

**AQS**

*Advanced Quality Solutions*

# **XML Roadmap**

Febrero 2001

© Copyright Advanced Quality Solutions 2001

---

# Índice

---

<b><u>XML</u></b>	<b><u>2</u></b>
<b>INTRODUCCIÓN</b>	<b>2</b>
<b>USO DE XML</b>	<b>5</b>
CREACIÓN	5
ANÁLISIS Y MANIPULACIÓN (SAX, DOM)	5
VALIDACIÓN	5
TRANSFORMACIÓN	6
<b>JAVA Y XML</b>	<b>6</b>
<b>HERRAMIENTAS</b>	<b>7</b>
ÍNDICES DE SOFTWARE	7
NAVEGADORES	7
EDITORES	8
PARSERS	8
VALIDACIÓN DE TIPOS	8
VARIOS	8
<b>ESPECIFICACIONES RELACIONADAS</b>	<b>9</b>
<b>APLICACIONES</b>	<b>9</b>
<b>TEMAS RELACIONADOS</b>	<b>11</b>
DATA BINDING	11
MESSAGE ORIENTED MIDDLEWARE	11
WEB PUBLISHING FRAMEWORKS	11
XML Y BASES DE DATOS	11
LENGUAJES XML	11
<b>RECURSOS</b>	<b>12</b>
<b><u>XSL</u></b>	<b><u>13</u></b>
<b>INTRODUCCIÓN</b>	<b>13</b>
<b>USO DE XSLT CON JAVA</b>	<b>14</b>
<b>ESPECIFICACIONES RELACIONADAS</b>	<b>14</b>
<b>HERRAMIENTAS</b>	<b>15</b>
MOTORES DE TRANSFORMACIÓN	15
VARIOS	15

# XML

---

## Introducción

---

El lenguaje extensible de marcas XML (eXtensible Markup Language) es un formato estándar para la estructuración de datos. La definición actual XML 1.0 (Second Edition: W3C Recommendation 6 October 2000 <http://www.w3.org/TR/2000/REC-xml>) es una recomendación del W3C (<http://www.w3.org/XML/>) desde Febrero de 1998 y está basado en el estándar SGML (Standard Generalized Markup Language, ISO 8879), que data de 1986. Habitualmente, por su origen en el mundo del procesado de documentos, se vincula a la definición de documentos estructurados, pero es de aplicación general para datos de cualquier tipo que puedan estructurarse jerárquicamente. En todo caso, el texto que representa los datos se suele denominar documento XML.

Como introducción a XML pueden utilizarse los siguientes libros y direcciones Web:

- Marchal, Benoît (2000). "XML by Example". Que. Indianapolis, USA.
- Café con leche XML News and Resources (Elliote Rusty Harold)  
<http://www.ibiblio.org/xml/slides/acmdc/introxml/>
- Zvon.org [http://www.zvon.org/index.php?nav\\_id=tutorials&mime=html](http://www.zvon.org/index.php?nav_id=tutorials&mime=html)
- En (<http://www.mulberrytech.com/quickref/index.html>) puede encontrarse una plantilla de referencia rápida de XML.

"Café con leche" es un sitio web que su autor mantiene constantemente actualizado. Incluye un índice de recursos relacionados con XML y provee diariamente de las últimas noticias sobre XML y temas afines por lo que es un buen punto de referencia para estar al día. El autor extracta también ciertos comentarios jugosos que aparecen en la lista de correo en la que se discute de forma genérica sobre XML (xml-dev mailing list <http://www.xml.org/xml-dev/index.shtml>).

"Zvon.org" es otro clásico sitio web de recursos XML y HTML. Su autor ha elaborado buenos tutoriales cargados de ejemplos.

Un documento XML aparece como una jerarquía estrictamente anidada de elementos. Los elementos tienen atributos y pueden contener texto u otros elementos como hijos. Por ejemplo, un documento que describe la estructura de delegaciones de una empresa podría tener el siguiente aspecto:

```
<?xml version="1.0" encoding="UTF-8"?>
<Company name="ABC, S.A." founded="2001-01-01" income="10000">
  <Delegation name="Central Madrid" numberOfEmployees="150">Texto asociado Madrid</Delegation>
  <Delegation name="Sucursal Jaén" numberOfEmployees="270">Texto asociado Jaén</Delegation>
</Company>
```

XML es en sí mismo un metalenguaje para definir lenguajes. Los elementos y atributos y su orden en la jerarquía forman el lenguaje usado por el documento. En el ejemplo anterior, puede inferirse, que el lenguaje contiene, al menos, dos elementos: Company y Delegation. El elemento 'Company' contiene elementos 'Delegation' y tiene atributos "name", "founded" y "income"; así mismo, "Delegation" contiene texto además de la información incluida en sus atributos. Las etiquetas utilizadas para describir los elementos, por ejemplo <Company> para el elemento "Company", conforman la sintaxis del documento XML. En XML no hay etiquetas predefinidas, como ocurre en HTML, el usuario define cuáles son sus elementos y las etiquetas asociadas. La semántica del documento XML la proporciona la aplicación o el usuario que utiliza el documento.

