2. Ficheros de marcas

Se puede decir que los archivos de **marcas** son una manera diferente de almacenar información en ordenadores, que añadimos a los modos de almacenar la información ya conocidos (archivos binarios o archivos de texto). El objetivo principal de los ficheros de marcas es intentar recoger las mejores características de los archivos de texto y binarios y esquivar sus problemas.

Los ficheros de marcas toman como base los archivos de texto para aprovecharse de las características más interesantes de este tipo de archivos:

- La facilidad de creación y lectura.
- El cumplimiento de estándares de almacenamiento definidos y públicos.

Como los archivos de texto siempre están almacenados en algún código de caracteres conocido (ASCII, UTF-8, etc.), se consigue que puedan ser transportados y leídos en cualquier plataforma, sistema operativo o programa que pueda interpretar estos códigos de caracteres. Por lo tanto, los lenguajes de marcas aprovecharán esta característica, al estar basados en el formato de texto.

Además, de rebote, también tendrán la ventaja de que podrán ser abiertos y creados con los programas de edición de texto estándar. Desde editores tan simples como el *Bloc de notas* de los sistemas Windows o *Gedit* de sistemas Unix hasta editores más complejos como el Microsoft *Word*, pasando por editores especializados en XML como el *Oxygen XML Editor*.

Los ficheros de marcas, por lo tanto, se aprovechan de una de las grandes ventajas de los archivos de texto sobre los archivos binarios, ya que estos últimos requieren ser abiertos con un programa específico que pueda interpretar el formato.

Pero los archivos de marcas no sólo intentan aprovechar las características de los ficheros de texto, sino que también intentan conseguir las características más interesantes de los **ficheros binarios**, como:

- La incorporación de **metadatos**.
- La definición de la **estructura** de los datos.

Esto hace que los lenguajes de marcas adquieran una de las características más interesantes de los ficheros binarios, que es la posibilidad de incorporar información sobre los datos -**metadatos**- pero intentando que afecte lo menos posible la legibilidad del documento.
También permiten definir los datos y su estructura de manera que sea sencillo para un programa poderlas interpretar.

Gracias a las ventajas que ofrecen los lenguajes de marcas, se han convertido rápidamente en una de las formas habituales de representar datos; se pueden encontrar continuamente en cualquier tarea con ordenadores:

- El exponente más popular es Internet -La Web-, que está basada totalmente en los lenguajes de marcas.
- Muchos de los programas de ordenador que se usan habitualmente, en algún momento, de alguna u otra forma, utilizan algún lenguaje de marcas para almacenar sus datos de configuración o de resultados:
    - Internamente los formatos de documentos de Microsoft Office o de OpenOffice/LibreOffice, están basados en lenguajes de marcas.
    - Microsoft Visual Studio guarda su configuración utilizando lenguajes de marcas.
    - etc.

2.1. Las Marcas

Las marcas son una serie de códigos que se incorporan a los documentos digitales para determinar su formato: la forma en que se han de imprimir, la estructura de los datos, etc. Por lo tanto, **son anotaciones que se incorporan a los datos** pero que no forman parte de ellos
Las marcas, deben ser fácilmente distinguibles del texto normal (por su posición, porque siguen algún tipo de sintaxis, etc.). Las marcas más usadas son las que están formadas por textos descriptivos y están rodeadas de los símbolos de "menor " (**<**) y "mayor" (**>**); normalmente suele haber una al principio y otra al final:

< nombre > Manel Puig Garcia </ nombre >

```
<persona >
    <nombre > Manel Perez Garcia </ nombre >
    <nombre > Pedro González Parada </ nombre >
    <nombre > María Pozos Calle </ nombre >
</ persona >
```

Estas marcas pueden ser ensambladas para **indicar estructuras de datos**.

```
\ Section {Personas} \ Begin {itemize}
\ Item Manel Perez Garcia
\ Item Pedro González Parada
\ Item María Pozos Calle
\ End {itemize}
```

Pero hay muchas otras formas de marcas. Otra idea consiste en encontrar alguna combinación de caracteres que salga raramente en el lenguaje habitual. El *TeX* utiliza las barras invertidas para indicar el inicio de las marcas:

Otros lenguajes de marcas usan caracteres no habituales en determinadas posiciones para indicar que son marcas. Por ejemplo, con *Wiki Markup* los caracteres "**=**" en la primera posición de una línea se usan para indicar que el texto es un título de apartado y el ***** para cada elemento de la lista:

```
= Personas =
* Manel Perez Garcia
* Pedro González Parada
* María Pozos Calle
```

La idea general es que es necesario que las marcas sean fácilmente identificables para poder aprovechar las ventajas que ofrecen los lenguajes de marcas.

