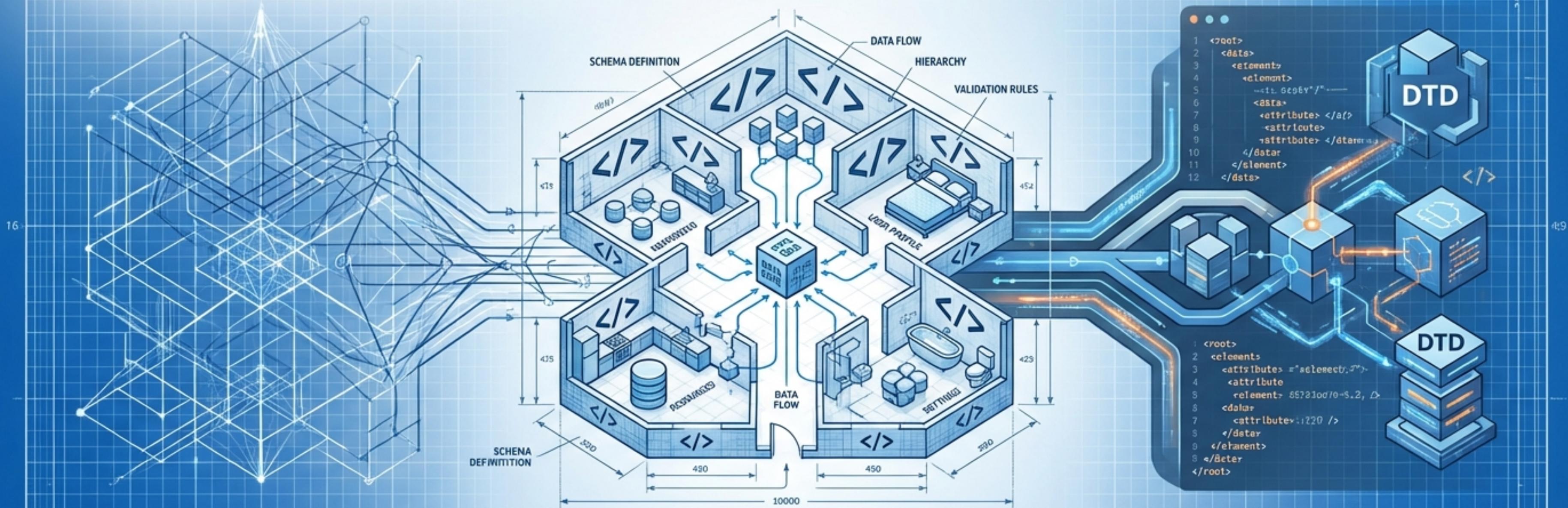


Estructura, Diseño y Validación en XML

El Arquitecto de Datos: De la Sintaxis al DTD



Definición de esquemas y vocabulario
para la interoperabilidad web.

El Desafío de WebMalagaDesign

El Contexto



Alberto necesita organizar el caos de información.
Gloria busca un método estándar de intercambio.



Inventario de Software: Catalogar licencias y versiones de forma estricta.



Comunicación: Estructurar correos con remitentes y destinatarios claros.



Datos Externos: Integrar información meteorológica (clima) en la web.

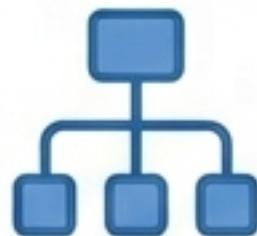
La Solución: XML como estándar independiente de la plataforma.

La Naturaleza del Material: ¿Qué es XML?

eXtensible Markup Language: Diseñado para transportar y guardar datos, no para mostrarlos.



LO QUE ES



Estructural: Árbol jerárquico con nodo raíz.



Semántico: Las etiquetas describen el significado (ej. <precio>).



Legible: Texto plano editable universalmente.



LO QUE NO ES



NO es HTML (no tiene formato visual intrínseco).



NO es una Base de Datos (es almacenamiento secuencial, no gestión).



NO es Programación (no ejecuta algoritmos por sí solo).