Esta gramática puede hacerse explícita en un esquema. Los autores de documentos XML (humanos o aplicaciones) podrían, usando este esquema, asegurar que sus documentos son válidos conforme a la gramática que éste representa. Además la gramática podría ampliarse sin que ello supusiera invalidar documentos que atendiesen a una especificación anterior más restringida. En este sentido, XML es adaptable: permite definir marcas propias y crear las relaciones estructurales necesarias, especificarlas en un esquema y compartirlo.

La propia especificación XML 1.0 incluye un tipo de esquema denominado DTD (Document Type Definition). Una DTD puede estar contenida en el propio documento o en un fichero externo o repartida en ambos. La DTD para el ejemplo anterior podría ser:

```
<!ELEMENT Company (Delegation+)>
<!ATTLIST Company
  name CDATA #REQUIRED
  founded CDATA #REQUIRED
  income CDATA #REQUIRED
>
<!ELEMENT Delegation (#PCDATA)>
<!ATTLIST Delegation
  name CDATA #REQUIRED
  numberOfEmployees CDATA #REQUIRED
>
```

Un esquema basado en una DTD tiene bastantes limitaciones. Una DTD no permite definir elementos locales que sólo sean válidos dentro de otros elementos. Por ejemplo, si queremos tener un elemento <Manager> que describa al gestor de una compañía o al de una delegación, y la definición de Manager es diferente en cada caso, con una DTD tendríamos que crear los elementos "CompanyManager" y "DelegationManager" para evitar el conflicto de nombres. Es decir, la falta de jerarquía en una DTD obliga a introducir una jerarquía a base de guiones o puntos en el espacio de nombres. En una DTD, es poco flexible la definición de elementos con contenido mixto, es decir, que incluyan otros elementos además de texto. Además no es posible indicar a qué tipo de dato (número, fecha, moneda) ha de corresponder un atributo o el texto de un elemento.

EL W3C Consortium promovió la especificación de un nuevo tipo de esquema: XML Schema (<http://www.w3.org/XML/Schema>) basándose a su vez en propuestas existentes que intentaban superar las limitaciones comentadas: *DDML/Xschema* (Document Definition Markup Language / Xschema <http://www.w3.org/TR/NOTE-ddml>), *DCD* (Document Content Description <http://www.w3.org/TR/NOTE-dcd>), *SOX* (Schema for Object-oriented XML <http://www.w3.org/TR/NOTE-SOX>), ...

Un esquema XML Schema es en sí mismo un documento XML (de lo que se infiere, que existirá a su vez un esquema que especifique la gramática del propio XML Schema: un esquema para esquemas); la DTD del ejemplo anterior no lo es. XML Schema ya incluye la noción de espacios de nombres lo que permite definir elementos con el mismo nombre con características diferenciadas siempre que correspondan a contextos diferentes o a distintos espacios de nombres.

Respecto a la definición de tipos simples de datos de atributos o del contenido textual de los elementos, XML Schema permite utilizar una gama amplia de tipos predefinidos, definir tipos propios, imponer restricciones, por ejemplo, patrones, etc. Además, XML Schema tiene ciertas características heredadas de la orientación a objetos, por ejemplo, la derivación de tipos por extensión o restricción, clases de equivalencia (o grupos de sustitución) y elementos abstractos, etc. Una presentación sobre esquemas del autor de "Café con Leche" expone estos puntos (<http://www.ibiblio.org/xml/slides/xmlonlondon2001/schemas/>).

La especificación XML Schema ha sido elevada por el W3C Consortium a recomendación final en Mayo de 2001 (<http://www.w3.org/XML/Schema>, XML Schema Part 1: Structures <http://www.w3.org/TR/xmlschema-1/>, XML Schema Part 2: Datatypes <http://www.w3.org/TR/xmlschema-2/>). Para iniciarse en XML Schema puede utilizarse un tutorial de Roger L. Costello (<http://www.xfront.com/xml-schema.html>) y finalmente recurrir al documento de la propia especificación XML Schema Part 0: Primer (<http://www.w3.org/TR/xmlschema-0/>).

