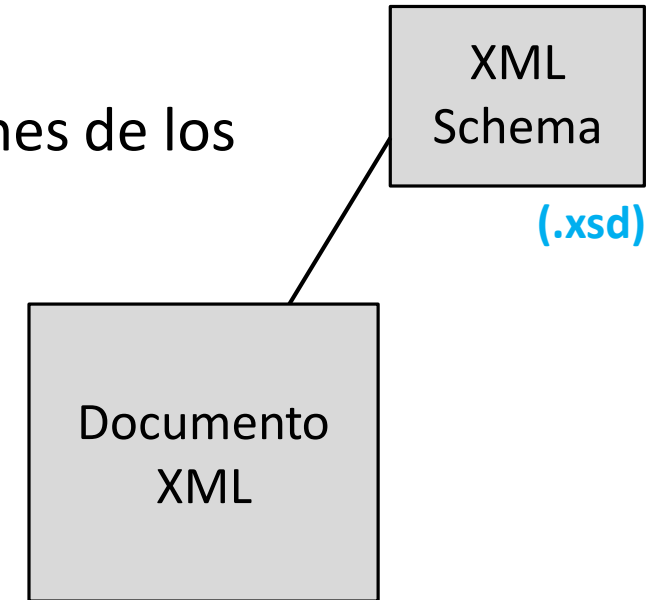


XML Schema (parte 1)

LENGUAJES DE MARCAS Y SISTEMAS DE GESTIÓN DE INFORMACIÓN

Introducción / Recordatorio

- XML sirve como estándar para el intercambio de datos entre aplicaciones, independientemente de su formato almacenado.
- **XML Schema** sirve para describir la estructura y las restricciones de los contenidos de los documentos XML (al igual que DTD).
- Sin embargo, XML Schema es más potente que DTD. Hace lo mismo, pero de una forma más detallada y describiendo mejor los documentos a nivel formal.



Referenciar un .xsd desde un XML

`<elemento_raiz xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="archivo.xsd">`