2.1.2. Características de los lenguajes de marcas

Los lenguajes de marcas son una manera de codificar un documento de texto, de manera que, por medio de las marcas (el equivalente de los metadatos de los archivos binarios) se incorpora información relativa a: cómo se debe representar el texto, qué estructura tienen los datos que contiene, etc.

Los lenguajes de marcas han destacado por una serie de características que los han convertido en los tipos de lenguajes más usados en la informática actual para almacenar y representar los datos.

Entre las características más interesantes que ofrecen los lenguajes de marcas se encuentran:

- Que se basan en el **texto plano**.
- Que permiten utilizar **metadatos**.
- Que son fáciles de **interpretar y procesar**.
- Que son fáciles de **crear** y suficientemente **flexibles** para representar datos muy diversos.

Las aplicaciones de Internet y muchos de los programas de ordenador que se utilizan habitualmente incorporan de alguna manera u otra algún lenguaje de marcas.

- **Basados en texto plano**

Los lenguajes de marcas se basan en texto plano sin formato. Estos caracteres pueden estar codificados en diferentes códigos de caracteres: ASCII, ISO-8859-1, UTF-8, etc.
Una de las ventajas que intentan aportar los lenguajes de marcas es que se pueden interpretar directamente y esto sólo es posible si usamos el formato de texto, ya que los binarios requieren un programa para interpretarlos. Pero además tienen la ventaja de que son independientes de la plataforma, del sistema operativo o del programa.
El hecho de que estén basados en formato de texto hace que sean fáciles de crear y modificar ya que sólo requieren un simple editor de textos.

- **Uso de metadatos**

Las **marcas** se intercalan entre el contenido del documento; generalmente estas etiquetas suelen ser descriptores de los datos a los que contienen. Son la forma en que se añaden los metadatos a los documentos de texto y cómo se consiguen superar las limitaciones del formato de texto (conseguir algunas de las ventajas de los ficheros binarios).

- **Facilidad de proceso**

Los lenguajes de marcas permiten que el procesamiento de los datos que contiene pueda ser automatizado de alguna manera, ya que el archivo incluye la estructura de esos datos. El hecho de incluir la estructura permitirá que un programa pueda interpretar cada uno de

los datos de un fichero de marcas para representarlo o tratarlo convenientemente.
Posteriormente un programa podrá interpretar, gracias a las marcas, qué es lo que significa cada uno de los datos del documento.

- **Facilidad de creación y representación de datos diversos**

A pesar de que fueron pensados para contener datos de texto, los lenguajes de marcas han demostrado que son capaces de contener datos de muchos tipos diferentes.
Actualmente se están utilizando archivos de marcas para representar imágenes vectoriales, fórmulas matemáticas, crear páginas web, ejecutar funciones remotas mediante servicios web, representar música o sonidos, etc.
Y sin importar qué tipo de datos se representen siempre habrá la posibilidad de crear estos archivos desde un editor de texto básico.

2.2. Clasificación de los lenguajes de marcas

Es complicado hacer una clasificación de los lenguajes de marcas que hay, pero generalmente, se acepta que tenemos dos grandes grupos seleccionados según cuál es su **objetivo** básico:

- **Lenguajes procedimentales y de presentación**, orientados a especificar cómo se debe representar la información.
- **Lenguajes descriptivos o semánticos**: orientados a describir la estructura de los datos que contiene.

Esta es la clasificación más aceptada, pero, como muy a menudo ocurre en el ámbito de la Informática, nos podemos encontrar lenguajes que tengan aspectos de los dos grupos y permitan tanto definir la manera de presentar la información como definir su estructura.