Anatomía de un Documento XML

- **Prólogo:** Declaración de versión y codificación.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

- **<biblioteca>**

```
    <libro>
```

```
        <titulo>Introducción a XML</titulo>
```

```
        <autor>Medac</autor>
```

```
    </libro>
```

```
</biblioteca>
```

- **Nodo Raíz:** Elemento padre único que contiene todo.

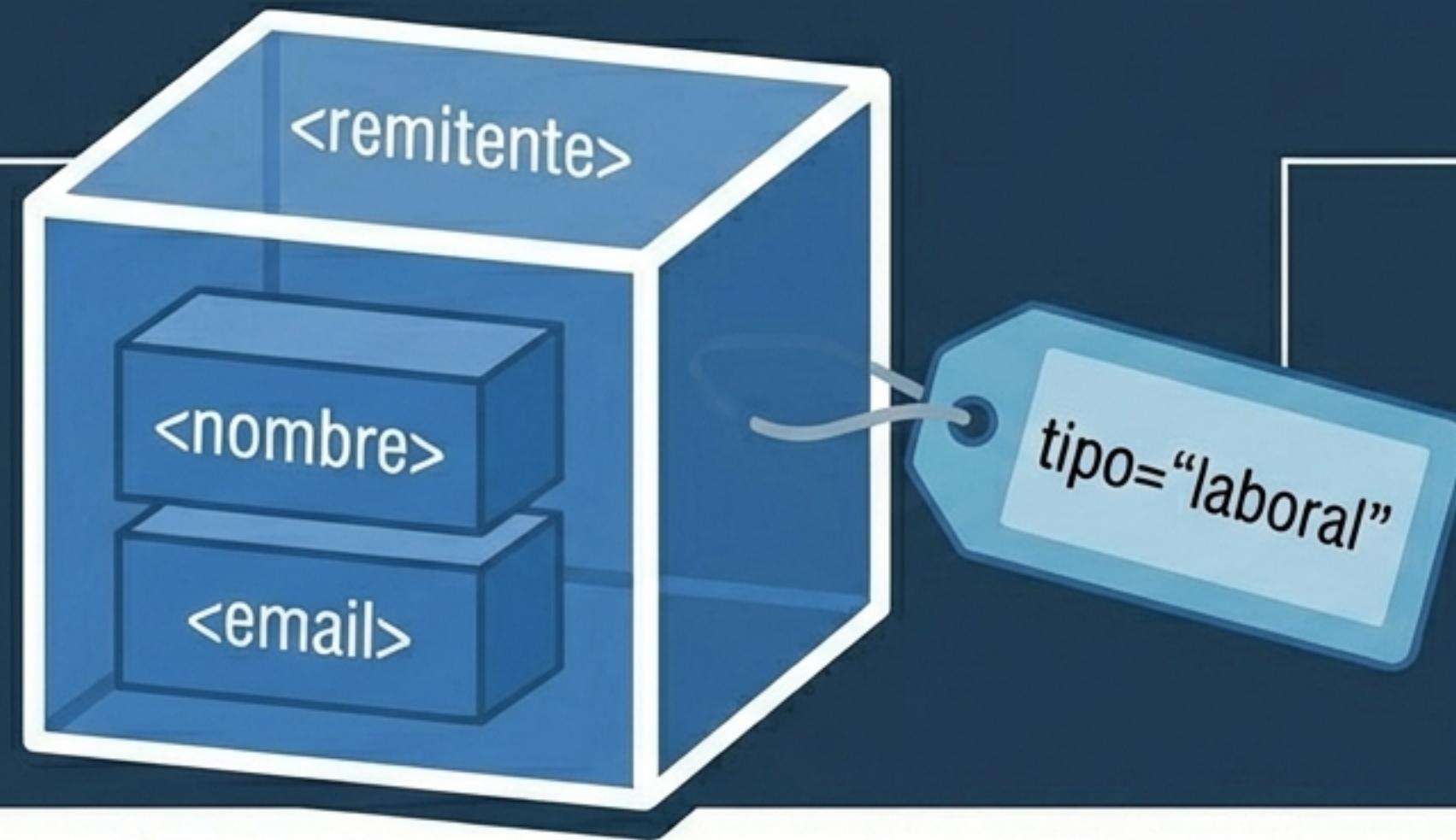
Cuerpo: Estructura jerárquica (Árbol).

