CURSO 2022-2023
CICLO SUPERIOR DE DESARROLLO DE APLICACIONES WEB
IES AGUADULCE

Lenguajes de Marcas y Sistemas de Gestión de la Información

Francisco Javier Sueza Rodríguez

6 de octubre de 2022

Índice general

Índice de figuras

Tema 1

Lenguajes de Marcas y Sistemas de Gestión de la Información

En esta unidad vamos a estudiar los aspectos básicos de los lenguajes de marcas y los sistemas de gestión de la información. Por un lado, veremos la evolución de los **lenguajes de marcas**, desde GML hasta HTML, así como sus elementos y atributos, haciendo especial énfasis en XML. A continuación, veremos en que consisten los **sistemas de gestión de la información**, en concreto los **ERP**, sus características, configuración básica, personalización,..etc.

1.1 Definición y Clasificación de los Lenguajes de Marcas

Los «lenguajes de marcas» sirven para **codificar un documentos**. Estos incorporan **etiquetas** o marcas con **información adicional** sobre como se estructura el texto o como se presenta. El lenguaje de marcas será el que defina que etiquetas se permiten, donde deben colocarse y que significado tienen.

Todo lenguaje de marcas esta definido en un documento denominado **DTD**, donde se establecen las marcas, los elementos utilizados por dicho lenguaje y sus correspondientes etiquetas y atributos, así como su sintaxis.

Los lenguajes de marcas se pueden clasificar, principalmente, en tres grupos:

- **Orientados a la presentación**: son los utilizados generalmente por los procesadores de texto y definen como debe presentarse el documento, es decir, el formato que tiene.
- **De procedimientos**: orientados también a la presentación, pero en este caso, dentro de un **marco procedural** que permite la definición de macros, es decir, el programa que representa el documento debe interpretar el código en el mismo orden que aparece. Algunos ejemplos son **TeX**, **LaTeX** y **Postscript**
- **Descriptivos o semánticos**: estos lenguajes no describen la presentación del documento, sino que **describen la información**, que es lo que se esta representando sin especificar como debe presentarse.

Algunos ejemplos de lenguajes de marcado agrupados por su ámbito de uso son los siguientes:

- **Documentación Electrónica**:
    - **RTF** (Rich Text Format): fue desarrollado por Microsoft en 1987 y permite el intercambio de documentos entre los diferentes procesador de texto.

- **TeX**: creado por Donald Knuth, este lenguaje esta especialmente enfocado en la creación de textos científicos. Es considerado la mejor forma de componer formulas matemáticas complejas. [1]
- **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, permitiendo que estos documentos puedan publicarse en diferentes formatos sin tener que modificar el documento original.

- **Tecnologías de Internet**:
  - **HTML**,**XHTML** (Hypertext Markup Language, eXtensible Hypertext Markup Language): estos lenguajes están orientados a la creación de páginas web.
  - **RSS** (Really Simple Sindication): permite la difusión de contenido web mediante la sindicación de contenidos.
- Otros lenguajes especializados:
  - **MathML** (Mathematica Markup Language): especializado en expresar los formalismos matemáticos de forma que puedan ser entendidos por diferentes aplicaciones.
  - **VoiceXML** (Voice eXtended Markup Language): permite el intercambio de información entre usuarios y una aplicación con capacidad de reconocer el habla.
  - **MusicXML**: permite el intercambio de partituras entre diferentes editores de partituras.

1.2 Evolución de los Lenguajes de Marcas

A finales de los **años 60** surgen unos lenguajes informáticos, diferentes de los lenguajes de programación, orientados a la gestión de la información. Con el desarrollo de los editores y procesadores de texto surgen los primeros lenguajes informáticos orientados a la descripción y estructuración de la información: **los lenguajes de marcas**. Paralelamente también surgen otros lenguajes orientados a la representación, almacenamiento y consultar de grandes cantidades de datos: lenguajes y sistemas de bases de datos.

Los lenguajes de marcas surgieron inicialmente como lenguajes formados por un conjunto de códigos que los procesadores de textos insertaban en los documentos para dirigir el proceso de presentación (impresión) mediante una impresora. Al igual que los lenguajes de programación, estos estaban **ligados** a las características de los **procesadores de texto**y las **impresoras** en los que se usaban y no permitían a los programadores abstraerse de dichas características.

Posteriormente se añadió como medio de presentación a la pantalla y se automatizó el proceso, teniendo ya solo que pulsar una combinación teclas para lograr los resultados deseados en vez de hacerlo a mano. Este marcado estaba orientado exclusivamente a la presentación de la información, aunque posteriormente se le dieron nuevos uso surgiendo con ello el **formato generalizado**.

1.2.1 El origen: GML y SGML

Uno de los problemas que ha tenido la informática ha sido la **falta de estandarización** en los formatos de información usados por los diferentes programas.

En los años 60, **IBM** encargo a **Charles F. Goldfarb** la construcción de un sistema de edición, almacenamiento y búsqueda de documentos legales. Después de analizar el funcionamiento de la empresa se

llego a la conclusión de que necesitaban un formato estándar a todos los departamentos para gestionar la documentación.

