Unidad 7.
Conversión y Adaptación de Documentos XML

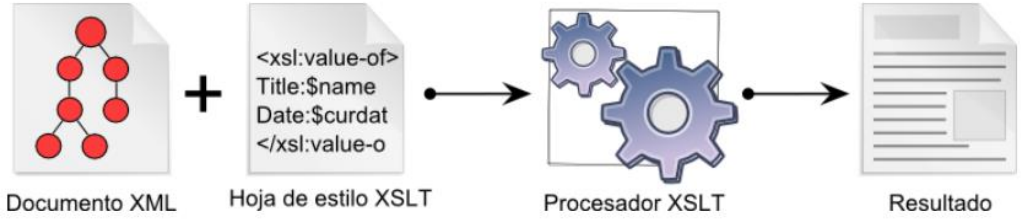
Objetivos

- Identificar la necesidad de la conversión de documentos XML.
- Analizar las tecnologías implicadas y su modo de funcionamiento.
- Conocer el funcionamiento de las transformaciones XSLT
- Identificar y caracterizar herramientas específicas relacionadas con la conversión de documentos XML.
- Realizar conversiones con distintos formatos de salida.

6.1 La familia de lenguajes de hojas de estilo extensibles: XSL (eXtensible Stylesheet Language)

XSL es un estándar aprobado por el W3C, que tiene como objetivo proporcionar herramientas para transformar los documentos XML.

La hoja de estilo XSLT es el documento que contiene el código fuente del programa, es decir, las reglas de transformación que se van a aplicar al documento inicial.



¿Una hoja XSLT es también un documento XML? Sí, XSLT es uno de los lenguajes derivados de XML, por tanto las hojas XSLT también son documentos XML (al igual que sucede con los canales RSS o los documentos XSD).

¿Qué transformaciones podemos realizar sobre un documento XML usando XSLT?

Se puede transformar en cualquier cosa que se pueda representar mediante cadenas de caracteres:

- A otro documento XML.
- A un documento HTML.
- A un documento de texto

La familia XSL está formada por tres elementos:

- XSLT: el lenguaje de transformación propiamente dicho
- XSL-FO: lenguaje en el que se indica el formato que debe tener un documento XML para representarse en diferentes dispositivos o medios.
- XPath: lenguaje que permite recorrer un documento XML utilizando expresiones. En XSLT se utiliza para seleccionar las diferentes partes de los documentos

6.2 XPath (XML Path Language)

XPath es un lenguaje para seleccionar nodos o conjuntos de nodos de un documento.

Se utiliza como soporte para otras tecnología o lenguajes anfitriones.

En sus últimas versiones permite navegar tanto por documentos XML como JSON.

Su sintaxis tiene una notación similar al establecimiento de rutas en sistemas de archivos o URL.

Podemos realizar selecciones con XPath:

- Con lenguajes de programación como Java o JavaScript
- Con herramientas, ejemplo BaseX
- Formando parte de transformaciones XSLT, etc.

XPath maneja los documentos XML como una estructura en forma de árbol.

El modelo de datos Xpath distingue siete **tipos de nodos** con diferentes funciones:

- Nodo de documento (Document):antes denominado “nodo raíz” . Representa al documento compelto
- Nodo de elemento (element): Representa los elementos XML
- Nodo atributo (attribute): Representa los atributos del documento XML
- Nodo de texto (text): Encapsula el contenido textual del documento XML
- Nodo de espacio de nombres (namespace): Representa el enlace de un URI de espacio de nombres a un prefijo de espacio de nombres o al espacio de nombres predeterminado
- Nodo de instrucción de procesamiento (processing instruction): Encapsula instrucciones de procesamiento
- Nodo de comentario (comment)

El concepto básico de XPath es **la expresión**.

Las **categorías de expresiones** son:

- Expresiones de ruta de localización
- Expresiones literales
- Expresiones condicionales
- Expresiones lógicas
- Expresiones de secuencias
- Expresiones “for”
- Expresiones cuantificadas
- Expresiones de comparación
- Expresiones aritméticas
- Expresiones de concatenación de cadenas
- Expresiones de asignación
- Expresiones en secuencias de tipo

La búsqueda de nodos se realiza con ayuda de las denominadas **rutas de localización**. Se usan para navegar a través del árbol y seleccionar el conjunto deseado de nodos.

Estas rutas se analizan de izquierda a derecha y pueden ser **absolutas o relativas**.

La expresión de ruta consta de pasos separados por barras diagonales (/), de forma similar al direccionamiento de ficheros en un sistema de registro.

