

Bloque 5 - XSD

[Introducción](#)

[Especificación](#)

[Documentación y herramientas online](#)

[Ejercicios iniciales](#)

[Ejercicios con horariosescolar.xml](#)

[Ejercicios finales](#)

[Ejercicios Opcionales](#)

Introducción

XSD (XML Schema Definition) es un lenguaje de esquema utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML. Fue desarrollado por el World Wide Web Consortium (W3C) y alcanzó el nivel de recomendación en mayo de 2001. Fuente: [XML Schema - Wikipedia, la enciclopedia libre](#)

Las principales características de XSD incluyen la capacidad de definir tipos de datos personalizados, definir la jerarquía de elementos y la relación entre ellos, establecer restricciones de valores para elementos y atributos, y especificar la cardinalidad de elementos.

Con un esquema XSD se garantiza que un documento XML es válido y coherente antes de procesarlo, lo que reduce errores y mejora la calidad de los datos.

Especificación

- [W3C XML Schema Definition Language \(XSD\) 1.1 Part 1: Structures](#)
- [W3C XML Schema Definition Language \(XSD\) 1.1 Part 2: Datatypes](#)

Documentación y herramientas online

- [Desarrollo de Aplicaciones Web - UOC \(pág 137 y sucesivas\)](#)
- [XML Schema Structure](#)
- [XML Schema Data type](#)
- [XML Schema Reference W3School](#)

- [Free Online XSD/XML Schema Generator - FreeFormatter.com](#)
- [Free Online XML to XSD Converter](#)
- [XML Schema \(XSD\) Validation online](#)
- [W3C XML Schema \(XSD\) Validation online](#)

Ejercicios iniciales

Un primer contacto para obtener una visión global.

1. Revisa si las extensiones de VSCode [XML - Visual Studio Marketplace](#) y [XML Tools - Visual Studio Marketplace](#) están instaladas y activadas.
2. Observa los ficheros XML y XSD siguientes:
 - a. <https://sede.ciemat.gob.es/SEDEportal/recursos/opendata/OfertasEmpleo.xsd>
(ofertasempleo.xsd está disponible en el repositorio)
 - b. <https://sede.ciemat.gob.es/SEDEportal/recursos/opendata/OfertasEmpleo.xml>
(ofertasempleo.xml está disponible en el repositorio)
 - c. Comprueba cómo enlaza el fichero XML con el XSD.

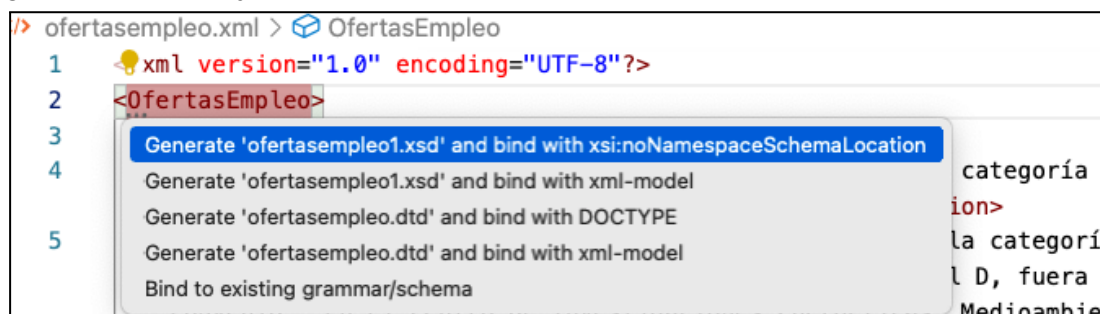
```
<OfertasEmpleo                                FechaCreacion="2022-02-02"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ofertasempleo.xsd">
```

- d. Comprueba en VSCode si ambos documentos son válidos.

En VSCode **Ver→Problemas** puedes comprobar los fallos. Aparecen fallos por repeticiones en valores del atributo id ([xs:ID — Definition of unique identifiers. - XML Schema \[Book\]](#)). Una vez corregidos, ambos documentos están bien formados y son válidos, el XSD con respecto a la norma XSD y el XML con respecto al esquema XSD al que enlaza. Recuerda que para que el fichero XML valide tiene que estar bien formado.

- e. Utiliza [Free Online XML Validator \(XSD\)](#) para comprobar si el documento XML es válido utilizando el esquema XSD
Muestra los mismos errores que VSCode
- f. Corrige los errores hasta que el documento XML sea válido.
Solo hay que cambiar los números en los id repetidos, una vez revisado el documento valida.

- Utiliza la herramienta online [Free Online XSD/XML Schema Generator - FreeFormatter.com](https://sede.ciemat.gob.es/SEDEportal/recursos/opendata/OfertasEmpleo.xml) para generar el XSD del documento XML <https://sede.ciemat.gob.es/SEDEportal/recursos/opendata/OfertasEmpleo.xml> (ofertasempleo.xml en el repositorio), utiliza las tres formas que permite (Venetian Blind, Salami Slice, Russian Doll). Compáralo con el <https://sede.ciemat.gob.es/SEDEportal/recursos/opendata/OfertasEmpleo.xsd> (ofertasempleo.xsd está disponible en el repositorio).
- Desde VSCode genera el fichero XSD para el fichero <https://sede.ciemat.gob.es/SEDEportal/recursos/opendata/OfertasEmpleo.xml> (ofertasempleo.xml está disponible en el repositorio). Compáralo con los generados en el ejercicio anterior.



