



XML: eXtensible Markup Languaje

Tecnologías XML

Esquemas XML (XML Schema)

Dr. Juan Manuel Cueva Lovelle
Departamento de Informática
Universidad de Oviedo
cueva@uniovi.es

Software y estándares para la Web

Esquema

**Grado en
Ingeniería
Informática
del Software**

- Introducción a XML Schema
- Ejemplos de XML Schema
- De DTDs a Schemas
- Definición de elementos simples
- Definición de atributos
- Definición de tipos
- Definición de elementos complejos
- Validación con XML Schema
- Limitaciones de XML Schema
- Generación automática de XML Schema
- Bibliografía
- Referencias Web
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- **Introducción a XML Schema**

- Ejemplos de XML Schema
- De DTDs a Schemas
- Definición de elementos simples
- Definición de atributos
- Definición de tipos
- Definición de elementos complejos
- Validación con XML Schema
- Limitaciones de XML Schema
- Generación automática de XML Schema
- Bibliografía
- Referencias Web
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Introducción a XML Schema (I)

Grado en
Ingeniería
Informática
del Software

- Problemas de los DTDs
 - Difíciles de manipular (**no son XML**)
 - **No son extensibles** (una vez definido, no es posible añadir nuevos vocabularios a un DTD)
 - **No soportan tipos de datos** (ej. enteros, float, etc.)
 - **No están soportados por muchas herramientas**
- XML Schema
 - Permite definir esquemas de documentos
 - La sintaxis utilizada **es XML**
 - La sintaxis de los DTD no es XML
 - Soporta la especificación de **tipos de datos y tipos definidos por el usuario**
 - Soporta comprobación de **restricciones numéricas**

Software y estándares para la Web

Introducción a XML Schema (II)

Grado en
Ingeniería
Informática
del Software

- XML Schema
 - Definición de la estructura de un conjunto de documentos XML
- Validar
 - Comprobar que un documento sigue un esquema
 - La principal ventaja es evitar de errores
 - Otras aplicaciones: edición, comprensión, enlaces de programación, etc.
 - Originalmente se utilizaron los DTDs
 - Posteriormente a los DTDs se ha desarrollado XML Schema
 - Existen otras formas de validar los documentos XML
 - RELAX-NG, Schematron, etc.

Software y estándares para la Web

Introducción a XML Schema (III): Características (a)

Grado en
Ingeniería
Informática
del Software

- Sintaxis XML
- Soporte para Espacios de Nombres
- Mayor expresividad
 - Restricciones numéricas
 - Integridad dependientes del contexto
- Tipos de datos
 - Gran cantidad de tipos de datos predefinidos
 - Creación de tipos de datos por el usuario
- Extensibilidad
 - Inclusión/Redefinición de esquemas
 - Herencia de tipos de datos
- Soporte a Documentación

- XML Schema permiten:
 - Describir estructura
 - Anidación
 - Multiplicidad
 - Ordenamiento
 - Describir tipos
 - Para velocidad operatoria
 - Para mejor almacenamiento
 - Para búsquedas
 - Para ingreso de datos
 - Para detectar errores
- Se almacenan en archivos **.xsd**

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Introducción a XML Schema
- **Ejemplos de XML Schema**
- De DTDs a Schemas
- Definición de elementos simples
- Definición de atributos
- Definición de tipos
- Definición de elementos complejos
- Validación con XML Schema
- Limitaciones de XML Schema
- Generación automática de XML Schema
- Bibliografía
- Referencias Web
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Ejemplos de XML Schema (I): *pizzas.xsd*

pizzas.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.uniovi.es"
xmlns="http://www.uniovi.es"
elementFormDefault="qualified">
<xs:element name="pizzas">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="pizza" minOccurs="1" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="pizza">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="ingrediente" minOccurs="1" maxOccurs="4" />
    </xs:sequence>
    <xs:attribute name="nombre" type="xs:ID" use="required"/>
    <xs:attribute name="precio" type="xs:integer" use="required"/>
  </xs:complexType>
</xs:element>
<xs:element name="ingrediente">
  <xs:complexType>
    <xs:attribute name="nombre" type="xs:string" use="required" />
  </xs:complexType>
</xs:element>
</xs:schema>
```

Elemento
raíz
schema

Permite especificar
rangos de inclusión

Permite especificar tipos

Asociación del archivo XML
con el esquema

pizzas.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<pizzas xmlns=http://www.uniovi.es
xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance
xsi:schemaLocation="http://www.uniovi.es pizzas.xsd"
...</pizzas>
```

Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

Ejemplos de XML Schema (II): pizzas.xsd (a)

pizzas.xsd

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

<xs:element name="pizzas">
<xs:complexType>
<xs:sequence>
<xs:element ref="pizza" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
</xs:element>
```

Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

Ejemplos de XML Schema (III): pizzas.xsd (b)

Grado en
Ingeniería
Informática
del Software

pizzas.xsd continuación

```
<xs:element name="pizza">
<xs:complexType>
<xs:sequence>
<xs:element ref="ingrediente" minOccurs="0" maxOccurs="5"/>
</xs:sequence>
<xs:attribute name="nombre" type="xs:ID" use="required"/>
<xs:attribute name="precio" type="xs:integer" use="required"/>
</xs:complexType>
</xs:element>
```

Software y estándares para la Web

Ejemplos de XML Schema (IV): pizzas.xsd (c)

Grado en
Ingeniería
Informática
del Software

pizzas.xsd continuación

```
<xs:element name="ingrediente">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="nombre" type="xs:string" use="required"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
</xs:schema>
```

Software y estándares para la Web

Ejemplos de XML Schema (V): pizzas.xml (a)

pizzas.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<pizzas xsi:schemaLocation="http://www.uniovi.es pizzas.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.uniovi.es">

<pizza nombre="Barbacoa" precio="8">
<ingrediente nombre="Salsa Barbacoa"/>
<ingrediente nombre="Mozzarella"/>
<ingrediente nombre="Pollo"/>
<ingrediente nombre="Bacon"/>
<ingrediente nombre="Tenera"/>
</pizza>
```

Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

Ejemplos de XML Schema (VI): pizzas.xml (b)

pizzas.xml continuación

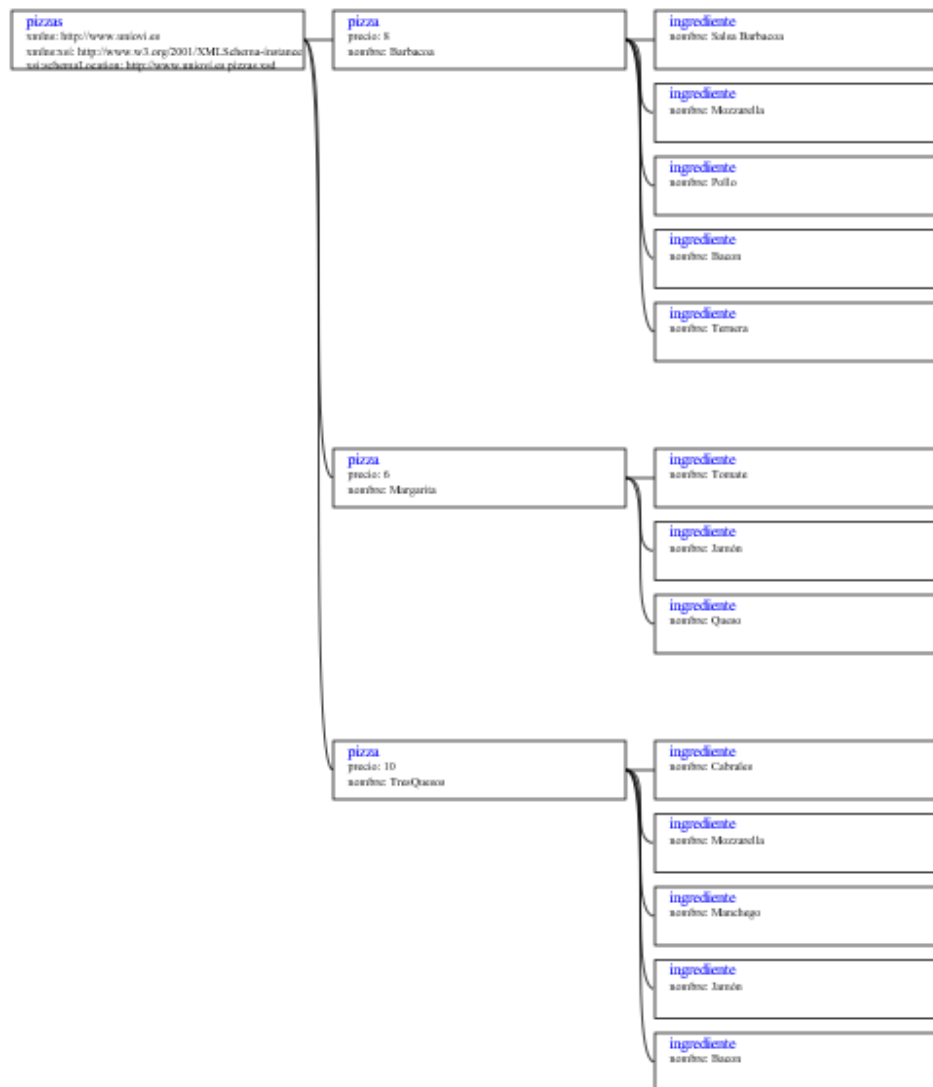
```
<pizza nombre="Margarita" precio="6">
<ingrediente nombre="Tomate"/>
<ingrediente nombre="Jamón"/>
<ingrediente nombre="Queso"/>
</pizza>
<pizza nombre="TresQuesos" precio="10">
<ingrediente nombre="Cabrales"/>
<ingrediente nombre="Mozzarella"/>
<ingrediente nombre="Manchego"/>
<ingrediente nombre="Jamón"/><ingrediente nombre="Bacon"/>
</pizza>
</pizzas>
```

Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

Ejemplos de XML Schema (VII): árbol pizzas.xml

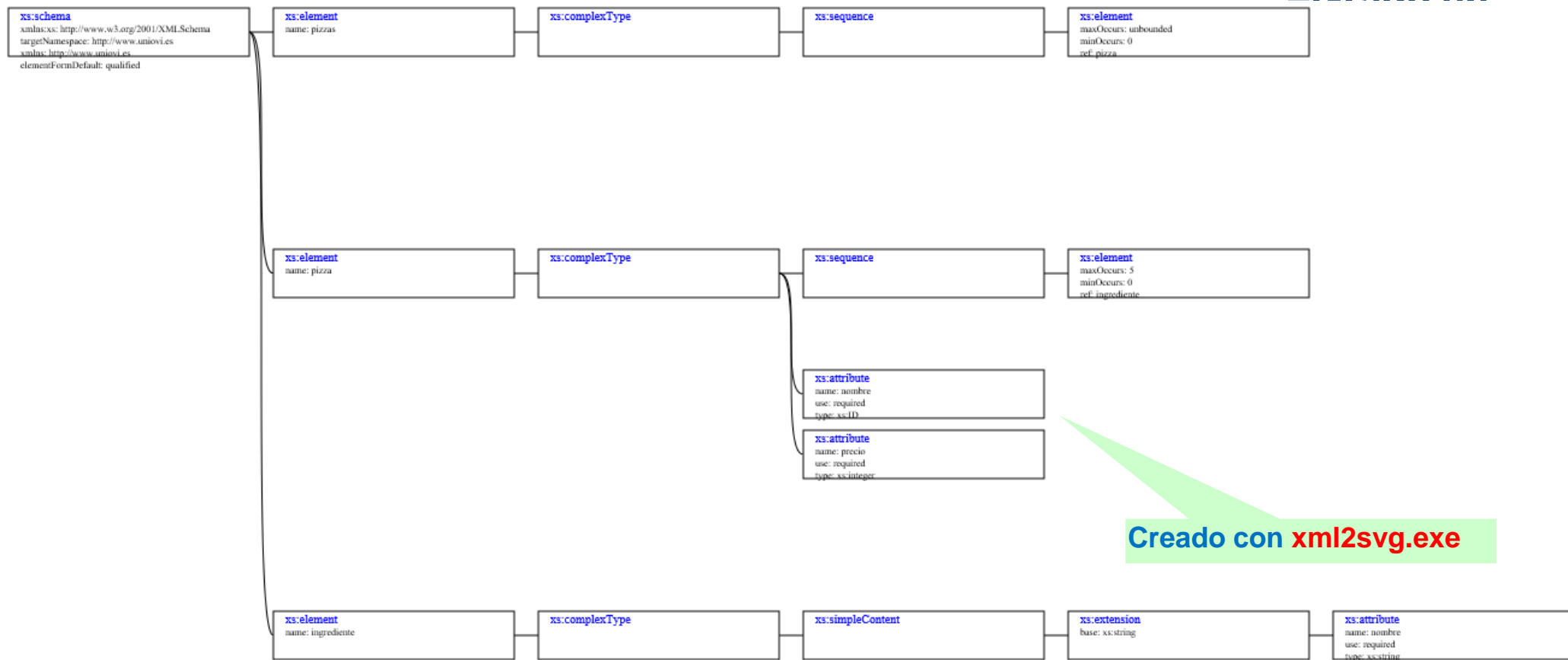
Grado en
Ingeniería
Informática
del Software



Creado con **xml2svg.exe**

Software y estándares para la Web

Ejemplos de XML Schema (VIII): árbol pizzas.xsd



Creado con **xml2svg.exe**

Software y estándares para la Web

Ejemplos de XML Schema (IX): poema.xsd (a)

Grado en
Ingeniería
Informática
del Software

poema.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema elementFormDefault="qualified" xmlns="http://www.uniovi.es" targetNamespace="http://www.uniovi.es"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  - <xs:element name="poesía">
    - <xs:complexType>
      - <xs:sequence>
        - <xs:element name="poema">
          - <xs:complexType>
            - <xs:sequence>
              <xs:element name="titulo" type="xs:string"/>
              <xs:element name="verso" type="xs:string" minOccurs="1" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="autor" type="xs:string" use="required"/>
            <xs:attribute name="fecha" type="xs:string" use="required"/>
            <xs:attribute name="lugar" type="xs:string" use="optional"/>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Software y estándares para la Web

