XML Schema

Introducción

XML Schema Definition Language (XSDL) es un estándar del *World Wide Web Consortium* (W3C) que permite describir la estructura y el contenido de un documento XML. Se trata, por tanto, de una tecnología similar a *Document Type Definition* (DTD), pero mucho más potente y versátil.

Durante un tiempo es de esperar que convivan ambas tecnologías, ya que existen muchos sistemas actuales que se han desarrollado sobre DTD y su uso es más sencillo que XML Schema.

Además de XML Schema, se han desarrollado otras tecnologías para definir esquemas de documentos XML, como *Document Content Description for XML*[1] (DCD), *XML Data Reduce*[2] (XDR), XML-Data[3], y BizTalk.

Ventajas

XML Schema ofrece múltiples ventajas frente a DTD:

1. Tipado fuerte. En un DTD, las posibilidades de limitar el contenido de un elemento o un atributo son limitadas. Por ejemplo, un elemento se puede definir como EMPTY, ANY, un modelo de grupo o contenido mixto (elementos y texto). Pero no hay forma de especificar que el contenido de un elemento tiene que ser un entero o que no puede exceder un cierto

[1] Disponible en `http://www.w3.org/TR/1998/NOTE-dcd-19980731.html`.

[2] Disponible en `http://www.ltg.ed.ac.uk/~ht/XMLData-Reduced.htm`.

[3] Disponible en `http://www.w3.org/TR/1998/NOTE-XML-data-0105/`.

número de caracteres. XML Schema supera las limitaciones de DTD que no permite especificar, por ejemplo, que un elemento debe contener un número entero y no una cadena de caracteres.

2. Tipos de datos similares a los lenguajes de programación y bases de datos. XML Schema incluye una serie de tipos de datos básicos similares a los que se puede encontrar en los lenguajes de programación o en las bases de datos relacionales u orientadas a objetos. Pero, además, el usuario puede definir sus propios tipos de datos a partir de los existentes.
3. Permite controlar con precisión el número de repeticiones. En un DTD, la repetición de un elemento se describe con los siguientes modificadores: nada para una sola vez, ? para cero o una vez, * para cero o más veces y + para una o más veces. En XML Schema se puede especificar cualquier valor para el número de ocurrencias mínimo o máximo.
4. Un XML Schema es un documento XML. Un DTD se escribe mediante una notación que no tiene nada que ver con la sintaxis de un documento XML. Sin embargo, un XML Schema es un documento XML válido, por lo que se pueden emplear con él todas las herramientas desarrolladas para trabajar con XML.
5. Verdadera representación de claves primarias y ajenas. Con DTD, si se quiere representar una base de datos en un documento XML, la única forma de especificar claves primarias y ajenas es con los tipos ID e IDREF. Sin embargo, presenta dos graves inconvenientes: ID tiene que ser único en todo el documento (en una base de datos, una clave primaria se puede repetir en distintas tablas) e IDREF tiene que referenciar un ID dentro del propio documento, pero no se puede especificar el tipo de elemento (puede ser una clave ajena a cualquier tabla). Con XML Schema, estos conceptos se pueden representar sin problemas.

Qué necesito para usar XML Schema

La extensión de un XML Schema suele ser `.xsd`. Se deben emplear nombres cortos y sencillos. Hay que evitar el uso de espacios o de caracteres especiales en el nombre del archivo y también controlar el uso de mayúsculas y minúsculas puesto que en Internet existen multitud de sistemas operativos, que no pueden aceptar los mismos nombres de archivo que acepta el nuestro. Por ejemplo, hay sistemas operativos en los que las mayúsculas y minúsculas se distinguen y otros donde no.

Diseño de un documento XML

En la actualidad, el principal uso que se le da a XML es como medio de integración entre distintas aplicaciones. La integración de XML con los sistemas gestores de bases de datos relacionales (SGBDR) es cada vez mayor y XML se ha convertido en la *lingua franca* para traspasar información de una aplicación a otra.