Ejercicios con horariosescolar.xml

Utiliza el documento XML de los horarios disponible en el repositorio para realizar los siguientes ejercicios.

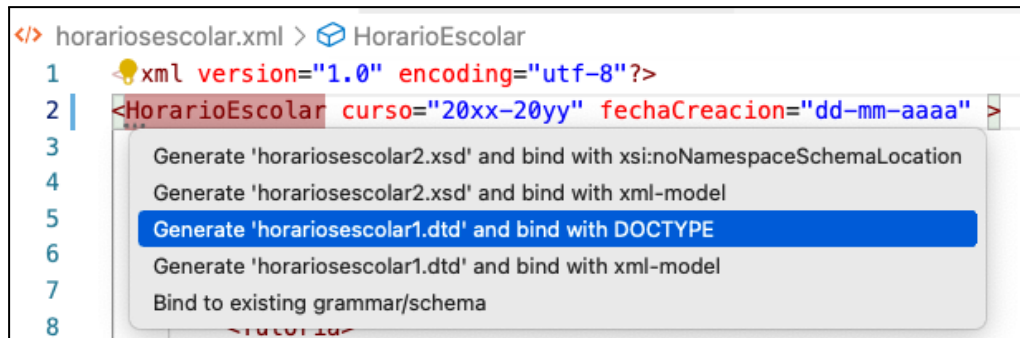
Se incluye la solución en la mayoría de los ejercicios.

- El fichero horariosescolar.xsd, disponible en el repositorio, ha sido generado desde el documento horariosescolar.xml utilizando la herramienta [Free Online XSD/XML Schema Generator - FreeFormatter.com](https://sede.ciemat.gob.es/SEDEportal/recursos/opendata/OfertasEmpleo.xml), opción "Salami Slice". Enlaza el documento XSD con el XML, utiliza la etiquetas marcadas en negrita:

```
<?xml version="1.0" encoding="utf-8"?>
<HorarioEscolar xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="horariosescolar.xsd" curso="20xx-20yy"
  fechaCreacion="dd-mm-aaaa">
  ...
</HorarioEscolar>
```

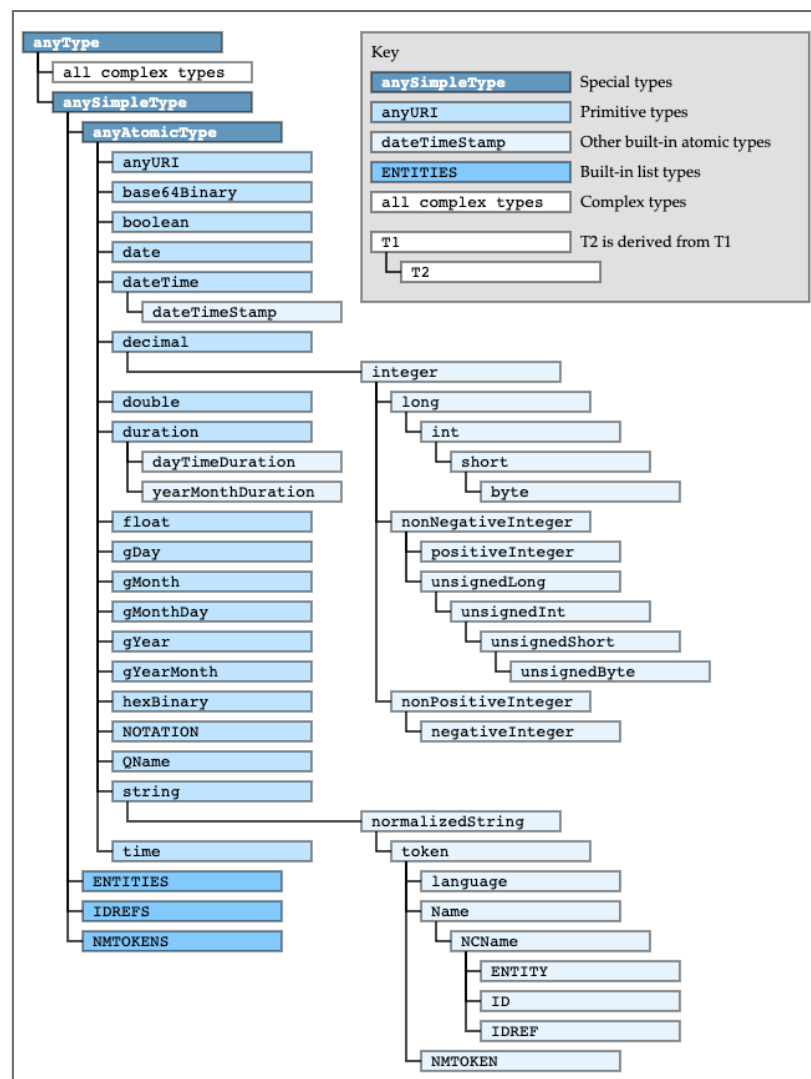
Y comprueba que valida en VSCode (ver panel “Problemas”) y en [XML Schema \(XSD\) Validation online](#)

6. Desde VSCode genera un DTD desde el XML y compáralo con `horariosescolar.xsd`



7. Explica los tipos de datos [xs:normalizedString](#), [xs:token](#). Más información en [W3C XML Schema Definition Language \(XSD\) 1.1 Part 2: Datatypes](#) (pulsar en **imagen resumen de tipos de datos**)

Ver enlaces en los propios tipos de datos.