Ejemplos de XML Schema (X): **alba.xml**

Grado en
Ingeniería
Informática
del Software

alba.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<poesía xsi:schemaLocation="http://www.uniovi.es poema.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://www.uniovi.es">
<poema fecha="Abril de 1915" lugar="Granada" autor="Federico García Lorca">
<titulo>Alba</titulo>
<verso>Mi corazón oprimido</verso>
<verso>siente junto a la alborada</verso>
<verso>el dolor de sus amores</verso>
<verso>y el sueño de las distancias.</verso>
</poema>
</poesía>
```

Software y estándares para la Web

Ejemplos de XML Schema (XI): árbol **alba.xml**

Creado en

poesia

xmlns: <http://www.uniovi.es>

xmlns:xsi: <http://www.w3.org/2001/XMLSchema-instance>

xsi:schemaLocation: <http://www.uniovi.es/poema.xsd>

poema

autor: Federico García Lorca

lugar: Granada

fecha: Abril de 1915

título

Alba

verso

Mi corazón oprimido

verso

siente junto a la alborada

verso

el dolor de sus amores

verso

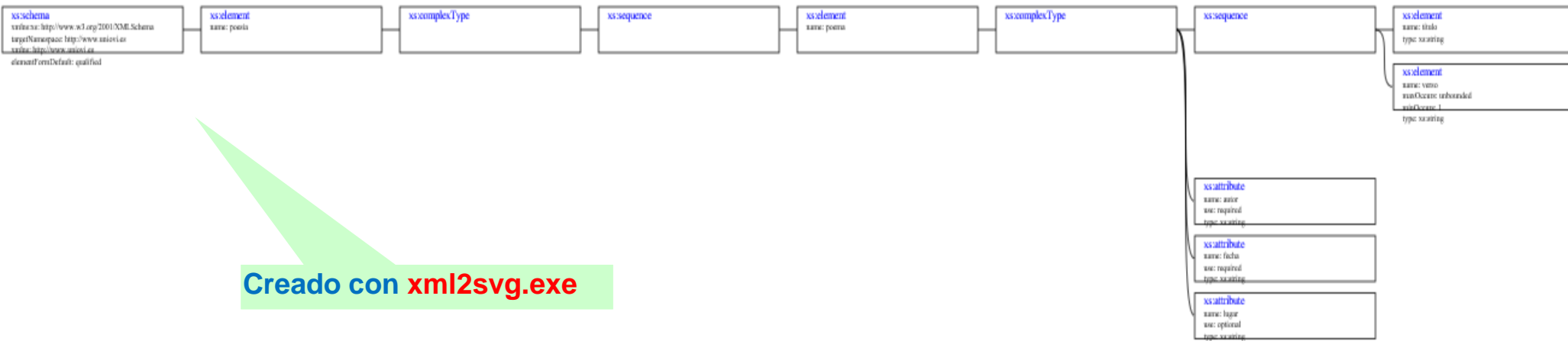
y el sueño de las distancias.

Creado con **xml2svg.exe**

Software y estándares para la Web

Ejemplos de XML Schema (XII): árbol poema.xsd

Grado en



Creado con **xml2svg.exe**

Software y estándares para la Web

Ejemplos de XML Schema (XIII): libros.xsd

Grado en
Ingeniería
Informática
del Software

```
libros.xsd
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema elementFormDefault="qualified" xmlns="http://www.uniovi.es" targetNamespace="http://www.uniovi.es" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  - <xs:element name="libros">
    - <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" minOccurs="1" maxOccurs="unbounded" type="xs:string"/>
              <xs:element name="any" type="xs:gYear"/>
            - <xs:element name="precio">
              - <xs:complexType>
                - <xs:simpleContent>
                  - <xs:extension base="xs:decimal">
                    <xs:attribute name="moneda" type="xs:string" use="required"/>
                  </xs:extension>
                </xs:simpleContent>
              </xs:complexType>
            </xs:sequence>
            <xs:element name="editorial" type="xs:string"/>
            <xs:element name="clasificacion" type="xs:string"/>
            <xs:element name="idioma" type="xs:string"/>
          </xs:sequence>
          <xs:attribute name="isbn" type="xs:ID" use="required"/>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
```

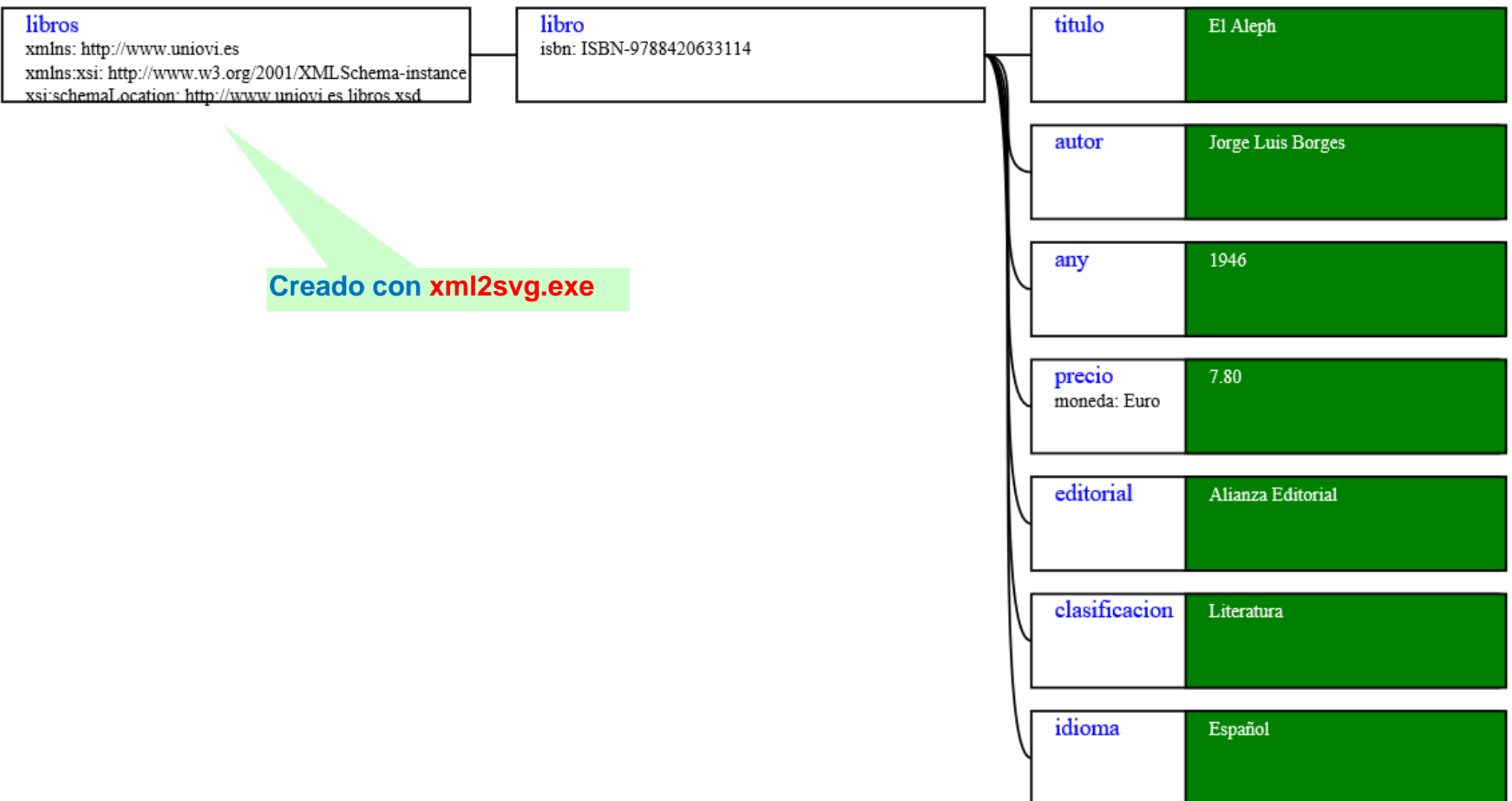
Software y estándares para la Web

Ejemplos de XML Schema (XIV): libros.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<libros xsi:schemaLocation="http://www.uniovi.es libros.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://www.uniovi.es">
  - <libro isbn="ISBN-9788420633114">
    <titulo>El Aleph</titulo>
    <autor>Jorge Luis Borges</autor>
    <any>1946</any>
    <precio moneda="Euro">7.80</precio>
    <editorial>Alianza Editorial</editorial>
    <clasificacion>Literatura</clasificacion>
    <idioma>Español</idioma>
  </libro>
</libros>
```

Software y estándares para la Web

Ejemplos de XML Schema (XV): árbol **libros.xml**



Ejemplos de XML Schema (XVI): árbol **libros.xsd**

Grado en Ingeniería Informática del Software



Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Introducción a XML Schema
- Ejemplos de XML Schema
- **De DTDs a Schemas**
- Definición de elementos simples
- Definición de atributos
- Definición de tipos
- Definición de elementos complejos
- Validación con XML Schema
- Limitaciones de XML Schema
- Generación automática de XML Schema
- Bibliografía
- Referencias Web
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

De DTDs a Schemas (I): Ejemplo dirección.dtd y dirección.xsd

Grado en
Ingeniería
Informática
del Software

```
<!ELEMENT dirección (empresa?, nombre, calle, ciudad, provincia, códigoPostal, teléfono+)>
<!ELEMENT      empresa      (#PCDATA)>
<!ELEMENT      nombre      (#PCDATA)>
<!ELEMENT      calle        (#PCDATA)>
<!ELEMENT      ciudad       (#PCDATA)>
<!ELEMENT      provincia    (#PCDATA)>
<!ELEMENT      códigoPostal (#PCDATA)>
<!ELEMENT      teléfono     (#PCDATA)>
```

dirección.xsd

```
<elementType name= "dirección" >
  <sequence>
    <elementType name = "empresa"      minOccurs="0"      maxOccur = "1"/>
    <elementType name = "nombre"       minOccurs="1"       maxOccur = "1"/>
    <elementType name = "calle"         minOccurs="1"       maxOccur = "1"/>
    <elementType name = "ciudad"        minOccurs="1"       maxOccur = "1"/>
    <elementType name = "provincia"     minOccurs="1"       maxOccur = "1"/>
    <elementType name = "códigoPostal" minOccurs="1"       maxOccur = "1"/>
    <elementType name = "teléfono"      minOccurs="1"       maxOccur = "unbounded"/>
  </sequence>
</elementType>
```

Software y estándares para la Web

De DTDs a Schemas (II): Reglas de conversión

Grado en
Ingeniería
Informática
del Software

```
*      minOccurs=0  maxOccurs=unbounded
+      minOccurs=1  maxOccurs=unbounded
?      minOccurs=0  maxOccurs=1
,      xs:sequence
|      xs:choice
X      xs:element
```

Software y estándares para la Web

De DTDs a Schemas (III): Espacio de nombres y elemento raíz

Grado en
Ingeniería
Informática
del Software

- El espacio de nombres del XML Schema es usualmente **xs:**

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

- **Schema** es el elemento raíz del documento

Ejemplo alumnos.xsd

Grado en
Ingeniería
Informática
del Software

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://www.uniovi.es/alumnos"
            xmlns="http://www.uniovi.es/alumnos">
  <xs:element name="alumnos">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="alumno" minOccurs="1" maxOccurs="200"
                    type="TipoAlumno"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="TipoAlumno">
    <xs:sequence>
      <xs:element name="nombre" type="xs:string"/>
      <xs:element name="apellidos" type="xs:string"/>
      <xs:element name="nacim" type="xs:gYear"/>
    </xs:sequence>
    <xs:attribute name="dni" type="xs:string"/>
  </xs:complexType>
</xs:schema>
```

Elemento raíz **schema** y
espacio de nombres
determinado

Permite especificar
rangos de inclusión

Permite especificar
tipos

Software y estándares para la Web

Validación `alumnos.xml`

`alumnos.xsd`

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
            targetNamespace="http://www.uniovi.es/alumnos"
            xmlns="http://www.uniovi.es/alumnos">
  <xs:element name="alumnos">
    ...
  </xs:schema>
```

`alumnos.xml`

```
<alumnos
  xmlns="http://www.uniovi.es/alumnos"
  xsi:SchemaLocation="http://www.uniovi.es/alumnos
                    alumnos.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  . . .
</alumnos>
```

Los espacios de nombres
deben coincidir.
También puede usarse:
`xsi:noNamespaceLocation`

Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

Vincular un Schema a un documento XML

Grado en
Ingeniería
Informática
del Software

alumnos.xml

```
<alumnos
  xmlns="http://www.uniovi.es/alumnos"
  xsi:SchemaLocation="http://www.uniovi.es/alumnos
                    alumnos.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  . . .
</alumnos>
```

pizzas.xml

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

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Introducción a XML Schema
- Ejemplos de XML Schema
- De DTDs a Schemas
- **Definición de elementos simples**
- Definición de atributos
- Definición de tipos
- Definición de elementos complejos
- Validación con XML Schema
- Limitaciones de XML Schema
- Generación automática de XML Schema
- Bibliografía
- Referencias Web
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Definición de elementos simples

- Sólo puede contener texto (cualquier tipo de dato) pero no otros elementos ni atributos
- Se define como

<xs:element name="nombre" type="tipo" />

donde

- **nombre** es el nombre del elemento
- Los valores más comunes de tipos de datos (**tipo**) son
 - xs:boolean**
 - xs:date**
 - xs:decimal**
 - xs:integer**
 - xs:string**
 - xs:time**
- Otros atributos:
 - **default="default value"** Valor por defecto de un atributo. Podría definirse otro valor.
 - **fixed="value"** Valor fijo de un atributo. Si no se define, se utiliza ése. Si se define, debe coincidir.
- *Ejemplo:* **<xs:element name="apellido" type="xs:string" />**

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Introducción a XML Schema
- Ejemplos de XML Schema
- De DTDs a Schemas
- Definición de elementos simples
- **Definición de atributos**
- Definición de tipos
- Definición de elementos complejos
- Validación con XML Schema
- Limitaciones de XML Schema
- Generación automática de XML Schema
- Bibliografía
- Referencias Web
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Definición de atributos (I)

- Los atributos se declaran:

```
<xs:attribute name="nombre" type="tipo" />
```

Donde:

- **nombre** y **tipo** es igual que en `xs:element`
- Otros atributos:
 - `default="default value"` *si no se especifica otro valor*
 - `fixed="value"` *no se puede especificar otro valor*
 - `use="optional"` *el atributo no es obligatorio (opcional)*
 - `use="required"` *el atributo debe estar presente (obligatorio)*
- *Ejemplo:*

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

Software y estándares para la Web

Definición de atributos (II): Ejemplos de atributos

Grado en
Ingeniería
Informática
del Software

```
<xs:complexType name="Círculo">
  <xs:attribute name="radio"
    type="xs:float"
    use="required" />

  <xs:attribute name="color"
    type="Color"
    default="255 0 0"/>

  <xs:attribute name="tipo"
    type="xs:string"
    fixed="jpeg" />
</xs:complexType>
```

Por defecto los atributos son opcionales. Indicar que son obligatorios: `use="required"`

Valor por defecto de un atributo. Podría definirse otro valor.

Valor fijo de un atributo. Si no se define, se utiliza ése. Si se define, debe coincidir.

