DTD Y SCHEMA XML  
(Mayo 2014)

Raúl Andrés Ortiz Fuentes, 112508.

***Índice de términos— POO: La programación orientada a objetos o POO es un paradigma de programación que usa los objetos en sus interacciones, para diseñar aplicaciones y programas informáticos.***

# INTRODUCCIÓN

E

l XML es un Lenguaje de Etiquetado Extensible muy simple, pero estricto que juega un papel fundamental en el intercambio de una gran variedad de datos. Por la definición anterior es claro entender que se requerí un estándar o formas de modelado para controlar la información que se envía así como su estructura; por tal motivo se han generada el DTD - Definición de Tipo de Documento y el XSD, uno antecesor del otro y que en su momento permitieron la validación de los procesos. Hoy por hoy el formato más utilizado es el Schema XML, que permite una mayor flexibilidad y da respuesta a gran cantidad de necesidades presentadas, cuando se requiere el envió de información. Ac continuación se describirán los conceptos, características y competentes fundamentales del DTD y del XSD.

# DTD

Una DTD es un documento que define la estructura de un documento XML: los elementos, atributos, entidades, notaciones y demás elementos, el orden y el número de veces que pueden aparecer, cuáles pueden ser hijos de cuáles y demás. Por lo anterior el propósito de la norma DTD, o definición del tipo de documento, es definir los elementos constituyentes de un documento en XML. Define la estructura del documento con una lista de elementos establecidos. Las ventajas de la DTD son muchas, permite que: todos los archivos XML tengan una descripción de su propio formato.

Es esencial que la estructura de los documentos que se producen en XML coincida exactamente con la estructura prevista en la DTD, por lo que crear una definición del tipo de documento (DTD) es como crear nuestro propio lenguaje de marcado, para una aplicación específica.

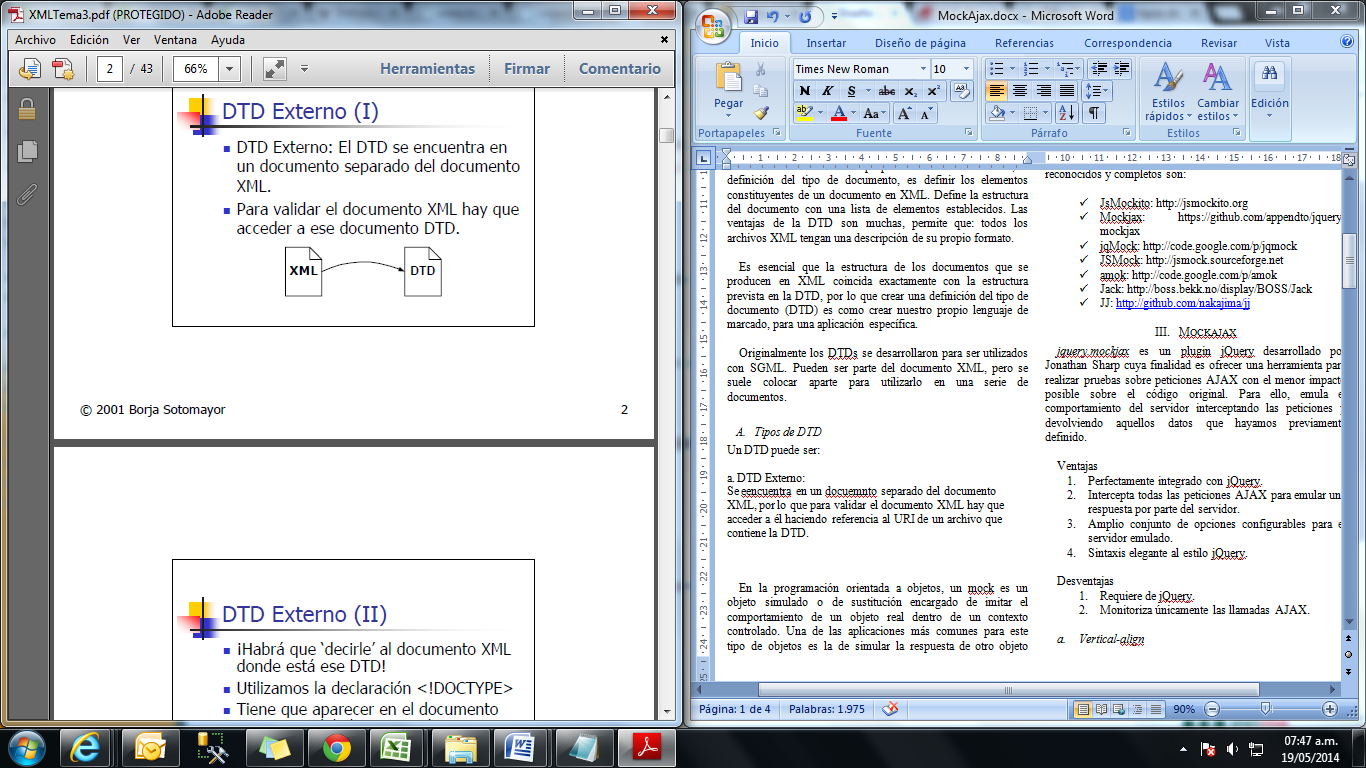
Originalmente los DTD’s se desarrollaron para ser utilizados con SGML. Pueden ser parte del documento XML, pero se suele colocar aparte para utilizarlo en una serie de documentos.