Los artículos "The W3C XML Schema specification in context" (<http://www.xml.com/lpt/a/2001/01/10/schemasincontext.html>) y "Comparative análisis of six schema languages" (<http://www.cobase.cs.ucla.edu/tech-docs/dongwon/sigmod-record-00.html>) comparan W3C XML Schema con otras propuestas de esquema alternativas: SOX, Relax, Schematron, XDR (Biztalk), etc.

A pesar de la potencia expresiva de XML Schema frente a una DTD, no siempre será descartable utilizar esta última. De hecho, no todo el software que permite analizar XML soporta esquemas XML Schema pero todos los analizadores soportan DTD. Por ejemplo, en aplicaciones donde el control de tipo de datos no sea relevante, conviene definir la o las correspondientes DTDs y validar los documentos XML frente a éstas. Utilizar una DTD es por ahora la solución más portable y la más eficiente puesto que los validadores para XML Schema son más "pesados".

Respecto al uso de XML Schema, su utilización va más allá de la validación de documentos XML. Parafraseando a Ronald Bourrent en la lista de correo xml-dev (<http://www.xml.org/xml-dev/index.shtml>): "La función principal de XML Schema es proveer de metadatos que pueden utilizarse para todo tipo de aplicaciones, por ejemplo: un editor de XML que lee un XML Schema y formatea el interfaz gráfico de usuario en consonancia o que genera las clases Java y el esquema de base de datos correspondientes.

---

## Uso de XML

---

### CREACIÓN

Cualquier documento XML puede representarse en formato texto utilizando caracteres prácticamente en cualquier codificación (encoding) (habitualmente UTF-8). La creación de documentos XML puede realizarse manualmente con cualquier editor de textos, aunque es preferible utilizar editores específicos de XML (véase apartado *Herramientas: Editores*) o mediante programación escribiendo en un fichero a través un stream de salida (véase apartado *Java y XML*).

La cuestión del encoding es suficientemente relevante dado que en una tecnología, como XML, orientada al desarrollo libre de plataforma es clave poder intercambiar la información con fiabilidad, al respecto son de interés los siguientes artículos:

- "How to encode XML data"  
(<http://msdn.microsoft.com/xml/articles/xmlencodings.asp>)
- "Unicode in XML and other markup languages"  
(<http://www.unicode.org/unicode/reports/tr20/tr20-3.html>)

### ANÁLISIS Y MANIPULACIÓN (SAX, DOM)

XML es independiente del lenguaje de programación. Los APIs DOM (Document Object Model, Level 1, 2 & 3, <http://www.w3.org/DOM/>) y SAX (SAX 2.0 Simple API for XML, <http://www.megginson.com/SAX/index.html>) son abiertos e independientes del lenguaje y definen cómo acceder, validar y modificar los documentos XML. Sobre la base de estos APIs se definen los procesadores de XML (parsers) para analizar y validar documentos XML.

SAX es un protocolo de acceso serie a un documento XML basado en eventos. El parser que implementa SAX genera eventos cada vez que se encuentra con una nueva marca XML o con un error (por documento mal formado o inválido). Es cuestión del manejador de eventos, que se registre en el parser, implementar los métodos apropiados para actuar según los eventos. Este API está pensado para leer con rapidez documentos XML y reaccionar en función de su contenido pero no para representarlos en memoria para su presentación o modificación. SAX es de aplicación sobre todo en intercambio de información entre aplicaciones y en algunas ocasiones en la transformación de documentos XML.

DOM es una representación estándar, en memoria, de la estructura de un documento XML y un API para acceder (acceso aleatorio), modificar, eliminar o insertar los elementos y atributos que componen dicho documento. DOM es de aplicación en editores de XML y como soporte de formularios de entrada de datos.

Véase el apartado *Java y XML* respecto a cómo utilizar SAX y DOM en Java.

### VALIDACIÓN

En la recomendación XML 1.0, se tiene en mente la necesidad de utilizar software procesador de XML para leer documentos y acceder a su contenido y estructura. Los programas procesadores o analizadores de XML (parsers) pueden ser no validadores si sólo comprueban que los documentos estén bien formados o validadores si comprueban que el documento sea válido en relación a su esquema.

Como se ha comentado, un esquema especifica cuáles son los elementos, y las etiquetas correspondientes, que pueden incluirse en un documento y cuál es la organización válida de estos elementos. Dada el esquema, se puede comprobar que un documento que se está generando o que se está leyendo tiene una estructura válida.