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Introducción a XML Schema
- Ejemplos de XML Schema
- De DTDs a Schemas
- Definición de elementos simples
- Definición de atributos
- **Definición de tipos**
- Definición de elementos complejos
- Validación con XML Schema
- Limitaciones de XML Schema
- Generación automática de XML Schema
- Bibliografía
- Referencias Web
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Definición de tipos (I)

Grado en
Ingeniería
Informática
del Software

- De los tipos básicos pueden derivarse nuevos tipos: **simples** o **complejos**.
 - Los tipos **simples** contienen texto y se definen mediante **xs:simpleType** .
 - y los **complejos** pueden contener cualquier combinación de contenido de elementos , información de caracteres y atributos y se definen con **xs:complexType**

Software y estándares para la Web

Definición de tipos (II): Tipos Simples

Grado en
Ingeniería
Informática
del Software

- No pueden contener elementos o atributos
- Pueden ser:
 - **Predefinidos** o *built-in* (definidos en la especificación)
 - Primitivos
 - Derivados
 - **Definidos por el usuario** (a partir de tipos predefinidos)

Software y estándares para la Web

Definición de tipos (III) Tipos Primitivos

Grado en
Ingeniería
Informática
del Software

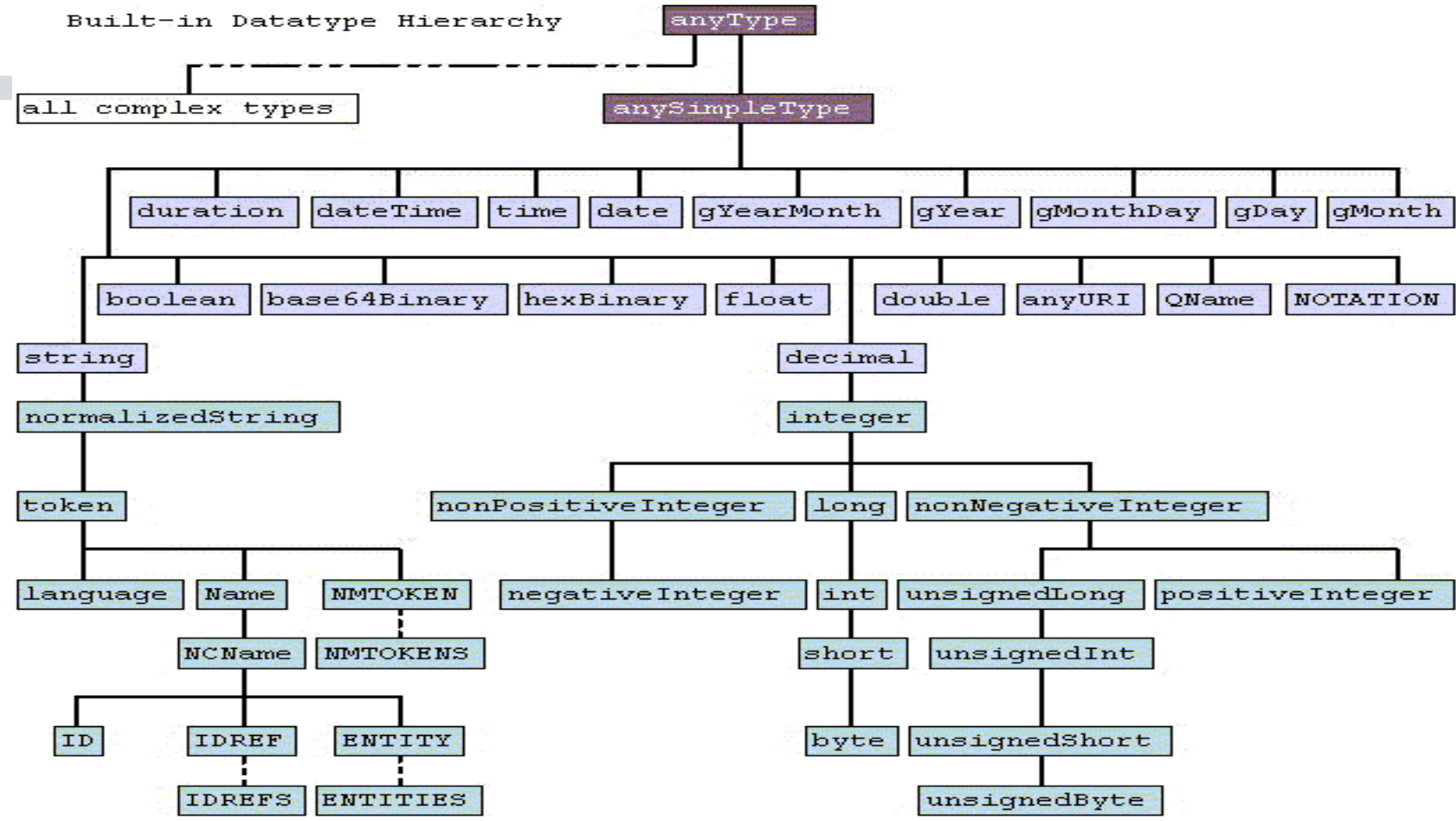
- string
- boolean
- number, float, double
- duration, dateTime, time, date, gYearMonth, gYear, gMonthDay, gDay, gMonth
- hexBinary, base64Binary
- anyURI
- QName = Nombre cualificado con espacio de nombres
- NOTATION = Notación binaria (similar a las NOTATION de los DTD)

Software y estándares para la Web

Definición de tipos (IV): Tipos Derivados

**Grado en
Ingeniería
Informática
del Software**

- `normalizedString`, `token`, `language`
- `IDREFS`, `ENTITIES`, `NMTOKEN`, `NMTOKENS`, `Name`, `NCName`, `ID`, `IDREF`, `ENTITY`
- `integer`, `nonPositiveInteger`, `negativeInteger`, `long`, `int`, `short`, `byte`, `nonNegativeInteger`, `unsignedLong`, `unsignedInt`, `unsignedShort`, `unsignedByte`, `positiveInteger`



Grado en Ingeniería Informática del Software

ur types

built-in primitive types

built-in derived types

complex types

—————

derived by restriction

derived by list

derived by extension or restriction

Software y estándares para la Web

Definición de tipos (VI): Tipos para fechas y tiempo

- Tipo fecha (**xs:date**)
 - Ejemplo
 - `<xs:element name="inicio" type="xs:date"/>`
 - `<inicio>2018-10-20</inicio>`
- Zonas horarias (por ejemplo Colombia, UTC-5)
 - Ejemplo
 - `<inicio>2018-10-20-05:00</inicio>`
- Tipo fecha y hora (**xs:dateTime**)
 - Ejemplo
 - `<xs:element name="inicioRuta" type="xs:dateTime"/>`
 - `<inicioRuta>2018-10-20T12:30:00</inicioRuta>`
- Tipo duración (**xs:duration**)
 - La duración, tipo de datos se utiliza para especificar un intervalo de tiempo.
 - El intervalo de tiempo se especifica en la siguiente forma "**PnYnMnDTnHnMnS**" donde:
 - P indica el período (obligatorio)
 - nY indica el número de años
 - nM indica el número de meses
 - nD indica el número de días
 - T indica el inicio de una sección de tiempo (necesario si se va a especificar horas, minutos o segundos)
 - nH indica el número de horas
 - nM indica el número de minutos
 - nS indica el número de segundos
 - Ejemplo
 - `<xs:element name="duracionRuta" type="xs:duration"/>`
 - `<duracionRuta>PT7H</duracionRuta>` indica una duración de 7 horas
 - `<duracionRuta>P1Y7M29DT12H</duracionRuta>` indica una duración de 1 año, 7 meses, 29 días y 12 horas

Software y estándares para la Web

Definición de tipos (VII): Facetas de Tipos

- Facetas fundamentales:
 - *equal*: Igualdad entre valores de un tipo de datos
 - *ordered*: Relaciones de orden entre valores
 - *bounded*: Límites inferiores y superiores para valores
 - *cardinality*: Define si es finito o infinito (numerable, no numerable)
 - *numeric*: Define si es numérico o no
- Facetas de restricción
 - *length, minlength, maxlength*: Longitud del tipo de datos
 - *pattern*: Restricciones sobre valores mediante expresiones regulares
 - *enumeration*: Restringe a una determinada enumeración de valores
 - *whitespace*: Define política de tratamiento de espacios (preserve/replace, collapse)
 - *(max/min)(in/ex)clusive*: Límites superiores/inferiores del tipo de datos
 - *totaldigits, fractionDigits*: número de dígitos totales y decimales

- **Restricciones:** Permiten restringir el valor que se le puede dar a un elemento o atributo XML.
- **Tipos:**
 - Sobre valores
 - Sobre un conjunto de valores
 - Sobre series de valores
 - Sobre espacios en blanco

Software y estándares para la Web

Definición de tipos (IX): Restricciones sobre valores

Grado en
Ingeniería
Informática
del Software

- La forma general de establecer una restricción sobre un valor es:

```
<xs:element name="name"> (o xs:attribute)
  <xs:restriction base="type">
    ... Las restricciones ...
  </xs:restriction>
</xs:element>
```

- Por ejemplo:

```
<xs:element name="edad">
  <xs:restriction base="xs:integer">
    <xs:minInclusive value="0">
    <xs:maxInclusive value="140">
  </xs:restriction>
</xs:element>
```

Software y estándares para la Web

Definición de tipos (X): Ejemplo

Grado en
Ingeniería
Informática
del Software

```
<xs:simpleType name="mes">  
  <xs:restriction base="xs:integer">  
    <xs:minInclusive value="1" />  
    <xs:maxInclusive value="31" />  
  </xs:restriction>  
</xs:simpleType>
```

Software y estándares para la Web

Definición de tipos (XI): Enumeration

Grado en
Ingeniería
Informática
del Software

- Restringe el valor a un conjunto de valores
- Ejemplo:

```
<xs:element name="estación">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:enumeration value="primavera"/>
      <xs:enumeration value="verano"/>
      <xs:enumeration value="otoño"/>
      <xs:enumeration value="invierno"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```


Software y estándares para la Web

Definición de tipos (XII): Ejemplo enumeración

Grado en
Ingeniería
Informática
del Software

```
<xs:simpleType name="Máster">  
  <xs:restriction base="xs:token">  
    <xs:enumeration value="Ingeniería Web"/>  
    <xs:enumeration value="Ciberseguridad"/>  
  </xs:restriction>  
</xs:simpleType>
```

Software y estándares para la Web

Definición de tipos (XIII): Restricciones sobre series de valores

Grado en
Ingeniería
Informática
del Software

- Para limitar el contenido del elemento XML definiendo las series de números y letras que se pueden usar usaremos patrones (expresiones regulares):

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

Software y estándares para la Web

Definición de tipos (XIV): Expresiones regulares

Grado en
Ingeniería
Informática
del Software

```
<xs:simpleType name="NIF">
  <xs:restriction base="xs:token">
    <xs:pattern value="\d{7,8}[A-Z]" />
  </xs:restriction>
</xs:simpleType>

<xs:element name="nif" type="NIF" />
```

```
<nif>9394173J</nif>
```

```
<nif>11079845M</nif>
```

Software y estándares para la Web

Definición de tipos (XV): Ejemplos de expresiones regulares (a)

Grado en
Ingeniería
Informática
del Software

EXPRESIÓN REGULAR

Elemento \d

a*b

[xyz]b

a?b

a+b

[a-c]x

POSIBLES VALORES

Elemento 2

b, ab, aab, aaab, ...

xb, yb, zb

b, ab

ab, aab, aaab, ...

ax, bx, cx

Software y estándares para la Web

Definición de tipos (XVI): Ejemplos de expresiones regulares (b)

Grado en
Ingeniería
Informática
del Software

<code>[a-c]x</code>	<code>ax, bx, cx</code>
<code>[^0-9]x</code>	Carácter ≠ dígito seguido de <code>x</code>
<code>\Dx</code>	Carácter ≠ dígito seguido de <code>x</code>
<code>(pa){2}rucha</code>	<code>paparucha</code>
<code>.abc</code>	Cualquier carácter seguido de <code>abc</code>
<code>(a b)+x</code>	<code>ax, bx, aax, bbx, abx, bax, ...</code>
<code>a{1,3}x</code>	<code>ax, aax, aaax</code>
<code>\n</code>	Salto de línea
<code>\p{Lu}</code>	Letra mayúscula
<code>\p{Sc}</code>	Símbolo de moneda

Software y estándares para la Web

Definición de tipos (XVII): Restricciones sobre espacios en blanco

Grado en
Ingeniería
Informática
del Software

- **whiteSpace** indica lo que hay que hacer con los espacios en blanco
 - **Value = "preserve"** Mantiene los espacios en blanco como están
 - **Value = "replace"** Cambia los espacios en blanco (saltos, tab...) por espacios
 - **Value = "collapse"** Reemplaza todas las secuencias de espacios en blanco por un sólo espacio en blanco

Software y estándares para la Web

Definición de tipos (XVIII): Restricciones en números

Grado en
Ingeniería
Informática
del Software

- **minInclusive:** El número debe ser \geq *value*
- **minExclusive:** El número debe ser $>$ *value*
- **maxInclusive:** El número debe ser \leq *value*
- **maxExclusive:** El número debe ser $<$ *value*
- **totalDigits:** El número debe tener exactamente *value* dígitos
- **fractionDigits:** El número no debe tener más de *value* dígitos después del punto decimal.

Software y estándares para la Web

Definición de tipos (XIX): Restricciones en strings

Grado en
Ingeniería
Informática
del Software

- **length** – el string debe contener exactamente **value** caracteres
- **minLength** – el string debe contener al menos **value** caracteres
- **maxLength** – el string no debe contener más de **value** caracteres
- **pattern** – el **value** es una expresión regular que el string debe cumplir

Software y estándares para la Web

Definición de tipos (XX): Listas

- Se pueden aplicar las facetas: `length`, `maxLength`, `minLength`, `enumeration`

```
<xs:simpleType name="ComponentesRGB">  
  <xs:list itemType="ComponenteRGB" />  
</xs:simpleType>
```

```
<xs:simpleType name="ComponenteRGB">  
  <xs:restriction base="xs:nonNegativeInteger">  
    <xs:maxInclusive value="255" />  
  </xs:restriction>  
</xs:simpleType>
```

```
<xs:simpleType name="ColorRGB">  
  <xs:restriction base="ComponentesRGB">  
    <xs:length value="3" />  
  </xs:restriction>  
</xs:simpleType>
```

```
<color>255 255 0</color>
```

Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

Definición de tipos (XXI): Listas

Grado en
Ingeniería
Informática
del Software

- En el ejemplo anterior **falla la restricción de `xs:length="3"`**
- La razón de que no funciona es que la faceta ***xs:length*** solamente se puede aplicar a los siguientes tipos simples:
 - `xs:anyURI`, `xs:base64Binary`, `xs:ENTITIES`, `xs:ENTITY`,
 - `xs:hexBinary`, `xs:ID`, `xs:IDREF`, `xs:IDREFS`, `xs:language`,
 - `xs:Name`, `xs:NCName`, `xs:NMTOKEN`, `xs:NMTOKENS`,
 - `xs:normalizedString`, `xs:NOTATION`, `xs:QName`,
 - `xs:string`, `xs:token`
- Se propone modificar el ejemplo para que funcione la restricción

Software y estándares para la Web

Definición de tipos (XXII): Uniones

Grado en
Ingeniería
Informática
del Software