Así fue como se creó **GML**, un formato que permitía describir los documentos de tal forma que el resultado fuese independiente de la plataforma o la aplicación utilizada. Este formato evolucionó hasta que en 1986 se creó el estándar **ISO 8879** donde se especificaba el formato **SGML**, un lenguaje muy complejo y que requería de unas herramientas de software caras, por lo que su uso quedó relegado a grandes aplicaciones industriales.

```
<email>
    <remitente>
        <nombre>Peter</nombre>
        <apellido>Pan</apellido>
    </remitente>
    <destinatario>
        <direccion>campanilla@paisdenuncajamas.com</direccion>
    </destinatario>
    <asunto>Paseo</asunto>
    <mensaje>¿Te apetece dar una vuelta?</mensaje>
</email>
```

Figura 1.2.1: Documento SGML simple

1.2.2 La Popularización: HTML

En 1990, **Tim Berners-Lee** creó el World Wide Web y conociendo SGML, se encontró con la necesidad de compatibilizar, enlazar y organizar gran cantidad de documentos procedentes de diversos sistemas. Como solución, a partir de la sintaxis de SGML, creó un lenguaje de descripción de documentos llamado **HTML**, combinando dos estándares ya existentes:

- **ASCII**: código basado en el alfabeto latino, tal como se usa en inglés moderno [2]. Cualquier procesador de textos simple puede reconocer y almacenar este formato, permitiendo la transferencia de datos entre dos ordenadores.
- **SGML**: lenguaje que permite dar estructura al texto aplicando diferentes formatos.

HTML es una **versión simplificada de SGML**, ya que solo utiliza las instrucciones absolutamente necesarias. Gracias a su simplicidad, tuvo un éxito rotundo en la World Wide Web, convirtiéndose rápidamente en el **estándar general** para la **creación de páginas web**. Actualmente, HTML es el tipo de documento más utilizado en el mundo.

```
<!DOCTYPE HTML>
<html>
    <head>
        <meta charset="utf-8" />
        <title>Ejemplo1</title>
    </head>
    <body>
        <p>Párrafo de ejemplo</p>
    </body>
</html>
```

Figura 1.2.2: Documento HTML simple

1.2.3 La Madurez: XML

Uno de los problemas que surgió con HTML es que la cantidad de documentos escritos en este lenguaje creció exponencialmente, muchos de los cuales no se ceñían a ningún estándar generando bastante caos. Como respuesta es ese problema, el **W3C** estableció en 1998 el estándar internacional **XML**, un lenguaje de marcas puramente estructural, que **no incluye información sobre el diseño**, y permite la creación de etiquetas adaptadas a las necesidades, convirtiéndose con rapidez en el estándar para intercambio de datos en la web.

XML es un **metalenguaje** con las siguientes características:

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

En realizad XML es un **conjunto de estándares** relacionados entre sí y que comprende los siguientes:

- **XLS** (eXtensible Style Language): permite definir hojas de estilo para XML e incluye capacidad de transformación de documentos.
- **XML Linking Language**: determina aspectos sobre los enlaces entre documentos XML e incluye **Xpath**, **Xlink** y **Xpointer**.
- **XML Namespaces**: proveen de un contexto donde se aplican las marcas del documento XML y que se diferencian de otras con el mismo nombre válidas en otros contextos.
- **XML Schemas**: permiten definir restricciones que se aplicarán a un documento XML. Actualmente las mas utilizadas son **DTD**.

```
<?xml version="1.0" encoding="UTF-8x°x"?>
<!DOCTYPE biblioteca>

<biblioteca>
    <ejemplar tipo="libro" isbn="978-2-7460-4958-1" edicion="1">
        <titulo>XML practico</titulo>
        <editorial>Ediciones Eni</editorial>
        <autor>Sebastien Lecomte</autor>
        <autor>Thierry Boulanger</autor>
        <autor funcion="traductor">Ángel Belinchon Calleja</autor>
        <prestamos>
            <lector inicio="13/05/2014" devolucion="15/05/2014">Pedro López</lector>
            <lector inicio="13/07/2015" devolucion="15/07/2015">Ali Méndez</lector>
        </prestamos>
     </ejemplar>
</biblioteca>
```

Figura 1.2.3: Documento XML simple

1.2.4 Comparación XML y SGML

Aunque XML esta basado en SGML, estos tienen muchas diferencias. A continuación se muestra una tabla con las principales diferencias de estos dos lenguajes de marcas.

XML	SGML
Su uso es sencillo	Su uso es my complejo
Trabaja con documentos bien formados	Solo Trabaja con documentos válidos
Desarrollo de aplicaciones a bajo coste	Aplicaciones para procesarlo costosas
Muy utilizado en informática y otras áreas	Se utiliza en sectores muy específicos
Compatibilidad e integración con HTML	No hay compatibilidad con HTML
Formateo y estilos fáciles de aplicar	Formateo y estilo relativamente complejos

Como vemos en esta tabla, SGML es un un lenguaje mas complejo y costoso que XML, además de imponer mas restricciones, haciéndolo un lenguaje menos flexible que XML y mas orientado a sectores concretos. Para obtener más información sobre XML, podemos consultar el Estándar XML publicado por la W3C.

1.2.5 Comparación XML y HTML

Aunque tanto XML como HTML se crearon a partir de SGML y su sintaxis es similar, son lenguajes diferentes con propósitos diferentes, como podemos ver en la siguiente tabla.