catalogo.xml

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<catalogo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="esquema.xsd">
    <libro>...</libro>
    ...
</catalogo>
```

esquema.xsd



XML Schema → Elemento raíz

Todo el contenido del XML Schema irá dentro de este elemento raíz:

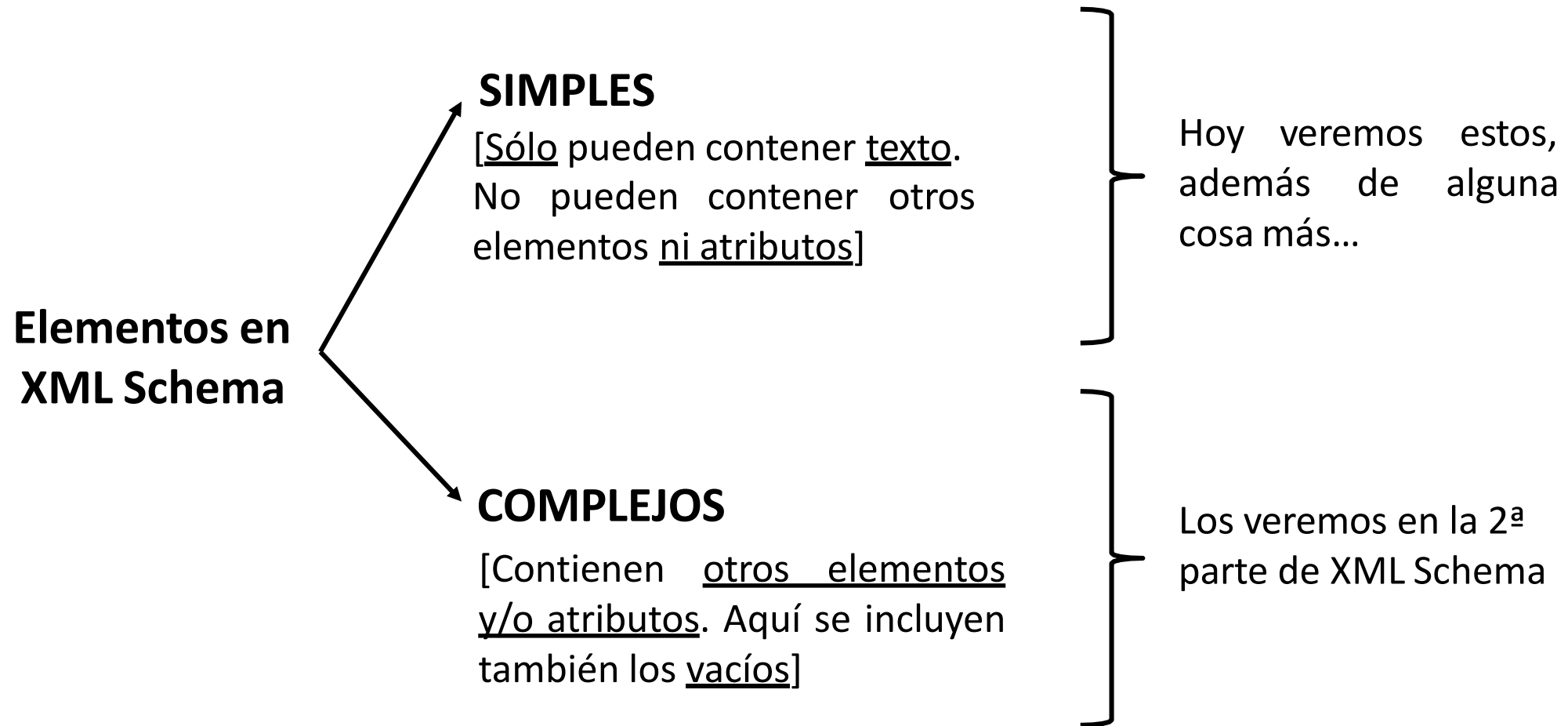
```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
```

```
...
```

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

Ahora iremos viendo todo lo que podemos declarar en nuestro XML Schema...

XML Schema → Elementos



XML Schema → Elementos simples

Según XML Schema, los elementos simples sólo pueden contener texto. No pueden contener otros elementos ni atributos.

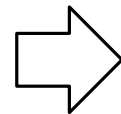
Dicho de otra manera, un elemento es simple si contiene texto y no tiene atributos.

<catalogo>

<libro>...</libro>

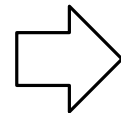
 ...

</catalogo>



¿Se considera **catalogo** un elemento simple? **NO**

<oferta/>



¿Se considera **oferta** un elemento simple? **NO**

XML Schema → Elementos simples

Según XML Schema, los elementos simples sólo pueden contener texto. No pueden contener otros elementos ni atributos.

Dicho de otra manera, un elemento es simple si contiene texto y no tiene atributos.

`<nombre idioma="inglés">Will</nombre>` ➡ ¿Se considera **nombre** un elemento simple? NO

`<apellido>Smith</apellido>` ➡ ¿Se considera **apellido** un elemento simple? SÍ

`<edad>23</edad>` ➡ ¿Se considera **edad** un elemento simple? SÍ

XML Schema → Elementos simples

Los elementos simples se definen así:

```
<xs:element name="nombre_elemento" type=" " />
```

Otra forma de verlo es: un elemento simple es aquel que sólo contiene un valor de uno de los tipos de datos simples definidos en XML Schema (ahí se muestran los principales, hay más).

El valor de un xs:decimal tendrá el formato *3.14*

El valor de un xs:boolean tendrá el formato *true*

El valor de un xs:date tendrá el formato *2018-05-23*

El valor de un xs:time tendrá el formato *17:54:31*

[Podrían modificarse, pero no vamos a hilar tan fino]

- xs:string
- xs:decimal
- xs:integer
- xs:boolean
- xs:date
- xs:time

XML Schema → Elementos simples

XML