La mayoría de los procesadores de XML permiten validar con respecto a un esquema basado en DTD, pero hasta la fecha, sólo el procesador Xerces de Apache permite validar frente a un esquema (véase apartado *Java y XML*).

[Nota: Algunos parsers validadores (Xerces), aunque la validación se desactive, cargan el documento y su esquema, particularmente su DTD. Parece razonable que de no requerirse validación, no haga falta procesar tampoco el esquema asociado al documento. Sin embargo, para obtener una representación canónica del documento XML, por ejemplo, al serializar, es necesario incluir los valores por defecto, si los hay, asociados a los atributos y estos sólo se pueden obtener a partir del esquema.]

## TRANSFORMACIÓN

En una estructura en capas, XML constituiría la capa más baja dentro del nivel de aplicación, sobre él los procesadores de XML dan servicio a aplicaciones varias. Estas determinan la presentación de la información. Independientemente del lenguaje y de la plataforma, las aplicaciones pueden compartir documentos a nivel XML.

XML permite separar la forma del contenido. Para un mismo documento XML se pueden definir distintas vistas, cada una apropiada para un fin o dispositivo. Para definir estas vistas puede utilizarse XSL (eXtensible Stylesheet Language) (<http://www.w3.org/Style/XSL/>). (véase apartado *Tecnologías XSL*)

---

## Java y XML

---

Sobre los APIs DOM y SAX se definen los procesadores de XML (parsers) para analizar, manipular y validar documentos XML y también los motores de transformación XSLT.

Los analizadores o parsers de XML pueden implementarse en cualquier lenguaje (Java, C++, Perl, ...). No obstante, hay una tendencia natural a utilizar Java para XML. De hecho, una característica compartida de Java y XML es la independencia de la plataforma. Al utilizar Java para implementar la tecnología XML se obtiene como valor añadido capacidad multiplataforma a nivel binario, de forma que incluso las herramientas que se utilizan para analizar y depurar código XML son independientes de la plataforma. La habilidad de XML para representar datos de forma portable entre plataformas se complementa con la portabilidad inherente de Java. Es más, la característica de carga dinámica de clases de Java permite cambiar de parser Java en tiempo de ejecución para ajustar el rendimiento. XML permite analizadores genéricos para chequeo de errores pues el código consiste en comprobar el documento en relación a un esquema.

Para el entorno Java dos de los parsers más conocidos son:

- Xerces (del proyecto Apache del mismo nombre, <http://xml.apache.org/xerces-j/index.html>). El sitio oficial contiene toda la información necesaria para utilizarlo. Además de un apartado de FAQ y de ejemplos de uso, existe un apartado sobre la implementación de XML Schema incluida en el parser. También se puede encontrar una buena introducción a Xerces en <http://ecerami.com/xerces/>.
- James Clark XP (<http://www.jclark.com/xml/xp/index.html>)

Puesto que el estándar DOM es independiente del lenguaje de programación, su implementación en Java no aprovecha las características de este lenguaje, lo que las hace “pesadas” y ciertamente algo incómodas de usar. Por ello, han surgido modelos de objetos XML en Java alternativos a DOM, entre éstos:

- JDOM (<http://www.jdom.org/>). JDOM utiliza un SAX parser o un DOM parser externo para construir los objetos JDOM que representan el documento XML en memoria. Una introducción a JDOM puede encontrarse en <http://www.ibiblio.org/xml/slides/nypc/jdom/01.html>.
- dom4j (<http://dom4j.org/>). dom4j incluye su propio SAX parser y a diferencia de JDOM también incluye soporte para XPath (véase apartado *Especificaciones relacionadas*).

JAXP (Java API for XML Parsing) ([http://java.sun.com/xml/xml\\_jaxp.html](http://java.sun.com/xml/xml_jaxp.html)) es una propuesta de Sun para permitir que las aplicaciones procesen y transformen documentos XML haciendo que su código sea independiente del procesador XML utilizado. En función de los requerimientos de la aplicación, los desarrolladores pueden intercambiar, en tiempo de ejecución, el procesador a utilizar, por ejemplo, pasar de uno de alto rendimiento a uno más eficiente en el uso de recursos de memoria, sin necesidad de realizar cambios en el código. JAXP está basado en los patrones de diseño Builder y Factory.

Las siguientes referencias de libros y direcciones web pueden ser un buen punto de referencia para iniciarse en la programación XML con Java.