2.2.1. Procedimentales y de presentación

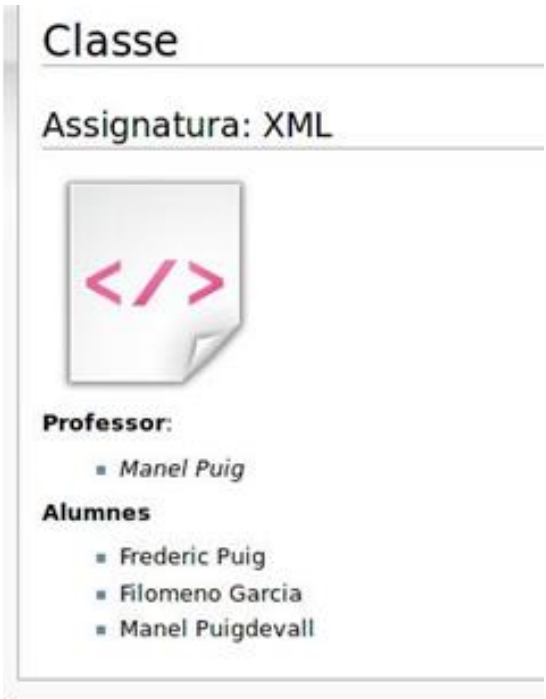
En estos lenguajes lo que se hace es indicar de qué manera se hará la presentación de los datos. Ya sea por medio de información para el diseño (marcar negritas, títulos, etc.) o de procedimientos que tiene que hacer el software de representación. El ejemplo más popular de estos lenguajes es el *HTML,* pero hay muchos más: *TeX*, *Wikitext* ...

En estos casos los documentos nos pueden servir para determinar de qué manera se mostrará el documento a quien lo lea.

Si tomamos un ejemplo sencillo utilizando el lenguaje de marcas ligero Wiki markup, que utiliza *Mediawiki* (programa con el que se ha desarrollado la Wikipedia), que nos mostrará lo que se ve en la Figura 1.7:

```
= Clase =

== Asignatura: XML ==
[[Archivo: xml.png]]

" 'Profesor' ":
* " Manel Puig "

" Alumnos " '

* Frederic Puig
* Filomeno García
* Manel Puigdevall
```

Figura 1.7 Representación del texto en formato wikicode



Se puede ver cómo el programa ha interpretado las marcas "**=**" o "**==**" para mostrar los diferentes niveles de los títulos; los símbolos "*****" indican los elementos de la lista que con la diferente cantidad de comillas indican negritas o cursiva.

2.2.2. Descriptivos o semánticos

En estos lenguajes se describe qué estructura lógica tiene el documento, ignorando de qué manera será representada en los programas. Sólo se ponen las marcas con el objetivo de definir las partes que dan estructura al documento. El ejemplo más importante es el *XML*, pero hay algún otro que está teniendo mucho uso, como por ejemplo *JSON*.

En el documento siguiente tenemos un ejemplo de un archivo de marcas que da información sobre personas:

```
<alumnos >
    <persona >
        <nombre > Pedro </ nombre >
        <apellido > Perez </ apellido >
    </ persona >
    <persona >
        <nombre > Manuel </ nombre >
        <apellido > Garcia </ apellido >
    </ persona >
```

Se puede ver claramente que la información de las marcas en el documento establece cuál es el contenido de los datos: una lista de alumnos. Con un simple vistazo resulta fácil determinar que Pedro y Manuel son nombres y que Perez y Garcia son apellidos. Pero por medio de la jerarquía de los datos se puede deducir que Pedro Perez y Manel Garcia son alumnos ya que tanto el nombre como el apellido están incluidos dentro de la marca de alumnos.

Este documento muestra cuál es **la estructura de los datos** que contiene y además también permite conocer su significado interpretando el contenido semántico de las etiquetas; se deduce que Pedro es el nombre de una persona que es un alumno.

2.2.3. Sistema de etiquetado

Tanto si el sistema es descriptivo como de presentación, **las marcas no han sido colocadas de cualquier manera,** sino que se ha ido siguiendo un sistema determinado.

A menudo, las marcas rodean el contenido que queremos que tenga un significado o que sea representado de una manera determinada. No se pueden colocar las marcas de cualquier manera, ya que hay una cosa que debemos evitar y son las posibles malinterpretaciones.

Por ello, además de definir las marcas que se pondrán, los lenguajes de marcas definen unas reglas de uso que especifican cómo deben ser las marcas, en qué condiciones se permite usarlas y, a veces incluso qué significan.

2.3. Historia

Se considera que el origen de los lenguajes de marcas está en las modificaciones que los impresores hacían con lápiz en manuscritos. Cuando alguien quería imprimir un libro que había escrito, los impresores, con un lápiz generalmente de color azul, escribían en el texto qué características debía tener cada parte del texto, si se tenía que hacer en negrita, si era el título del libro, etc. Se cree que estos son los antecedentes de las marcas.