XML	HTML
Es un perfil de SGML	Es una aplicación de SGML
Permite definir conjuntos de etiquetas	Aplica un conjunto limitado de etiquetas
Modelo de hiperenlaces complejo	Modelo de hiperenlaces simple
Navegador como plataforma de desarrollo	Navegador como visor de páginas
Compatibilidad e integración con HTML	No hay compatibilidad con HTML
Fin de las etiquetas propietarias	Problema de la no compatibilidad en navegadores

1.3 Etiquetas, Elementos y Atributos

Los lenguajes de marcas usan una serie de etiquetas intercaladas en un documento sin formato, las cuales serán posteriormente por el interprete del lenguaje.

Existen tres términos ampliamente utilizados en los lenguajes de marcas:

- **Etiquetas**: una etiqueta, también llamada **tag**, es un pequeño bloque de código que se escribe encerrado entre los símbolos **menor que** (**<**) y **mayor que** (**>**). Normalmente se utilizan **dos etiquetas**, una de **inicio** y otra de **fin**, para indicar que el efecto que queríamos conseguir ha finalizado, con la única diferencia que a la etiqueta de fin se le añade el carácter **/** antes del nombre.
- **Elemento**: representan estructuras mediante las que se organiza el contenido del documento, o acciones que se desencadenan cuando el navegador lo interpreta. Está **compuesto** de la **etiqueta de apertura**, la **etiqueta de cierre** y el **contenido entre ambas**.

- **Atributo**: es un par **nombre-valor**, que se encuentra al inicio de un elemento e indican diferentes propiedades asociadas a ese elemento

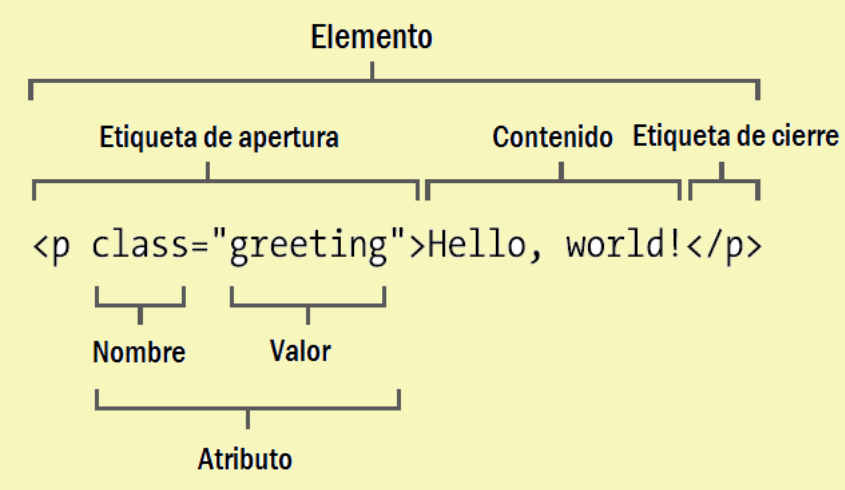

Figura 1.3.1: Partes de un elemento HTML

1.4 Herramientas de Edición

Para trabajar con XML es necesario, por un lado, editar los documentos, y por otro procesarlos. Por ello, necesitaremos dos tipos de herramientas para trabajar con él, estas son:

- **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 con usar cualquier editor de texto para construir un documento XML.

  Aunque podemos usar cualquier editor, cuando elaboramos documentos XML complejos es conveniente usar algún software de edición XML. Estos nos ayudan a crear estructuras y etiquetas de los elementos usados en XML, resaltan las etiquetas para diferenciarlas más cómodamente y ademas incluyen ayudas para la creación de otros elementos como DTD, hojas de estilo CSS o XLS,.. El W3C ha desarrollado un editor HTML, XHTML, CSS y XML gratuito llamado Amaya, pero existen otros muchos gratuitos como: Notepad++, VSCode, Sublime Text, Netbeans,..etc

- **Procesadores XML**

  Para interpretar un documento XML puede usarse cualquier navegador. Los procesadores XML permiten visualizar 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**, que comprueba que el documento cumple las normas establecidas para que pueda abrirse. Estás normas pueden corresponderse a las necesarias para trabajar con documentos de tipo válido o solo exigir que el documento este bien formado. A los primeros se le conocen como **procesadores validadores** y a los segundos como **procesadores no validadores**.

  Para publicar documentos XML en internet se usan **procesadores XSLT**, que permiten generar archivos HTML a partir de XML.

  Puesto que XML se usa para el intercambio de archivos entre aplicaciones, hay que recurrir a motores independientes como «XML para Java» de IBM, JAXP de Sun, etc

1.5 XML

XML, que significa *eXtensible Markup Language*, es un lenguaje que permite definir lenguajes de marcas desarrollado por el W3C y utilizado para almacenar datos de forma legible. Proviene del lenguaje SGML y permite definir la gramática de lenguajes específicos para estructurar documentos. [3]

Su importancia radica en que permite **compartir datos** entre diferentes equipos y aplicaciones de forma **segura**, **fiable** y **sencilla**. El hecho de que diferentes equipos y aplicaciones puedan leer y generar archivos en este formato lo convierte en una herramienta muy útil para el envío de información a través de la Web. Aunque a veces suele confundirse con HTML, podemos decir que HTML esta diseñado para mostrar datos en nuestras pantallas mientras que XML esta diseñado para almacenar y compartir esos datos.