Data Type	Description	Example Representation
xsd:string	A character string	Hello World
xsd:boolean	A true/false value	True
xsd:decimal	Those real numbers which can be represented decimal format	42.1
There are many built-in types which derive from xsd:decimal. These include:		
xsd:integer, xsd:positiveInteger, xsd:negativeInteger, xsd:nonNegativeInteger, xsd:long, xsd:int, xsd:short, xsd:byte, xsd:unsignedLong, xsd:unsignedInt, xsd:unsignedShort, xsd:unsignedByte		
xsd:float	Patterned after single-precision IEEE 32-bit floating point	6.0235e-23
xsd:double	Patterned after IEEE 64-bit floating point	6.0235e-23
xsd:duration	A duration of time. For example, a period of 5 years, 2 months, and 10 days.	P5Y2M10D
xsd:dateTime	A date and time together	2002-05-30T09:30:10.5
xsd:time	An instant of time that recurs every day	13:04:00
xsd:date	Intervals of exactly one day in length	1889-09-24
xsd:gYearMonth	A specific Gregorian month in a specific Gregorian year	1999-05
xsd:gYear	A Gregorian calendar year	1999
xsd:gMonthDay	A Gregorian date that recurs, specifically a day of the year such as the third of May	-05-03
xsd:gDay	A Gregorian day that recurs, specifically a day of the month such as the 27th of the month	-27
xsd:gMonth	A Gregorian month that recurs every year. For example, June.	-06-
xsd:hexBinary	Arbitrary hex-encoded binary data	0047dedbef
xsd:base64Binary	Arbitrary Base64-encoded binary data	VGhpcyBpcyBzb211IHJleHQh
xsd:anyURI	A Uniform Resource Identifier Reference (URI)	http://www.cambridgesemantics.com

Fuente

8. Realiza los siguientes ajustes en el schema XSD:

Nota: otras soluciones pueden ser válidas, por ejemplo utilizar type en lugar de ref. Las elegidas buscan tener una representación de los diferentes recursos que nos ofrece la especificación.

- a. Añade una anotación al esquema con los siguientes textos:
 - i. En xs:appinfo, IES VJP - PLASENCIA - LMSGI - 1.0
 - ii. En xs:documentation, Schema para validación de documentos XML con horarios escolares por grupo, elaborado por Nombre alumno y en otro xs:documentation el texto Schema for validating XML documents with school schedules by group, prepared by Student name

Observa que no es necesario el uso de CDATA en xs:documentation aunque utilicemos símbolos como < >

```
<xs:schema attributeFormDefault="unqualified"
  elementFormDefault="qualified"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:annotation>
    <xs:appinfo>IES VJP - PLASENCIA - LMSGI - 1.0</xs:appinfo>
    <xs:documentation xml:lang="es">
      Schema para validación de documentos XML con horarios
      escolares por grupo, elaborado por <strong>Nombre alumno</strong>
    </xs:documentation>
    <xs:documentation xml:lang="en">
      Schema for validating XML documents with school schedules
      by group, prepared by <strong>Student name</strong>
    </xs:documentation>
  </xs:annotation>
```

- b. Configura el tipo de datos más correcto para el atributo //Modulo/@horas

```
<xs:element name="Modulo">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Docente"/>
    </xs:sequence>
    <xs:attribute type="xs:string" name="abbr" use="optional"/>
    <xs:attribute type="xs:string" name="nombre" use="optional"/>
    <xs:attribute type="xs:unsignedShort" name="horas" use="optional"/>
  </xs:complexType>
```

```
</xs:element>
```

- c. Configura el tipo de datos más correcto para los atributos de fechas y hora (atributos `//Periodo/@desde`, `//Periodo/@hasta` y `//HorarioEscolar/@fechaCreacion`). Cambia el documento XML para ajustarlo a los nuevos tipos de datos.