2.3.1 SGML

Al principio de los años ochenta a IBM necesitaba alguna manera de almacenar y compartir una gran cantidad de información entre diferentes plataformas, que permitiera integrar los datos en sistemas de datos, editores, etc., y desarrollaron *GML*, que posteriormente acabó con el nombre *SGML* cuando fue estandarizado en 1986 por la organización de estándares internacional ISO (**I**nternational **O**rganization for **S**tandardization). La especificación se encuentra bajo el nombre *ISO-8879*.

A pesar de que no se considera el primer lenguaje de marcas, fue el primer lenguaje reconocido como estándar ISO.
SGML es un lenguaje basado en los datos de texto que se pueden usar para añadir metadatos a los datos. Es un sistema para organizar y etiquetar elementos de un documento, poniendo énfasis en los aspectos de la estructura del documento y dejando que sea el intérprete el que se encarga de hacer la representación visual de estos datos. Lo hace definiendo unas reglas estrictas que especifican de qué manera se pueden usar las etiquetas.

SGML se diseñó para ser una manera estándar de etiquetar datos genéricos, de manera que no importara si los datos para etiquetar provenían del mundo de las matemáticas o bien eran los resultados financieros de una empresa. Todos los datos se podían etiquetar con sentido utilizando SGML.

> El hecho de que SGML fuera tan complejo no lo hacía ideal para intercambiar datos a través de Internet. Si sólo un grupo de personas hubiera generado información, el crecimiento del entorno Web habría sido muy inferior.

SGML se utilizaba sobre todo en documentos que debían tener muchos cambios y que posteriormente habían de representarse en formatos diferentes.
Por lo tanto, con SGML tenemos las siguientes ventajas:

- Tenemos una manera de **reutilizar** los datos.
- Permite un mayor control sobre los datos y garantiza la **integridad**.
- Es **portable**.
- Es **flexible**.
- Nos garantiza la **perdurabilidad** de la información.

Pero no todo son ventajas en el SGML:

- La mayoría de los documentos que se creaban sólo estaban destinados a la impresión.
- Es terriblemente complejo, por lo que no se utiliza en ordenadores personales.

2.3.2. HTML

En 1989, Tim Berners-Lee y Anders Berglund, dos investigadores del CERN (acrónimo de **E**uropean **C**ouncil for **N**uclear **R**esearch, Organización Europea para la Investigación Nuclear), crearon un lenguaje de etiquetas basado en SGML destinado a compartir información por Internet: HTML (Hyper **T**ext **M**arkup **L**anguage). HTML se basa en la manera de definir e interpretar etiquetas de SGML, pero no es totalmente compatible con SGML (algunas de sus reglas incumplen las reglas SGML).

HTML se centra en definir un formato para describir la visualización de la información en una página web y es muy sencillo. Su sencillez ha sido uno de los factores que ha llevado a su rápida popularidad en **W**orld **W**ide **W**eb, en Internet. Es uno de los motivos por los que cada día se generan millones de páginas web nuevas.

El gran éxito de las tecnologías basadas en HTML ha hecho que no paren de evolucionar y, por tanto, que HTML debiera evolucionar muy rápidamente para adaptarse cada vez a más cambios y a las nuevas necesidades de los usuarios. Esto, sumado al propósito de no incrementar la dificultad del lenguaje, ha provocado que no siempre se hayan hecho las cosas de la misma manera y que, por tanto, la creación de intérpretes de HTML (en

especial los navegadores) sea cada vez sea más compleja.

A todo esto, hay que sumarle que, pese a estar pensado para representar la información, HTML no define muy estrictamente algunas de las reglas de cómo se debe visualizar la información, y por lo tanto a menudo los navegadores han de hacer interpretaciones que no siempre coinciden con las que hacen los otros navegadores. Es conocido por todos los diseñadores de páginas web, que las páginas no siempre se ven igual en todos los navegadores (Figura 1.8).

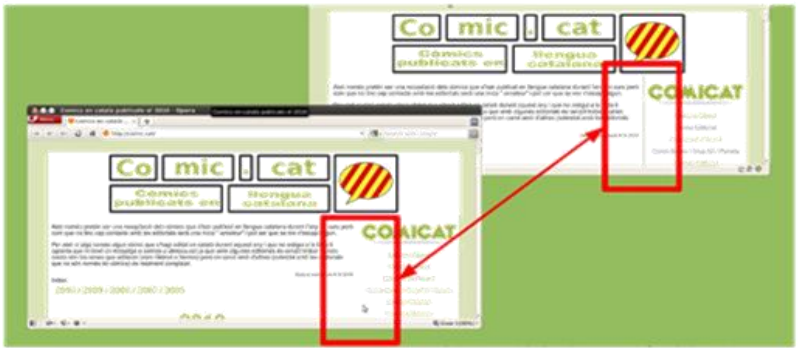


Figura 1. 8. Visualización de páginas en navegadores

A veces las diferencias son mínimas, pero en otros casos las diferencias pueden ser muy importantes.