- McLaughlin, B. (2000). “Java and XML”. O’Reilly & Associates, Inc.
- Working with XML: The Java XML tutorial (<http://java.sun.com/xml/jaxp-1.1/docs/tutorial/index.html>)
- Developerlife (<http://65.1.136.127/developerlife/>)
- “Programming XML in Java” (<http://www.javaworld.com/javaworld/jw-03-2000/jw-03-xmlox.html>)

---

## Herramientas

---

En este apartado se hace un compendio de herramientas útiles para la generación, manipulación y gestión de documentos XML.

### ÍNDICES DE SOFTWARE

Free XML Tools and Software (<http://www.garshol.priv.no/download/xmltools/>)

WDVL XML Software guide (<http://wdvl.com/Software/XML/>)

XMLSoftware (<http://www.xmlsoftware.com/>)

### NAVEGADORES

El navegador Internet Explorer desde la versión 5.0 (<http://www.microsoft.com/windows/ie/default.htm>) permite leer documentos XML.

---

## EDITORES

XML Spy (<http://www.xmlspy.com/>) (Hasta la fecha, el más completo editor/validador de XML)

XML Authority y XML Instance (<http://www.extensibility.com/>)

Athens ([http://www.swiftinc.co.jp/index\\_en\\_frame.html](http://www.swiftinc.co.jp/index_en_frame.html))

XML Writer ([http://xmlwriter.net/about\\_us.shtml](http://xmlwriter.net/about_us.shtml))

xmloperator (<http://www.xmloperator.org/>)

## PARSERS

Apache Xerces Java Parser (<http://xml.apache.org/xerces-j/index.html>) (Hasta la fecha, el más completo parser de XML que incluye además soporte de XML Schema)

IBM Java XML Parser (<http://www.alphaworks.ibm.com/formula/xml>) (básicamente una versión comercial de Xerces)

Java NanoXML Parser (<http://nanoxml.sourceforge.net/>) (Un parser con bastantes limitaciones pero muy útil en aplicaciones que no requieran un complejo proceso de validación pero si un tratamiento SAX rápido de documentos XML)

## VALIDACIÓN DE TIPOS

Como se ha comentado más arriba, la utilización de XML Schema puede ir más allá de la validación de documentos XML por un parser. Los metadatos que XML Schema aporta pueden utilizarse para otras aplicaciones, por ejemplo, para configurar el GUI de un editor de XML, o para validar datos cuyos tipos y restricciones se han definido utilizando precisamente XML Schema, sin recurrir a un parser. Es para esta última aplicación para la que pueden resultar de interés los siguientes enlaces:

- IBM Alphaworks Extensible Types (<http://www.alphaworks.ibm.com/tech/etypes>)
- Sun XML Datatypes Library (<http://www.sun.com/software/xml/developers/xsdlib>)
- Sun Multi-Schema XML Validator (<http://www.sun.com/software/xml/developers/multischema>)

## VARIOS

A continuación se relacionan sin orden predefinido algunas utilidades que pueden resultar de interés para incorporarlas en aplicaciones XML o como fuente de inspiración:

- Herramientas de procesamiento de XML y un servidor de datos "open source" 4Suite.org (<http://4suite.org/index.html>)
- Tests de comprobación de conformidad con respecto a especificaciones para XML y tecnologías relacionadas (<http://xmlconf.sourceforge.net/>)
- Un motor de almacenamiento y consulta de documentos XML (GMD-IPSI XQL Engine, <http://xml.darmstadt.gmd.de/xql/>). Según se comenta en la página indicada, GMD-IPSI XQL es un motor basado en Java para almacenamiento de documentos XML y una aplicación de consulta de los mismos. Se fundamenta en dos tecnologías: una implementación para hacer persistentes W3C-DOM objetos Document y una implementación completa del lenguaje de consulta XQL.
- Una herramienta de modelado UML que permite obtener como salida el XML Schema equivalente al modelo definido (<http://www.softera.com/products.htm>)
- Una propuesta de formularios en XML (<http://www.webslingerz.com/balld/xmlform/>)

- Un motor de consulta XML (<http://www.fatdog.com/#30000>)

En IBM Alphaworks pueden encontrarse bastantes utilidades para XML entre ellas:

- Utilidad para obtener una estimación de DTD a partir de documentos XML bien formados (Alphaworks Data Descriptors DDbE, <http://www.alphaworks.ibm.com/tech/DDbE>)
- Utilidad de conversión de DTD a XML Schema (DTD2Schema, [http://www.w3.org/2000/04/schema\\_hack/](http://www.w3.org/2000/04/schema_hack/))