```
<xs:simpleType name="TipoNota">  
  <xs:union>  
    <xs:simpleType>  
      <xs:restriction base="xs:float">  
        <xs:maxInclusive value="10" />  
        <xs:minInclusive value="0" />  
      </xs:restriction>  
    </xs:simpleType>  
    <xs:simpleType>  
      <xs:restriction base="xs:string">  
        <xs:enumeration value="No presentado" />  
      </xs:restriction>  
    </xs:simpleType>  
  </xs:union>  
</xs:simpleType>  
  
<xs:element name="nota" type="TipoNota" />
```

<nota> 5.75 </nota>

<nota> No presentado </nota>

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Introducción a XML Schema
- Ejemplos de XML Schema
- De DTDs a Schemas
- Definición de elementos simples
- Definición de atributos
- Definición de tipos
- **Definición de elementos complejos**
- Validación con XML Schema
- Limitaciones de XML Schema
- Generación automática de XML Schema
- Bibliografía
- Referencias Web
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Definición de elementos complejos (I)

Grado en
Ingeniería
Informática
del Software

- Elementos que contienen otros elementos hijo o que tienen atributos.
- Se suelen dividir en 4 **tipos**:
 - Elementos vacíos
 - Elementos no vacíos con atributos
 - Elementos con elementos hijos
 - Elementos con elementos hijos y con “texto” o valor propio

Software y estándares para la Web

Definición de elementos complejos (II): Sintaxis

Grado en
Ingeniería
Informática
del Software

- Se definen como:

```
<xs:element name="nombre">
  <xs:complexType>
    ...
  </xs:complexType>
</xs:element>
```

- Ejemplo:

```
<xsd:element name="empleado">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="nombre" type="xsd:string"/>
      <xsd:element name="apellidos" type="xsd:string"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
```

Software y estándares para la Web

Definición de elementos complejos (III): Indicadores

Grado en
Ingeniería
Informática
del Software

- De Orden:
 - En un determinado orden o secuencia: **sequence**
 - Pudiendo el autor del documento escoger alguno de los elementos del grupo: **choice**
 - Dar al autor total libertad tanto respecto al orden como a la selección de elementos: **all**
- De ocurrencias: **cardinalidades**

Software y estándares para la Web

Definición de elementos complejos (IV): Secuencia

<xs:sequence>

- *Secuencia*: Construcción básica mediante enumeración de elementos
 - Elementos que contienen elementos

```
<xs:complexType name="TipoAlumno">
  <xs:sequence>
    <xs:element name="nombre" type="xs:string"/>
    <xs:element name="apellidos" type="xs:string"/>
    <xs:element name="nacim" type="xs:gYear"
      minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
  <xs:attribute name="dni" type="xs:integer"/>
</xs:complexType>
```

```
<alumno dni="10399390">
  <nombre>Manuel</nombre>
  <apellidos>Cueva Norniella</apellidos>
  <nacim>1924</nacim>
</alumno>
```


Software y estándares para la Web

Definición de elementos complejos (V): Alternativa

Grado en
Ingeniería
Informática
del Software

`<xs:choice>`

- *Choice* : Representa alternativas
 - ¡ Atención ! : Es una o-exclusiva

```
<xs:complexType name="Transporte">  
  <xs:choice>  
    <xs:element name="coche" type="xs:string"/>  
    <xs:element name="tren" type="xs:string"/>  
    <xs:element name="avión" type="xs:string"/>  
  </xs:choice>  
</xs:complexType>
```

```
<transporte>  
<coche>Renault R23</coche>  
</transporte>
```

Software y estándares para la Web

Definición de elementos complejos (VI): Secuencias no ordenadas

<xs:all>

- *all* = Todos los elementos en cualquier orden
- En los DTDs se requería enumerar las combinaciones:
(A,B,C)|(A,C,B)|...|(C,B,A)

```
<xs:complexType name="TipoLibro">
  <xs:all>
    <xs:element name="autor" type="xs:string"/>
    <xs:element name="título" type="xs:string"/>
  </xs:all>
</xs:complexType>
<xs:element name="libro" type="TipoLibro" />
```

```
<libro>
  <autor>Juanita la Loca</autor>
  <título>No estoy loca</título>
</libro>
```

```
<libro>
  <título>El Quijote</título>
  <autor>Cervantes</autor>
</libro>
```

Software y estándares para la Web

Definición de elementos complejos (VII): Cardinalidades

Grado en
Ingeniería
Informática
del Software

- **minOccurs** y **maxOccurs**
- Se utilizan para indicar el número máximo y mínimo de veces que puede aparecer un elemento hijo de un elemento complejo
 - **minOccurs**= **0** : Opcionalidad
 - **maxOccurs** = **unbounded**: no existe valor límite

- Transforma el modelo de contenido

$(a \mid (b, c?))^*$

en una definición mediante XMLSchema

Software y estándares para la Web

Definición de elementos complejos (IX): Solución

Grado en
Ingeniería
Informática
del Software

```
<xs:complexType>  
  <xs:choice minOccurs='0' maxOccurs='unbounded'>  
    <xs:element ref='a' />  
    <xs:sequence>  
      <xs:element ref='b' />  
      <xs:element ref='c'  
        minOccurs='0' maxOccurs='1' />  
    </xs:sequence>  
  </xs:choice>  
</xs:complexType>
```

Software y estándares para la Web

Definición de elementos complejos (X): Contenido Mixto

Grado en
Ingeniería
Informática
del Software

- El contenido *Mixto* permite mezclar texto con elementos
- Se añade `mixed="true"` al elemento `xs:complexType`
- El texto no se menciona en el elemento y puede ir en cualquier sitio (básicamente se ignora)

```
<xs:complexType name="TCom" mixed="true">
  <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element name="emph" type="xs:string"/>
  </xs:choice>
</xs:complexType>

<xs:element name="comentarios" type="TCom" />
```

```
<comentarios>
  Es un poco <emph>listillo</emph>
</comentarios>
```

Software y estándares para la Web

Definición de elementos complejos (XI): Agrupaciones

Grado en
Ingeniería
Informática
del Software

- Es posible nombrar agrupaciones de elementos y de atributos para hacer referencias a ellas

```
<xs:group name="nombreApellidos">
  <xs:sequence>
    <xs:element name="nombre" type="xs:string"/>
    <xs:element name="apellidos" type="xs:string"/>
  </xs:sequence>
</xs:group>
```

```
<xs:complexType name="TipoAlumno">
  <xs:group ref="nombreApellidos" />
  <xs:element name="carrera" type="xs:string"/>
</xs:complexType>
```

Software y estándares para la Web

Definición de elementos complejos (XII): Tipos “Anónimos” versus “Con nombre”

Grado en
Ingeniería
Informática
del Software

Anónimo

```
<xs:element name="alumno">  
  <xs:sequence>  
    <xs:element name="nombre" type="xs:string"/>  
    <xs:element name="apellidos" type="xs:string"/>  
  </xs:sequence>  
</xs:element>
```

+ legible

Con nombre

```
...  
<xs:element name="alumno" type="TipoAlumno"/>  
...  
<xs:ComplexType name="TipoAlumno">  
  <xs:sequence>  
    <xs:element name="nombre" type="xs:string"/>  
    <xs:element name="apellidos" type="xs:string"/>  
  </xs:sequence>  
</xs:ComplexType>
```


Software y estándares para la Web

Definición de elementos complejos (XIII): Otra posibilidad “Referencias”

Grado en
Ingeniería
Informática
del Software

```
<xs:element name="alumno">
  <xs:sequence>
    <xs:element name="nombre" type="xs:string"/>
    <xs:element name="apellidos" type="xs:string"/>
  </xs:sequence>
</xs:element>
```

```
<xs:element name="alumnos">
  <xs:sequence>
    <xs:element ref="alumno" />
  </xs:sequence>
</xs:element>
```

Software y estándares para la Web

Tipos Derivados por Extensión (I)

- Similar a las subclases de POO. Consiste en heredar elementos de un tipo base

```
<xs:complexType name="Figura" >
  <xs:attribute name="color" type="Color"/>
</xs:complexType>

<xs:complexType name="Rectángulo">
  <xs:complexContent>
    <xs:extension base="Figura">
      <xs:attribute name="base" type="xs:float" />
      <xs:attribute name="altura" type="xs:float" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="Círculo">
  <xs:complexContent>
    <xs:extension base="Figura">
      <xs:attribute name="radio" type="xs:float" />
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Software y estándares para la Web

Tipos Derivados por Extensión (II)

Grado en
Ingeniería
Informática
del Software

- Los tipos derivados pueden utilizarse en los mismos sitios que la clase base

```
<xs:element name="figuras">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="figura" type="Figura"
        maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

```
<figuras>
<figura base="23" altura="3" xsi:type="Rectángulo" />
<figura radio="3" xsi:type="Círculo" />
</figuras>
```

Es necesario especificar el tipo mediante **xsi:type**

Software y estándares para la Web

Tipos Abstractos

- Al igual que en la POO se pueden declarar tipos abstractos
- Mediante `abstract="true"` se declara un tipo como abstracto.
 - Ese tipo no puede usarse directamente
- También es posible limitar la derivación de tipos
`final="restriction"`

```
<xs:complexType name="Figura" abstract="true">  
  <xs:attribute name="color" type="Color"/>  
</xs:complexType>
```

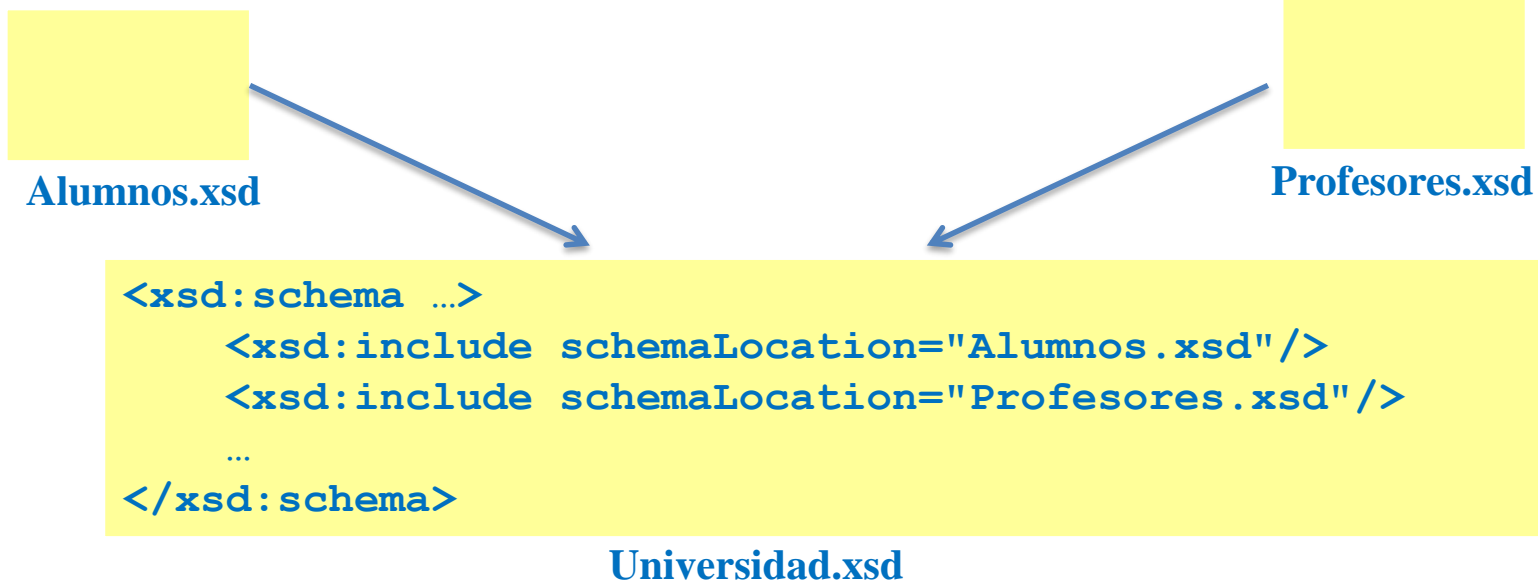
Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

Inclusión de Esquemas

Grado en
Ingeniería
Informática
del Software

- `<xsd:include schemaLocation="...">` permite incluir elementos de otros esquemas
 - Los elementos deben estar en el mismo espacio de nombres
 - Es como si se hubiesen tecleado todos en un mismo archivo



Software y estándares para la Web

Importación de Esquemas

Grado en
Ingeniería
Informática
del Software

- `<xsd:import namespace=...>` permite incluir elementos de otros esquemas con distintos espacios de nombres

Espacio de
nombres A

Espacio de
nombres P

```
<xsd:schema ...>
  <xsd:import namespace="A" schemaLocation="Alumnos.xsd"/>
  <xsd:import namespace="P" schemaLocation="Profes.xsd"/>
  ...
</xsd:schema>
```

Software y estándares para la Web

Redefinición de Esquemas

- `<xs:redefine ...>` es similar a *include* pero permite modificar los elementos incluidos.

Grado en
Ingeniería
Informática
del Software

Alumnos.xsd



Añade el elemento nota

```
<xs:redefine schemaLocation="Alumnos.xsd">
  <xs:complexType name="TipoAlumno">
    <xs:complexContent>
      <xs:extension base="TipoAlumno">
        <xs:sequence>
          <xs:element name="nota" type="Nota" />
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:redefine>
```

AlumnosConNotas.xsd

Software y estándares para la Web

Claves y Unicidad (I)

Grado en
Ingeniería
Informática
del Software


- Los DTDs proporcionaban el atributo **ID** para marcar la unicidad (un valor ID era único en todo el documento)
- XML Schema tiene más posibilidades:
 - Indicar que un elemento es único (**unique**)
 - Definir atributos únicos
 - Definir combinaciones de elementos y atributos como únicos
 - Distinción entre unicidad y claves (**key**)
 - Clave = además de ser único, debe existir y no puede ser nulo.
 - Declarar el rango de un documento en el que algo es único

Software y estándares para la Web

Claves y Unicidad (II)

Grado en
Ingeniería
Informática
del Software

```
<xs:complexType name="Alumnos">
  <xs:sequence>
    <xs:element name="Alumno" type="TipoAlumno"/>
  </xs:sequence>
  <xs:key name="DNI">
    <xs:selector xpath="a:alumno"/>
    <xs:field xpath="a:dni"/>
  </xs:key>
</xs:complexType>
```



Es necesario incluir el espacio de nombres (XPath)

La clave puede formarse para atributos y elementos

```
<xs:key name="DNI">
  <xs:selector xpath="a:alumno"/>
  <xs:field xpath="a:nombre"/>
  <xs:field xpath="a:apellidos"/>
</xs:key>
```

Una clave puede estar formada por varios elementos

Software y estándares para la Web

Claves y Unicidad (III)

Grado en
Ingeniería
Informática
del Software

```
<xs:complexType name="Alumnos">
  <xs:sequence>
    <xs:element name="Alumno" type="TipoAlumno"/>
  </xs:sequence>
  <xs:unique name="DNI">
    <xs:selector xpath="a:alumno"/>
    <xs:field xpath="a:dni"/>
  </xs:unique>
</xs:complexType>
```

Unique especifica que debe ser único, pero podría no existir

Software y estándares para la Web

Referencias a Claves

- **keyref** especifica que debe hacer referencia a una clave (Claves Externas)