Tanto XML como las bases de datos relacionales se asocian con la representación de entidades (objetos) de negocio, pero lo hacen de forma diferente, tal como se muestra en la siguiente tabla.

	SGBDR	**XML**
Entidad	Tabla	Documento
Instancia de entidad	Fila	Elemento
Atributo de entidad	Columna	Atributo Valor de elemento Subelemento

La principal diferencia entre SGBDR y XML reside en la representación de las características (atributos) de una entidad. En un SGBDR, las características de la entidad se reprendan por columnas de una tabla. En un documento XML, las características pueden ser atributos, valores de elemento o subelementos.

Por ejemplo, la siguiente tabla relacional se puede representar de distintas formas mediante XML:

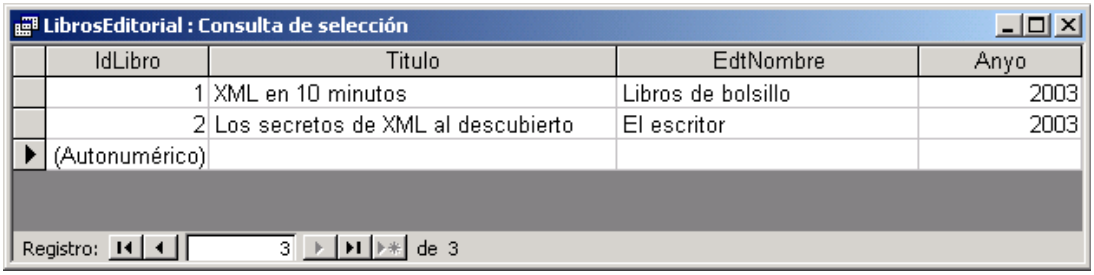

Figura 1

Para representar esta tabla en XML, se podría crear un documento XML llamado `Libros` con dos elementos `Libro`. Las columnas podrían representarse *centradas en el atributo* de forma que las columnas de una tabla se asocian a los atributos de un documento XML, como se muestra en el siguiente ejemplo:

```
<Libros>
  <Libro IdLibro="1" Titulo="XML en 10 minutos"
EdtNombre="Libros de bolsillo" Anyo="2003" />
  <Libro IdLibro="2" Titulo="Los secretos de XML al descubierto"
EdtNombre="El escritor" Anyo="2003" />
</Libros>
```

Como alternativa, se podría utilizar una relación *centrada en el elemento*. En esta relación, se devuelven todas las columnas como subelementos del elemento representado en la tabla a la que pertenece, como se muestra en el siguiente ejemplo:

```
<Libros>
 <Libro>
   <IdLibro>1</IdLibro>
   <Titulo>XML en 10 minutos</Titulo>
   <EdtNombre>Libros de bolsillo</EdtNombre>
   <Anyo>2003</Anyo>
 </Libro>
 <Libro>
   <IdLibro>2</IdLibro>
   <Titulo>Los secretos de XML al descubierto</Titulo>
   <EdtNombre>El escritor</EdtNombre>
   <Anyo>2003</Anyo>
 </Libro>
</Libros>
```

Por último, también se podrían mezclar ambas aproximaciones:

```
<Libros>
 <Libro IdLibro="1">
   XML en 10 minutos
   <EdtNombre>Libros de bolsillo</EdtNombre>
```

```
   <Anyo>2003</Anyo>
 </Libro>
 <Libro IdLibro=”2”>
   Los secretos de XML al descubierto
   <EdtNombre>El escritor</EdtNombre>
   <Anyo>2003</Anyo>
 </Libro>
</Libros>
```

En este ejemplo se utilizan las tres formas de representar las características de una entidad. `IdLibro` se representa como un atributo, `Titulo` se representa como el valor del elemento que representa la instancia de la entidad, y `EdtNombre` y `Anyo` como subelementos.

El primer XML Schema

Vamos a definir un XML Schema para el siguiente documento XML. Para ello, vamos a seguir un método incremental.