---

## Especificaciones relacionadas

---

En torno a XML hay una extensa lista de especificaciones, no todas implementadas en las aplicaciones de tratamiento de XML, entre ellas:

- XLinks, XPointers, XInclude: XLink define como un documento se puede enlazar con otro. XPointer define como partes concretas de un documento son referenciadas.
  - (<http://www.w3.org/XML/Linking>)
  - (<http://www.ibiblio.org/xml/books/bible/updates/16.html>)
  - XLinks (<http://www.ibiblio.org/xml/books/bible2/chapters/ch19.html>,
  - XPointer (<http://www.ibiblio.org/xml/books/bible2/chapters/ch20.html>)
- XPath (<http://www.w3.org/TR/xpath>) es la sintaxis usada en XSLT y XPointers para obtener nodos particulares del árbol que representa el documento XML en base a ciertos criterios. Puede considerarse casi un lenguaje de consulta.
- XForms (<http://www.w3.org/MarkUp/Forms/>) es un paso hacia una nueva generación de formularios Web basados en XML.
- XML Signature (<http://www.w3.org/Signature/>) tiene que ver con la obtención de una representación canónica de un documento XML. Puede ser de interés "Hash values of XML objects" (<http://www.faqs.org/rfcs/rfc2803.html>)
- XQuery (<http://www.w3.org/TR/xquery/>) es una propuesta de lenguaje de consulta para XML sujeta a evaluación (versión draft). En el artículo "XQuery: Reinventing the wheel?" se hace una crítica constructiva muy interesante de XQuery indicando como pueden resolverse alternativamente consultas en base a XSLT 1.0 y la venidera XSLT 2.0 (<http://www.xmlportfolio.com/xquery.html>).

---

## Aplicaciones

---

XML puede utilizarse en cualquier aplicación en la que se pretenda guardar, recuperar o tratar información estructurada y validar su estructura y contenido, lo que incluye prácticamente cualquier aplicación informática. Parafraseando a Michael Brennan en la lista de correo xml-dev (<http://www.xml.org/xml-dev/index.shtml>):

“El éxito de XML corresponde a un hecho, que los desarrolladores mas avezados conocen desde el principio: la orientación a objetos (OO) no es una panacea que permita resolver todos los problemas. OO es, por ahora, el mejor paradigma de desarrollo de software que se conoce y XML no ha cambiado esto. Pero OO no responde bien en aquellas situaciones en las que la información tiene que ser externa a la aplicación y compartida con otras aplicaciones (que pueden estar ejecutándose en diferentes plataformas, usando lenguajes de programación diferentes, con diferentes librerías disponibles en tiempo de ejecución, etc). Tampoco OO se ajusta bien a modelos de información más dinámicos que no están muy explícitamente definidos en tiempo de desarrollo. XML es muy útil para estas situaciones. OO es sin embargo muy útil como paradigma de programación para trabajar con XML.”

A continuación se citan posibles aplicaciones de XML:

- Publicación

- Front-end de una base de datos
- XML sustituyendo a HTML

XML está listo para ser el formato común en la Web. Toda la infraestructura existente es válida con mínimas modificaciones (véase por ejemplo XHTML).

- XML como formato de documentos científicos y de oficina

- Intercambio de información entre aplicaciones

En aplicaciones distribuidas y, en general, en el intercambio de información entre aplicaciones, se puede considerar XML como protocolo nativo de comunicación. Sobre una capa XML, se monta una capa de procesadores XML que ofrecen a las aplicaciones servicios básicos de tratamiento de los datos XML que intercambian.

Los datos pueden validarse en ambos extremos de la comunicación. Para el intercambio de información con sistemas heredados, se desarrollan adaptadores para que la información de los sistemas heredados se codifique en XML.

Esto es especialmente relevante para aplicaciones de e-commerce.

- Persistencia

- Configuración de aplicaciones

Los datos de configuración de aplicaciones pueden guardarse con formato XML. Esto facilitaría el intercambio de información entre aplicaciones. Por ejemplo, los datos de cuentas de correo o libretas de direcciones se importarían y exportarían con facilidad entre aplicaciones de correo electrónico.

- Serialización de objetos

Para guardar el estado de un objeto se podrían utilizar documentos XML. Esto facilitaría el intercambio de objetos por valor entre aplicaciones distribuidas en distintas plataformas, por ejemplo entre objetos CORBA y objetos RMI.

