2.8.- XML-SCHEMA.

2.8.1. Introducción y características.

XML Schema se encarga de definir la estructura de documentos XML que estén asignados a dicho esquema y los tipos de datos válidos para cada elemento y atributo. De esta forma, las posibilidades de control sobre la estructura y los tipos de datos son muy amplias. Al restringir el contenido de los ficheros XML se facilita el intercambio de información entre aplicaciones.

Los ficheros XML Schema se escriben en lenguaje XML y sus funciones son las de definir:

- Los elementos y atributos que pueden aparecer en un documento.
- Los tipos de datos de elementos y atributos.
- Los valores por defecto y fijos para elementos y atributos.
- Qué elementos son elementos hijos.
- El orden y número de los elementos hijos.
- Si un elemento puede estar vacío o puede incluir texto.

XML Schema tiene un enfoque modular que recuerda a la programación orientada a objetos y que facilita la reutilización de código. Los tipos de datos tienen en XML Schema la función de las clases en la POO (Programación Orientada a Objetos). El usuario puede construir tipos de datos a partir de tipos predefinidos, agrupando elementos y atributos de una determinada forma y con mecanismos de extensión parecidos a la herencia. Los tipos de datos se clasifican en función de los elementos y atributos que contienen y pueden ser simples o complejos.

XML Schema incluye el uso de "*espacios de nombres*". Los "*namespaces*" permiten definir elementos con igual nombre dentro del mismo contexto, siempre y cuando se anteponga un prefijo al nombre del elemento. El uso de "*namespaces*" también evita confusiones en la reutilización de código.

No debe confundirse los XML Schemas con los DTDs (Document Type Definitions). Los DTDs son también un lenguaje de esquema pero se diferencian principalmente con los XML Schemas en lo siguiente:

- Los XML Schema poseen un lenguaje propio de escritura, especificado en lenguaje XML.
- Los DTDs tienen un tipado para los datos del documento extremadamente limitado, no permite definir que un elemento pueda ser de un tipo numeral o de un tipo de fecha, sólo presenta variaciones limitadas sobre strings.

- Los XML Schemas son extensibles.

También hay que diferenciar entre un *documento XML bien formado* y *un documento XML válido*. Un *documento XML bien formado* es aquél que se ajusta a las normas de sintáxis XML, tales como que debe comenzar con la declaración XML, debe tener un único elemento raíz, todos los elementos deben cerrarse... Incluso si los documentos están bien formados, pueden contener errores.

Los lenguajes de esquema permiten verificar que un documento XML está bien formado y, además, que siga una estructura definida. A estos últimos se les denomina *documentos XML válidos*.

2.8.2. Estructura de un XML Schema.

Un documento XML Schema [21][22] tiene la extensión *".xsd"* e incluye (entre otros) lo siguientes elementos:

- *<xsd:schema>:* este elemento se utiliza como elemento raíz del documento y actúa como contenedor del resto del contenido del esquema.
- *xmlns:xsd* : todos los elementos en el esquema tienen el prefijo *xsd:* (está asociado al espacio de nombre del XML Schema a través de la declaración *xmlns:xsd)*.

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

2.8.3. Tipos XML Schema.

En un documento XML Schema los elementos pueden ser de dos tipos:

1) Tipos simples: se trata de elementos que no contienen elementos ni atributos, sólo texto. La sintaxis para definir un elemento simple es:

```
<xs:element name="xxx" type="zzz"/>
```

Donde "xxx" es el nombre del elemento y "zzz" es el tipo de datos del elemento.

Son tipos simples:

- Tipos predefinidos de XML: string, double, boolean, etc.
- List : lista de datos separados por espacios.
- Union: tipo de dato derivado de la unión de tipos predefinidos.