El XML ahorra tiempos de desarrollo y proporciona ventajas, dotando a webs y aplicaciones de un método muy potente para almacenar y compartir información. Por ello, se ha convertido en un formato universal usado por todo tipo de sistemas operativos y dispositivos.

1.5.1 Estructura y Sintaxis

Un documento XML es un documento de de texto, con la **extensión .xml**, compuesto de un **conjunto de etiquetas**, **estructuradas en árbol**, que describen la organización del documento y que es interpretado por un navegador Web.

La **características básicas** de XML son las siguientes:

- Es directamente **compatible** con protocolos usados en en la Web como **HTTP** y **URL**.
- Todo documento que verifique las reglas de XML esta **conforme con SGML**.
- **No se requieren conocimientos de programación** para realizar tareas sencillas en XML.
- Los documentos XML son **fáciles de crear**.
- **La difusión** de documentos XML **está asegurada**, ya que cualquier procesador de XML puede leer documentos XML.
- El marcado de XML es **legible para los humanos**.
- El diseño de 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** de **datos de marcado** y **datos carácter** entremezclados.

El **proceso de creación** de un documento 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:

1. Especificación de requisitos.
2. Diseño de etiquetas.
3. Marcado de los documentos.

El **marcado** son etiquetas que se añaden para estructurar los documentos y permitir a los ordenadores interpretar los textos. Los **datos carácter** son los que componen la verdad información del documento.

Los documentos XML también pueden **contener comentarios**, que son interpretados por el intérprete XML y que comienzan con la cadena <!– y finalizan con –>. Pueden estar en cualquier posición del documento salvo en:

- El prólogo
- Dentro de una etiqueta

Los XML están **formados** por dos partes, **el prólogo** y **el ejemplar**, los cuales veremos a continuación.

1.5.2 El Prólogo

El prólogo es la primera parte que nos encontramos en cualquier documento XML y siempre debe preceder al ejemplar del documento. Éste se divide en dos partes, la **declaración XML** y la **declaración de la codificación** empleada, los cuales explicamos a continuación.

1. **La declaración XML**: es la primera linea del documento, de no ser así, se genera un error que impide que el documento sea procesado. En caso de que sea opcional, se permite el procesamiento de documentos HTML y SGML como si fueran documentos XML, si es obligatoria, estos deberán incluir la declaración de la versión XML. Esta declaración permite indicar de forma explicita que el documento es de tipo XML.

   El prologó puede tener **tres funciones**:

   a) **Declaración de la versión** utilizada de XML:

   ```
   <?xml version= "1.0"?>
   ```

   b) **Declaración de la codificación empleada**:

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

   En este caso se usa el código código iso-8859-1 (Latin-1) que permite el uso de tildes o caracteres como la «ñ». Otro de los códigos a tener en cuenta es **UTF-8** (**unicode**). Esta codificación soporta más caracteres que iso-8859-1 y permite que estos se visualicen correctamente en más sistemas. Es la **codificación estándar** recomendada para todos los documentos y la usaremos siempre, salvo que por algún motivo no pueda ser empleada.

   Para consultar más información sobre codificación de caracteres en sistemas informáticos y cuales son los más empleados podemos visitar página de Wikipedia sobre codificación de caracteres.

   c) **Declaración de autonomía del documento**:

   Indica 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" ?>
   ```

   La opción *standalone* indica al procesador XML si un documento es independiente (*standalone=ɛes"*), o si depende de un documento externo, como declaraciones de marcas externas o una DTD externa (*standalone="no"*). Por defecto el documento se considera independiente.

2. **La declaración del tipo de documento**:

   Esta declaración define el tipo de documento que estamos creando para que sea procesado correctamente. Toda declaración de tipo de documento comienza con la cadena:

```
<!DOCTYPE Nombre_tipo ...>
```

1.5.3 El Ejemplar

Es la parte más importante de un documento XML ya que contiene los **datos reales** del documento. Es el **elemento raíz** de un documento XML y este se nombrará igual que la declaración del tipo de documento (*!DOCTYPE*). Suele estar compuesto de **elementos anidados**.

En el siguiente ejemplo, el ejemplar es el elemento <*libro*>, que a su vez esta compuesto de los elementos <*titulo*>, <*autoria*>, <*editorial*>, <*isbn*>, <*edicion*> y <*paginas*>.

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!DOCTYPE libro>

<libro>
  <titulo>XML práctico</titulo>
  <autoria>
      <autor>Sebastien Lecomte</autor>
      <autor>Thierry Boulanger</autor>
  </autoria>
  <editorial>Ediciones Eni</editorial>
  <isbn>978-2-7460-4958-1</isbn>
  <edicion>1</edicion>
  <paginas>347</paginas>
</libro>
```

Figura 1.5.1: Ejemplo del *ejemplar* en un documento XML

Es recomendable **establecer un criterio** y mantenerlo durante todo el documento, por ejemplo, que las etiquetas vayan escritas **siempre en minúsculas**.

Por otro lado, es conveniente anidar los elementos para una visualización óptima del documento, esto se hará **indentando** o **tabulando** el código. A continuación se muestran dos figuras, una sin indentación (errónea) y otra con indentación (correcta).