Para la horas utilizamos `xs:time` y en el documento XML habría que poner `hh:mm:ss`

Tenemos varias opciones, la más sencilla es indicar que utilice el tipo `date` (`yyyy-mm-dd`). Tendríamos que cambiar la fecha en el documento XML para que cumpla con el tipo de datos `date`.

```
<xs:element name="HorarioEscolar">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="IES"/>
      <xs:element ref="Grupo" maxOccurs="unbounded" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute type="xs:string" name="curso"/>
    <xs:attribute type="xs:date" name="fechaCreacion"/>
  </xs:complexType>
</xs:element>
<xs:element name="Periodo">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute type="xs:time" name="desde" use="optional"/>
        <xs:attribute type="xs:time" name="hasta" use="optional"/>
        <xs:attribute type="xs:string" name="abbr" use="optional"/>
        <xs:attribute type="xs:string" name="tipo" use="optional"/>
        <xs:attribute type="xs:string" name="descripcion" use="optional"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

Otra opción es utilizar un patrón que coincida con el formato que estamos utilizando (`dd-mm-aaaa`):

***Nota:** por defecto, sin necesidad de indicarlo, todos los atributos son opcionales (`use="optional"`).*

```
<xs:element name="HorarioEscolar">
```

```

<xs:complexType>
  <xs:sequence>
    <xs:element ref="IES"/>
    <xs:element ref="Grupo" maxOccurs="unbounded" minOccurs="0"/>
  </xs:sequence>
  <xs:attribute type="xs:string" name="curso"/>
  <xs:attribute type="fecha" name="fechaCreacion"/>
</xs:complexType>
</xs:element>
<xs:simpleType name="fecha">
  <xs:restriction base="xs:string">
    <xs:pattern value="\d{2}[-]\d{2}[-]\d{4}"></xs:pattern>
    <xs:length value="10"/>
  </xs:restriction>
</xs:simpleType>

```

- d. El elemento //Dia contiene al menos 1 elemento //Dia/Periodo y un máximo de 7

```

<xs:element name="Dia">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Periodo" minOccurs="1" maxOccurs="7"/>
    </xs:sequence>
    <xs:attribute type="xs:string" name="nombre" use="optional"/>
  </xs:complexType>
</xs:element>

```

- e. El elemento //Horario contendrá exactamente 5 elementos //Horario/Dia

```

<xs:element name="Horario">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Dia" minOccurs="5" maxOccurs="5" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

- f. El atributo //Dia/@nombre solo permite los siguientes valores: Lunes, Martes, Miércoles, Jueves y Viernes y es obligatorio (observa que la primera letra está en mayúsculas).

Con la siguiente configuración hay que escribir exactamente los nombres como se indica, la primera letra en mayúscula. Los espacios a izquierda y derecha no supondría error al utilizar `whiteSpace`. Hay que cambiar los valores del atributo en el documento XML al estar escritos en minúsculas.

```
<xs:element name="Dia">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Periodo" maxOccurs="unbounded" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="nombre" use="required" type="diaSemana" />
  </xs:complexType>
</xs:element>
<xs:simpleType name="diaSemana">
  <xs:restriction base="xs:string">
    <xs:whiteSpace value="collapse" />
    <xs:enumeration value="Lunes" />
    <xs:enumeration value="Martes" />
    <xs:enumeration value="Miércoles" />
    <xs:enumeration value="Jueves" />
    <xs:enumeration value="Viernes" />
  </xs:restriction>
</xs:simpleType>
```

- g. El atributo `//Periodo/@tipo` no es obligatorio y el valor por defecto es módulo. Los valores permitidos son módulo o descanso

Nota: por defecto, sin necesidad de indicarlo, todos los atributos son opcionales (use="optional").

```
<xs:element name="Periodo">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute type="xs:time" name="desde" use="optional"/>
        <xs:attribute type="xs:time" name="hasta" use="optional"/>
        <xs:attribute type="xs:string" name="abbr" use="optional"/>
        <xs:attribute type="tipoPeriodo" default="módulo" name="tipo" />
        <xs:attribute type="xs:string" name="descripcion" use="optional"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
<xs:simpleType name="tipoPeriodo">
  <xs:restriction base="xs:string">
    <xs:whiteSpace value="collapse" />
    <xs:enumeration value="módulo" />
```

```

    <xs:enumeration value="descanso" />
  </xs:restriction>
</xs:simpleType>

```

- h. Añade un atributo al elemento //IES para guardar el código del IES, el cual es un valor numérico positivo sin decimales con un máximo de 10 dígitos y es opcional.

```

<xs:element name="IES">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Nombre"/>
      <xs:element ref="Url"/>
    </xs:sequence>
    <xs:attribute name="codigo" use="optional" type="tipoCodigo"/>
  </xs:complexType>
</xs:element>
<xs:simpleType name="tipoCodigo">
  <xs:restriction base="xs:positiveInteger">
    <xs:totalDigits value="10"/>
  </xs:restriction>
</xs:simpleType>

```

- i. Añade al elemento //Tutoria tres nuevos elementos, todos sin atributos. Uno de nombre //Tutoria/Ubicacion, con contenido de tipo texto con una longitud comprendida entre 10 y 100 caracteres, otro de nombre //Tutoria/Observaciones, con contenido de tipo texto sin restricciones, y un último de nombre //Tutoria/DocenteTitular de tipo vacío. Los elementos pueden aparecer en cualquier orden. //Tutoria/Dia y //Tutoria/Docente tienen que aparecer obligatoriamente una vez, el resto pueden aparecer o no, pero con un máximo de 1.