```
<xs:element name="clase">
  <xs:sequence>
    <xs:element name="alumnos" ...
    <xs:element name="delegado" ...
  </xs:sequence>

  <xs:key name="DNI">
    <xs:selector xpath="a:alumnos/a:alumno"/>
    <xs:field xpath="a:dni"/>
  </xs:key>

  <xs:keyref name="Delegado" refer="DNI">
    <xs:selector xpath="a:delegado"/>
    <xs:field xpath="a:dni"/>
  </xs:keyref>
```

Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

Valores Nulos

Grado en Ingeniería Informática del Software

- Indicar que un elemento puede ser nulo sin estar vacío.
 - Vacío (Empty)**: Un elemento sin contenido
 - Nulo (Nil)**: Un elemento que indica que no hay valor

```
<xsd:element name="Persona">
  <xsd:complexType>
    <xsd:element name="nombre" type="xsd:NMTOKEN"/>
    <xsd:element name="primerApellido" type="xsd:NMTOKEN"/>
    <xsd:element name="segundoApellido" type="xsd:NMTOKEN"
      nillable="true"/>
  </xsd:complexType>
</xsd:element>
```

```
<persona>
  <nombre>John</nombre>
  <primerApellido>Smith</primerApellido>
  <segundoApellido xsi:nil="true"/>
</persona>
```

El segundo apellido puede ser un **NMTOKEN** o estar indefinido

Software y estándares para la Web

Incluir cualquier contenido

- **any** indica cualquier contenido de un determinado espacio de nombres
- **anyAttribute** cualquier atributo de un espacio de nombres

Grado en
Ingeniería
Informática
del Software

```
<xs:complexType name="Comentario">
  <xs:sequence>
    <xs:any namespace="http://www.w3.org/1999/xhtml"
      minOccurs="1"
      processContents="skip" />
  </xs:sequence>
  <xs:anyAttribute
    namespace="http://www.w3.org/1999/xhtml" />
</xs:complexType>
```

También puede declararse
##any, ##local, ##other

```
<comentarios>
  <html:p>Es un
    <html:emph>Listillo</html:emph>
  </html:p>
</comentarios>
```

Otros valores
strict = obliga a validar
lax = valida si es posible

Software y estándares para la Web

Generalización de comentarios

Grado en
Ingeniería
Informática
del Software

- *annotation*: mecanismo para documentar los componentes de un esquema, haciendo una función similar a los comentarios
- *appinfo*: proporciona información a otras aplicaciones, lo que supone un procesamiento externo al propio documento
- *documentation*: texto o referencia a texto dentro del elemento *annotation*

Software y estándares para la Web

Ejemplo

Grado en
Ingeniería
Informática
del Software

```
<xsd:complexType nombre="elemento">
  <xsd:annotation">
    <xsd:documentation>
      este elemento es una descripción de ...
    </xsd:documentation>
  </xsd:annotation">
</xsd:complexType>
```

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

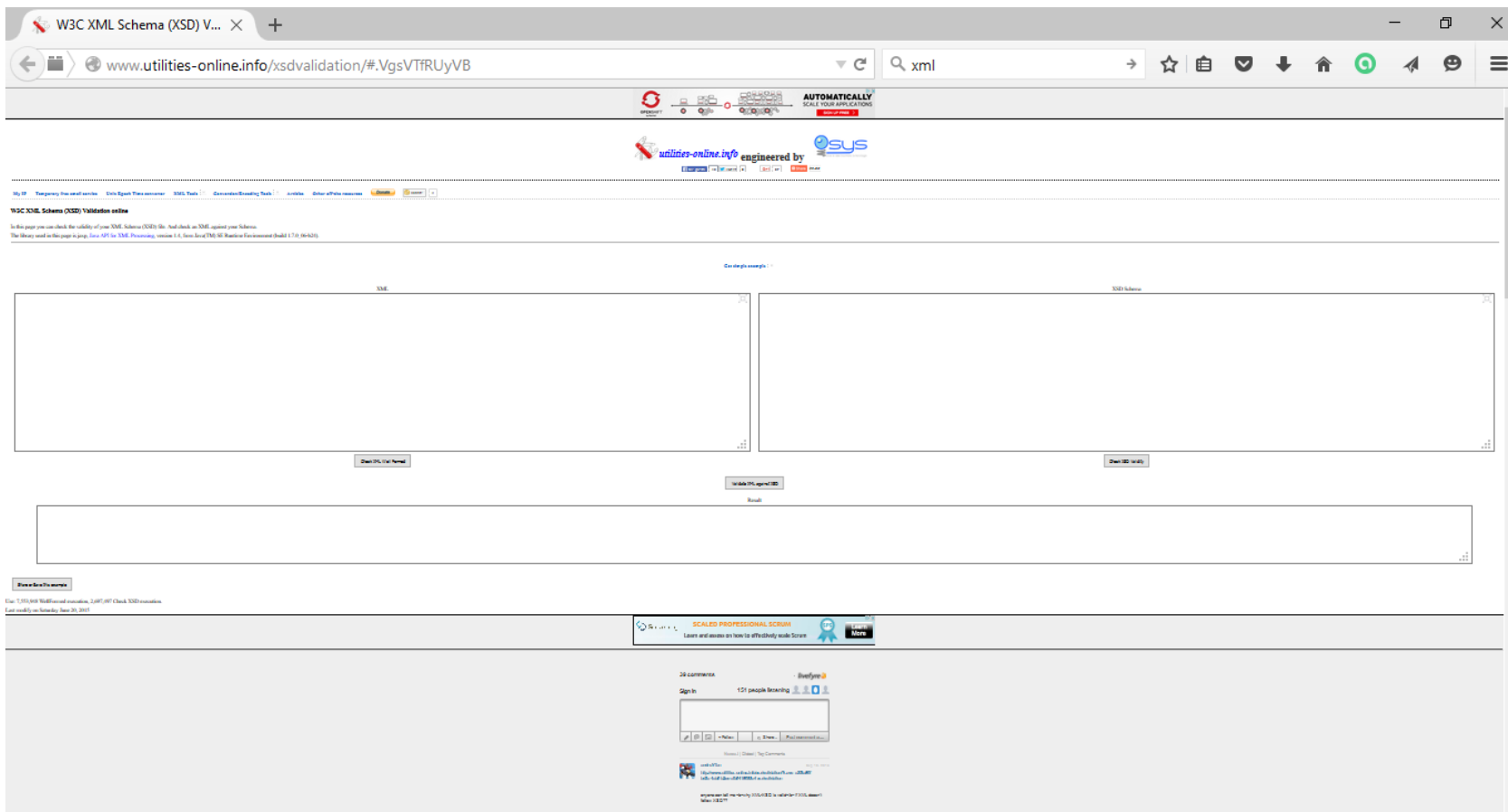
- Introducción a XML Schema
- Ejemplos de XML Schema
- De DTDs a Schemas
- Definición de elementos simples
- Definición de atributos
- Definición de tipos
- Definición de elementos complejos
- **Validación con XML Schema**
- Limitaciones de XML Schema
- Generación automática de XML Schema
- Bibliografía
- Referencias
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Validación con XML Schema(I): validador on-line

Grado en
Ingeniería
Informática
del Software

- <http://www.utilities-online.info/xsdvalidation/#.VgsVTfRUyVB>

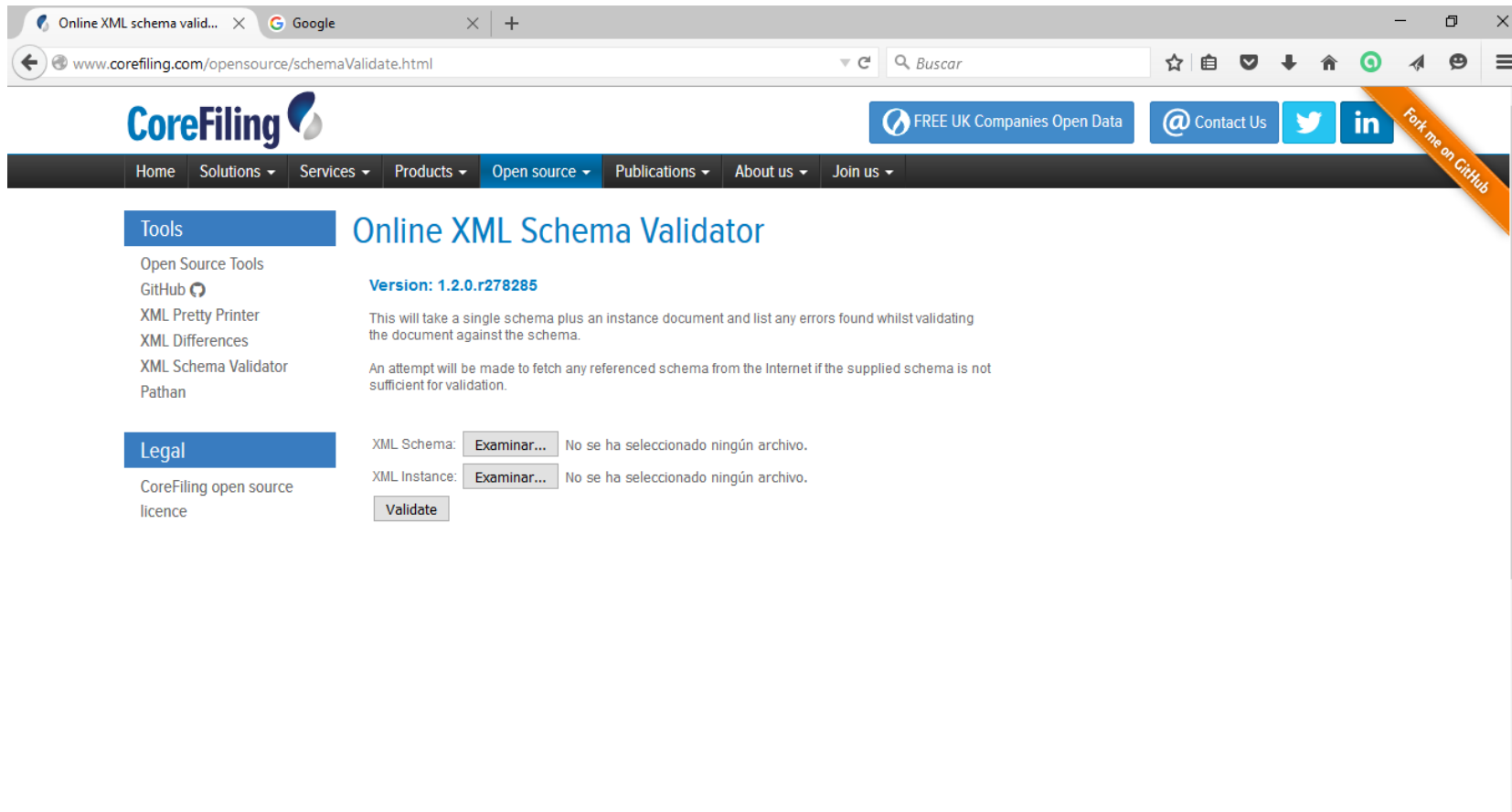


Software y estándares para la Web

Validación con XML Schema(II): validador on-line

Grado en
Ingeniería
Informática
del Software

- <http://www.corefiling.com/opensource/schemaValidate.html>



Software y estándares para la Web

Validación con XML Schema(III): validador on-line

Grado en
Ingeniería
Informática
del Software

- <http://www.freeformatter.com/xml-validator-xsd.html>

The screenshot shows a web browser window with the address bar displaying 'www.freeformatter.com/xml-validator-xsd.html'. The page has a dark header with the site name 'FREEFORMATTER.COM' and navigation links for 'HTTPS', 'FreeDataGenerator.com', and 'Contact'. Social media icons for Facebook, Twitter, and Google+ are also present. On the left, a sidebar lists various tools under categories like 'Formatters', 'Validators', 'Encoders & Decoders', 'Code Minifiers', 'Converters', and 'Cryptography'. The main content area is titled 'XML Validator - XSD (XML Schema)' and includes a description of the tool's function. Below the description, there are input sections for 'XML Input' and 'XSD Input', each with two options: 'Option 1: Copy-paste your [XML/XSD] string here' and 'Option 2: Or type in the URL to your [XML/XSD] file'. The 'XML Input' section has a text area with the placeholder 'http://www.example.com/myfile.xml'.

Software y estándares para la Web

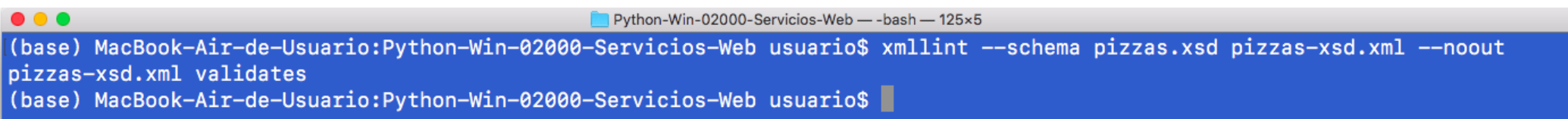
Validación con XML Schema(IV): en línea de comandos

Grado en
Ingeniería
Informática
del Software

- En **Linux** se puede validar un archivo **.xml** con un **.xsd** con el siguiente comando:
 - `$xmllint --schema pizzas.xsd pizzas-xsd.xml --noout`



```
cueva@Tikal: /mnt/c/Users/Juan Manuel Cueva/Dropbox/Python-Win-02000-Servicios-Web
cueva@Tikal:/mnt/c/Users/Juan Manuel Cueva/Dropbox/Python-Win-02000-Servicios-Web$ xmllint --schema pizzas.xsd pizzas-xsd.xml --noout
pizzas-xsd.xml validates
cueva@Tikal:/mnt/c/Users/Juan Manuel Cueva/Dropbox/Python-Win-02000-Servicios-Web$
```