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<!DOCTYPE BIBLIOTECA SYSTEM "biblioteca.dtd">
<BIBLIOTECA>
   <LIBRO COD="1">
      <TITULO>XML para todos</TITULO>
      <AUTOR>Sergio Lujan Mora</AUTOR>
      <ANYO>2001</ANYO>
      <EDITORIAL>UA Prensa</EDITORIAL>
   </LIBRO>
   <LIBRO COD="11">
      <TITULO>Como aprobar una oposición</TITULO>
      <AUTOR>Marisa Zayas Fornieles</AUTOR>
      <AUTOR>Sergio Lujan Mora</AUTOR>
      <ANYO>1999</ANYO>
      <EDITORIAL>Prensa Editorial</EDITORIAL>
   </LIBRO>
</BIBLIOTECA>
```

El DTD de este documento es:

```
<!ELEMENT BIBLIOTECA (LIBRO+)>
<!ELEMENT LIBRO (TITULO, AUTOR*, ANYO?, EDITORIAL?)>
<!ATTLIST LIBRO COD CDATA #REQUIRED>
<!ELEMENT AUTOR (#PCDATA)>
<!ELEMENT TITULO (#PCDATA)>
<!ELEMENT EDITORIAL (#PCDATA)>
<!ELEMENT ANYO (#PCDATA)>
```

Un esquema para este documento se puede escribir simplemente siguiendo la estructura del documento y definiendo cada elemento conforme se encuentran. Un XML Schema es un documento XML con la siguiente estructura básica (el elemento raíz es `xs:schema`):

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
...
...
</xs:schema>
```

Para el elemento raíz del documento (BIBLIOTECA), definimos un elemento llamado BIBLIOTECA. Este elemento no tiene atributos y no contiene texto, así que lo consideramos de tipo `complexType` (ya que el otro tipo `simpleType` se emplea para elementos que sólo contienen valores y no poseen ni elementos ni atributos. La lista de elementos que puede contener BIBLIOTECA se define mediante `sequence`:

```
<xs:element name="BIBLIOTECA">
 <xs:complexType>
   <xs:sequence>
   …
   </xs:sequence>
 </xs:complexType>
</xs:element>
```

El elemento `sequence` define una secuencia ordenada de sub-elementos. Otros elementos similares de XML Schema son `choice` que define un grupo de elementos mutuamente excluyentes y `all` que define un grupo no ordenado de elementos.

A continuación, se puede definir el elemento LIBRO, que se tiene que definir como un tipo complejo, ya que contiene subelementos:

```
<xs:element name="LIBRO" minOccurs="1" maxOccurs="unbounded">
 <xs:complexType>
   <xs:sequence>
```

```
   …
   </xs:sequence>
 </xs:complexType>
</xs:element>
```

Al contrario que otros lenguajes de definición de esquemas, XML Schema permite definir la cardinalidad de un elemento (el número posible de ocurrencias) con gran precisión. Se pueden especificar tanto el número de ocurrencias (`minOccurs`) como el máximo número de ocurrencias (`maxOccurs`). En este ejemplo, `maxOccurs` toma el valor `unbounded`, que significa que pueden existir tantas ocurrencias de este elemento como se quiera (no hay un límite superior). Si no se indica un valor, ambos atributos tienen 1 como valor por defecto.

A continuación, se pueden definir los elementos TITULO, AUTOR, ANYO y EDITORIAL como tipos simples, ya que no poseen atributos o subelementos y se pueden describir directamente con un elemento `element` degenerado (vacío). El tipo de estos elementos es `xs:string` (una cadena cualquiera de caracteres) y aparece prefijado con el prefijo del espacio de nombres asociado con XML Schema, para indicar que se trata de un tipo de dato predefinido de XML Schema:

```
<xs:element name="TITULO" type="xs:string" minOccurs="1"
maxOccurs="1" />
<xs:element name="AUTOR" type="xs:string" minOccurs="0"
maxOccurs="unbounded" />
<xs:element name="ANYO" type="xs:string" minOccurs="0"
maxOccurs="1" />
<xs:element name="EDITORIAL" type="xs:string" minOccurs="0"
maxOccurs="1" />
```