Observa que minOccurs="1" no es necesario ponerlo al ser el valor por defecto, en el caso de Dia no se ha puesto. En //Observaciones no se ha utilizado ref, por lo que este elemento solo podrá ser utilizado dentro del elemento //Tutoria. Los elementos vacíos necesitan utilizar complexType.

```

<xs:element name="Tutoria">
  <xs:complexType>
    <xs:all>
      <xs:element ref="Dia" />
      <xs:element ref="Docente" minOccurs="1"/>
      <xs:element ref="Ubicacion" minOccurs="0"/>
    </xs:all>
  </xs:complexType>

```

```

    <xs:element name="Observaciones" type="xs:string" minOccurs="0"/>
    <xs:element name="DocenteTitular" minOccurs="0">
      <xs:complexType />
    </xs:element>
  </xs:all>
</xs:complexType>
</xs:element>
<xs:element name="Ubicacion">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="collapse" />
      <xs:minLength value="10" />
      <xs:maxLength value="100" />
    </xs:restriction>
  </xs:simpleType>
</xs:element>

```

- j. Añade las siguientes Observaciones a la tutoría de uno de los grupos: "Concertar cita previa para evitar esperas."

Observa que en el documento XML utilizamos CDATA para incluir los símbolos <> en el elemento Observaciones.

```

<Observaciones><![CDATA[Concertar <em>cita previa</em> para evitar
esperas.]]> </Observaciones>

```

EL K NO EN EL EXAMEN

- k. Añade al elemento //IES y al elemento //Docente dos elementos opcionales, //Telefono y //Mail. Pueden aparecer cero o más veces y en cualquier orden. Ambos elementos pueden tener los atributos //(IES,Docente)/(Telefono, Mail)/@corporativo, con valores true o false, e //(IES,Docente)/(Telefono, Mail)/@idiomas, que permite indicar una lista de idiomas descritos por [RFC1766](#). El valor por defecto del atributo idiomas es "es-ES".

El elemento //Docente podrá incluir elementos //Docente/Dia con un máximo de 7 para indicar los días y horas en los que atenderá consultas, para lo que en //Docente/Dia/Periodo podrá indicar también el valor Activo. Añade una **anotación** a //Docente con el texto Días y horas en las que contesta a los mensajes que reciba en los contactos.

```

<xs:element name="IES">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="Nombre"/>
      <xs:element ref="Url"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

```

```

    <xs:group ref="Contacto" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="codigo" use="optional" type="tipoCodigo"/>
</xs:complexType>
</xs:element>
<xs:element name="Profesor">
  <xs:complexType mixed="true">
    <xs:sequence>
      <xs:element ref="Dia" minOccurs="0" maxOccurs="7">
        <xs:annotation>
          <xs:documentation>Días y horas en las que contestará a los mensajes que
reciba en los contactos</xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:group ref="Contacto" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:simpleType name="tipoPeriodo">
  <xs:restriction base="xs:string">
    <xs:whiteSpace value="collapse" />
    <xs:enumeration value="módulo" />
    <xs:enumeration value="descanso" />
    <xs:enumeration value="activo" />
  </xs:restriction>
</xs:simpleType>

<xs:group name="Contacto">
  <xs:choice>
    <xs:element name="Telefono">
      <xs:complexType>
        <xs:simpleContent>
          <xs:extension base="xs:positiveInteger">
            <xs:attributeGroup ref="attrContacto" />
          </xs:extension>
        </xs:simpleContent>
      </xs:complexType>
    </xs:element>
    <xs:element name="Mail">
      <xs:complexType>
        <xs:simpleContent>
          <xs:extension base="emailAddress">
            <xs:attributeGroup ref="attrContacto" />
          </xs:extension>
        </xs:simpleContent>
      </xs:complexType>
    </xs:element>
  </xs:choice>
</xs:group>

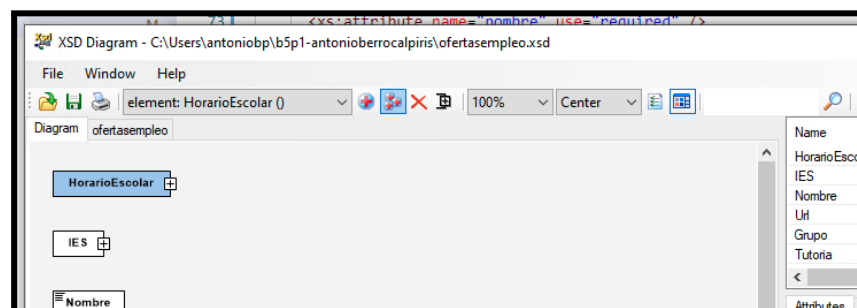
```

```

    </xs:element>
  </xs:choice>
</xs:group>
<xs:simpleType name="emailAddress">
  <xs:restriction base="xs:string">
    <xs:pattern
value=" [A-Za-z0-9_]+([-.'] [A-Za-z0-9_]+)*@[A-Za-z0-9_]+([-.] [A-Za-z0-9_]+)*\.[
[A-Za-z0-9_]+([-.] [A-Za-z0-9_]+)*" />
  </xs:restriction>
</xs:simpleType>
<xs:attributeGroup name="attrContacto">
  <xs:attribute name="corporativo" type="xs:boolean"/></xs:attribute>
  <xs:attribute name="idiomas" type="listaIdiomas" default="es-ES"/>
</xs:attributeGroup>
<xs:simpleType name="listaIdiomas">
  <xs:list itemType="xs:language"/>
</xs:simpleType>