Regla de Oro: Bien Formado

1. Etiquetas cerradas
2. Sensible a mayúsculas (Case Sensitive)
3. Anidación correcta (No cruzar etiquetas) (Case Sensitive)
3. Anidación correcta (No cruzar etiquetas)

Ladrillos y Mortero: Elementos vs. Atributos

Elemento: Unidad estructural. Contiene datos complejos.



Atributo: Metadatos simples. Siempre entre comillas.

```
<correo>
  <remitente>
    <nombre>Pedro Martínez</nombre>
    <email>pmartinez@gmail.com</email>
  </remitente>
  <!-- Atributos añaden metadatos sin romper la estructura --&gt;
&lt;/correo&gt;</pre>
```

Decoración de Interiores: Visualización con CSS

```
<?xml version="1.0" encoding="utf-8"?>
<?xmlstylesheet href="meteo.css"
type="text/css"?>
<temperatura>
  <lugar>
    <poblacion>Málaga</poblacion>
    <fecha>26/06/2020</fecha>
    <maxima>Máxima: 24 °C</maxima>
    <minima>Mínima: 16 °C</minima>
  </lugar>
  <!-- ... other content ... -->
</temperatura>
```

Datos Puros (XML)



Weather Card

NORTA: A diferencia de HTML, en XML el CSS debe definir `display: block` para cada elemento personalizado.

El Código de Edificación: La Validación

BIEN FORMADO



Cumple la sintaxis XML básica.
El navegador lo puede leer.

VÁLIDO



Cumple una reglas de negocio externas
(DTD). La empresa lo acepta.

Un documento válido siempre está bien formado,
pero un documento bien formado no siempre es válido.



DTD (Document Type Definition)
- Estructura básica.



XSD (XML Schema)
- Tipado avanzado.

Ubicación del DTD: <!DOCTYPE>

Elemento Raíz
definido.

Tipo de ubicación.

 Ruta del fichero
de reglas.

```
<!DOCTYPE biblioteca SYSTEM "biblio.dtd">
```

Tipo	Descripción	Ejemplo
Interno	Reglas dentro del mismo XML.	standalone="yes"
Externo Privado	Fichero local de la organización.	 SYSTEM "fichero.dtd"
Externo Público	Estándar compartido en la web.	 PUBLIC "identificador" "url"

Redactando las Reglas: Definición de Elementos

Declaración de Elemento

```
<!ELEMENT nombre_elemento (modelo_contenido)>
```

Nombre del Elemento

Regla de Contenido

```
<ELEMENT biblioteca (libro)+>
<ELEMENT libro (nombre, autor+, editorial?, paginas, (precio|precio_con_iva))>

<ELEMENT nombre (#PCDATA)>
<ELEMENT autor (#PCDATA)>
<ELEMENT editorial (#PCDATA)>
<ELEMENT paginas (#PCDATA)>
<ELEMENT precio(#PCDATA)>
<ELEMENT precio_con_iva (#PC
```

#PCDATA

```
<!ELEMENT titulo (#PCDATA)>
```

Parsed Character Data (Texto plano).

EMPTY

```
<!ELEMENT img EMPTY>
```

Elemento vacío (solo atributos).

Hijos

```
<!ELEMENT biblioteca (libro)>
```

Contiene otros elementos anidados.

Reglas de Juego: Cardinalidad y Secuencia

Cardinalidad (Repetición)

?

0 o 1 (Opcional)

→ Ej: *¿Segundo apellido?*

*

0 a N (Opcional repetible)

→ Ej: *¿Lista de alumnos?*

+

1 a N (Obligatorio repetible)

→ Ej: *¿Autores del libro?*

Secuencia (Orden)

,

Secuencia estricta
(A seguido de B).

|

Selección (A o B, no ambos).

```
<!ELEMENT libro (titulo, autor+, editorial?, (precio|precio_iva))>
```

Un libro TIENE QUE tener título, AL MENOS un autor, PUEDE tener editorial,
y OBLIGATORIAMENTE un precio (con o sin IVA).