- Bases de datos XML

A veces no es fácil modelar las relaciones estructurales mediante un diagrama E/R que luego se traduzca en un modelo físico para un cierto RDBMS. Y aunque sea posible la estructura de relaciones puede ser poco intuitiva. Se puede utilizar toda la potencia expresiva de XML para modelar datos fuertemente estructurados como por ejemplo, productos compuestos, incluyendo las relaciones de integridad referencial. Como un documento XML es texto plano es fácil de almacenar en cualquier base de datos.

---

## Temas relacionados

---

### DATA BINDING

JOX (Java Objects in XML) (<http://www.wutka.com/jox.html>)

JXML (<http://www.jxml.com/index.html>)

Castor JO (Java Data Objects) (<http://castor.exolab.org/jdo.html>)

Java Serialization to XML (JSX) (<http://www.csse.monash.edu.au/~bren/JSX/>)

### MESSAGE ORIENTED MIDDLEWARE

XML/CORBA based Message Oriented Middleware (MOM) for Java  
(<http://www.xmlblaster.org/>)

### WEB PUBLISHING FRAMEWORKS

Ver apartado correspondiente en Tecnologías

### XML Y BASES DE DATOS

Artículos

- "XML APIs for databases"  
(<http://developer.java.sun.com/developer/technicalArticles/xml/api/>)
- "XML representation of a relational database" (<http://www.w3.org/XML/RDB.html>)

Soluciones:

- XML-DBMS (Middleware for Transferring Data between XML Documents and Relational Databases) (<http://www.rpbourret.com/xmldbms/index.htm>)
- XML Lightweight extractor (<http://www.alphaworks.ibm.com/tech/xle>)
- RAX Record API for XML (<http://www.oreillynet.com/~rael/data/xml/rax/>)
- Apache Cocoon ESQl Taglib (<http://xml.apache.org/cocoon/esql.html>)
- "Using XML and Relational Databases for Internet Applications"  
(<http://technet.oracle.com/tech/xml/info/index2.htm?Info&htdocs/relational/index.htm>)
- Tamino XML Database (<http://www.software-ag.com/tamino/product/strategy.htm>)

### LENGUAJES XML

Voice XML (<http://www.voicexml.org/>)

WML (<http://www.wapforum.org/>)

## Recursos

---

Los enlaces siguientes corresponden a sitios Web que tienen índices actualizados de recursos sobre XML y que son buenos puntos de referencia para mantenerse al día.

- Café con leche XML News and Resources (<http://www.ibiblio.org/xml/>)
- Sun Java Technology and XML (<http://java.sun.com/xml/>)
- Apache XML (<http://xml.apache.org/>) y las listas de correo de Xerces () y Xalan ()
- XML.org The XML Industry Portal (OASIS)  
([http://www.xml.org/xmlorg\\_resources/index.shtml](http://www.xml.org/xmlorg_resources/index.shtml))
- XML Patterns (<http://www.xmlpatterns.com/patterns.shtml>)
- XML.com (O'Reilly) (<http://www.xml.com/>)
- The XML Cover Pages (OASIS – Robin Cover) (<http://xml.coverpages.org/siteIndex.html>)

# XSL

---

## Introducción

---

XSL Extensible Stylesheet Language (XSL) (<http://www.w3.org/TR/xsl/>) es un lenguaje para transformar documentos XML (XSLT, XSL Transformations Version 1.0 <http://www.w3.org/TR/xslt>) y un vocabulario XML para especificar semántica de formato de documentos (XSL-FO)<sup>1</sup>. Al igual que CSS (Cascade Style Sheets).

XSLT permite decidir la presentación y estilo de los elementos del documento y añade una sintaxis (XPath) para poder procesar los documentos XML de forma más cómoda. Una hoja de transformación XSLT no deja de ser un documento XML cuya sintaxis está definida en la especificación.

Las siguientes referencias de páginas Web son tutoriales de XSLT.

- XML Bible Chapter 14: XSL Transformations (<http://www.ibiblio.org/xml/books/bible/updates/14.html>)
- Roger L. Costello XSL Tutorial (<http://www.xfront.com/xsl.html>)
- Zvon XSL Tutorial (<http://www.zvon.org/HTMLonly/XSLTutorial/Books/Book1/index.html>)
- A guide to XML and XSL for designers ([http://www.webslingerz.com/balld/xsl/designer\\_manual.xml](http://www.webslingerz.com/balld/xsl/designer_manual.xml))
- "Improve your XSLT coding five ways" (<http://www-106.ibm.com/developerworks/library/x-xslt5.html?dwzone=xml>)

Se sugiere utilizar la primera cita como iniciación y las restantes como referencia. En (<http://www.mulberrytech.com/quickref/index.html>) puede encontrarse una plantilla de referencia rápida de XSLT y XPath muy útil en el desarrollo de hojas de transformación.