```
...  
<fechaNac>1987-04-18</fechaNac>  
...
```



XML Schema

```
<xs:element name="fechaNac" type="xs:date"/>
```

XML Schema → Elementos simples

A los elementos simples se les podría añadir:

- **default="valor_por_defecto"** → Para darle un contenido por defecto
- **fixed="valor_fijo"** → Para darle un contenido fijo

Ejemplos:

- `<xs:element name="notaMedia" type="xs:decimal" default="5.0"/>>`
En este caso, el elemento notaMedia, tendría de contenido por defecto 5.0
- `<xs:element name="enVenta" type="xs:boolean" fixed="true"/>>`
En este caso, el elemento enVenta siempre debe tener de contenido true

XML Schema → Elementos

Elementos en XML Schema

SIMPLES

[Sólo pueden contener texto.
No pueden contener otros
elementos ni atributos]

```
<xs:element name="nombre_elemento" type=" " />
```

Se puede añadir

- default="valor_por_defecto"
- fixed="valor_fijo"

xs:string
xs:decimal
xs:integer
xs:boolean
xs:date
xs:time

COMPLEJOS

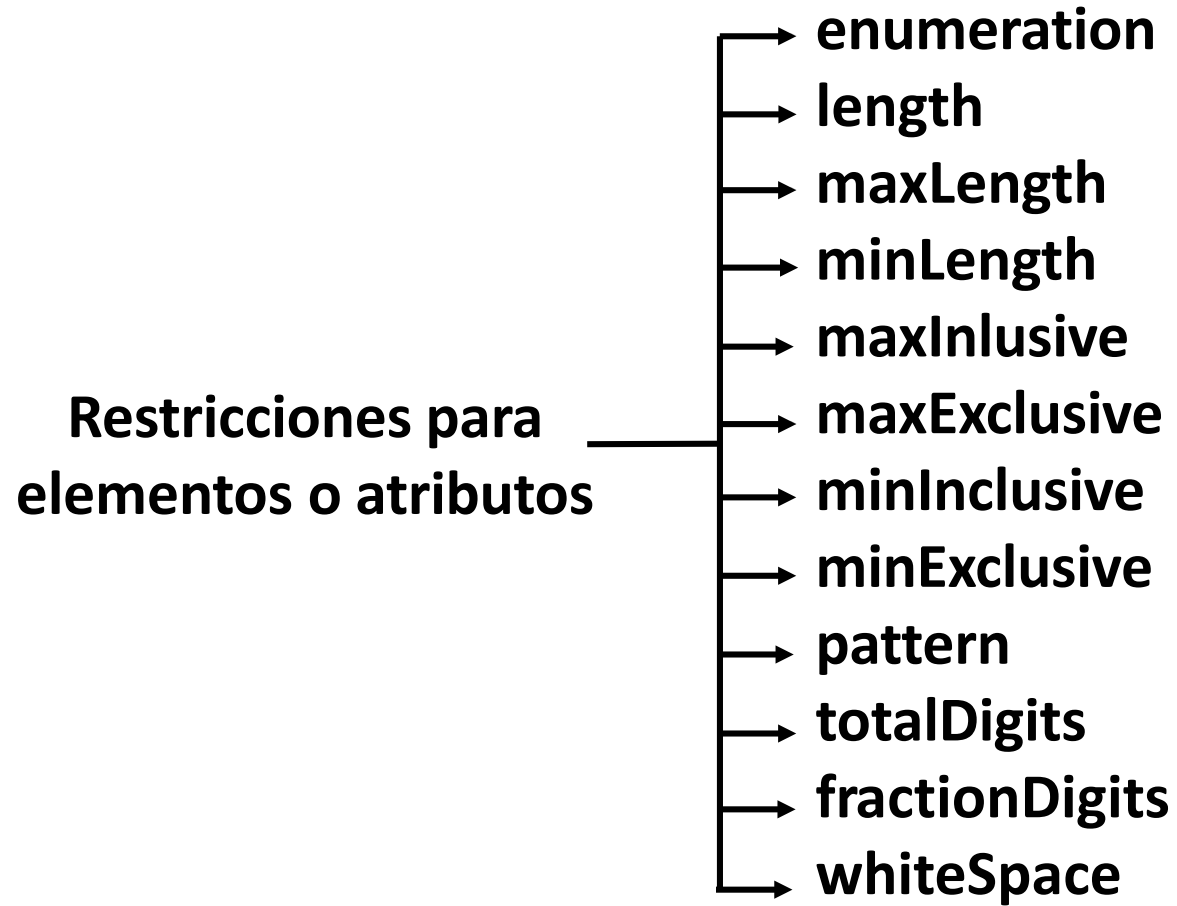
[Contienen otros elementos y/o atributos. Aquí se incluyen también los vacíos]

XML Schema → Restricciones

Las restricciones (**restriction**) se utilizan para definir valores aceptables de los elementos o atributos XML. Dicho de otra manera, y como su propio nombre indica, para restringir el contenido de un elemento/atributo.

Hay muchos tipos de restricciones (¡12!), que veremos a continuación...

XML Schema → Restricciones



XML Schema → Restricciones → enumeration

Con **enumeration** restringimos el valor del elemento/atributo a un conjunto de valores.