Control de Calidad: Definición de Atributos

The diagram illustrates the structure of an XML attribute list definition and its application. At the top, a clipboard icon contains a table with four columns: 'Nombre del Elemento', 'Nombre del Atributo', 'Tipo de Dato', and 'Regla de Comportamiento'. Below the table is a code snippet: '<!ATTLIST elemento atributo tipo comportamiento>'. Blue arrows point from each column header to the corresponding part of the code. To the right, a blue box contains an example: '<!ATTLIST empleado sexo (v|h) #REQUIRED sueldo CDATA #IMPLIED>'. An arrow points from this example to a statement below it: 'El sexo es obligatorio (v o h), el sueldo es opcional.'.

	Nombre del Elemento	Nombre del Atributo	Tipo de Dato	Regla de Comportamiento
<!ATTLIST elemento atributo tipo comportamiento>				
Detalles de Atributos				
1	Tipos de Datos	<input checked="" type="checkbox"/> CDATA : Texto libre. <input checked="" type="checkbox"/> ID : Identificador único. <input checked="" type="checkbox"/> (a b c) : Lista de opciones permitidas.		
2	Comportamiento	<input checked="" type="checkbox"/> #REQUIRED : Obligatorio (Falla si no está). <input checked="" type="checkbox"/> #IMPLIED : Opcional. <input checked="" type="checkbox"/> #FIXED : Valor fijo inamovible.		

```
<!ATTLIST empleado  
sexo (v|h) #REQUIRED  
sueldo CDATA #IMPLIED>
```

El sexo es obligatorio (v o h), el sueldo es opcional.

Atajos y Constantes: Entidades

&asignaturas;

DTD: <!ENTITY asignaturas '6'>

6

Code Snippet

XML: <contenido>El curso tiene **&asignaturas;** asignaturas</contenido>

Resultado: El curso tiene **6** asignaturas

Entidades Predefinidas:

< (**<**) | **>** (**>**) | **&** (**&**) | **"** (**"**) | **'** (**'**)

Caso Práctico: Estructura de "Cursos"

El Plano (DTD)

```
<!ELEMENT curso (profesor+, alumno*, contenido)>
<!ATTLIST curso codigo CDATA #REQUIRED>

<!ELEMENT profesor (profesor+, profesor)>
<!ATTLIST alumno (profesor+, alumno)>
<!ELEMENT contenido (codigo, contenido)>

<!ELEMENT dni #dni>
<!ELEMENT nombre nombre>
<!ATTLIST ap1 #tring>
<!ATTLIST ap2 #tring>
<!ELEMENT descripcion descripcion>
<!ELEMENT horas #horas>
<!ATTLIST creditos #tring>
```

El Edificio (XML)

```
<curso codigo="LM">
  <profesor>...</profesor>
  ...
  <alumno>...</alumno>
  <contenido>...</contenido>
</curso>
```

Las Limitaciones del DTD

1. Sintaxis No-XML

DTD tiene su propio lenguaje extraño, no usa etiquetas XML.