```

- l. Vuelve a validar el documento XML, resuelve todos los errores.
- m. Genera un gráfico del esquema XSD con el software [XSD Diagram](#).
Para ver el gráfico pulsamos "Add all top level elements"



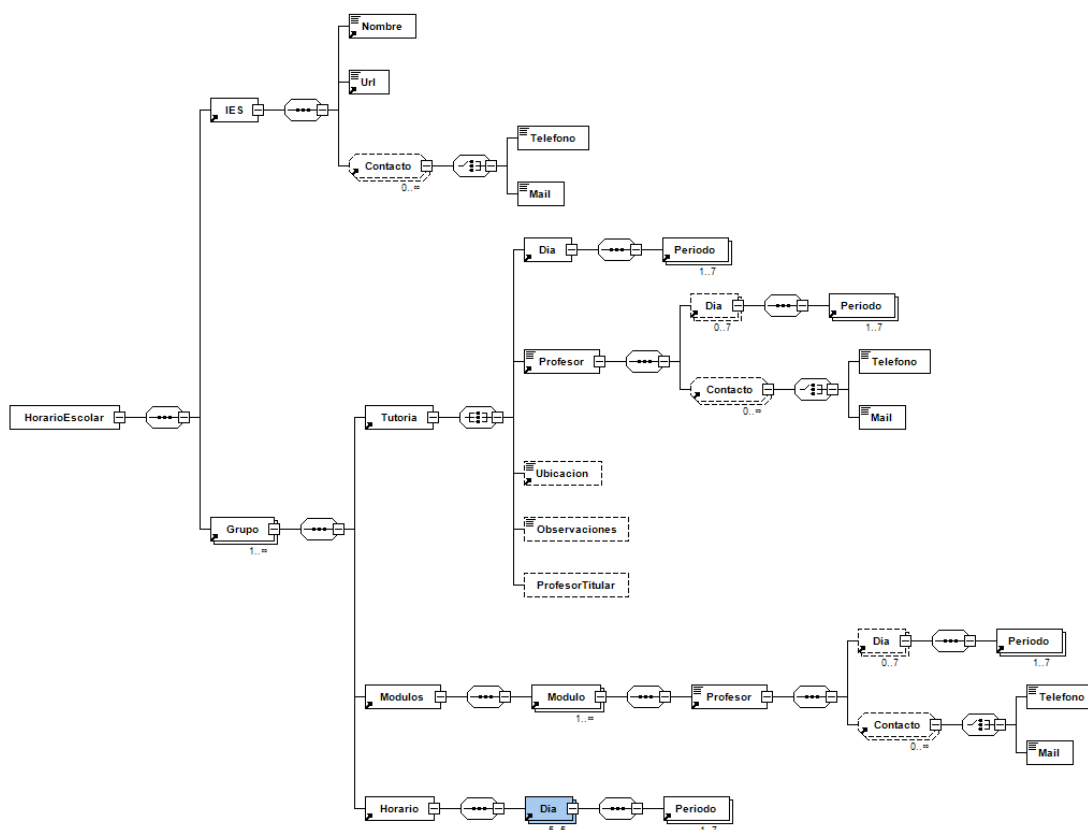


Diagrama generado con XSD Diagram

Ejercicios finales

Comprobar ejemplos reales de uso de XML.

9. Crea un documento XML válido con datos inventados basado en el schema prueba.xsd disponible en el repositorio. Observa el elemento `//gard` y `//line/example_schema/length`. Explica el uso de `simpleContent` y `complexContent`.

Utiliza una herramienta para generar un XML y después se personaliza, por ejemplo: <http://xsd2xml.com/> o <https://www.liquid-technologies.com/online-xsd-to-xml-converter>

`simpleContent` permite definir restricciones o extensiones a elementos que solo contienen datos, es decir, no contienen otros elementos. Se puede utilizar para añadir atributos.

```
<xs:element name="length">
```

```

<xs:complexType>
  <xs:simpleContent>
    <xs:extension base="xs:integer">
      <xs:attribute name="unit_of_length" type="xs:string" />
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
</xs:element>

```

Ejemplo de elemento length en un documento XML:

```
<length unit_of_length="string">-266654</length>
```

complexContent se utiliza para definir extensiones o restricciones a un tipo complejo.

```

<xs:complexType name="personinfo">
  <xs:sequence>
    <xs:element name="firstname" type="xs:string"/>
    <xs:element name="lastname" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="fullpersoninfo">
  <xs:complexContent>
    <xs:extension base="personinfo">
      <xs:sequence>
        <xs:element name="address" type="xs:string"/>
        <xs:element name="city" type="xs:string"/>
        <xs:element name="country" type="xs:string"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:element name="gard" type="fullpersoninfo"/>

```

Ejemplo del elemento gard en un documento XML:

```

<gard>
  <firstname>Nombre</firstname>
  <lastname>Apellidos</lastname>
  <address>Calle nueva, 25</address>
  <city>Plasencia</city>
  <country>España</country>
</gard>

```

10. Observa algunos de los documentos XSD de la Secretaría de Estado de Presupuestos y Gastos relacionados con [Fondos 2020](#). Observa como prescinde del uso de atributos.
11. Realiza los siguientes ejercicios con una publicación del BOE:

Directiva 2000/78/CE del Consejo, de 27 de noviembre de 2000, relativa al establecimiento de un marco general para la igualdad de trato en el empleo y la ocupación.