Cada paso de búsqueda se puede a su vez dividir en tres partes:

- **Eje(axis):** determina la dirección de la navegación en la estructura de árbol a partir de los nodos de contexto o los nodos de documento.
- **Prueba de nodo (node test):** es un filtro con el que se delimita a un conjunto de nodos entre todos los que se sitúan en el eje.
- **Predicados (predicates):** Mecanismo para hacer un filtro sobre los nodos proporcionados por el eje y la prueba de nodo.

La sintaxis es: ***axis::node_test[predicate]***

Eje en XPath

El eje (axis) nos permite seleccionar un subconjunto de nodos del documento.

Los **nodos elemento** se indican mediante el nombre del elemento.

Los **nodos atributo** se indican mediante @ y el nombre del atributo.

/ si está al principio indica el nodo raíz, si no, indica "hijo".

// indica "descendiente" (hijos, hijos de hijos, etc.).

.. selecciona el elemento padre.

| permite elegir varios recorridos.

Predicados XPath

Un predicado (predicate) permite restringir la selección realizada por el nodo y limitarla a los elementos que cumplan determinadas condiciones.

Los predicados se escriben entre corchetes

- **[@atributo]** selecciona los elementos que tienen el atributo. Se puede invertir, es decir los que no tengan el elemento, con **not()**

  (Ejemplo: que no tenga el atributo id *[not(@id)]*)

- **[número]** si hay varios resultados selecciona uno de ellos por número de orden. **last()** selecciona el último de ellos.
- **[condicion]** selecciona los nodos que cumplen la condición. La condición puede utilizar el valor de un atributo (utilizando @) o el texto que contiene el elemento usando **text()**.

Condiciones XPath

En las condiciones se pueden utilizar los siguientes operadores:

- operadores lógicos: **and, or, not()**
- operadores aritméticos: **+, -, *, div, mod**
- operadores de comparación: **=, !=, <, >, <=, >=**

Se pueden escribir varios predicados seguidos, teniendo en cuenta que cada uno restringe los resultados del anterior, como si estuvieran encadenados por la operación lógica and.

Selección de Nodos XPath

La selección de nodos se escribe a continuación del eje y el predicado.

Si el eje y el predicado han seleccionado unos nodos, la selección de nodos indica con qué parte de esos nodos nos quedamos.

- **node()** selecciona todos los nodos (elementos y texto).
- **text()** selecciona el contenido del elemento (texto).
- ***** selecciona todos los elementos.
- **@*** selecciona todos los atributos.

Expresiones XPath Anidadas

Las expresiones XPath pueden anidarse, lo que permite definir expresiones más complicadas.

Ejemplo: Obtener los títulos de los libros publicados el mismo año que la novela "Ubik".

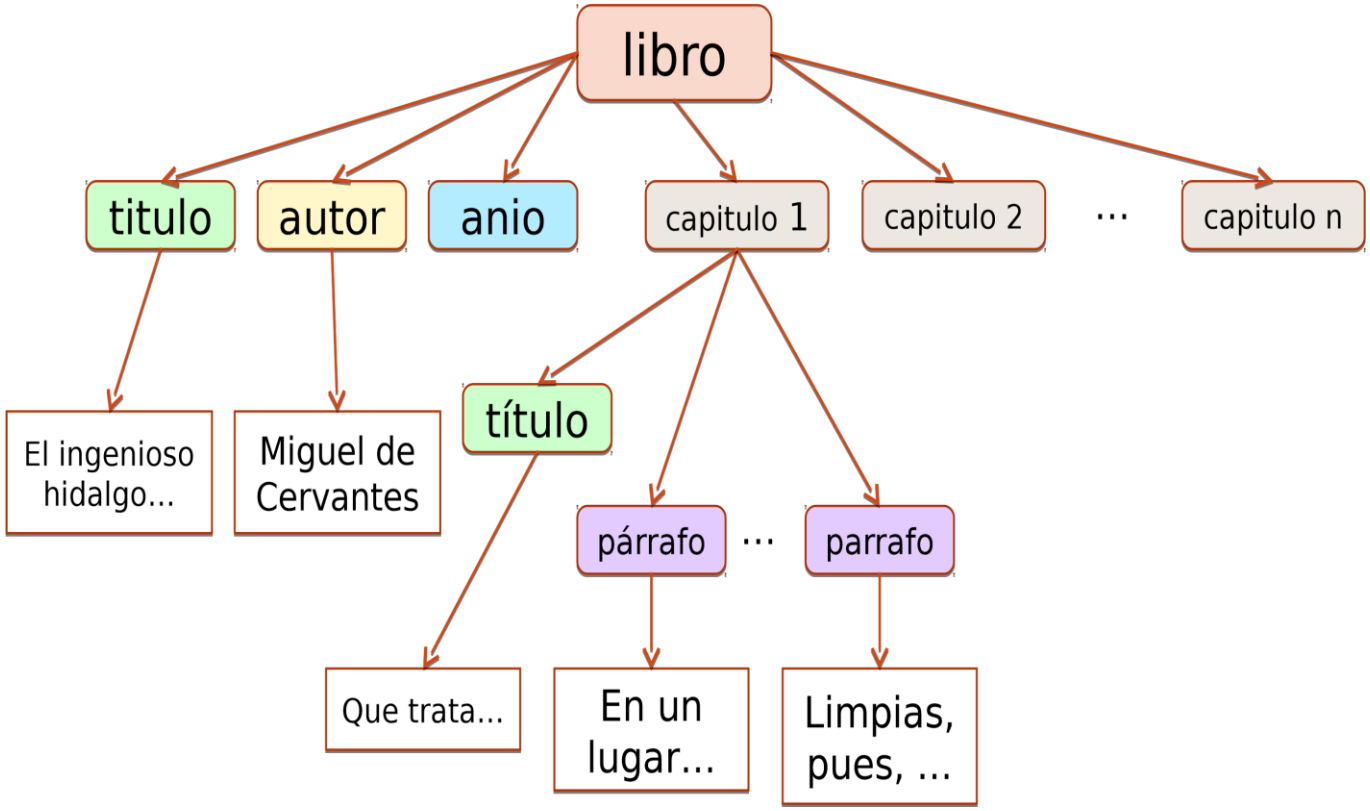
Esta información no está directamente almacenada en el documento, pero se puede obtener la respuesta en dos pasos:

1. Obtener el año en que se publicó la novela "Ubik", que devuelve 1969:
   /libros/libro[titulo="Ubik"]/fechaPublicacion/@año
2. Obtener los títulos de los libros publicados en 1969:
   /libros/libro[fechaPublicacion/@año=1969]/titulo
3. Unimos ambas expresiones

   *libros/libro[fechaPublicacion/@año=***/libros/libro[titulo="Ubik"]/fechaPublicacion/@año***]/titulo*

EJEMPLO:

Supongamos la siguiente estructura para un documento XML:



Elementos XPath

- Nodo raíz: **/**
- Camino a un elemento: **/libro/capitulo/titulo**
- Atributos

  /libro/capitulo[@number="2"]/titulo

  Selecciona el título del segundo capítulo del libro

- **/libro/capitulo/***

  Selecciona todos los elementos del capítulo

- Todos los descendientes (no necesariamente, los directos)

  /libro//titulo

  Descendientes título de libro

- Elementos en cierta posición

  //capitulo/parrafo[2]

  Segundo párrafo de los capítulos

- Recuperar el texto

  /libro/capitulo/titulo/text()

  Texto del elemento título del capítulo del libro

- Condiciones en Elementos

  /libro/capitulo[titulo="Que trata de..."]/parrafo

  Párrafos del capítulo cuyo título es "Que trata de..."

6.3 XSLT (eXtensible Stylesheet Language)

Con XSLT se puede transformar un documento XML en un documento equivalente expresado en otro formato, normalmente HTML.

El proceso de transformación consiste en seleccionar diferentes partes del documento XML de entrada e indicar en qué se quieren transformar. La selección se realiza mediante XPath y la transformación mediante XSLT. Las transformaciones XSLT se almacenan en ficheros con extensión **.xsl**

Los componentes que forman parte de una transformación son los siguientes:

- Un documento XML de entrada
- Un fichero de transformación XSLT
- Un procesador con capacidad para aplicar las transformaciones (un editor o un navegador web)

Para indicar la hoja XSLT asociada a un documento XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="ejemplo1.xsl" type="text/xsl"?>
```

La estructura del documento XSL debe incluir la declaración del espacio de nombres de XSL y, habitualmente, el elemento **<template match="/">,** que indica que se debe aplicar la transformación a todo el documento XML.

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
...
</xsl:template>
</xsl:stylesheet>
```

En un documento XSL se alterna texto libre con elementos XSL. En general, podemos encontrar:

Elementos XSLT, están precedidos del prefijo xsl:, pertenecen al espacio de nombres xsl, están definidos en el estándar del lenguaje y son interpretados por cualquier procesador XSLT.

Elementos LRE, no pertenecen a XSLT, sino que se repiten en la salida sin más.

Elementos de extensión, al igual que los anteriores, no pertenecen al espacio de nombres xsl. Son manejados por implementaciones concretas del procesador. Estos normalmente no son usados.

