Conversión y adaptación de documentos XML. XPATH.

Los documentos XML son documentos de texto con etiquetas que contienen exclusivamente información sin entrar en detalles de formato. Esto implica la necesidad de algún medio para expresar la transformación de un documento XML, para que una persona pueda utilizar directamente los datos para leer, imprimir, etc.

Las tecnologías que entran en juego en la transformación de documentos son:

- **XSLT**: permite definir el modo de transformar un documento XML en otro.
- **XSL-FO**: Se utiliza para transformar XML en un formato legible e imprimible por una persona, por ejemplo en un documento PDF.
- **Xpath**: permite el acceso a los diversos componentes de un documento XML

La parte de transformaciones ganó en importancia, y se llega a la terminología actual que es XSL y comprende las anteriores.

Los documentos XML se pueden transformar en otro tipo de documento, por ejemplo, una página web XHTML. Para realizar esta tarea necesitamos indicar que datos son los que queremos incluir en la página Web.

Ejemplo:

El siguiente documento XML:

```
<?xml version="1.0" encoding="utf-8"?>
<catalogo>
	<libro>
		<titulo>Manual imprescindible de C/C++</titulo>
		<isbn>9788441526143</isbn>
		<autores>
			<autor>Miguel Angel</autor>
		</autores>
		<paginas>416</paginas>
		<editorial>Anaya Multimedia</editorial>
	</libro>
	<libro>
		<titulo>Redes locales</titulo>
		<isbn>9788441519800</isbn>
		<autores>
			<autor>Jim Doherty</autor>
			<autor>Neil Anderson</autor>
		</autores>
		<paginas>544</paginas>
		<editorial>Rama</editorial>
	</libro>
</catalogo>
```

Podemos transformarlo en el siguiente documento XHTML, que muestre la lista de títulos de los libros:

```
<!DOCTYPE html PUBLIC "-//W3C/DTD XHTML 1.1//EN"
        "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="es">
        <head>
                <title>Libros</title>
        </head>
        <body>
                <h1>Libros</h1>
                <ul>
                        <li>Manual imprescindible de C/C++</li>
                        <li>Redes locales</li>
                </ul>
        </body>
</html>
```

En esta unidad aprenderemos como hacer esto. Pero dividiremos nuestro aprendizaje en partes:

1- Como seleccionar los datos del documento XML, mediante XPath (cuya sintaxis permite localizar y obtener elementos o datos concretos de un documento XML.

2- Como crear una plantilla del documento que queremos obtener, en el ejemplo XHTML, mediante XSL.

2. XPath

Recomendación del W3C para XPath: http://www.w3.org/TR/xpath/

Aunque a lo largo del curso hemos hablado de ello, es importante recordar el orden de los elementos en un documento XML:

Ejemplo: En el caso del xml anterior, sabemos que <catalogo> es el ejemplar, que alberga al elemento <libro>, que a su vez contiene otros elementos.

Es importante recordarlo, porque para seleccionar un elemento, debemos conocer su ruta o camino.

También es importante saber el orden en el que aparecen los elementos.

Los términos o nomenclatura que se utiliza en XPath es la siguiente:

- **Nodo raíz**, es el nodo que contiene al ejemplar del fichero XML. Se identifica por "**/**".

- **Nodos elemento**, son cada uno de los elementos del documento XML. Todos ellos tienen un elemento padre, el padre del elemento raíz, es decir del ejemplar, es el nodo raíz del documento.