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!DOCTYPE libro>

<libro>
<titulo>XML práctico</titulo>
<autoria>
<autor>Sebastien Lecomte</autor>
<autor>Thierry Boulanger</autor>
</autoria>
<editorial>Ediciones Eni</editorial>
<isbn>978-2-7460-4958-1</isbn>
</libro>
```

Figura 1.5.2: Código XML sin indentación (incorrecto)

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<!DOCTYPE libro>

<libro>
    <titulo>XML práctico</titulo>
    <autoria>
        <autor>Sebastien Lecomte</autor>
        <autor>Thierry Boulanger</autor>
    </autoria>
    <editorial>Ediciones Eni</editorial>
    <isbn>978-2-7460-4958-1</isbn>
</libro>
```

Figura 1.5.3: Código XML con indentación (correcto)

Como podemos observar, en la segunda figura se reconoce más fácilmente la estructura del documento y se facilita la lectura de este.

Por último, es recomendable **anidar grupos de datos relacionados**, como se ha hecho en el ejemplo anterior con los elementos <*autor*> dentro del elemento <*autoria*, ya que el documento que **mas limpio** y **ordenado**.

1.5.3.1 Elementos

Todos los datos de un documento XML deben **pertenecer** a un **elemento del mismo**.

Los elementos son los diferentes **bloques de información** que permiten definir la **estructura** del documento XML. Estos está delimitados por una **etiqueta de apertura** y una **etiqueta de cierre**. Pueden estar **compuestos** por **otros elementos**.

Los **nombres** de las etiquetas deben ser **autodescriptivos**, es decir, que ilustren su contenido. Por ejemplo, si estamos con datos relativo a un libro, una etiqueta no debería ser <*caracteristicas*>, ya que es demasiado ambiguo, deberíamos utilizar nombres como <*titulo*>, <*isbn*, etc...

La **formación de elementos** debe cumplir ciertas **normas** para que queden bien definidos y que el documento XML al que pertenecen pueda ser procesado sin generar ningún error fatal. Estas normas son las siguientes:

- En todo documento XML debe existir **un documento raíz** y **solo uno**.
- **Todos** los elementos tienen una **etiqueta de inicio** y **otra de cierre**. En caso de que en el documento existan **elementos vacíos**, se pueden sustituir las etiquetas de apertura y cierre por una de elemento vacío. Esta se construye como una etiqueta de inicio pero añadiendo **/**, es decir, <*elemento/*> en vez de <*elemento*>. Esta considerada una etiqueta de apertura y cierre.
- Al anidar hay que tener en cuenta que **no puede cerrarse** ningún **elemento** que **contenga otro elemento** que aún **no se haya cerrado**.
- Los **nombres** de las **etiquetas de apertura y cierre** deben ser **idénticos**, respetando las mayúsculas y minúsculas. Pueden ser cualquier cadena alfanumérica que no contenga espacios ni tildes, y que no comience por dos puntos (**:**), ni por la cadena **xml**.
- El contenido de los elementos **no puede contener** la cadena «***]]>***», por compatibilidad con SGML. Además, no se pueden utilizar los caracteres **mayor que** (**>**), **menor que** (**<**), **ampersand** (**&**), **dobles comillas** (**"**) y **apostrofe** (**‘**).

En caso de tener que utilizar alguno de estos caracteres, se deben sustituir por las siguientes cadenas:

Caracteres	Cadena
>	>
<	<<
&	&

Caracteres	Cadena
"	"
’	'

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

  En los siguientes enlaces puedes consultar tanto los códigos ASCII y su equivalente en HTML, como el código correspondiente a cada carácter en UNICODE.

  - ASCII - https://www.asciitable.com/
  - UTF-8 - https://www.charset.org/utf-8

1.5.3.2 Atributos

Las etiquetas pueden tener **atributos**, que **permiten definir propiedades** a los elementos de un documento. Los atributos, a diferencia de los elementos, no puede organizarse en ninguna jerarquía o estructura de árbol, no pueden contener ningún otro elemento, no pueden contener valores múltiples y no reflejan ninguna estructura lógica. En definitiva, los atributos **no se podrán extender fácilmente** en futuros cambios.

Un elemento puede tener varios atributos, pero ninguno de estos puede estar vacío, además todos los atributos dentro de un elemento **deben ser únicos**. Los atributos se codifican de la siguiente forma:

```
<etiqueta atributo="valor_atributo"></etiqueta>
```

No debe usarse un atributo para almacenar información susceptible de ser dividida, sino para proporcionar información adicional sobre el elemento. A continuación vamos a ver un ejemplo de la utilización de atributos en un documento XML.

```
<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<centro_educativo>
    <alumno sexo="Varón" fecha_nacimiento="05/06/1990">
        <nombre>Pablo</nombre>
        <apellido>Pérez</apellido>
        <telefono tipo="Móvil">666666666</telefono>
        <direccion tipo="Calle">Policar, 34</direccion>
    </alumno>
 </centro_educativo>
```

Figura 1.5.4: Uso de atributos en documentos XML

Por norma general, intentaremos **evitar el uso** de atributos o procurar **no abusar de ellos**. Normalmente **los usaremos** para **metadatos** o información que no sea relevante para los datos. Se puede ver como una manera de incorporar características o propiedades a los elementos, como en el siguientes

ejemplo: *<perro raza="Pastor">Lolo</perro>*. También se suelen para especificar unidades de medida y similares: *altura unidad_altura="cm">178</altura>*. Hay que destacar que toda la información que se almacena en atributos se podría almacenar igualmente en elementos.