## Tipos de DTD

Un DTD puede ser:

1. *DTD Externo:*

Se encuentra en un documento separado del documento XML, por lo que para validar el documento XML hay que acceder a él haciendo referencia al URI de un archivo que contiene la DTD.

  
Fig. 1 DTD Externo

Entre las ventas del uso de un DTD externo tenemos la posibilidad de que varios documentos XML puedan validarse contra un único documento DTS, sin embargo la mayor desventaja que ofrece es que al estar separado se requiere de más tiempo en el proceso de lectura debido a que se debe realizar la búsqueda y el acceso al mismo siguiendo la ruta definida.

1. *DTD interno:*

El DTD se encuentra en el propio documento XML, por lo que para realizar la validación no hay que acudir a ningún documento separado.

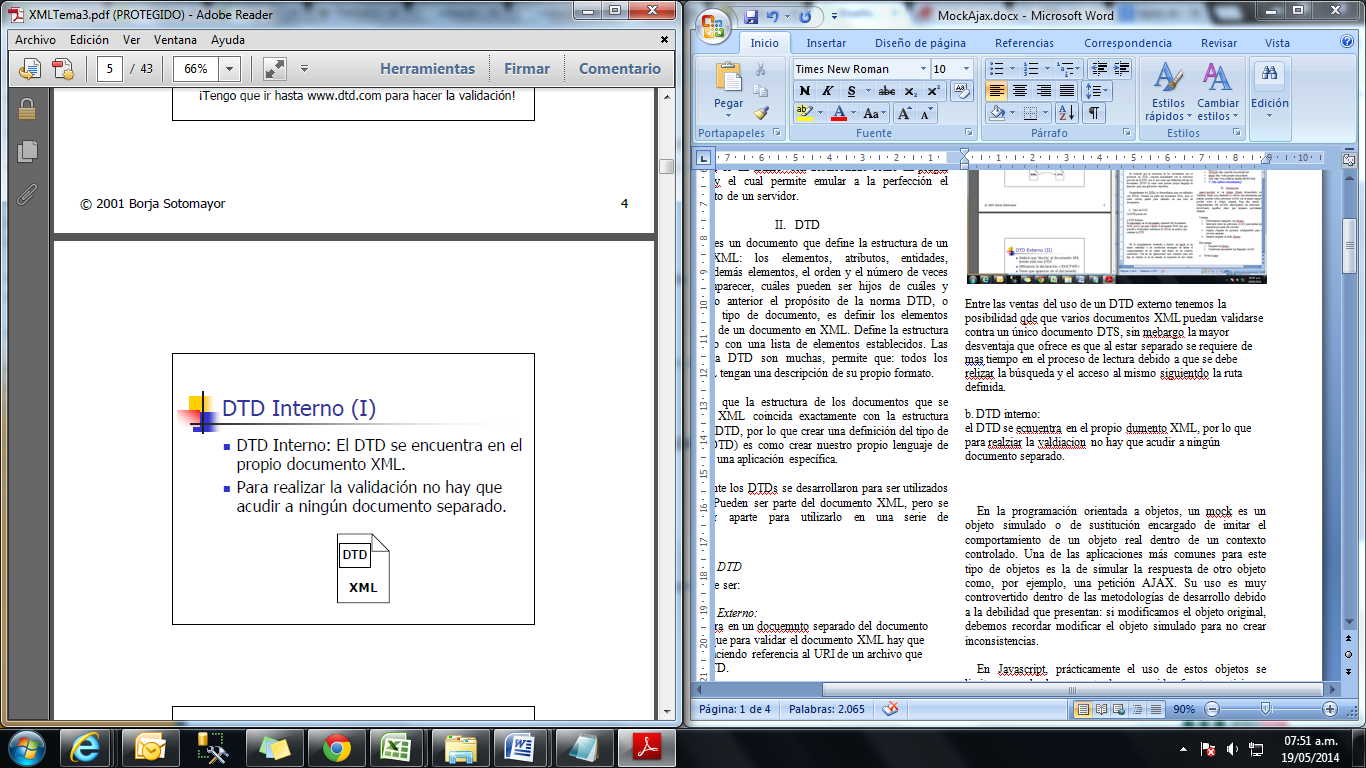


Fig. 2 DTD Interno

A diferencia del DTD externo, este tipo de DTD no puede ser accedido por otros documentos XML, por lo que si se requiere la validación de otros XML habría que repetir el DTD en cada uno de ellos.

1. *DTD Público:*

Cuando DTD goza de la suficiente aceptación se le puede asignar un identificar público. Algunos programas pueden reconocer ese identificador público y es posible que ya tengan almacenado el DTD y no tenga que acudir a Internet para obtenerlo.

En cuanto a su sintaxis se requiere colocar el tag *“PUBLIC*”, seguido del nombre que se le quiera dar.