```
Python-Win-02000-Servicios-Web — -bash — 125x5
(base) MacBook-Air-de-Usuario:Python-Win-02000-Servicios-Web usuario$ xmllint --schema pizzas.xsd pizzas-xsd.xml --noout
pizzas-xsd.xml validates
(base) MacBook-Air-de-Usuario:Python-Win-02000-Servicios-Web usuario$
```

Software y estándares para la Web

Validación con XML Schema(V): Editores con validación

Grado en
Ingeniería
Informática
del Software

- **Notepad++**
 - Editor libre para Windows
 - <https://notepad-plus-plus.org/>
 - Debe instalarse un plug-in para validar XML
 - XML Tools
 - Plugins >> Plugin Manager >> Show Plugin Manager >> XML Tools
- **OxygenXML**
 - Editor comercial
 - <http://www.oxygenxml.com/>
- **XMLSpy**
 - Editor comercial
 - <http://www.altova.com/xmlspy.html>

Software y estándares para la Web

Validación con XML Schema(VI): Notepad++

Grado en
Ingeniería
Informática
del Software

- Es importante usar **múltiples** validadores
- Los validadores dan distintos mensajes de error
- Cada validador tiene una implementación diferente
- **Se debe probar con varios validadores**

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Introducción a XML Schema
- Ejemplos de XML Schema
- De DTDs a Schemas
- Definición de elementos simples
- Definición de atributos
- Definición de tipos
- Definición de elementos complejos
- Validación con XML Schema
- **Limitaciones de XML Schema**
- Generación automática de XML Schema
- Bibliografía
- Referencias
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Limitaciones de XML Schema

Grado en
Ingeniería
Informática
del Software

- **No soporta entidades.** Mecanismo para crear macros
 `<!ENTITY &texto; "Esto texto se repite muchas veces" >`
 - Es necesario seguir usando los DTDs ☹️
- **Lenguaje de restricciones limitado**
 - Ejemplo: ¿Verificar valor total = suma de valores parciales?
- **Sensibilidad al contexto limitada**
 - Por ejemplo: Especificar que el contenido depende del valor de un atributo
 `<transporte tipo="coche"> ...</transporte>`
 `<transporte tipo="avión"> ...</transporte>`
- **Tamaño de archivos XML Schema puede ser excesivo**
- Legibilidad de las especificaciones. XML Schema **no siempre es muy legible**
- **Complejidad de la especificación:**
 - Muchas situaciones/combinaciones excepcionales
- Otras propuestas: Relax-NG, Schematron, etc.

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Introducción a XML Schema
- Ejemplos de XML Schema
- De DTDs a Schemas
- Definición de elementos simples
- Definición de atributos
- Definición de tipos
- Definición de elementos complejos
- Validación con XML Schema
- Limitaciones de XML Schema
- **Generación automática de XML Schema**
- Bibliografía
- Referencias
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

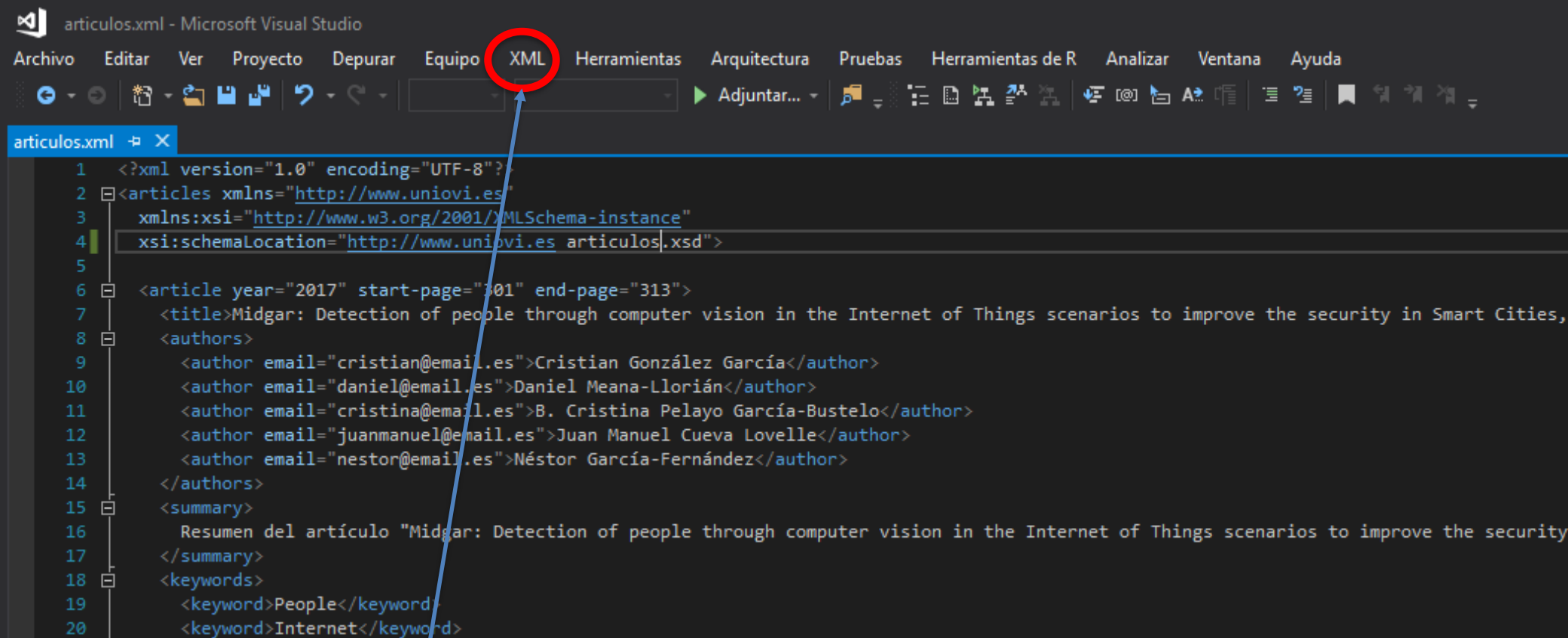
Generación automática de XML Schema

Grado en
Ingeniería
Informática
del Software

- Algunas herramientas generan automáticamente XML Schema a partir de un archivo XML o un archivo DTD
- El archivo generado nos ahorra tiempo de escritura
- Pero **es necesario modificarlo** debido a que es obligatorio
 - **Ajustar bien los tipos de datos**. Suelen aparecer demasiados *xs:string*
 - Algunos **elementos o atributos opcionales** pueden no aparecer si no se ha elegido lo suficientemente genérico el archivo de entrada a la herramienta
 - También es posible que no genere la codificación que deseemos

Software y estándares para la Web

Abrir archivo XML con Visual Studio 2019



Menú XML

Software y estándares para la Web

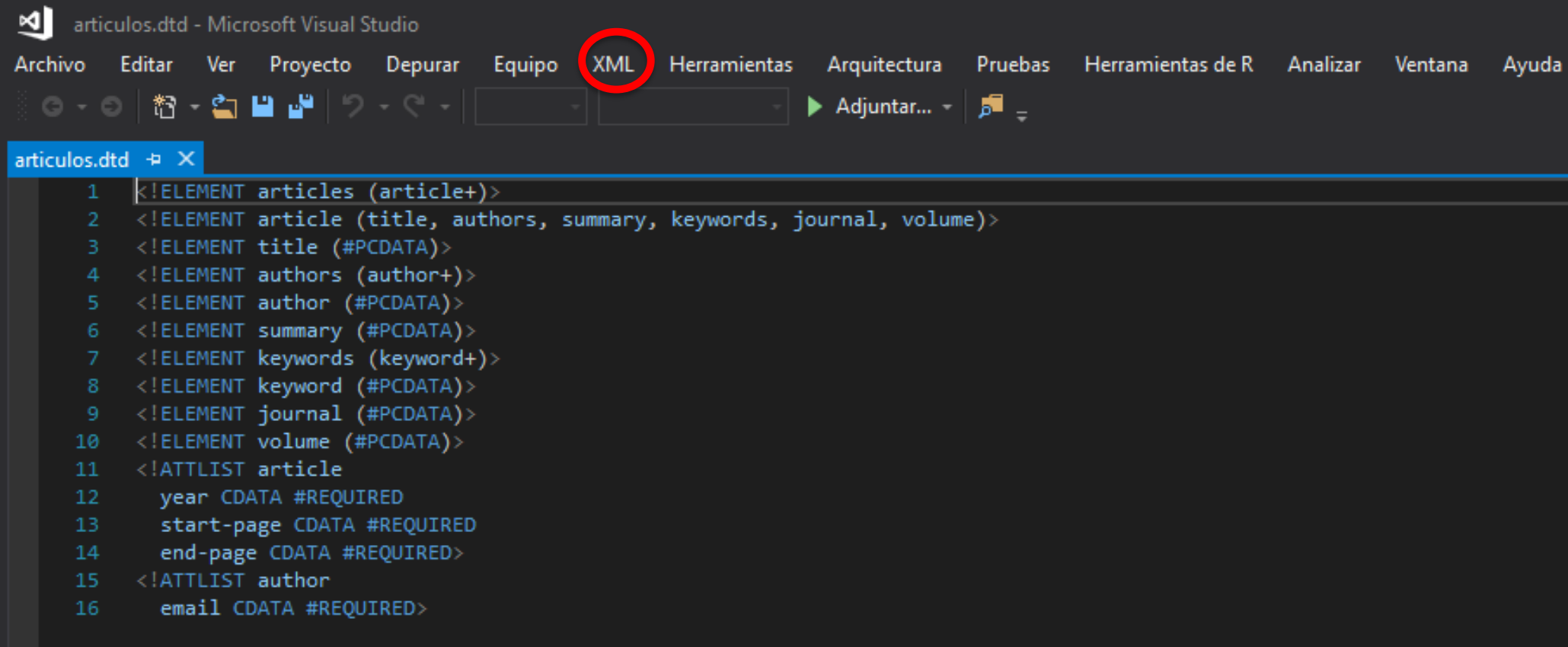
Archivo generado con Visual Studio 2019 a partir de un XML

The image shows a screenshot of the Microsoft Visual Studio IDE. The title bar at the top reads "articulos.xsd - Microsoft Visual Studio". The menu bar includes "Archivo", "Editar", "Ver", "Proyecto", "Depurar", "Equipo", "XML", "Herramientas", "Arquitectura", "Pruebas", "Herramientas de R", "Analizar", "Ventana", and "Ayuda". The toolbar contains various icons for file operations, editing, and development. The main editor window displays an XML schema file named "articulos.xml". The schema is written in Spanish and defines an XML structure for articles. It includes elements for title, authors, keywords, summary, journal, volume, year, start-page, and end-page. The schema uses the XML Schema language (XSD) and includes namespaces for XML Schema and the target namespace "http://www.uniovi.es".

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xi="http://www.w3.org/2001/XMLSchema-instance" attributeFormDefault="unqualified" elementFormDefault="qualified" targetNamespace="http://www.uniovi.es" xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="articles">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" name="article">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="title" type="xs:string" />
              <xs:element name="authors">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element maxOccurs="unbounded" name="author">
                      <xs:complexType>
                        <xs:simpleContent>
                          <xs:extension base="xs:string">
                            <xs:attribute name="email" type="xs:string" use="required" />
                          </xs:extension>
                        </xs:simpleContent>
                      </xs:complexType>
                    </xs:element>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="summary" type="xs:string" />
        <xs:element name="keywords">
          <xs:complexType>
            <xs:sequence>
              <xs:element maxOccurs="unbounded" name="keyword" type="xs:string" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="journal" type="xs:string" />
        <xs:element name="volume" type="xs:unsignedByte" />
      </xs:sequence>
      <xs:attribute name="year" type="xs:unsignedShort" use="required" />
      <xs:attribute name="start-page" type="xs:unsignedShort" use="required" />
      <xs:attribute name="end-page" type="xs:unsignedShort" use="required" />
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Software y estándares para la Web

Abrir DTD con Visual Studio 2019



The screenshot shows the Microsoft Visual Studio 2019 interface. The title bar reads "articulos.dtd - Microsoft Visual Studio". The menu bar includes "Archivo", "Editar", "Ver", "Proyecto", "Depurar", "Equipo", "XML", "Herramientas", "Arquitectura", "Pruebas", "Herramientas de R", "Analizar", "Ventana", and "Ayuda". The "XML" menu is circled in red. Below the menu bar is a toolbar with icons for opening, saving, and other file operations, followed by a dropdown menu and an "Adjuntar..." button. The main editor area displays the content of "articulos.dtd" with line numbers 1 through 16. The code defines XML elements and attributes for an articles document.

```
1 <!ELEMENT articles (article+)>
2 <!ELEMENT article (title, authors, summary, keywords, journal, volume)>
3 <!ELEMENT title (#PCDATA)>
4 <!ELEMENT authors (author+)>
5 <!ELEMENT author (#PCDATA)>
6 <!ELEMENT summary (#PCDATA)>
7 <!ELEMENT keywords (keyword+)>
8 <!ELEMENT keyword (#PCDATA)>
9 <!ELEMENT journal (#PCDATA)>
10 <!ELEMENT volume (#PCDATA)>
11 <!ATTLIST article
12   year CDATA #REQUIRED
13   start-page CDATA #REQUIRED
14   end-page CDATA #REQUIRED>
15 <!ATTLIST author
16   email CDATA #REQUIRED>
```

Software y estándares para la Web

Archivo generado con Visual Studio 2019 a partir de un DTD

Grado en
Ingeniería
Informática
del Software