```
<xs:element name="marcaCoche">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="Audi"/>
      <xs:enumeration value="KIA"/>
      <xs:enumeration value="BMW"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

En este caso, el elemento **marcaCoche** sólo podrá tener como contenido uno de estos textos: *"Audi"*, *"KIA"* o *"BMW"*

Si te fijas, marcaCoche es de tipo **xs:string**, pero eso ya no se pone arriba. Cuando hay restricción, el tipo se pone dentro de la restricción. Cuidado...

XML Schema → Restricciones → enumeration

XML

```
...  
<marcaCoche>Mercedes</marcaCoche>  
...
```



XML Schema

```
<xs:element name="marcaCoche">  
  <xs:simpleType>  
    <xs:restriction base="xs:string">  
      <xs:enumeration value="Audi"/>  
      <xs:enumeration value="KIA"/>  
      <xs:enumeration value="BMW"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

XML Schema → Restricciones → length, maxLength...

Para limitar la longitud del contenido de un elemento/atributo, tenemos las restricciones **length**, **maxLength** y **minLength**.

```
<xs:element name="iniciales">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:length value="3"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

El contenido del elemento **iniciales** (tipo **xs:string**) deberá ser de **3 caracteres**

```
<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:minLength value="5"/>
      <xs:maxLength value="8"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

(Como ves, se pueden agrupar las restricciones)

El contenido del elemento **password** (tipo **xs:string**) deberá tener **entre 5 y 8 caracteres**

XML Schema → Restricciones → length, maxLength...

XML

```
...  
<password>hola123</password>  
...
```



XML Schema

```
<xs:element name="password">  
  <xs:simpleType>  
    <xs:restriction base="xs:string">  
      <xs:minLength value="5"/>  
      <xs:maxLength value="8"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

XML Schema → Restricciones → maxInclusive...

Para límites superiores/inferiores en contenidos numéricos, tenemos las restricciones **maxInclusive**, **maxExclusive**, **minInclusive** y **minExclusive**.

```
<xs:element name="edad">  
  <xs:simpleType>  
    <xs:restriction base="xs:integer">  
      <xs:minInclusive value="18"/>  
      <xs:maxInclusive value="99"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

El elemento **edad** (tipo **xs:integer**) deberá
contener **un número entre 18 y 99**

```
<xs:element name="edad">  
  <xs:simpleType>  
    <xs:restriction base="xs:integer">  
      <xs:minExclusive value="18"/>  
      <xs:maxExclusive value="99"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

El elemento **edad** (tipo **xs:string**) deberá
contener **un número entre 19 y 98**

XML Schema → Restricciones → maxInclusive...