Como el valor por defecto de `minOccurs` y `maxOccurs` es 1, también se podría haber escrito:

```
<xs:element name="TITULO" type="xs:string" />
<xs:element name="AUTOR" type="xs:string" minOccurs="0"
maxOccurs="unbounded" />
<xs:element name="ANYO" type="xs:string" minOccurs="0" />
<xs:element name="EDITORIAL" type="xs:string" minOccurs="0" />
```

Finalmente, la declaración de los atributos de los elementos tiene que aparecer al final. No existen unas razones justificadas para este orden, pero parece que el *W3C XML Schema Working Group* consideró que era más simple imponer un orden relativo a la definición de la lista de elementos y atributos dentro de un tipo complejo, y que era más natural definir los atributos después de los elementos. De este modo, la definición del atributo COD de LIBRO es:

```
<xs:attribute name="COD" type="xs:string" />
```

Este diseño de un XML Schema se conoce como "Diseño de muñecas rusas" (*Russian Doll Design*), ya que sigue estrictamente la estructura del documento XML: cada elemento y atributo se define en el contexto donde se emplea en el documento XML.

Juntando todas las piezas, el XML Schema quedaría de la siguiente forma:

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="BIBLIOTECA">
<xs:complexType>
<xs:sequence>
<xs:element name="LIBRO" minOccurs="1" maxOccurs="unbounded">
<xs:complexType>
<xs:sequence>
<xs:element name="TITULO" type="xs:string" />
<xs:element name="AUTOR" type="xs:string" minOccurs="0"
maxOccurs="unbounded" />
<xs:element name="ANYO" type="xs:string" minOccurs="0" />
<xs:element name="EDITORIAL" type="xs:string" minOccurs="0" />
</xs:sequence>
<xs:attribute name="COD" type="xs:string" />
</xs:complexType>
</xs:element>
</xs:sequence>
</xs:complexType>
</xs:element>
```

```
</xs:schema>
```

Tipos de datos

En la construcción del XML Schema anterior, se ha empleado el atributo `type` para definir el contenido de los elementos que contienen texto y el tipo de los atributos. En XML Schema existe una jerarquía de tipos de datos, tal como se muestra en la Figura 2.

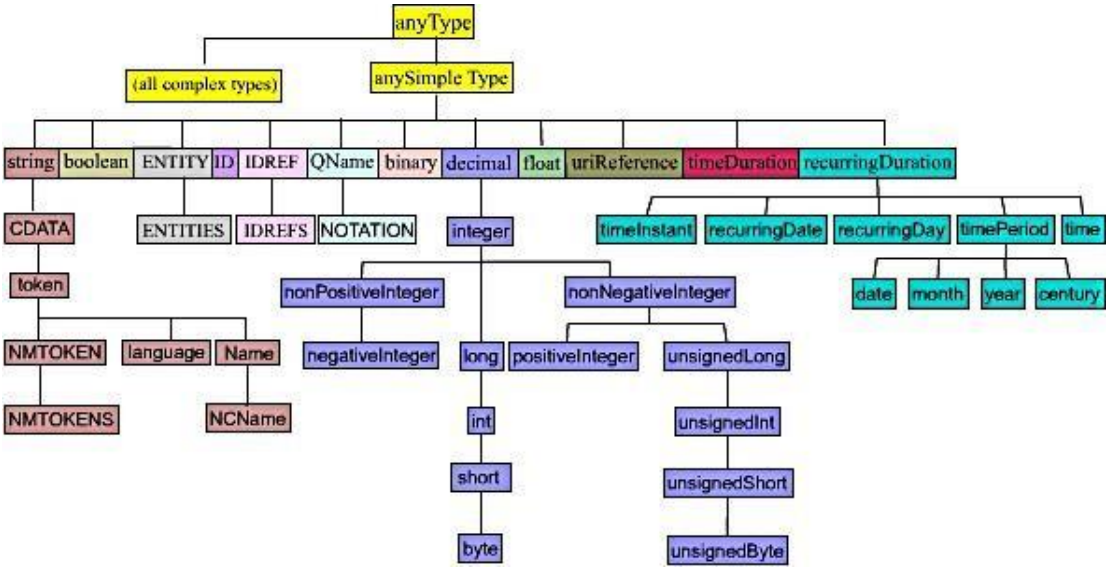

Figura 2

Ejercicio Esquemas XML: Biblioteca

Ejemplo 1: Tipos simples

ejemplo1.xml