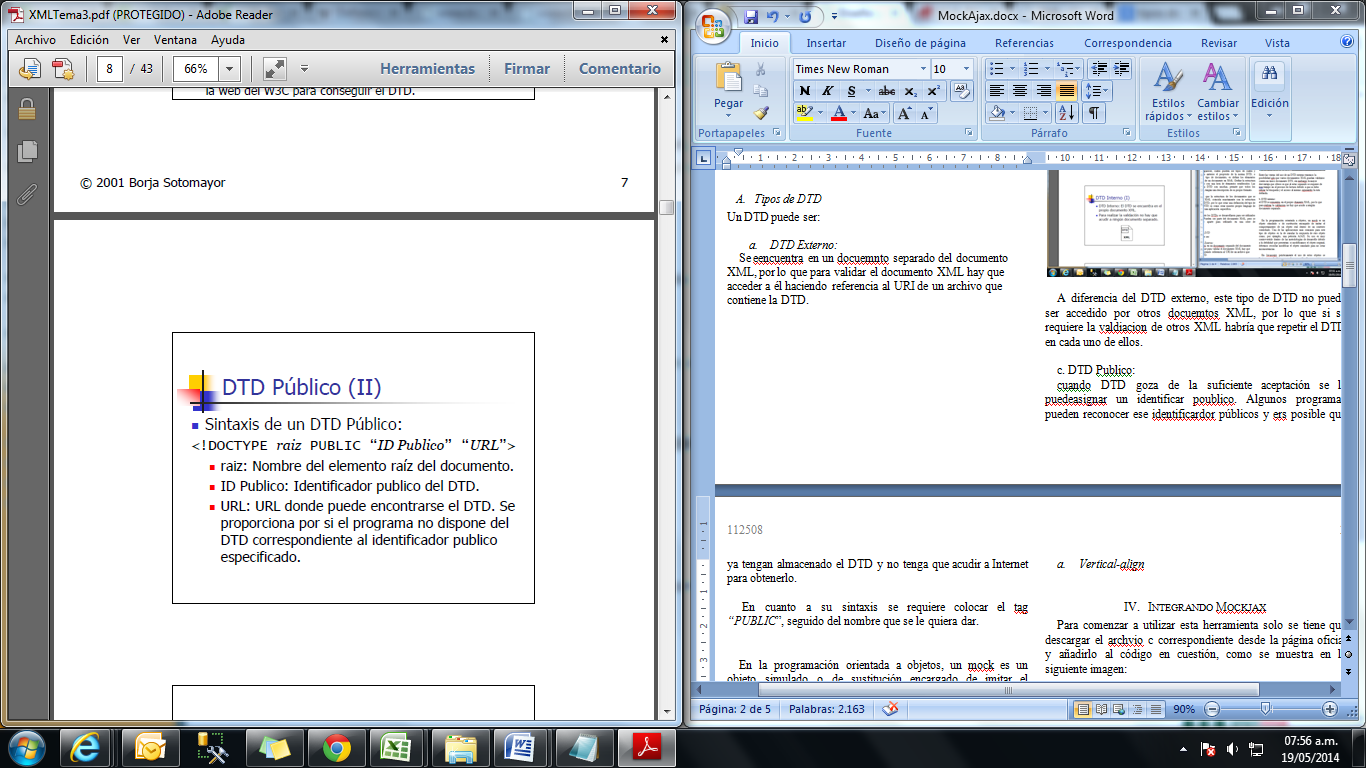


Fig. 3 DTD Interno

## Estructura DTD

Un DTD se compone de una serie de declaraciones. Cada declaración va encerrada entre los caracteres <! y >, y se clasifican en cuatro tipos de declaraciones:

* Elemento (ELEMENT).
* Atributo (ATTLIST).
* Entidad (ENTITY)
* Notación (NOTATION).

1. *Declaraciones de elementos:*

Las declaraciones de los elementos siguen la siguiente sintaxis:

*<!ELEMENT nombreElemento (contenido)>*

En la que "nombreElemento" es el nombre del elemento, y "(contenido)" una expresión que describe el contenido del elemento. Para definir el contenido del elemento se pueden utilizar los términos EMPTY, (#PCDATA) o ANY o escribir expresiones más complejas. [1]

1. *Declaraciones de atributos:*

Indican los atributos permitidos en cada elemento y el tipo o valores permitidos de cada elemento. Una declaración de atributos sigue la siguiente sintaxis:

*<!ATTLIST nombreElemento nombreAtributo tipoAtributo valorInicialAtributo >*

En la que:

* "nombreElemento" es el nombre del elemento para el que se define un atributo.
* "nombreAtributo" es el nombre del atributo.
* "tipoAtributo" es el tipo de datos .
* "valorInicialAtributo" es el valor predeterminado del atributo (aunque también puede indicar otras cosas). [1]

1. *Declaración de entidades*

Una entidad consiste en un nombre y su valor. Con algunas excepciones, el procesador XML sustituye las referencias a entidades por sus valores antes de procesar el documento. Una vez definida la entidad, se puede utilizar en el documento escribiendo una referencia a la entidad, que empieza con el carácter "&", sigue con el nombre de la entidad y termina con ";". [1]

Las entidades pueden ser internas o externas y tanto unas como otras pueden ser generales o paramétricas.

Las declaraciones de entidades internas (generales) siguen la siguiente sintaxis:

*<!ENTITY nombreEntidad "valorEntidad">*

Las declaraciones de entidades externas hacen referencia a un fichero de texto y en ese caso la entidad se sustituye por el contenido del archivo.