XML

```
...  
<estrellas>5</estrellas>  
...
```



XML Schema

```
<xs:element name="estrellas">  
  <xs:simpleType>  
    <xs:restriction base="xs:integer">  
      <xs:minInclusive value="1"/>  
      <xs:maxInclusive value="5"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

XML Schema → Restricciones → pattern

pattern define la secuencia exacta de caracteres que es aceptable (el patrón que debe seguir el contenido del elemento/atributo). Difícil de manejar, pero muchas posibilidades...

```
<xs:element name="letra">  
  <xs:simpleType>  
    <xs:restriction base="xs:string">  
      <xs:pattern value="[a-z]"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

El elemento **letra** (tipo **xs:string**) sólo acepta como contenido **una letra minúscula**.

```
<xs:element name="codigopin">  
  <xs:simpleType>  
    <xs:restriction base="xs:integer">  
      <xs:pattern value="[0-9][0-9][0-9][0-9]"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

El elemento **codigopin** (tipo **xs:integer**) sólo acepta como contenido **cuatro números**.

XML Schema → Restricciones → pattern

pattern define la secuencia exacta de caracteres que es aceptable (el patrón que debe seguir el contenido del elemento/atributo). Difícil de manejar, pero muchas posibilidades...

```
<xs:element name="genero">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="hombre|mujer"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

El elemento **genero** (tipo **xs:string**) sólo acepta como contenido **"hombre" o "mujer"**.


```
<xs:element name="password">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-zA-Z0-9]{8}"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

El elemento **password** (tipo **xs:string**) sólo acepta **ocho caracteres (números y letras)**.

XML Schema → Restricciones → pattern

XML

```
...  
<provincia>AL</provincia>  
...
```



XML Schema


```
<xs:element name="provincia">  
  <xs:simpleType>  
    <xs:restriction base="xs:string">  
      <xs:pattern value="[A-Z]{2}"/>  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```


XML Schema → Restricciones → totalDigits...


Para especificar el número exacto de dígitos de un número y el número máximo de decimales permitidos, existen las restricciones **totalDigits** y **fractionDigits**.


```
<xs:element name="numero">
  <xs:simpleType>
    <xs:restriction base="xs:decimal">
      <xs:totalDigits value="5"/>
      <xs:fractionDigits value="2"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

El elemento **numero** (tipo **xs:decimal**) deberá contener **un número de 5 cifras, de las cuales 2 como mucho sean decimales**.

999.99 

-123.50 

74252.88 

1.2345 

XML Schema → Restricciones → whiteSpace

whiteSpace sirve para especificar cómo se tratarán los espacios en blanco (finales de línea, tabuladores, espacios...). Tiene tres valores: **preserve**, **replace** y **collapse**.

```
<xs:element name="direccion">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="preserve"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

En este caso, como pone **"preserve"**, en el contenido del elemento **direccion** (tipo **xs:string**) **no se quitarán los espacios en blanco**.

Si en vez de "preserve" pusiera **"replace"**, **se cambiarían todos los espacios en blanco** (tabuladores, finales de línea...) **por espacios**.

Y, si pusiera **"collapse"**, **se eliminarían todos los espacios en blanco**.

XML Schema → Restricciones → whiteSpace

XML

```
...  
<provincia>    AL    </provincia>  
...
```



XML Schema

```
<xs:element name="provincia">  
  <xs:simpleType>  
    <xs:restriction base="xs:string">  
      <xs:pattern value="[A-Z]{2}" />  
      <xs:whiteSpace value="collapse" />  
    </xs:restriction>  
  </xs:simpleType>  
</xs:element>
```

XML Schema → Restricciones