```
<?xml version="1.0" encoding="utf-8"?>
<nota>hola</nota>
```

ejemplo1.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
        <xs:element name="nota" type="xs:string"/>
</xs:schema>
```

Ejemplo 2: Tipos complejos

```
<libro>
        <autor>Miguel de Cervantes Saavedra</autor>
        <titulo>El Quijote de la Mancha</titulo>
</libro>
```

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="libro">
        <xs:complexType>
                <xs:sequence>
                        <xs:element name="autor" type="xs:string" />
                        <xs:element name="titulo" type="xs:string" />
                </xs:sequence>
        </xs:complexType>
        </xs:element>
</xs:schema>
```

Ejemplo 3: Tipos complejos II

```
<biblioteca>
        <libro>
                <autor>Miguel de Cervantes Saavedra</autor>
                <titulo>El Quijote de la Mancha</titulo>
        </libro>
        <libro>
                <autor>Pablo Neruda</autor>
                <titulo>Veinte poemas de amor y una canción desesperada</titulo>
        </libro>
</biblioteca>
```

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="biblioteca">
        <xs:complexType>
                <xs:sequence>
                        <xs:element name="libro" minOccurs="0"
maxOccurs="unbounded">
                                <xs:complexType>
                                        <xs:sequence>
                                                <xs:element name="autor" />
                                                <xs:element name="titulo" />
                                        </xs:sequence>
                                </xs:complexType>
                        </xs:element>
                </xs:sequence>
        </xs:complexType>
</xs:element>
</xs:schema>
```

Ejemplo 4: Restricciones

```
<?xml version="1.0" encoding="utf-8"?>
<biblioteca>
        <libro>
                <autor>Miguel de Cervantes Saavedra</autor>
                <titulo>El Quijote de la Mancha</titulo>
                <codigo>123</codigo>
    </libro>
        <libro>
                <autor>Pablo Neruda</autor>
                <titulo>Veinte poemas de amor y una canción desesperada</titulo>
                <codigo>124</codigo>
        </libro>
</biblioteca>
```

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="biblioteca">
        <xs:complexType>
                <xs:sequence>
                        <xs:element name="libro" minOccurs="0"
maxOccurs="unbounded">
                                <xs:complexType>
                                        <xs:sequence>
                                                <xs:element name="autor" />
                                                <xs:element name="titulo" />
                                                <xs:element name="codigo">
                                                        <xs:simpleType>

        <xs:restriction base="xs:integer">

        <xs:minInclusive value="1"/>

        <xs:maxInclusive value="9999"/>

        </xs:restriction>
                                                                </xs:simpleType>
                                                </xs:element>
                                        </xs:sequence>
                                </xs:complexType>
                        </xs:element>
                </xs:sequence>
        </xs:complexType>
</xs:element>
</xs:schema>
```

Ejemplo 5: Restricciones II

```xml
<?xml version="1.0" encoding="utf-8"?>
<biblioteca>
        <libro>
                <autor>Miguel de Cervantes Saavedra</autor>
                <titulo>El Quijote de la Mancha</titulo>
                <codigo>123</codigo>
                <ubicacion>estantería 1</ubicacion>
        </libro>
        <libro>
                <autor>Pablo Neruda</autor>
                <titulo>Veinte poemas de amor y una canción desesperada</titulo>
                <codigo>124</codigo>
                <ubicacion>estantería 11</ubicacion>
        </libro>
</biblioteca>
```