```
articulos.xsd - Microsoft Visual Studio
Archivo  Editar  Ver  Proyecto  Depurar  Equipo  XML  Herramientas  Arquitectura  Pruebas  Herramientas de R  Analizar  Ventana  Ayuda
articulos.xsd
1  <?xml version="1.0" encoding="Windows-1252"?>
2  <xs:schema xmlns="http://tempuri.org/articulos" elementFormDefault="qualified" targetNamespace="http://tempuri.org/articulos" xmlns:xs="http://www.w3.org/2001/XMLSchema">
3    <xs:element name="articles">
4      <xs:complexType>
5        <xs:sequence>
6          <xs:element minOccurs="1" maxOccurs="unbounded" ref="article" />
7        </xs:sequence>
8      </xs:complexType>
9    </xs:element>
10   <xs:element name="article">
11     <xs:complexType>
12       <xs:sequence>
13         <xs:element ref="title" />
14         <xs:element ref="authors" />
15         <xs:element ref="summary" />
16         <xs:element ref="keywords" />
17         <xs:element ref="journal" />
18         <xs:element ref="volume" />
19       </xs:sequence>
20       <xs:attribute name="year" type="xs:string" use="required" />
21       <xs:attribute name="start-page" type="xs:string" use="required" />
22       <xs:attribute name="end-page" type="xs:string" use="required" />
23     </xs:complexType>
24   </xs:element>
25   <xs:element name="title" type="xs:string" />
26   <xs:element name="authors">
27     <xs:complexType>
28       <xs:sequence>
29         <xs:element minOccurs="1" maxOccurs="unbounded" ref="author" />
30       </xs:sequence>
31     </xs:complexType>
32   </xs:element>
33   <xs:element name="author">
34     <xs:complexType>
35       <xs:simpleContent>
36         <xs:extension base="xs:string">
37           <xs:attribute name="email" type="xs:string" use="required" />
38         </xs:extension>
39       </xs:simpleContent>
40     </xs:complexType>
41   </xs:element>
42   <xs:element name="summary" type="xs:string" />
43   <xs:element name="keywords">
44     <xs:complexType>
45       <xs:sequence>
46         <xs:element minOccurs="1" maxOccurs="unbounded" ref="keyword" />
47       </xs:sequence>
48     </xs:complexType>
49   </xs:element>
50   <xs:element name="keyword" type="xs:string" />
51   <xs:element name="journal" type="xs:string" />
52   <xs:element name="volume" type="xs:string" />
53 </xs:schema>
```

Software y estándares para la Web

Esquema

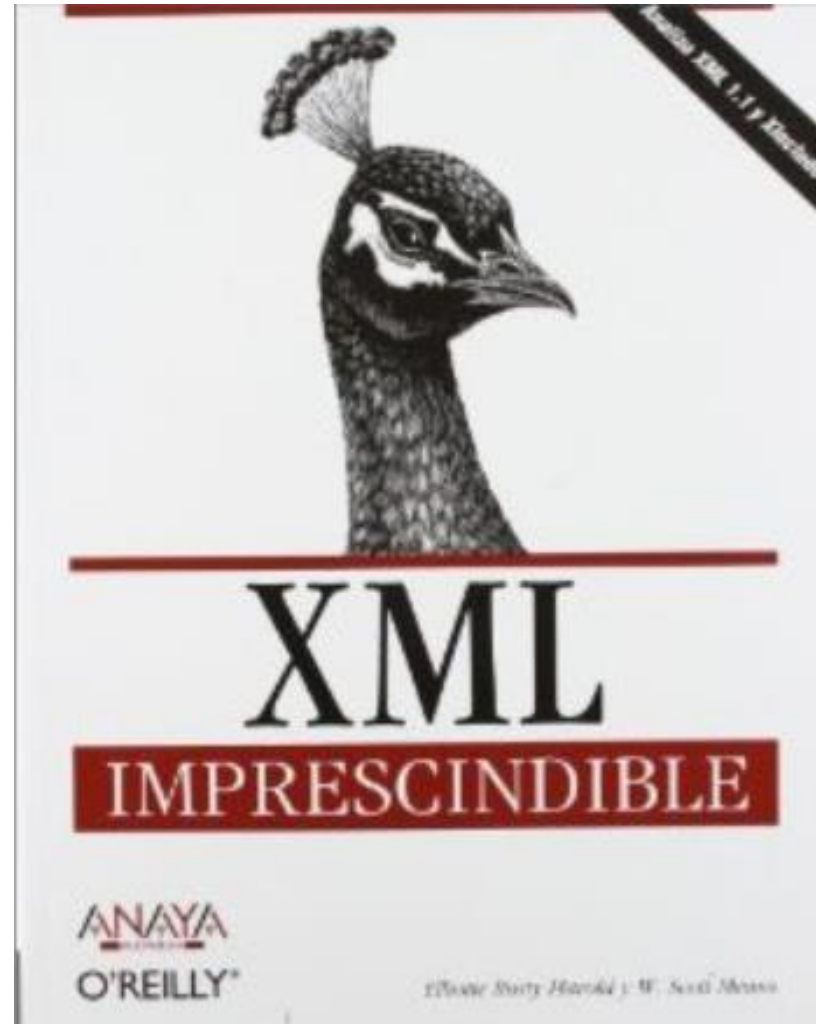
Grado en
Ingeniería
Informática
del Software

- Introducción a XML Schema
- Ejemplos de XML Schema
- De DTDs a Schemas
- Definición de elementos simples
- Definición de atributos
- Definición de tipos
- Definición de elementos complejos
- Validación con XML Schema
- Limitaciones de XML Schema
- Generación automática de XML Schema
- **Bibliografía**
 - Referencias
 - Ejercicios resueltos
 - Ejercicios propuestos

Software y estándares para la Web

Bibliografía (I)

- Libro recomendado de lectura y consulta:
 - “**XML imprescindible**”
 - ANAYA/O'Reilly (2005)
 - E. Rusty Harold y W. Scott Means



Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

Bibliografía (II)

- Libro recomendado de consulta:
 - “**Beginning XML**”
 - John Wiley & Sons (2012)
 - Joe Fawcett, Liam R.E. Quin, and Danny Ayers

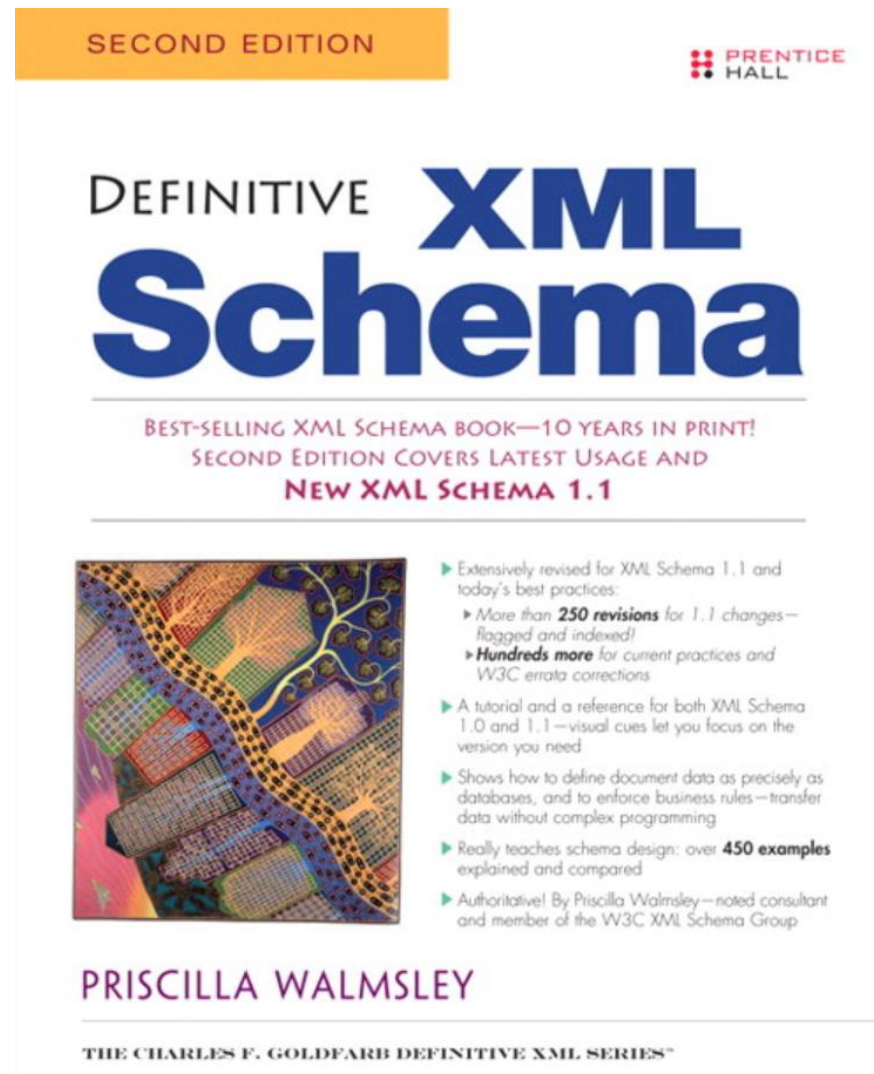


Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

Bibliografía recomendada

- Libro “**Definitive XML Schema**”.
Second Edition.
- Autora: Priscilla Walmsley
- Editorial: Prentice Hall (2012)



Grado en
Ingeniería
Informática
del Software

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Introducción a XML Schema
- Ejemplos de XML Schema
- De DTDs a Schemas
- Definición de elementos simples
- Definición de atributos
- Definición de tipos
- Definición de elementos complejos
- Validación con XML Schema
- Limitaciones de XML Schema
- Generación automática de XML Schema
- Bibliografía
- **Referencias Web**
- Ejercicios resueltos
- Ejercicios propuestos

Software y estándares para la Web

Referencias Web (I)

**Grado en
Ingeniería
Informática
del Software**

- W3C recomendación XML Schema:
 - Introducción:
 - <http://www.w3.org/TR/xmlschema-0/>
 - Estructuras:
 - <http://www.w3.org/TR/xmlschema-1/>
 - Tipos de datos:
 - <http://www.w3.org/TR/xmlschema-2/>
 - Herramientas y recursos
 - <http://www.w3.org/XML/Schema>

Software y estándares para la Web

Referencias Web (II)

**Grado en
Ingeniería
Informática
del Software**

- ¿Cómo validar XML con Schema?
 - <http://www.adrianmouat.com/bit-bucket/2013/11/xml-schema-validation/>
- Tutoriales
 - https://www.w3schools.com/xml/schema_intro.asp
 - <https://www.abrirllave.com/xsd/guion-del-tutorial.php>
- Validadores de XML con XML Schema
 - <http://www.utilities-online.info/xsdvalidation/#.Vgl5jTahccA>
 - <http://www.corefiling.com/opensource/schemaValidate.html>
 - <http://www.freeformatter.com/xml-validator-xsd.html>

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Introducción a XML Schema
- Ejemplos de XML Schema
- De DTDs a Schemas
- Definición de elementos simples
- Definición de atributos
- Definición de tipos
- Definición de elementos complejos
- Validación con XML Schema
- Limitaciones de XML Schema
- Generación automática de XML Schema
- Bibliografía
- Referencias Web
- **Ejercicios resueltos**
- Ejercicios propuestos

Software y estándares para la Web

Ejercicios propuestos (I)

Grado en
Ingeniería
Informática
del Software

- Construir un **documento XML bien formado y válido usando un XML Schema** para contener artículos de revistas con los siguientes requisitos mínimos:
 - Título del artículo
 - Autores y su correo electrónico
 - Resumen
 - Palabras clave
 - Nombre de la revista
 - Número o volumen de la revista
 - Página de inicio del artículo
 - Página final del artículo
 - Año

Software y estándares para la Web

Ejercicios propuestos (I) – Solución - articulos.xsd

Grado en
Ingeniería
Informática
del Software

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

  <xs:element name="articles">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="article"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="article">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="title" minOccurs="1" maxOccurs="1" type="xs:string"/>
        <xs:element minOccurs="1" maxOccurs="1" ref="authors"/>
        <xs:element name="summary" minOccurs="1" maxOccurs="1" type="xs:string"/>
        <xs:element minOccurs="1" maxOccurs="1" ref="keywords"/>
        <xs:element name="journal" minOccurs="1" maxOccurs="1" type="xs:string"/>
        <xs:element name="volume" minOccurs="1" maxOccurs="1" type="xs:integer"/>
      </xs:sequence>
```

Software y estándares para la Web

Ejercicios propuestos (I) Solución - articulos.xsd - continuación

Grado en
Ingeniería
Informática
del Software

```
<xs:attribute name="year" use="required" type="xs:gYear"/>
<xs:attribute name="start-page" use="required" type="xs:integer"/>
<xs:attribute name="end-page" use="required" type="xs:integer"/>
</xs:complexType>
</xs:element>

<xs:element name="authors">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="unbounded" ref="author"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="author">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="email" type="xs:string" use="required"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

Software y estándares para la Web

Ejercicios propuestos (I) Solución - `articulos.xsd` - continuación

Grado en
Ingeniería
Informática
del Software

```
<xs:element name="keywords">
<xs:complexType>
<xs:sequence>
<xs:element name="keyword" minOccurs="0" maxOccurs="unbounded" type="xs:string"/>
</xs:sequence>
</xs:complexType>
</xs:element>

</xs:schema>
```

Software y estándares para la Web

Ejercicios propuestos (I) Validación

E:\Dropbox\Asignaturas\MIW\Lenguajes-y-estándares-Web\XML\2018\03-Ejemplos\articulos.xml - Notepad++

Archivo Editar Buscar Vista Codificación Lenguaje Configuración Herramientas Macro Ejecutar Plugins Ventana ?

articulos.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <articles xmlns="http://www.uniovi.es"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xsi:schemaLocation="http://www.uniovi.es articulos.xsd">
5
6   <article year="2017" start-page="301" end-page="313">
7     <title>Midgar: Detection of people through computer vision in the Internet of Things scenarios to improve the security in Smart Cities, Smart Towns, and Smart Homes.</title>
8     <authors>
9       <author email="cristian@email.es">Cristian González García</author>
10      <author email="daniel@email.es">Daniel Meana-Llorián</author>
11      <author email="cristina@email.es">B. Cristina Pelayo García-Bustelo</author>
12      <author email="juanmanuel@email.es">Juan Manuel Cueva Lovelle</author>
13      <author email="nestor@email.es">Néstor García-Fernández</author>
14    </authors>
15    <summary>
16      Resumen del artículo "Midgar: Detection of people through computer vision in the Internet of Things scenarios to improve the security in Smart Cities, Smart Towns, and Smart Homes.".
17    </summary>
18    <keywords>
19      <keyword>People</keyword>
20      <keyword>Internet</keyword>
21      <keyword>Things</keyword>
22    </keywords>
23    <journal>Future Generation Comp. Syst.</journal>
24    <volume>76</volume>
25  </article>
26
27  <article year="2016" start-page="299" end-page="337">
28    <title>Combining the Continuous Integration Practice and the Model-Driven Engineering Approach.</title>
29    <authors>
30      <author email="vicente@email.es">Vicente García-Díaz</author>
31      <author email="jordan@email.es">Jordán Pascual Espada</author>
32      <author email="edward@email.es">Edward Rolando Núñez-Valdéz</author>
33      <author email="cristina@email.es">B. Cristina Pelayo García-Bustelo</author>
34      <author email="juanmanuel@email.es">Juan Manuel Cueva Lovelle</author>
35    </authors>
36    <summary>
37      Resumen del artículo "Combining the Continuous Integration Practice and the Model-Driven Engineering Approach.".
38    </summary>
39    <keywords>
40      <keyword>Integration</keyword>
41      <keyword>Practice</keyword>
42      <keyword>Model-Driven</keyword>
43    </keywords>
44    <journal>Computing and Informatics</journal>
45    <volume>35</volume>
46  </article>
```

XML Tools plugin

XML Schema validation:
XML is valid.