Elementos XSLT

El elemento raíz de una hoja XSLT es **xsl:stylesheet** o **xls:transform**, que son prácticamente equivalentes.

xmlns:xsl se utiliza para declarar el espacio de nombres http://www.w3.org/1999/XSL/Transform

Los elementos más destacados son:

xsl:attribute	añade un atributo a un elemento en el árbol de resultados.
xsl:choose	permite decidir qué parte de la hoja XSL se va a procesar en función de varios resultados.
xsl:decimal-format	define un patrón que permite convertir en cadenas de texto números en coma flotante.
xsl:for-each,	aplican sentencias a cada uno de los nodos del árbol que recibe como argumento.
xsl:if	permite decidir si se va a procesar o no una parte del documento XSL en función de una condición
xsl:import	importa una hoja de estilos XSLT localizada en una URI dada.

xsl:key	define una o varias claves que pueden ser referenciadas desde cualquier lugar del documento.
xsl:output	define el tipo de salida que se generará como resultado.
xsl:preserve-space	especifica cuales son los elementos del documento XML que no tienen espacios en blanco eliminados antes de la transformación.
xsl:sort	permite aplicar un template a una serie de nodos ordenándolos alfabético numéricamente.
xsl:strip-space	especifica cuales son los elementos del documento XML que tienen espacios en blanco eliminados antes de la transformación.
xsl:value-of	calcula el valor de una expresión XPath dada y lo inserta en el árbol de resultados del documento de salida.
xsl:variable	asigna un valor a una etiqueta para usarlo cómodamente.
xsl:template	permite establecer una plantilla y determinar sobre qué conjunto de elementos se va a realizar la transformación. Es el bloque fundamental de una hoja XSLT, por lo que veremos su descripción en el apartado siguiente.

Puedes consultar la lista completa en : https://developer.mozilla.org/es/docs/Web/XSLT/Element

Elementos XSLT

El elemento **xsl:template** permite controlar el formato de salida que se aplica a ciertos datos de entrada. Dicho formato se especifica utilizando sentencias XHTML

Tiene un atributo denominado **match**, que se utiliza para seleccionar los nodos del árbol de entrada.

```
<xsl:template match=”propietario”>
<p>Agenda de </p>
</xsl:template>
```

En una plantilla podemos tener más de una regla que afecte a distintos elementos.

Por defecto ese orden es el que el intérprete utiliza al leer el documento XML, es decir, de arriba abajo, aunque puede ser modificado en la plantilla, para ello utilizaremos el elemento **xsl:apply-templates**.

```
<xsl:template match=”contactos”>
<xsl:apply-templates select=”identificadores”/>
</xsl:template>
```

6.4 XSLT-FO

Permite describir el modo de mostrar un conjunto de datos en un soporte determinado, ya sea papel, la pantalla u otros medios alternativos.

En los objetos XSL-FO se definen dimensiones, márgenes, párrafos, listas, tablas, etc.

Con XSL-FO se pueden crear documentos PS (PostScript), RTF, PNG, PCL o PDF.

Los ficheros tienen la extensión **.fo** y . **xml**

Para su uso es necesario disponer de un programa que sea capaz de procesar XSL-FO. La más utilizada es Apache FOP

Nota: XSLT y XSL-FO forman parte de la misma familia de lenguajes y se complementan, pero no son los mismo. XSLT es un lenguaje relativo a las transformaciones y XSL-FO es un lenguaje relativo al formato.

EJEMPLO XSL-FO

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Creación de una tarjeta de presentación en PDF -->
<fo:root xmlns="http://www.w3.org/1999/XSL/Format">
    <fo:layout-master-set>
        <!--Tamaño y configuración de la página -->
        <fo:simple-page-master master-name="tarjeta-presentacion" page-height="5.5cm"
        page-width="8.5cm" margin-top="2.2cm" margin-bottom="0,5cm">
            <fo:region-body/>
            <fo:region-before extent="0.5cm"/>
            <fo:region-after extent="0.5cm"/>
        </fo:simple-page-master>
    </fo:layout-master-set>
    <!--Página -->
    <fo:page-sequence master-reference="tarjeta-presetacion">
        <fo:flow flow-name="xsl-region-body">
            <!--Nombre de la persona. Tamaño de la fuente 16 puntos -->
            <fo:block font-size="16pt" text-align="center">
                Ana Herrero
            </fo:block>
            <!--Profesión de la persona. Tamaño de la fuente 8 puntos Color gris -->
            <fo:block font-size="8pt" text-align="center" color="#888888">
                Analista
            </fo:block>
        </fo:flow>
    </fo:page-sequence>
</fo:root>
```