---

<sup>1</sup> El capítulo 15 de XML Bible trata XSL-FO y está disponible on-line (XML Bible Chapter 15: XSL FO Formatting objects, <http://www.ibiblio.org/xml/books/bible/updates/15.html>).

---

## Uso de XSLT con Java

---

Para transformar un documento XML en otro documento (que puede ser también XML o texto o HTML) basta especificar dicha transformación en lenguaje XSLT en un documento XML (cuya extensión por conveniencia suele ser .xsl) que se denomina hoja de transformación.

A partir del documento fuente y de la hoja de transformación, un motor de transformación XSLT permite obtener el documento de salida. En Java, los motores de transformación (véase apartado *Herramientas: Motores de transformación*) disponen generalmente de una clase para ejecutarla en la máquina virtual desde línea de comandos (algunos como XT incluso disponen de un ejecutable que no requiere una máquina virtual Java).

Obviamente no todas las transformaciones van a realizarse de este modo, sino que se resolverán programáticamente como parte de un componente de la aplicación.

Puesto que XSLT se sirve de XPath, los motores de transformación incluyen un paquete de tratamiento de XPath. Por ejemplo, en el motor de transformación Xalan de Apache se incluye un API XPath que puede utilizarse de forma independiente del motor lo cual resulta útil para recuperar nodos de un árbol DOM en base a determinados criterios.

[Nota: No siempre hay que utilizar XSLT para transformar documentos. A veces, mejora la expresividad y la eficiencia una implementación ad hoc de un ContentHandler que se sirva de un SAX parser (véase el ejemplo Construplaza) para generar un documento resultado a partir del fuente.]

---

## Especificaciones relacionadas

---

XPath (<http://www.w3.org/TR/xpath>) es la sintaxis usada en XSLT para obtener nodos particulares del árbol que representa el documento XML en base a ciertos criterios.

TRAX (Transformation API for XML, <http://xml.apache.org/xalan-j/trax.html>) es una propuesta de API estándar en Java para transformaciones, independiente de XSLT. Uno de los retos de un API de transformación es como tratar las múltiples combinaciones de entradas y salidas sin perder su generalidad.

---

## Herramientas

---

### MOTORES DE TRANSFORMACIÓN

La elección del motor de transformación seguramente vendrá dada por cuestiones de rendimiento o de consumo de recursos. A continuación una lista de los motores más usados:

- Xalan (versión 2.2.D10 <http://xml.apache.org/xalan-j/index.html>)
- Saxon (<http://users.iclway.co.uk/mhkay/saxon/#Scope>)
- XT (<http://www.jclark.com/xml/xt.html>)
- jd.xslt (<http://www.aztecrider.com/xslt/>)

En nuestra experiencia XT ha resultado más rápido que Xalan para ciertas transformaciones y más eficiente en el uso de memoria. Xalan tiene el respaldo de un grupo de desarrollo en Apache y XT está mantenido por su autor James Clark. Comentar también que XT presenta alguna limitación en cuanto al encoding de los documentos generados. Si el documento de salida es XML, entonces sólo se soporta UTF-8 (<http://www.dpawson.co.uk/xsl/N9457.html>).

Una comparativa de motores de transformación puede encontrarse en [http://www.datapower.com/XSLTMark/res\\_2000\\_10\\_22.html#ChartOverall](http://www.datapower.com/XSLTMark/res_2000_10_22.html#ChartOverall)

### VARIOS

A continuación se relacionan sin orden predefinido algunas utilidades que pueden resultar de interés para incorporarlas en aplicaciones de transformación de XML o como fuente de inspiración:

- Utilidades XSLT Jeni's (<http://www.jenitennison.com/xslt/utilities/>), en particular, XSLTDoc que permite explorar hojas de transformación XSLT mostrando resúmenes y explicando cada instrucción XSLT en detalle.
- RenderX XEP Rendering Engine (<http://www.renderx.com/FO2PDF.html>) es un motor que convierte XSL FO en un documento imprimible (PDF o Postscript)
- Sun XSLT Compiler (<http://www.sun.com/xml/developers/xsltc/>) es una herramienta para crear clases Java ligeras y rápidas para transformar documentos XML a partir de una hoja de transformación.