Por otra parte, HTML funciona bien a la hora de presentar información a los humanos, pero tiene algunos problemas que lo hacen poco eficiente para las nuevas aplicaciones actuales:

- es muy difícil reutilizar la información que contiene para generar resultados en formatos distintos de los que ha definido el diseñador
- es muy complejo para los programas interpretar de qué tipo son los datos contenidos en un documento HTML.

Por lo tanto, era necesario algún modo de poder realizar búsquedas inteligentes en los documentos HTML y seleccionar los resultados según criterios personalizables.

> Había una manera de buscar, mover, visualizar y manipular la información contenida en los documentos HTML.

Y por este motivo apareció el XML.

2.3.4. XML

El consorcio *W3C* desarrolló una alternativa al HTML que pudiera satisfacer las necesidades futuras de la web. En 1996 el consorcio *W3C* se propuso introducir el poder y la flexibilidad de SGML en la web.

SGML ofrecía tres ventajas que el HTML no tenía:

- Extensibilidad
- Estructura
- Validación

> La versión más usada es la 1.0, ya que la versión 1.1 no aporta muchas novedades interesantes para la mayoría de usos (básicamente sobre las versiones de Unicode posteriores a XML 1.0)

En febrero de 1998 se lanza la especificación 1.0 de XML (http://www.w3.org/TR/2004/REC-xml-20040204/) y posteriormente en 2004 salió la versión 1.1 (www.W3.org/TR/xml11). Estas especificaciones se han ido revisando periódicamente.

El XML es un lenguaje simple de **descripción** de información:

- Es un estándar que permite diseñar y desarrollar lenguajes de marcas.
- Es un formato de texto estandarizado que sirve para representar y transportar información estructurada.

- **Número de etiquetas**

Al HTML le ha ido bien con un número finito de etiquetas y, por tanto, a la hora de diseñar el XML se hicieron varios intentos de crear un número finito de etiquetas. Pero todos los intentos de crear un conjunto finito de etiquetas fallaron porque restringir las etiquetas restaba flexibilidad al lenguaje.

Se vio, que cada conjunto de usuarios necesita un subconjunto de etiquetas diferente y, que a menudo, eran divergentes (los matemáticos usaban uno, los químicos necesitaban

otro, etc.), o sea, que la solución final adoptada fue la más lógica: si restringir las etiquetas resta flexibilidad, lo más fácil es no restringirlas.

El XML define un número **infinito** de etiquetas.

Por lo tanto, el XML permitirá que cada persona pueda definir las etiquetas que le hagan falta para poder representar los datos más adecuadamente.

- **Estructura de los datos**

Otra idea que se tuvo en cuenta a la hora de desarrollar el XML era que los datos que contuviera, se pudieran reutilizar para generar otros resultados y, por tanto, pudiera ser interpretado fácilmente por medio de programas de ordenador. Por lo tanto, los datos contenidos en los documentos debían tener una estructura.

XML se diseñó con la idea de dar estructura a los datos y no preocuparse de cómo se presentarán a los usuarios. Para ello ya se desarrollarían otras alternativas: *CSS*, *XML-FO*, etc.

Una de las ideas más importantes de XML es separar los datos de la presentación.

A la hora de crear un documento XML se debe pensar en cómo debemos estructurar los datos y nunca especificar nada de cómo se deberán representar.

- **Transporte de los datos**

El hecho de que el XML se concentre en la estructura de los datos y que, por tanto, sea relativamente fácil determinar qué datos contiene, lo hace un sistema ideal para el transporte de datos entre diferentes plataformas.

Por lo tanto, si tenemos un documento XML como este:

```
<alumnos >
    <persona >
        <nombre > Manel </ nombre >
        <apellido > Garcia </ apellido >
    </ persona >
    <persona >
        <nombre > Pedro </ nombre >
        <apellido > González </ nombre >
    </ persona >
</ alumnos >
```

Podemos ver, que, observando este documento, es relativamente sencillo responder las preguntas:

- ¿Qué información contiene el archivo?
- ¿Cuál es la estructura de la información?
- ¿Qué etiquetas se han creado para describir la información?