Aceptar

length : 9.484 lines : 226 Ln : 1 Col : 1 Sel : 0 | 0 Windows (CR LF) UTF-8 INS

Software y estándares para la Web

Ejercicios propuestos (II)

Grado en
Ingeniería
Informática
del Software

- Construir un **documento XML bien formado y válido usando un XML Schema** para contener recetas de cocina con los siguientes requisitos mínimos:
 - Nombre de la receta (por ejemplo “Fabada Asturiana”)
 - Tipo de plato (postre, primer plato, entrante,. . .)
 - Ingredientes con cantidades (por ejemplo “Fabes 500 gramos”)
 - Calorías del plato (opcional)
 - Proceso de elaboración, especificado en pasos, por ejemplo:
 - Paso 1: Poner *les fabes* a remojo la noche anterior
 - Paso 2: Poner *les fabes* a cocer con agua y laurel
 - Paso 3: etc...
 - Dificultad del proceso de elaboración (por ejemplo “Fácil”, “Medio”, “Difícil”,...)
 - Tiempo de elaboración (por ejemplo “45 minutos”)
 - Elementos utilizados para la elaboración (microondas, wok, horno, freidora,. . .)
 - Origen de la receta (por ejemplo “Receta de mi abuela”, “Libro de M^a Luisa”, “Libro de las 1001 recetas”, “www.recetasMUYricas.com”)

Software y estándares para la Web

Ejercicios propuestos (II) – Solución - recetas.xsd (a)

Grado en
Ingeniería
Informática
del Software

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

  <xs:element name="cooking-recipes">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="cooking-recipe"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="cooking-recipe">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="name" minOccurs="1" maxOccurs="1" type="xs:string"/>
        <xs:element minOccurs="1" maxOccurs="1" ref="ingredients"/>
        <xs:element minOccurs="1" maxOccurs="1" ref="elaboration-process"/>
        <xs:element name="origin" minOccurs="1" maxOccurs="1" type="xs:string"/>
      </xs:sequence>
      <xs:attribute name="type" type="dish-type" use="required"/>
      <xs:attribute name="vegan" type="vegan-type" use="required"/>
    </xs:complexType>
  </xs:element>
```

Software y estándares para la Web

Ejercicios propuestos (II) – Solución - recetas.xsd (b)

Grado en
Ingeniería
Informática
del Software

```
<xs:simpleType name="dish-type">
<xs:restriction base="xs:string">
<xs:enumeration value="Entrante"/>
<xs:enumeration value="Primer plato"/>
<xs:enumeration value="Segundo plato"/>
<xs:enumeration value="Postre"/>
</xs:restriction>
</xs:simpleType>

<xs:simpleType name="vegan-type">
<xs:restriction base="xs:string">
<xs:enumeration value="Si"/>
<xs:enumeration value="No"/>
</xs:restriction>
</xs:simpleType>

<xs:element name="ingredients">
<xs:complexType>
<xs:sequence>
<xs:element minOccurs="0" maxOccurs="unbounded" ref="ingredient"/>
</xs:sequence>
</xs:complexType>
</xs:element>
```

Software y estándares para la Web

Ejercicios propuestos (II) – Solución - recetas.xsd (c)

Grado en
Ingeniería
Informática
del Software

```
<xs:element name="ingredient">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="quantity" type="quantity-type" use="required"/>
        <xs:attribute name="measure" type="xs:string" use="required"/>
        <xs:attribute name="calories" type="xs:integer" use="optional"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>

<xs:simpleType name="quantity-type">
  <xs:union memberTypes="xs:integer fraction-type" />
</xs:simpleType>

<xs:simpleType name="fraction-type">
  <xs:restriction base="xs:string">
    <xs:pattern value="-?\d+/-?\d+"/>
  </xs:restriction>
</xs:simpleType>
```


Software y estándares para la Web

Ejercicios propuestos (II) – Solución - recetas.xsd (d)

Grado en
Ingeniería
Informática
del Software

```
<xs:element name="elaboration-process">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="unbounded" ref="steps"/>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="elements"/>
    </xs:sequence>
    <xs:attribute name="difficulty" type="difficulty-process" use="required"/>
    <xs:attribute name="time-minutes" type="xs:integer" use="required"/>
  </xs:complexType>
</xs:element>

<xs:simpleType name="difficulty-process">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Fácil"/>
    <xs:enumeration value="Medio"/>
    <xs:enumeration value="Difícil"/>
  </xs:restriction>
</xs:simpleType>

<xs:element name="steps">
  <xs:complexType>
    <xs:sequence>
      <xs:element minOccurs="1" maxOccurs="unbounded" ref="step"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Software y estándares para la Web

Ejercicios propuestos (II) – Solución - recetas.xsd (e)

Grado en
Ingeniería
Informática
del Software

```
<xs:element name="step">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="number" type="xs:integer" use="required"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>

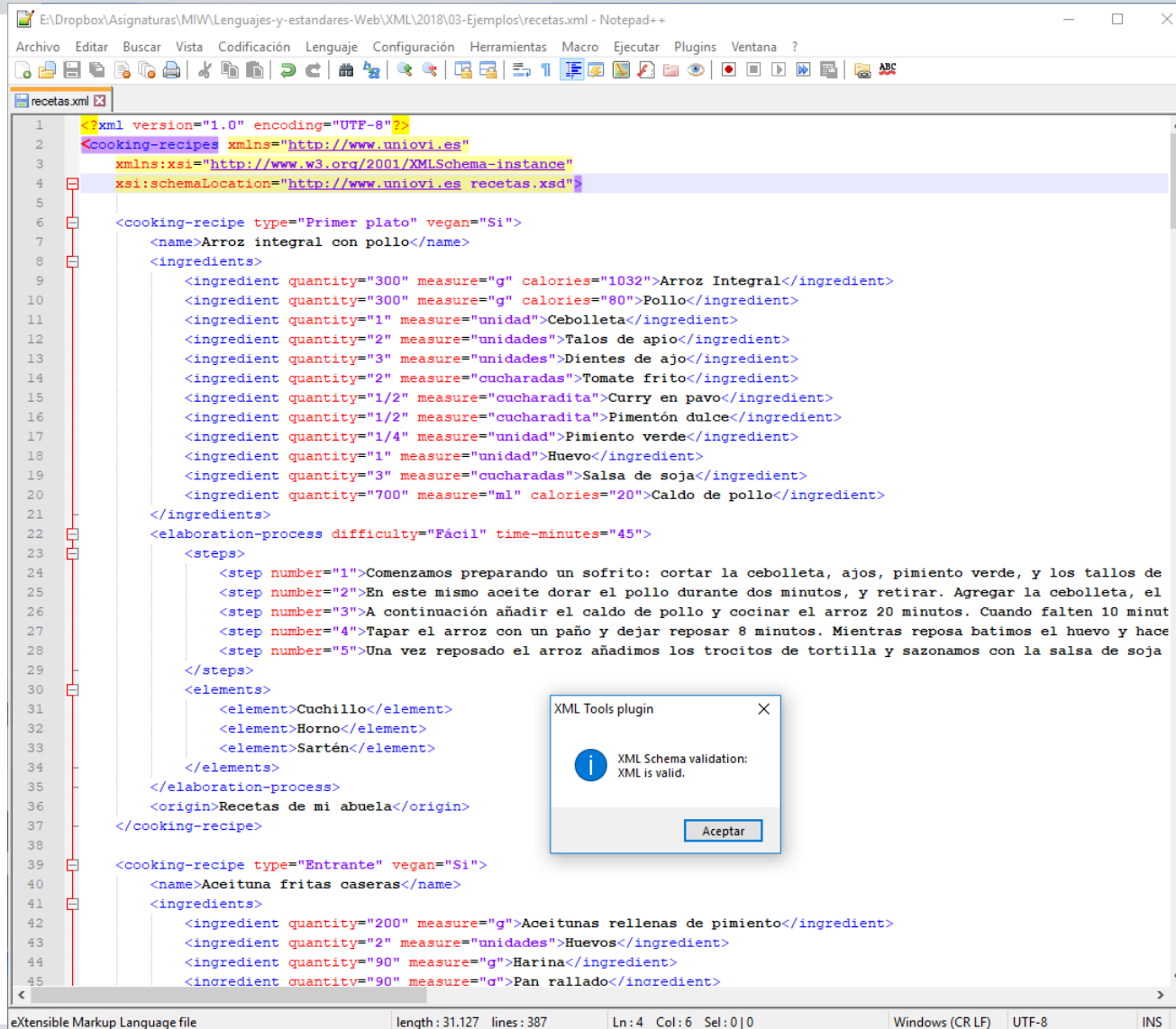
<xs:element name="elements">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="element" minOccurs="0" maxOccurs="unbounded" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

Software y estándares para la Web

Ejercicios propuestos (II) Validación

Grado en
Ingeniería
Informática
del Software



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <cooking-recipes xmlns="http://www.uniovi.es"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xsi:schemaLocation="http://www.uniovi.es recetas.xsd">
5
6   <cooking-recipe type="Primer plato" vegan="Si">
7     <name>Arroz integral con pollo</name>
8     <ingredients>
9       <ingredient quantity="300" measure="g" calories="1032">Arroz Integral</ingredient>
10      <ingredient quantity="300" measure="g" calories="80">Pollo</ingredient>
11      <ingredient quantity="1" measure="unidad">Cebolleta</ingredient>
12      <ingredient quantity="2" measure="unidades">Talos de apio</ingredient>
13      <ingredient quantity="3" measure="unidades">Dientes de ajo</ingredient>
14      <ingredient quantity="2" measure="cucharadas">Tomate frito</ingredient>
15      <ingredient quantity="1/2" measure="cucharadita">Curry en pavo</ingredient>
16      <ingredient quantity="1/2" measure="cucharadita">Pimentón dulce</ingredient>
17      <ingredient quantity="1/4" measure="unidad">Pimiento verde</ingredient>
18      <ingredient quantity="1" measure="unidad">Huevo</ingredient>
19      <ingredient quantity="3" measure="cucharadas">Salsa de soja</ingredient>
20      <ingredient quantity="700" measure="ml" calories="20">Caldo de pollo</ingredient>
21    </ingredients>
22    <elaboration-process difficulty="Fácil" time-minutes="45">
23      <steps>
24        <step number="1">Comenzamos preparando un sofrito: cortar la cebolleta, ajos, pimiento verde, y los tallos de
25        <step number="2">En este mismo aceite dorar el pollo durante dos minutos, y retirar. Agregar la cebolleta, el
26        <step number="3">A continuación añadir el caldo de pollo y cocinar el arroz 20 minutos. Cuando falten 10 minut
27        <step number="4">Tapar el arroz con un paño y dejar reposar 8 minutos. Mientras reposa batimos el huevo y hace
28        <step number="5">Una vez reposado el arroz añadimos los trocitos de tortilla y sazonamos con la salsa de soja
29      </steps>
30      <elements>
31        <element>Cuchillo</element>
32        <element>Horno</element>
33        <element>Sartén</element>
34      </elements>
35    </elaboration-process>
36    <origin>Recetas de mi abuela</origin>
37  </cooking-recipe>
38
39  <cooking-recipe type="Entrante" vegan="Si">
40    <name>Aceituna fritas caseras</name>
41    <ingredients>
42      <ingredient quantity="200" measure="g">Aoeitunas rellenas de pimiento</ingredient>
43      <ingredient quantity="2" measure="unidades">Huevos</ingredient>
44      <ingredient quantity="90" measure="g">Harina</ingredient>
45      <ingredient quantity="90" measure="g">Pan rallado</ingredient>
```

Software y estándares para la Web

Esquema

Grado en
Ingeniería
Informática
del Software

- Introducción a XML Schema
- Ejemplos de XML Schema
- De DTDs a Schemas
- Definición de elementos simples
- Definición de atributos
- Definición de tipos
- Definición de elementos complejos
- Validación con XML Schema
- Limitaciones de XML Schema
- Generación automática de XML Schema
- Bibliografía
- Referencias Web
- Ejercicios resueltos
- **Ejercicios propuestos**

Software y estándares para la Web

Ejercicios propuestos (I): Monumentos prerrománicos

Grado en
Ingeniería
Informática
del Software

- Construir un **documento XML bien formado y válido usando un XML Schema** para contener monumentos del prerrománico asturiano (5 monumentos) con los siguientes requisitos mínimos:
 - Nombre del monumento (por ejemplo “Fuente de Foncalada”)
 - Tipo de monumento (por ejemplo “Arquitectura hidráulica”, “Iglesia”, “Palacio”)
 - Año de construcción aproximado (por ejemplo “1096”)
 - Constructor (por ejemplo “Desconocido”, “Ramiro I”)
 - Descripción del monumento
 - Estado del monumento (por ejemplo “bien conservado”, “mal conservado”, etc)
 - Municipio (por ejemplo “Oviedo”)
 - Dirección opcional (por ejemplo “calle Foncalada”)
 - Coordenadas geográficas: longitud, latitud y altitud
 - Galería de fotografías:
 - Fotografía 1: Por ejemplo Foncalada-01.jpg
 - Fotografía 2: Por ejemplo Foncalada-02.jpg
 - Fotografía 3: etc...

Software y estándares para la Web

Ejercicios propuestos (I): Monumentos prerrománicos - continuación

**Grado en
Ingeniería
Informática
del Software**

- Galería de vídeos:
 - Video 1: Por ejemplo Foncalada-01.mpeg
 - Vídeo 2: Por ejemplo Foncalada-02.mpeg
 - Video 3: etc...
- Recomendación de visita de 0 a 10 (por ejemplo “7”)
- Horario de visita
- Días de visita
- ¿Quién lo enseña?
- Referencias y bibliografía con información del monumento
 - Referencia 1: por ejemplo <https://es.wikipedia.org/wiki/Foncalada>
 - Referencia 2: <http://prerromancoasturiano.es/>
 - Referencia 3. etc.

Software y estándares para la Web

Ejercicios propuestos (II): Rutas turísticas

Grado en
Ingeniería
Informática
del Software

- Construir un **documento XML bien formado y válido usando un XML Schema** para contener rutas turísticas (5 rutas mínimo) con los siguientes requisitos mínimos:
 - Nombre de la ruta turística (por ejemplo “Ruta por Oviedo”)
 - Tipo de ruta (por ejemplo “Arquitectura y monumentos”, “Gastronómica”, “Paisajística”, “Mixta tapas y monumentos”, “Escalada”, “Senderismo”, etc.)
 - Medio de transporte (por ejemplo “A pie”, “Automóvil”, “Bicicleta”, “Canoa”, “Mixta a pie y tren”, etc.)
 - Fecha de inicio de la ruta (opcional)
 - Hora de inicio de la ruta (opcional)
 - Tiempo de duración de la ruta (por ejemplo “2 horas”, “3 días”, “2 semanas”, “3 meses”)
 - Agencia que gestiona la ruta (por ejemplo “Sin agencia”, “NaturAller”)
 - Descripción de la ruta
 - Personas adecuadas para la ruta (por ejemplo “Se puede ir con niños”, “Personas en buena forma física”, “tercera edad”, etc.)
 - Lugar de inicio de la ruta (por ejemplo “Oviedo”)
 - Dirección de inicio de la ruta (por ejemplo “calle Foncalada”)
 - Coordenadas geográficas de inicio de la ruta: longitud, latitud y altitud

Software y estándares para la Web

Ejercicios propuestos (II): Rutas turísticas - continuación

**Grado en
Ingeniería
Informática
del Software**

- Referencias y bibliografía con información de la ruta (mínimo 3)
 - Referencia 1: por ejemplo <https://es.wikipedia.org/wiki/Foncalada>
 - Referencia 2: <http://prerromancoasturiano.es/>
 - Referencia 3. etc.
- Recomendación de la ruta de 0 a 10 (por ejemplo “7”)
- Hitos de la ruta (mínimo 3 hitos):
 - Nombre del sitio
 - Descripción del sitio
 - Coordenadas geográficas del sitio: longitud, latitud, altitud
 - Distancia desde el hito anterior (las unidades se expresarán como atributos)
 - Galería de fotografías del hito (mínimo 1, máximo 5)
 - Fotografía 1: Por ejemplo Monumento.jpg
 - Fotografía 2: Por ejemplo Panorama.jpg
 - Fotografía 3: etc...
 - Galería de vídeos del hito (opcional). Mínimo 0 y máximo 3.
 - Video 1: Por ejemplo Paisaje360.mpeg o enlace a YouTube, Vimeo, etc.
 - Video 2: Por ejemplo Modelo3D.mpeg
 - Video 3: etc...



XML: eXtensible Markup Languaje

Tecnologías XML

Esquemas XML (XML Schema)

Dr. Juan Manuel Cueva Lovelle
Departamento de Informática
Universidad de Oviedo
cueva@uniovi.es