XML Schema

Juan Ignacio Raggio

May 3, 2025

Contents

1 De donde surge el XML Schema?

- **Contenido** en el DTD no lo puedo restringir ni puedo `especificarle` un tipo
- Las mejores restricciones de contenido solo se logran expresar a travez de `atributos`
- El DTD `no es un documento XML bien formado` (doble parser)
- La `frecuencia repetitiva` solo permite + o * (no se puede dar una cantidad exacta)
- No se puede expresar la `falta de orden` en un conjunto de elementos
- Una regla en un DTD no puede reusarse en otro DTD

$$\Rightarrow$$

`no hay extensibilidad`

1.1 Recomendacion W3C oficial

- `https://www.w3.org/XML/Schema`

1.2 Valida con 2 metodos el XML Schema:

1. `Content model validation`: Tags correctos, secuencia indicada, etc.
2. `DayaTypeValidation`: Tipo de un dato y su rango

1.3 Vocabulario:

1. `schema`: Raiz del xsd. Tag para declarar el principio y el fin del documento. Sus atributos sirven para declarar el namespace
2. `element`: Declaracion de un elemento.
   (a) `name`: Nombre del elemento
   (b) `minOccurs`, `maxOccurs`: cardinalidad (el default es 1)
   (c) `type`: tipo del dato del elemento
   (d) `default`, `fixed`
3. `attribute`: declaracion de un atributo
   (a) `name`
   (b) `type`
   (c) `use`
4. Ejemplo:

```
<element name="elemento" minOccurs="1" maxOccurs="unbouned" type="string">
```

1.4 Data type validation:

1.4.1 `simpleType`: Sin subelementos ni atributos

```
<xsd:element name="fechaPublicacion" type="xsd:date" />
```

- Ejemplo que valida:

```
<fechaPublicacion>2025-12-30</fechaPublicacion>
```

- Ejemplo que no valida:

```
<fechaPublicacion location="USA">2025-12-30</fechaPublicacion>
```

El motivo es que tiene location que no corresponde a la especificacion de la declaracion

1. Tipos built-in mas usados (hay que investigar mas):
   - Fecha: dateTime, duration, date, time
   - Texto: string, ID, IDREF, NCName
   - Numerico: decimal, integer
   - Logico: boolean
2. Tipos definidos por el usuario:
   (a) restriction FACET_VALUES
   - Ejemplo para restringir nroEdicion entre 1000 y 1005 inclusive -> `Restriccion anonima / Restriccion local`

```
<element name="nroEdicion">
  <simpleType>
    <restriction base="integer" > <---Aca va un simple type--->
    <minInclusive value="1000" /> -> Facet value
    <maxInclusive value="1005" /> -> Facet value
    </restriction>
  </simpleType>
</element>
...
```

   - Restriccion con nombre de otro simple type

```
<simpleType name="tipoDefinido">
  <restriction base="otroSimpleType">
    . . . -> Facet values
  </restriction>
</simpleType>
...y luego se lo referencia desde un elemento
<element name="nombreElemento" type="tipoDefinido"/>
```

   i. `Facet values` para integers:

```
<totalDigits value="nro"/> -> Cantidad exacta de digitos que debe haber
<enumeration value="valor"/> -> Valores que puede tener. Si aparecen varic
<pattern value="expresion_regular"/> -> Si aparecen varios se interpreta c
<minInclusive value="nro"/>
<maxInclusive value="nro"/>
<minExclusive value="nro"/>
<minExclusive value="nro"/>
```

ii. `Facet values` para decimales

```
<fractionDigits value="nro"/> -> Cantidad de numeros atras de la coma
y todo lo mismo que para integers
```

iii. `Facet values` para string

```
<length value="nro"/>
<minLength value="nro"/>
<maxLength value="nro"/>
<enumeration value="nro"/>
<pattern value="expresion_regular"/>
<whiteSpace value="preserve"/> -> Indica como manejar los espacios en blar
```

- Ejemplo para validar un telefono (string)

```
<element name="telefono">
  <simpleType>
    <restriction base="string">
        <pattern value="\d{4}-\d{4}"/>
        <pattern value="\(011\)\d{4}-\d{4}"/>
    </restriction>
  </simpleType>
</element>
```

(b) union Se puede hacer de las dos formas primero creando varias tipos con restricciones y la union va a probar con cada uno si matchea, mientras que tambien se puede crear directamente dentro del cuerpo de la union

- Opcion 1, crear una union con algunos previamente creados

```
<schema>
  <element name="sorpresa">
    <simpleType>
      <union memberType="xsd:date xsd:decimal"/> -> Aca podria ser un xsd:mi_
    </simpleType>
  </element>
</schema>
```

- Opcion 2, crearlos directamente dentro de la union

```
<schema>
  <element name="sorpresa">
    <simpleType>
      <union>
        <restriction base="un simpleType">
          ... Facet
        </restriction>
      </union>
    </simpleType>
  </element>
</schema>
```

(c) list Se usa para validar de un elemento tiene varios valores de un tipo especificado

```
<schema>
  <element name="precios">
    <simpleType>
      <list itemType="float"/>
    </simpleType>
  </element>
</schema>
```

El siguiente si verifica pues tiene varios valores float y se llama `precios` como el tipo indicado

```
<precios>12.3 110.2 0.4</precios>
```

1.4.2 `complexType`: El contenido puede tener subelementos o atributos

1. Creation
   (a) Sequence
   (b) Choise
   (c) All
2. extension or restriction
   (a) A partir de un simpleType
   (b) A partir de un complexType

3. Un elemento con solo atributos es un tipo complejo que tiene una <extension base...> `"aqui irian los atributos y sus tipos"` </extension>