```xml
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="biblioteca">
        <xs:complexType>
                <xs:sequence>
                        <xs:element name="libro" minOccurs="0"
maxOccurs="unbounded">
                                <xs:complexType>
                                        <xs:sequence>
                                                <xs:element name="autor" />
                                                <xs:element name="titulo" />
                                                <xs:element name="codigo">
                                                        <xs:simpleType>

        <xs:restriction base="xs:integer">

        <xs:minInclusive value="1"/>

        <xs:maxInclusive value="9999"/>

        </xs:restriction>
                                                        </xs:simpleType>
                                                </xs:element>
                                                <xs:element name="ubicacion">
                                                        <xs:simpleType>

        <xs:restriction base="xs:string">

        <xs:enumeration value="estantería 1"/>

        <xs:enumeration value="estantería 2"/>

        <xs:enumeration value="estantería 3"/>

        <xs:enumeration value="estantería 4"/>

        <xs:enumeration value="estantería 5"/>

        <xs:enumeration value="estantería 6"/>

        <xs:enumeration value="estantería 7"/>

        <xs:enumeration value="estantería 8"/>
```

```
        <xs:enumeration value="estantería 9"/>

        <xs:enumeration value="estantería 10"/>

        <xs:enumeration value="estantería 11"/>

        <xs:enumeration value="estantería 12"/>

        <xs:enumeration value="estantería 13"/>

        <xs:enumeration value="estantería 14"/>

        </xs:restriction>
                                                                </xs:simpleType>
                                                        </xs:element>
                                                </xs:sequence>
                                        </xs:complexType>
                                </xs:element>
                        </xs:sequence>
        </xs:complexType>
</xs:element>
</xs:schema>
```

Ejemplo 6: Atributos

```
<?xml version="1.0" encoding="utf-8"?>
<biblioteca>
        <libro codigo="123" ubicacion="estantería 1">
                <autor>Miguel de Cervantes Saavedra</autor>
                <titulo>El Quijote de la Mancha</titulo>
        </libro>
        <libro codigo="1023" ubicacion="estantería 3">
                <autor>Pablo Neruda</autor>
                <titulo>Veinte poemas de amor y una canción desesperada</titulo>
        </libro>
</biblioteca>
```

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="biblioteca">
        <xs:complexType>
                <xs:sequence>
                        <xs:element name="libro" minOccurs="0"
maxOccurs="unbounded">
                                <xs:complexType>
                                        <xs:sequence>
                                                <xs:element name="autor" />
                                                <xs:element name="titulo" />
                                        </xs:sequence>
                                        <xs:attribute name="codigo">
                                                <xs:simpleType>
                                                        <xs:restriction
base="xs:integer">

        <xs:minInclusive value="1"/>
```

```
        <xs:maxInclusive value="9999"/>
                                                                </xs:restriction>
                                                        </xs:simpleType>
                                                </xs:attribute>
                                                <xs:attribute name="ubicacion">
                                                        <xs:simpleType>
                                                                <xs:restriction
base="xs:string">

        <xs:enumeration value="estantería 1"/>

        <xs:enumeration value="estantería 2"/>

        <xs:enumeration value="estantería 3"/>

        <xs:enumeration value="estantería 4"/>

        <xs:enumeration value="estantería 5"/>

        <xs:enumeration value="estantería 6"/>

        <xs:enumeration value="estantería 7"/>

        <xs:enumeration value="estantería 8"/>

        <xs:enumeration value="estantería 9"/>

        <xs:enumeration value="estantería 10"/>

        <xs:enumeration value="estantería 11"/>

        <xs:enumeration value="estantería 12"/>

        <xs:enumeration value="estantería 13"/>

        <xs:enumeration value="estantería 14"/>
                                                                </xs:restriction>
                                                        </xs:simpleType>
                                                </xs:attribute>
                                        </xs:complexType>
                                </xs:element>
                        </xs:sequence>
        </xs:complexType>
</xs:element>
</xs:schema>
```