2. Tipado Débil

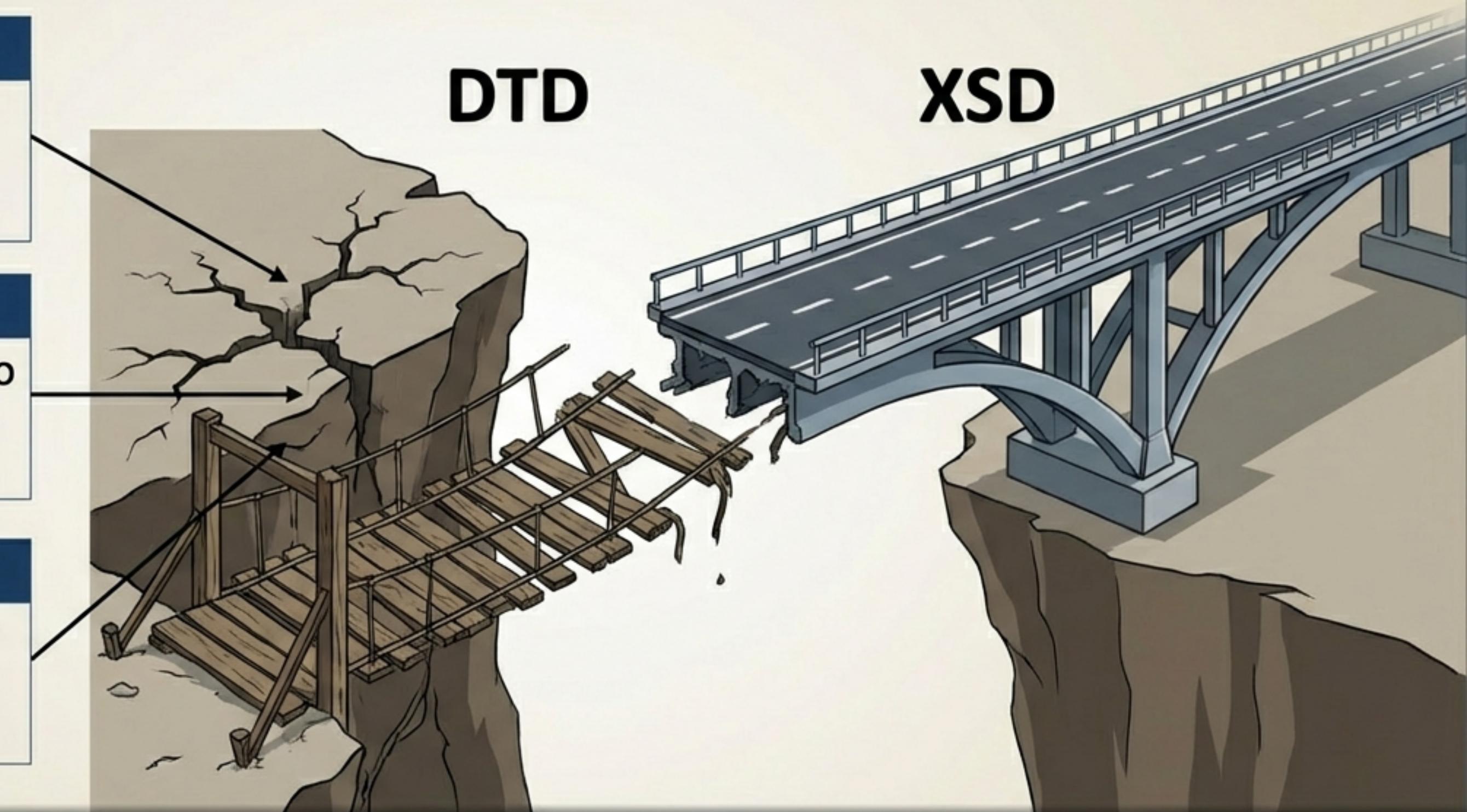
Todo es texto (PCDATA). No distingue entre números, fechas o booleanos.

3. Cardinalidad Simple

Solo permite ?, *, +. No se puede definir “mínimo 2, máximo 5”.

DTD

XSD



Estas limitaciones impulsaron la creación de **XML Schema (XSD)**.

Resolución: El inventario de Software

El Plano (DTD)

```
<!ELEMENT software (programa)+>
<!ELEMENT programa (titulo, (español|ingles),
precio, categoria, descripcion,
departamentos, web?, caratula?)>
<!ELEMENT titulo (#PCDATA)>
<!ELEMENT ingles EMPTY>
<!ELEMENT español EMPTY>

<!ATTLIST programa codigo CDATA #REQUIRED
version CDATA #REQUIRED
año CDATA #IMPLIED
sistema_operativo (windows|mac) "windows">
```

Elemento #PCDATA Selección de elementos

Atributo #REQUIRED

El Edificio (XML)

```
<software>
<programa codigo="p1"
version="15.0" año="2015"
sistema_operativo="mac">
<titulo>Adobe Photoshop</titulo>
<ingles/>
<precio>120€</precio>
...
</programa>
</software>
```

VALIDATED

“Una buena estructura hoy evita el caos de datos mañana.”