En la siguiente ilustración se muestra los tipos simples predefinidos:

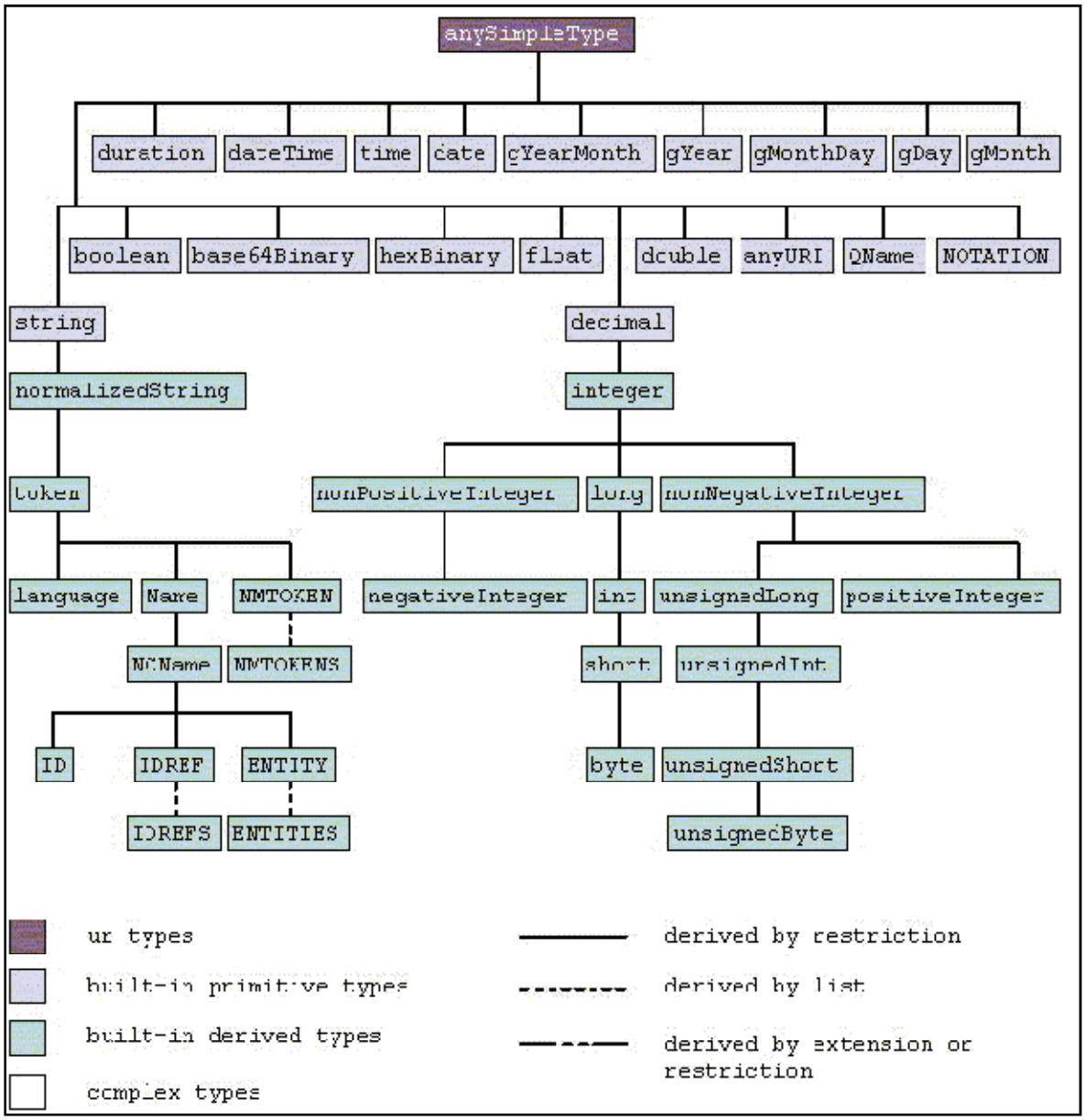

Ilustración 27.- Tipos simples predefinidos de XML Schema.

Los elementos de tipo simple son:

<xsd:element>	
<xsd:element name= **"nombreElemento"** type=**"tipoSimple/tipoCompuesto"** minOccurs=**"valor"** maxOccurs=**"valor"** **/>**	Atributos: *-name:* nombre del elemento que tomará cualquier instancia del documento asociada a este esquema. *-type:* indica el tipo de elemento: simple o complejo. *-minOccurs y maxOccurs:* la combinación de ambos indica el número de ocurrencias de un elemento.

Tabla 50.- Elemento complejo *<xsd:element>*.