Lo que si es recomendable es que **una vez elegido un estilo**, mantenerlo dentro de todo el documento XML, teniendo en cuenta que los **nombres de los atributos** tienen que cumplir las **mismas normas** que los **elementos**, y no pueden contener el carácter menor que "<".

1.6 Documentos XML bien formados

Los documentos XML deben de estar **bien formados**, es decir, ser documentos **válidos**.

Los **documentos bien formados** son aquellos que cumplen las reglas sintácticas de creación de documentos XML ya mencionados en los puntos anteriores, como por ejemplo, que usan caracteres válidos para la creación de nombres de etiquetas o atributos, que las etiquetas estén cerradas correctamente, etc...

Los **documentos válidos** son aquellos que, ademas de estar bien formados, cumplen los requisitos de una definición de estructura (DTD, Schema,..), que veremos en la siguientes unidad.

Por lo tanto, para que un documento esté bien formado, debe **verificar** las **reglas sintácticas** que define la recomendación de la W3C para el **estándar XML**. Estas normas básicas se pueden resumir en las siguientes:

- El documento ha de tener **definido** una **declaración XML** en el prólogo:

  ```
  <?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
  ```

  Hay que tener en cuenta que aunque es posible omitir el prólogo, hay navegadores que nos pueden devolver un error al procesar el documento, por lo que siempre hay que incluirlo para evitar problemas. En caso de omitirlo, los valores por defecto son los mostrados encima de este párrafo.

- Existe un **único elemento raíz** por cada documento: es un solo elemento en el que todos los demás contenidos y elementos se encuentran anidados.

- Los elementos se organizan entre sí en un **estructura jerárquica** y **no se permite** el **solapamiento** de éstos.

- Hay que **cumplir reglas sintácticas** a la hora de definir el nombre de elementos y atributos. Estas normas se pueden resumir en:

  - El **nombre de elementos** puede contener como **primer carácter** los siguientes: *[A-Z]*, *[a-z]* y *"_"*. Para el resto de caracteres, ademas de estos, se pueden emplear: *[0-9]*, *"-"* y *"."*.

  - Las **etiquetas de apertura y cierre** deben ser **idénticas**, teniendo en cuenta que XML es sensible a mayúsculas y minúsculas.

  - Los **valores de los atributos** se escribirán siempre entre **comillas dobles o simples**. Si quisiéramos usar alguno de estos caracteres dentro del nombre usaríamos " (") y ' (’). También hay que tener en cuenta que hay nombres reservados para el uso del lenguaje.

  - **Los comentarios** en XML se escriben así: *<!– Comentario –>*

1.6.1 Espacios de Nombres

Los **espacios de nombres** permiten definir la pertenencia de los elementos y nombres a un contexto de vocabulario XML. De esta forma se resuelven las ambigüedades que se pueden producir al juntar dos documentos donde os autores han usado nombres similares para diferentes elementos.

Los espacios de nombres, también conocidos como **XML namespaces**, permiten dar nombre único a un elemento, indexándolo según un nombre de vocabulario adecuado. Además, están asociados a una **URI**.

En el documento, las etiquetas ambiguas se sustituyen por otras con el nombre precedido de un **prefijo**, que determina el contexto al que pertenece dicho nombre:

```
<prefijo:nombre_etiqueta></prefijo:nombre_etiqueta>
```

Esta etiqueta de denomina **nombre cualificado**. Al definir el prefijo hay que tener en cuenta que no se pueden usar espacios ni caracteres especiales y que no puede comenzar por un dígito.

Antes de poder usar el espacio de nombres hay que declararlo, es decir, asociar un índice con la URI asignada al espacio de nombres, mediante un atributo **xmlns**. Esto se hace entre el prólogo y el ejemplar y tiene la siguientes sintaxis:

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

Si queremos consultar más información acerca de los espacios de nombres, podemos consultar el estándar en la recomendación en XML de la W3C.

1.7 Sistemas de Gestión Empresarial

Lo primero que debemos de entender antes de meternos de lleno en los sistemas de gestión empresarial es como fluye el flujo de información en una empresa, es decir, conocer sus recursos empresariales y gestionarlos eficientemente. Estos flujos de información son los siguientes:

- Entre los empleados de la empresa.
- Entre los empleados y la empresa.
- Entre la empresa con sus clientes y proveedores.

Estos **flujos de información** se puede clasificar en dos tipos:

- Informales y no estructurados.
- Formales y estructurados, que se centran en información de procesos críticos de la empresa.

Para facilitar estos flujos de información es conveniente tener instalado un **Sistema de Información**, que facilite el conocimiento propio de la empresa para mejorar la **planificación**, la **gestión** y el **control**.

Un **Sistema de Información** se define como un conjunto organizado, de elementos relacionados, orientados al tratamiento y administración de la información. Esta compuesto por elementos físicos, humanos y por un conjunto de normas y protocolos.

La utilización de sistema de información ofrece **ventajas competitivas** a las empresas, mejorando su eficiencia, calidad del producto o mejorando los servicios ofrecidos a los clientes. También ayudan, entre otras cosas, a la captación de clientes.

La automatización de los flujos de información cambió sustancialmente con el surgimiento de los **ERP** y los **CRM**, en los que se integran las diferentes aplicaciones que soportan los procesos de la empresa.