La entidad puede ser una entidad de sistema, con la siguiente sintaxis:

*<!ENTITY nombreEntidad SYSTEM "uri">*

1. *Declaración de notaciones*

Las notaciones se usan en XML para definir las entidades externas que no va a analizar el procesador XML. Para hacer referencia estas entidades no se utiliza la notación habitual, sino que se utiliza el nombre de la entidad directamente. [1]

## Limitaciones de la DTD

Una DTD no permite definir elementos locales que sólo sean válidos dentro de otros elementos. Por ejemplo:

* Posee un lenguaje propio de escritura lo que ocasiona problemas a la hora del aprendizaje, pues no sólo hay que aprender XML, sino también conocer el lenguaje de las DTDs. [3]
* Para el procesado del documento, las herramientas y analizadores (parsers) empleados para tratar los documentos XML deben ser capaces de procesar también las DTDs. [3]
* No permite el uso de namespaces y estos son muy útiles ya que permiten definir elementos con igual nombre dentro del mismo contexto, siempre y cuando se anteponga un prefijo al nombre del elemento. [3]
* Tiene una tipología para los datos del documento extremadamente limitada, pues no permite definir el que un elemento pueda ser de un tipo número, fecha, etc. sino que sólo presenta variaciones limitadas sobre cadenas. [3]
* El mecanismo de extensión es complejo y frágil ya que está basado en sustituciones sobre cadenas y no hace explicitas las relaciones, es decir, que dos elemento que tienen definido el mismo modelo de contenido no presentan ninguna relación. [3]

# Schema XML

Al igual que las DTDs, los Schemas describen el contenido y la estructura de la información, pero de una forma más precisa. Los esquemas indican tipos de dato, número mínimo y máximo de ocurrencias y otras características más específicas. En este sentido las posibilidades de control sobre la estructura y los tipos de datos son muy amplias. [3]

Al restringir el contenido de los ficheros XML es posible intercambiar información entre aplicaciones con gran seguridad. Disminuye el trabajo de comprobar la estructura de los ficheros y el tipo de los datos. XML Schema tiene un enfoque modular que recuerda a la programación orientada a objetos y que facilita la reutilización de código. [3]

## Estructura Schema XML

El primer elemento del schema define dos espacios de nombre. El primero "xml-data" le dice al analizador (parser) que esto es un schema y no otro documento XML cualquiera. El segundo "datatypes" nos permite definir el tipo de elementos y atributos utilizando el prefijo "dt". [3]

1. *ElementType:* Define el tipo y contenido de un elemento, incluyendo los sub-elementos que pueda contener.
2. *AttributeType:* Asigna un tipo y condiciones a un atributo.
3. *attribute:* Declara que un atributo previamente definido por AttributeType puede aparecer como atributo de un elemento determinado.
4. *element:* Declara que un elemento previamente definido por ElementType puede aparecer como contenido de otro elemento.

## Tipos de datos

Los tipos de datos tienen en XML Schema la función de las clases en la POO. El usuario puede construir tipos de datos a partir de tipos predefinidos, agrupando elementos y atributos de una determinada forma y con mecanismos de extensión parecidos a la herencia. Los tipos de datos se clasifican en función de los elementos y atributos que contienen. [3]

1. *Tipos simples*

Tipos simples son aquellos que no tienen ni elementos hijos ni atributos.

Específicamente tipos simples son:

* Tipos predefinidos de XML: string, double, boolean, etc.
* List (lista de datos separados por espacios).
* Union (tipo de dato derivado de la unión de tipos predefinidos).

1. *Tipos complejos*

Son tipos complejos aquellos que tienen elementos hijos y/o atributos. Pueden tener nombre o ser anónimos. Si tienen nombre pueden ser reutilizados dentro del mismo XML Schema o por otros XML Schemas. [3]

## Recomendaciones para el uso de Schema XML

Las Recomendaciones establecidas por el W3C en relación a los esquemas XML son las siguientes:

1. *XML Schema Part 0:* Primer. Es un documento no normativo que pretende ofrecer una fácil descripción de las funcionalidades de XML Schema y que está orientado a comprender de forma rápida cómo crear esquemas utilizando el lenguaje XML Schema. [4]
2. *XML Schema Part 1:* Structures. Especifica la definición del lenguaje XML Schema y ofrece las herramientas para describir la estructura y constreñir el contenido de los documentos de XML 1.0, incluyendo las que tratan los espacios de nombre (XML Namespace). El lenguaje de esquemas que se representa en XML usa espacios de nombre, reconstruye sustancialmente y extiende las capacidades de las DTDs de los documentos del lenguaje XML 1.0. [5]
3. *XML Schema Part 2:* Datatypes. Establece las herramientas para definir los tipos de datos que se usan en los esquemas XML y en otras especificaciones XML. El lenguaje de tipo de datos que se representa en XML 1.0, ofrece un conjunto de capacidades que se encontraban en las DTDs en XML 1.0 para especificar tipos de datos sobre elementos y atributos. [6]

## Ventajas del uso de Schema XML

* XML Schema presenta una estructura de tipos mucho más rica. [3]
* Permite tipos definidos por el usuario, llamados Arquetipos. Dando un nombre a estos arquetipos, se pueden usar en distintas partes dentro del Schema. [3]
* Es posible agrupar atributos, haciendo más comprensible el uso de un grupo de aspectos de varios elementos distintos, pero con un denominador común, que deben ir juntos en cada uno de estos elementos. [3]
* Es posible trabajar con espacios de nombre, según la Especificación XML Schema Part 0: Primer, permitiendo validar documentos con varios namespaces. [3]
* Con XML Schema es posible extender Arquetipos de un modo específico, es decir permite lo que en términos de orientación a objetos se llama herencia. [3]

# DTD VS XSD

# Tabla I. Diferencias entre DTD y XSD. [2]

|  |  |
| --- | --- |
| **DTD** | **XSD** |
| Usa una sintaxis distinta al XML. | Usa la misma sintaxis que el XML. |
| Capacidad limitada para expresar tipos de datos. | Gran flexibilidad para expresar tipos de datos. |
| Tipos de datos soportados (similares a los que se encuentran en las bases de datos): 10. | Tipos de datos soportados (similares a los que se encuentran en las bases de datos): más de 44. |
| No permiten crear nuestros propios tipos de datos | Nos permiten crear nuestros propios tipos de datos. |
| No orientado a objetos | Orientado a objetos (pueden extender o restringir un tipo) |
| No puede definir conjuntos | Puede definir conjuntos. |
| No puede especificar elementos únicos (clave) | Puede especificar elementos únicos (clave). |
| No puede definir elementos de contenido nulo | Puede definir elementos de contenido nulo. |
| No puede definir elementos subtitulables. | Puede definir elementos subtitulables. |
| El espacio de nombres en DTD está preasignado, no podemos usar el nuestro propio. | El vocabulario definido por nosotros en los esquemas XML está asociado a un espacio de nombres propio. |
|  | El espacio de nombres de esquemas de XML se toma de www.w3.org. |

# Conclusiones

Podemos concluir que el Schema XML es la evolución y la solución a cada una de las limitaciones que en su momento presento DTD. Esta nueva forma de definición y validación de la estructura del XML, asegura y aumenta la coherencia, precisión y seguridad de la información a enviar, por alta motivo la implementación de dichos estándar es una prioridad cuando se requiera el envió y almacenamiento de información a través de un XML.

Referencias

1. B. Marco, (2014), DTD: Definición de Tipo de Documento. [Online]. Available: http://www.mclibre.org/consultar/xml/lecciones/xml\_dtd.html#L2173.
2. I. Lopez (2008) DTD vs XSD. [Online]. Available: <http://www.tic2.org/WebTecnica/Programacion/DTD/DTDContenido/DTDvsXSD.htm>
3. M. Lamarca Lapuente, (2013), Hipertexto: El nuevo concepto de documento en la cultura de la imagen. [Online]. Available: <http://www.hipertexto.info/documentos/dtds.htm>
4. WC3, (2008), [Online]. Available: <http://www.w3.org/TR/xmlschema-0/>
5. WC3, (2008), [Online]. Available: <http://www.w3.org/TR/xmlschema-1/>
6. WC3, (2008), [Online]. Available: <http://www.w3.org/TR/xmlschema-2/>

**Primer autor** Raúl Andrés Ortiz Fuentes, 112508, Aplicaciones Web Avanzadas.