<xsd:attribute>	
<xsd:attribute name="**nombreAtributo**" type="**tipoSimple**" use="**valor**" default="**valor**" fixed="**valor**" **/>**	Atributos: *-name:* nombre del atributo que está incluido en el elemento. *-type:* los atributos sólo permiten tipo simple. *-use: (opcional)* puede tomar los valores: *required* (el atributo debe aparecer), *optional* (el atributo puedo o no aparecer) y *prohibited* (el atributo no debe aparecer). *-default: (opcional)* si el atributo no aparece en el documento, el parser lo añade con dicho valor. *-fixed*: *(opcional)* si el atributo existe en el documento, su valor debe ser el mismo que el que indica *fixed.* Si no aparece, el parser lo añadirá con dicho valor. *Los atributos default y fixed no pueden coexistir.

Tabla 51.- Elemento complejo *<xsd:attribute>.*

Los valores que un tipo simple puede tomar pueden restringirse utilizando el elemento "*xsd:restriction":*

<xsd:restriction>	
<xsd:restriction base="**tipoSimple**"> **Propiedades** </xsd:restricion>	Propiedades: - Para limitar valores numéricos: *<xsd:minExclusive>,<xsd:minInclusive>,* *<xsd:maxExclusive>, <xsd:maxInclusive>.* - Para limitar la longitud de un string: *<xsd:length>, <xsd:minLength>, <xsd:maxLength>.* - Para limitar un tipo simple a un conjunto acotado de valores: *<xsd:enumeration>.* - Para aplicar expresiones regulares: *<xsd:pattern>.*
Engloba una secuencia o un grupo de elementos secundarios.	

Tabla 52.- Elemento simple *<xsd:restriction>.*

2) Tipos complejos: son aquellos elementos que pueden contener cualquier combinación de elementos, atributos y/o texto. Pueden tener nombre o ser anónimos. Si tienen nombre pueden ser reutilizados dentro del mismo XML Schema o por otros XML Schemas.

   Se definen utilizando el elemento *"<xsd:complexType>"* el cual puede contener los subelementos *"<xsd:sequence>", "<xsd:choice>"* y *"<xsd:all>.*

   Los elementos complejos más importantes se muestran en la Tabla 53.

3) Elemento global: no es un tipo de elemento propiamente dicho. Se utiliza en el caso de que un elemento se repita varias veces en el XML Schema y se crea un elemento global para no repetirlo. Un elemento global debe cumplir:

   - Tiene que estar declarado como un subelemento del elemento *<xsd:schema>*, pero nunca como parte de un elemento de tipo complejo.
   - No puede contener referencias, es decir, no pueden albergar el atributo ref.
   - No puede indicar el número de ocurrencias de un elemento.

   La sintaxis de una declaración local que hace referencia a un elemento global es:

*<xsd:element **ref**="nombreElementoGlobalYaExistente" **minOccurs**="valor">*

Donde el atributo *ref* hace referencia a un elemento global. En la declaración local se puede indicar la cardinalidad del elemento global con los atributos *minOccurs* y *maxOccurs*.

<xsd:secuence>, <xsd:choice>, <xsd:all>	
<xsd:secuence> Indica que la secuencia de elementos anidados tienen que aparecer en el documento XML y en el mismo orden.	<xsd:element name="camiseta"> <xsd:complexType> **<xsd:sequence>** <xsd:element name="color" type="xsd:string"/> <xsd:element name="talla" type="xsd:string"/> **<xsd:sequence>** </xsd:complexType> </xsd:element name="camiseta">
<xsd:choice> Indica una lista de elementos y de entre los cuales sólo puede aparecer uno en el documento.	<xsd:element name="vehiculoMotor"> <xsd:complexType> **<xsd:choice>** <xsd:element name="coche" type="xsd:string"/> <xsd:element name="moto" type="xsd:string"/> <xsd:element name="fugoneta" type="xsd:string"/> <xsd:element name="camion" type="xsd:string"/> **<xsd:choice>** </xsd:complexType> </xsd:element name="camiseta">
<xsd:all> Igual que *<xsd:element>* pero en este caso no es obligatorio que la secuencia de elementos anidados aparezcan en el documento en el mismo orden	<xsd:element name="camiseta"> <xsd:complexType> **<xsd:all>** <xsd:element name="color" type="xsd:string"/> <xsd:element name="talla" type="xsd:string"/> **<xsd:all>** </xsd:complexType> </xsd:element name="camiseta">

Tabla 53.- Elementos complejos *<xsd:secuence>, <xsd:choice> y <xsd:all>*.