>
  <nombre>Tomás Rodríguez Hernández</nombre>
</profesores>
```

Al hacer un documento sobre los miembros del ciclo INF no se distinguirían los profesores de los alumnos, para resolverlo definiremos un espacio de nombres en el elemento raíz del documento de la siguiente forma:

```
<?xml version="1.0" encoding="iso-8859-1" standalone="yes" ?>
<!DOCTYPE miembros>

<miembros xmlns:alumnos="http://INF/alumnos"
xmlns:profesores="http://INF/profesores">
    <alumnos:nombre>Fernando Fernández González</alumnos:nombre>
    <alumnos:nombre>Isabel González Fernández</alumnos:nombre>
```

```
    <alumnos:nombre>Ricardo Martínez López</alumnos:nombre>
    <profesores:nombre>Pilar Ruiz Pérez</profesores:nombre>
    <profesores:nombre>Tomás Rodríguez Hernández</profesores:nombre>
</miembros>
```

También se puede definir un espacio de nombres para cada contexto:

```
<?xml version="1.0" encoding="iso-8859-1" standalone="yes" ?>
<!DOCTYPE miembros>
<miembros>
<alumnos xmlns:alumnos="http://INF/alumnos">
    <alumnos:nombre>Fernando Fernández González</alumnos:nombre>
    <alumnos:nombre>Isabel González Fernández</alumnos:nombre>
    <alumnos:nombre>Ricardo Martínez López</alumnos:nombre>
</alumnos>

<profesores xmlns:profesores="http://INF/profesores">
    <profesores:nombre>Pilar Ruiz Pérez</profesores:nombre>
    <profesores:nombre>Tomás Rodríguez Hernández</profesores:nombre>
</profesores>
</miembros>
```

Recursos recomendados:

http://www.abrirllave.com/xml/apuntes-de-xml.php