Publicado en: «DOCE» núm. 303, de 2 de diciembre de 2000, páginas 16 a 22 (7 págs.)

Departamento: Comunidades Europeas

Referencia: DOUE-L-2000-82357

Otros formatos:



PDF



XML

- a. Descarga en formato XML [DOUE-L-2000-82357 Directiva 2000/78/CE del Consejo, de 27 de noviembre de 2000, relativa al establecimiento de un marco general para la igualdad de trato en el empleo y la ocupación](#). Nombre fichero: boe.xml
- b. Revisa el manual oficial sobre sumarios BOE [XML SUMARIOS: BOE](#)
- c. Revisa [BOE.es - Datos abiertos \(OpenData\)](#)

12. Revisa los documentos:

- a. [Manual de usuario de esquemas XML para intercambio de documentos electrónicos y expedientes electrónicos](#) ¿para qué sirve xsd:import? Abre algunos de los xsd referenciados en el documento.



El elemento **xsd:import** en un archivo XSD se utiliza para importar definiciones de elementos, tipos complejos, grupos de elementos y atributos definidos en otro archivo XSD. Cuando se utiliza xsd:import, se crea una referencia a los componentes definidos en el archivo XSD importado, lo que permite utilizarlos en el archivo XSD actual.

El uso de `xsd:import` es útil en situaciones en las que se quiere definir un conjunto de tipos de datos comunes en un archivo XSD y reutilizarlos en varios archivos XSD diferentes. Por ejemplo, si tienes un archivo XSD que define los tipos de datos para una aplicación de contabilidad y quieres utilizar esos tipos de datos en otro archivo XSD que define los elementos para una aplicación de nómina, puedes utilizar `xsd:import` para importar los tipos de datos del archivo XSD de contabilidad en el archivo XSD de nómina.

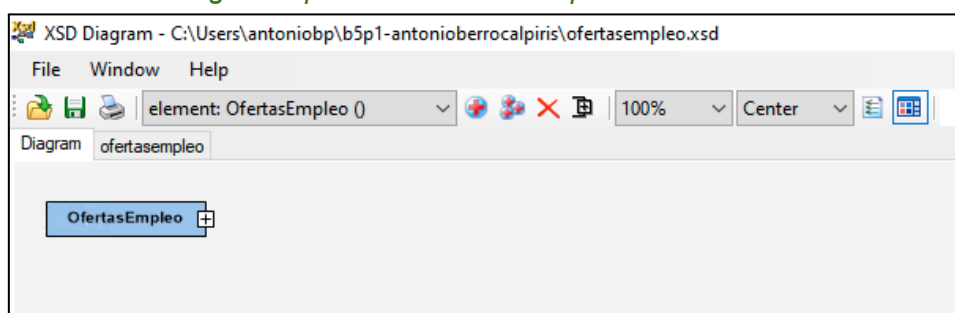
Es importante tener en cuenta que, al utilizar `xsd:import`, se crea una dependencia entre los archivos XSD, por lo que cualquier cambio en el archivo XSD importado puede afectar a los archivos XSD que lo utilizan.

- b. [BOE-A-2011-13169 Resolución de 19 de julio de 2011, de la Secretaría de Estado para la Función Pública, por la que se aprueba la Norma Técnica de Interoperabilidad de Documento Electrónico.](#)
- c. [BOE-A-2011-13170 Resolución de 19 de julio de 2011, de la Secretaría de Estado para la Función Pública, por la que se aprueba la Norma Técnica de Interoperabilidad de Expediente Electrónico.](#)

Ejercicios Opcionales

13. Instala el software [XSD Diagram](#) y genera un gráfico del esquema XSD <https://sede.ciemat.gob.es/SEDEportal/recursos/opendata/OfertasEmpleo.xsd> (`ofertasempleo.xsd` en el repositorio)

Para ver el gráfico pulsamos “Add all top level elements”