1.7.1 ERP

En una empresa, salvo que esta sea muy pequeña, es necesario usar algún sistema automatizado de gestión para controlar todos los recursos empresariales y procesos. Aquí es donde entran en juego los ERP.

Un **ERP** es un **sistema de gestión de la información** que se caracteriza por ser **una aplicación** donde hay **varias partes integradas** y se **especializa en manejar** todos los **datos relevantes** para la continuidad de la empresa. Estas partes gestionan diferentes procesos, por ejemplo producción, ventas, compras, pedidos, nóminas, etc...

Un **CRM** es un tipo de ERP que se centra en la **relación con los clientes** que tiene una empresa, es decir, información de contacto orientada a ventas.

Los **objetivos principales** de un ERP son los siguientes:

- **Optimizar** los procesos empresariales.
- **Acceder** a la información de forma confiables y precisa.
- **Permitir** compartir información entre los componentes de la organización.
- **Eliminar** los datos y operaciones innecesarias.

Las **características** que diferencia a los ERPs de otras aplicaciones, es que deben ser **sistemas integrales**, **modulares** y **adaptables**. Sus principales características son:

- Ser un programa con **acceso de una base de datos**.
- Sus **componentes interactúan** entre sí.
- Las **datos** deben ser **consistentes**, **completos**.

Suelen ser **sistemas complejos** de implantar ya que necesitan un desarrollo personalizado para las necesidades de la empresa a partir del paquete básico. Estas adaptaciones, suelen estar a cargo de **consultorías**.

Las **consultorías** en materia de **ERP** suelen ser de dos tipos:

- **Consultoría de negocios**: estudia los procesos de negocio de la compañía y personaliza el ERP para ajustarlo a las necesidades de la organización.
- **Consultoría técnica**: estudia los recursos tecnológicos existente.

En la actualidad, la mayoría de sistemas ERP poseen **interfaz web** que mejora la accesibilidad al sistema desde prácticamente cualquier lugar y dispositivo.

1.7.1.1 Características

Las características que distinguen a un sistema de gestión empresarial son las siguientes:

- **Integración**: un sistema ERP **integra todos los procesos de la empresa**, considerándolos como un serie de áreas que se relacionan entre sí para conseguir una mayor eficiencia, reduciendo tiempo y costes.

- **Modularidad**: **cada módulo** de un sistema ERP se corresponde con un **área funcional** de la empresa. Gracias a una **base de datos centralizada**, todos los módulos pueden compartir información entre sí, facilitando la adaptabilidad, personalización e integración. Cada módulo suele usar un software específico para su funcionalidad.
- **Adaptabilidad**: aunque las dos características anteriores facilitan la adaptabilidad del ERP a los requisitos de la empresa, a veces se utiliza una solución más genérica, para abaratar costes, y se modifican algunos procesos para adaptarlos al sistema ERP.

Gracias a la modularidad y la capacidad de integración de las funcionalidades, los ERPs se pueden adaptar fácilmente a las necesidades de cada empresa. Los diferentes módulos se interconectan entre sí de forma que una única herramienta ERP puede adaptarse a diferentes empresas cambiando el conjunto de módulos activos y sus relaciones.

Los **módulos principales** que forman un ERP son los siguientes:

- **Ventas/Marketing**: interfaz pública que interactúa con los clientes, pedidos, estrategias de ventas, precios, promociones, publicidad, etc..
- **Finanzas**: es la base de cada ERP, donde se almacenan las transacciones facilitando las auditorías.
- **Inventario/Logística**: controla el stock y los flujos de entrada y salida.
- **Recursos Humanos**: gestión de personal, nómina, productividad, incentivos, etc..
- **Producción**: en núcleo que se encarga de los movimientos físicos del producto, planificación de materiales, etc..

Estos módulos son considerados los básicos, pero se pueden añadir muchos más como proyectos, planificación de ventas, configuración de productos a medida, etc..

No todos los **trabajadores** accederán al ERP de la misma forma, ya que cada grupo tiene su **rol**, que será supervisado por un administrador, por lo que dependiendo de este rol el trabajador tendrá unos u otros módulos habilitados.

Mención especial requieren los **CRM** (Customer Relationship Management), que surgen como consecuencia de la aplicación especifica de los ERP a las interacciones con los clientes. Están centrados en mantener, crear y potencias las relaciones con clientes sirviendo de apoyo a las políticas de marketing de la empresa.

En la actualidad los sistemas globales de CRM se pueden dividir en:

- **Aplicaciones** electrónicas para los **canales de distribución** de la empresa.
- Centros de **atención telefónica**.
- **Autoservicio** para los clientes.
- **Gestión** electrónica de las **actividades** que afecten a los clientes.
- **Ventas**.

Entres sus **principales características** se encuentran la facilidad de **toma de decisiones** en **tiempo real**, **incremento de la rentabilidad** del clientes gracias a que se obtiene información muy útil a través de datos complejos, por ejemplo, identificando a los clientes que compran o que no están interesados y actuar en consecuencia.

1.7.1.2 Ventajas e Inconvenientes

Contar con un **sistema ERP personalizado**, permite a la empresa tener integradas diferentes utilidades que facilitan la gestión de la información. Aunque estos sistemas ofrecen muchas ventajas para una empresa, también tiene sus inconvenientes, como podemos ver en la siguiente lista.