- **Creación de lenguajes**

Es evidente que la libertad que da tener un número infinito de etiquetas, no es necesaria en la mayoría de los ámbitos de actuación. Por este motivo, normalmente cuando alguien quiera almacenar información definirá un número finito de etiquetas y en qué orden deben aparecer.

Para poder solucionar estos problemas, en XML se pueden definir archivos que definan cuál será la estructura del documento, y que, por tanto, se pueda comprobar si el documento sigue la estructura correcta o no. Esto a su vez permite que si definimos el vocabulario de manera pública cualquier nos pueda enviar documentos y detectar si están bien formados o no.

De hecho, ya hay toda una serie de documentos basados en XML que se han convertido en estándares públicos en diferentes ámbitos, algunos de los cuales se pueden ver en la Tabla .1.9.

Tabla 1.9. Algunos lenguajes estándar basados en XML

nombre	uso
SVG	Pensado para gráficos vectoriales en 2D con animaciones o sin
MathML	Lenguaje para representar fórmulas matemáticas
CML	Lenguaje para el intercambio de información química
SMIL	Tratamiento de información multimedia
SSML	Síntesis de voz
ChessGML	Para representar partidas de ajedrez
XFRML	Para hacer informes financieros
SML	Usado en la industria del acero

Pero hay más, ya que la lista es inmensa: SMBXML, CIML, Namlo, TML, SCORM, LMML, OpenMath, PetroXML, ProductionML, GeophysicsML, X3D, MML, SMDL, BGML, etc.

- **Extensible**

Otra de las ventajas de XML es, que es fácilmente extensible y adaptable a las necesidades que tengamos. Permite que se mezclen diferentes vocabularios en el mismo documento.

Podemos definir un documento XML con un vocabulario creado por nosotros que defina una lista de alumnos, que añada una imagen con el logotipo de la escuela en formato SVG (un estándar XML de gráficos vectoriales) y que defina la presentación en XHTML.

Por lo tanto, tenemos suficiente flexibilidad para representar los datos que nos sean necesarias en cada momento.

- **Uso de XML**

Actualmente los usos de XML son muy diversos:

- Mostrar el contenido de páginas web. Uno de los lenguajes XML es el XHTML, que intenta modificar el HTML para hacerlo más sencillo de interpretar.
- Comunicar sistemas distribuidos que incluso ejecuten sistemas operativos diferentes o estén en plataformas totalmente diferentes.
- En comercio electrónico, en un sistema conocido como Bussines2Bussines que permite a las empresas compartir datos de forma automática.
- Reducir la carga de servidores distribuyéndola entre servidores.

Muchos programas que utilizaban formatos binarios para almacenar sus datos han pasado a algún tipo de XML:

- Microsoft Office: pasó de guardar los documentos en binario con extensión *.doc* a documentos XML con extensión *.docx* (OOXML).
- OpenOffice.Org guarda sus documentos en un formato XML.

Podemos ver que muchos programas usan XML para guardar su configuración o sus datos con una simple búsqueda en el sistema operativo. Por ejemplo, en Linux podemos ejecutar lo siguiente para ver el número de ficheros XML que tenemos:

```
$ Locate .xml | wc -l
```

En Windows podemos hacer lo mismo con:

```
c: \> dir /ad /s *.xml | find /c /v ""
```

- **Problemas**

A pesar de las múltiples ventajas que ofrece el XML, también se le han hecho críticas, como el hecho de que los archivos XML tienen la tendencia a ser **muy grandes**. Casi siempre ocupan una cantidad mucho mayor de espacio en disco que sus equivalentes en formato binario.

El hecho de utilizar archivos muy grandes puede tener un impacto importante en el

rendimiento de los programas, ya que antes de poder trabajar deben cargar el archivo o descargarlo de la red.

Hay gente que considera que el problema del tamaño de los archivos a veces es compensado por:

- La facilidad de interoperabilidad entre programas.
- El precio del almacenamiento es cada vez más bajo y por ahora parece que la tendencia es que aún baje más.

Pero no todos están de acuerdo, y por este motivo han aparecido toda una serie de alternativas al XML, que se conocen como lenguajes de marcas ligeros, que normalmente tienen como objetivo conseguir que los archivos de marcas ocupen mucho menos espacio:

- Al ocupar menos espacio ahorran ancho de banda y espacio de disco.
- Normalmente se pueden convertir a XML sin problemas.
- Ocupan menos memoria RAM cuando son procesados.
- Los lenguajes de marcas ligeros más usados actualmente son **JSON** (JavaScript Object Notation) y los lenguajes de marcas de los wikis.