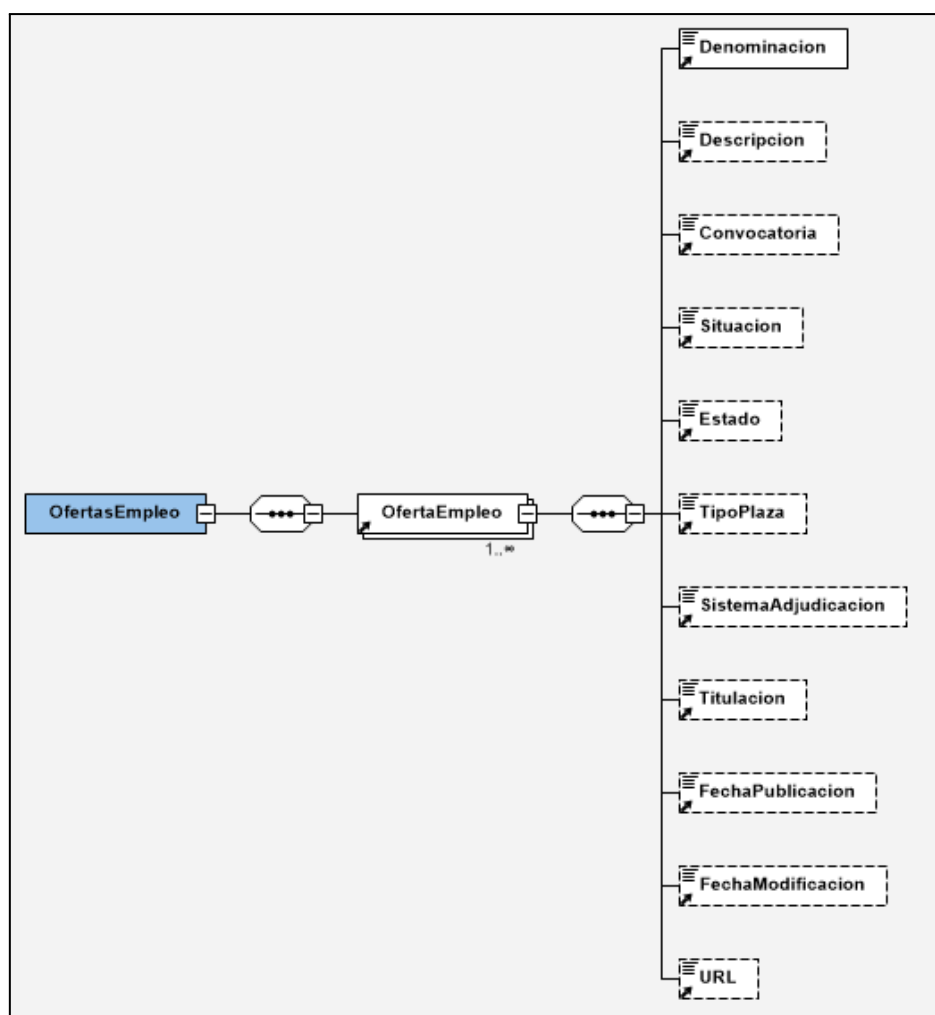
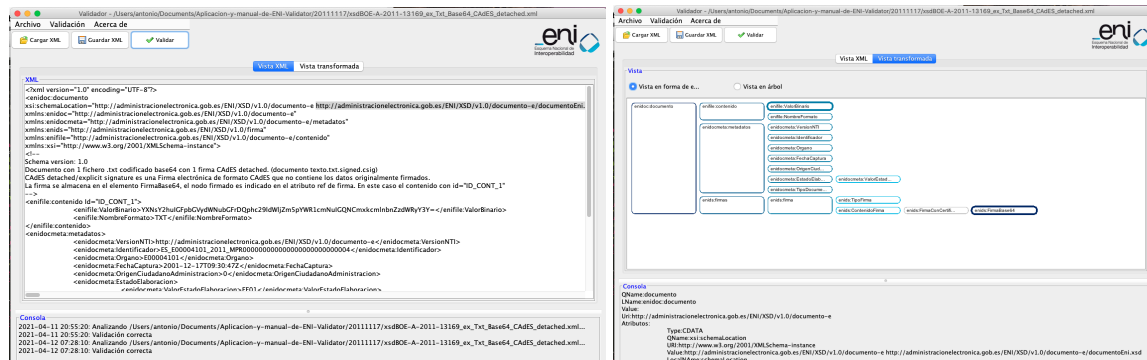


Diagrama generado con XSD Diagram

14. Prueba [XML Notepad](#)
15. Utiliza una de las herramientas de generación de documentación de forma automática (ver [Bloque 5 - XML](#))
16. Investiga **any**, **anyAttribute** y **union**
 - any** permite añadir elementos no definidos en XSD
 - anyAttribute** permite añadir atributos no definidos en XSD
 - union** permite poder asignar más de un tipo de datos simple
17. Investiga la [aplicación de validación de los esquemas XSD](#) de documento y expediente electrónico creada por el [Centro de Transferencia de Tecnología](#)
 Aplicación de escritorio (.jar y .exe) que permite:
 - Seleccionar los ficheros XML que contengan una ocurrencia o instancia de un documento o expediente electrónico
 - Comprobar su validez contra los esquemas XSD fijados por las Normas Técnicas de Interoperabilidad correspondientes.

- Editar los citados ficheros XML
- Visualizar los ficheros incluidos en ellos



18. Investiga “[Esquema Nacional de Interoperabilidad](#)”

19. Investiga “[Facturae](#)” del Ministerio de Hacienda