- **Ventajas**
  - **Aumento** de la **información** que tiene la empresa sobre sus **potenciales clientes**. Los ERP que incluyen CRM aportan beneficios relacionados con la gestión de clientes de la empresa. Algunos incluyen control de calidad de los productos, permitiendo enfocar la oferta en las necesidades y deseos de los clientes, con la consecuente mejora de la satisfacción de estos.
  - **Aumento** de las **ventas**.
  - Permiten **resolver problemas** relacionados con el **tratamiento de la información** derivado del uso de sistemas anteriores.
  - **Aumenta** la **eficiencia operativa**.
  - **Facilitan** el **acceso de la información** y constituyen una mejora en las herramientas de tratamiento de la misma.
  - **Reducción** de **costes empresariales**, especialmente los relacionados con las operaciones de las tecnologías de la información y comunicaciones.
  - Permiten mayor **facilidad de configuración** de los sistemas de la empresa.
  - Mejoran el **entorno de integración** de todas sus acciones.
- **Inconvenientes**
  - El ERP ha de ser usado y realizado por **personal capacitado**.
  - La **instalación** del ERP es **muy costosa**.
  - Los ERP son vistos como **sistemas muy rígidos** y difíciles de adaptar al modo de trabajo de las empresas.
  - Son sistemas que sufren de problemas de **cuello de botella**, es decir, todos los usuarios se pueden ver afectados por la ineficiencia en uno de los departamentos.
  - Altos **coste de modificación** de un ERP ya implantado.

1.7.1.3 ERP de Software Libre

Dentro de los ERP de software libre encontramos una gran variedad de aplicaciones para la gestión empresarial. Entre ellas podemos destacar **Openbravo**, que es una iniciativa de origen español y **OpenERP**, actualmente conocido como **Odoo**, de origen belga y que se carateriza por tener una gran cantidad de módulos.

Openbravo es una aplicación de código abierto de planificación de recursos empresariales. Utiliza una arquitectura **cliente/servidor** web y esta escrita en **Java**. Se ejecuta sobre un servidor web y ofrece **soporte** para la diferentes bases de datos **Oracle** y **PostgreSQL**. Consta de dos versiones:

- **Openbravo Community Edition**: versión libre y gratuita desde la que no se puede acceder a los módulos comerciales. Tiene licencia **OBPL**.

- **Openbravo Network Edition**: versión bajo licencia **OBCL** que ofrece actualizaciones de código y si ofrece acceso a los módulos comerciales.

Además de estas dos versiones, Openbravo ofrece dos soluciones diferentes:

- **Suite de comercio Openbravo**: solución de comercio para minoristas.
- **Suite de negocio Openbravo**: solución global para empresas.

Por otro lado tenemos **Odoo** (anteriormente **OpenERP**) que resuelve problemas complejos haciendo uso de soluciones sencillas. Esta escrita en **Python** y hace uso de la base de datos **PostgreSQL**.

OpenERP fue creado en el año 2005 por un joven informático belga llamado **Fabien Pickaers**, causando gran sorpresa de que creara un programa de estas características y lo pusiera de forma gratuita en internet, mientras otras empresas venden sus productos a precios desorbitados. El modelo de Odoo esta basado en los servicios prestados en torno al software y tiene colaboradores alrededor de todo el mundo. En este enlace podemos ver el motivo del cambio de nombre de OpenERP a Odoo, así como la respuesta a algunas preguntas interesantes.

A continuación se muestran algunos enlaces de interese sobre estos ERPs:

- **Página oficial Openbravo** - `www.openbravo.com/es/`
- **Wiki sobre OpenBravo** - `http://wiki.openbravo.com/wiki/Main_Page`
- **Página oficial de Odoo** - `http://www.openerp.com/es`
- **Doc. de usuario de Odoo** - `https://www.odoo.com/documentation/8.0/`
- **Doc. ténica de Odoo** - `https://www.odoo.com/documentation/user/`

1.7.1.4 Instalación

Glosario

ASCII American Standard Code for Information Interchange. 6, 19

CRM Customer Relationship Management. 17, 19

DTD Document Type Definition. 4, 19

ERP Enterprise Resourcer Planning. 17, 19

GML Generalized Markup Language. 6, 19

HTML Hypertext Markup Language. 6, 19

metalenguaje Lenguaje que permite la definición de otros lenguajes. 7, 19

SGML Standard Generalized Markup Language. 6, 19

texto plano Es aquel texto formado solo por datos sin formato, es decir, solo por caractéres.. 9, 19

unicode Es un código que permite el tratamiento informático de textos en cualquier lenguaje y disciplina técnica, ya que incluye todos los carácteres conocidos para cualquier lengua. Es compatible con ASCII. 11, 19

URI Uniform Resource Identifier. Son hypervínculos que dan acceso a un recurso remoto. 16, 19

UTF-8 8-bit Unicode Transformation Format. 11, 19

W3C Word Wide Web Consortium. 7, 19

XML eXtensible Markup Language. 7, 19

Bibliografía

[1] Wikipedia - Tex. ttps://es.wikipedia.org/wiki/TeX.

[2] Wikipedia - ASCII. https://es.wikipedia.org/wiki/ASCII.

[3] Wikipedia - XML. https://es.wikipedia.org/wiki/XML.