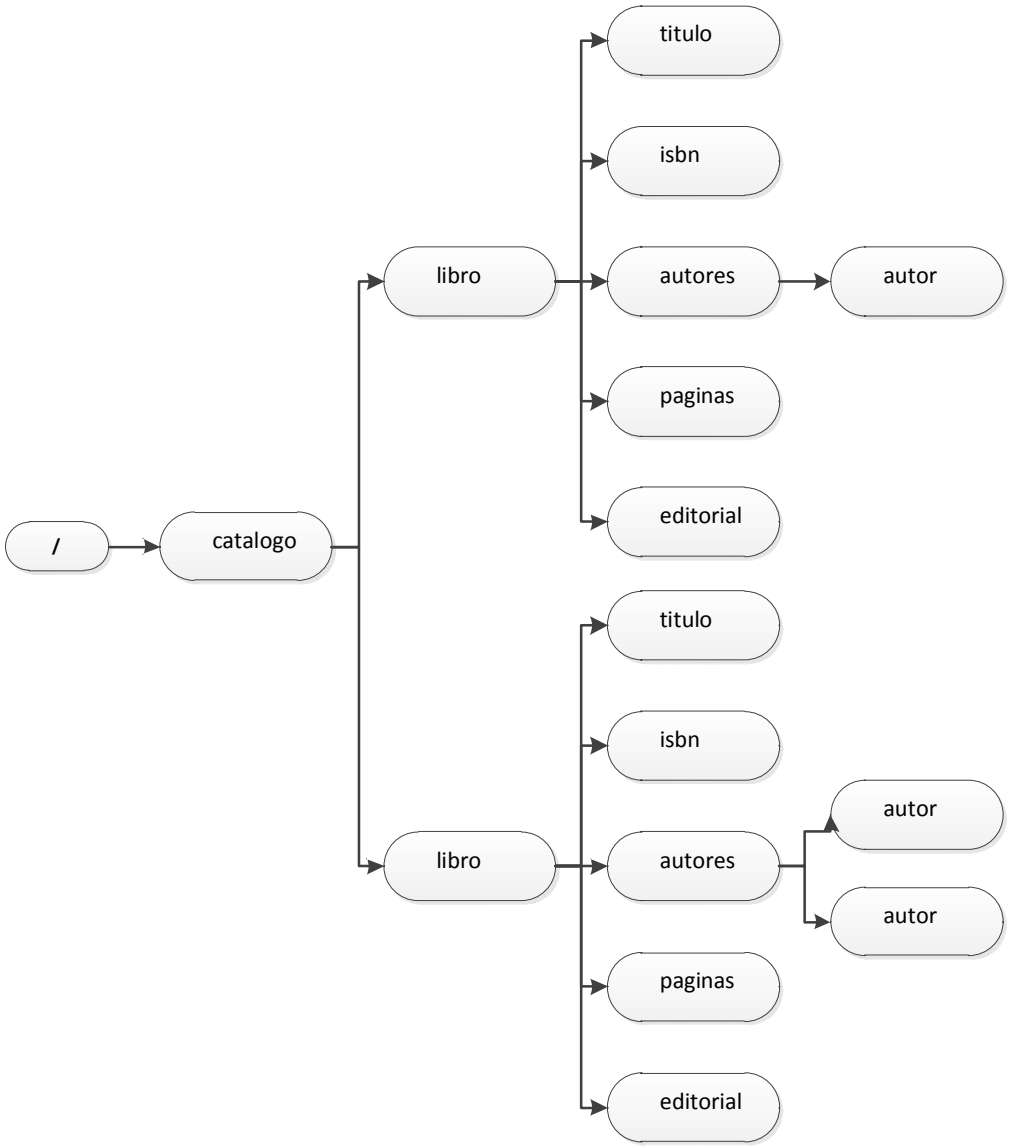
Pueden tener identificadores únicos, para ello es necesario que un atributo esté definido de ese modo en un DTD o SCHEMA asociado, que permite referenciar dicho elemento de forma mucho más directa.

- **Nodos texto**, son aquellos caracteres del documento que no están marcados con ninguna etiqueta. Este tipo de nodos no tienen hijos.
- **Nodos atributo**, no se consideran hijos del elemento al que están asociados sino etiquetas añadidas al nodo elemento.
    - Aquellos atributos que tengan un valor asignado en el esquema asociado, se tratarán como si ese valor se le hubiese dado al escribir el documento XML.
    - Para las definiciones de los espacios de nombre y para aquellos atributos que se han definido con la propiedad **#IMPLIED** en su **DTD** no se crean nodos.
- **Nodos de comentario y de instrucciones de proceso**, son los nodos que se generan para los elementos con comentarios e instrucciones de proceso.
- **Nodo actual**, es aquél al que nos referimos cuando se evalúa una expresión en Xpath.
- **Nodo contexto**, cada expresión está formada por sub-expresiones que se van evaluando antes de resolver la siguiente. El conjunto de nodos obtenido tras evaluar una expresión y que se utiliza para evaluar la siguiente es el nuevo contexto.
- **Tamaño del contexto**, es el número de nodos que se están evaluando en un momento dado en una expresión Xpath.

2.1. Raíz y nodos.

Vamos a tomar como referencia el ejemplo del principio. Sabiendo cuál es su estructura, podemos definirla gráficamente.

Raíz: / (no confundir con el ejemplar).



Elemento raíz: **/**

Ejemplar**: /catalogo**.

Para referirme a los siguientes elementos, utilizaremos la ruta, por ejemplo:

Libro: /catalogo/libro

Titulo: /catalogo/libro/titulo

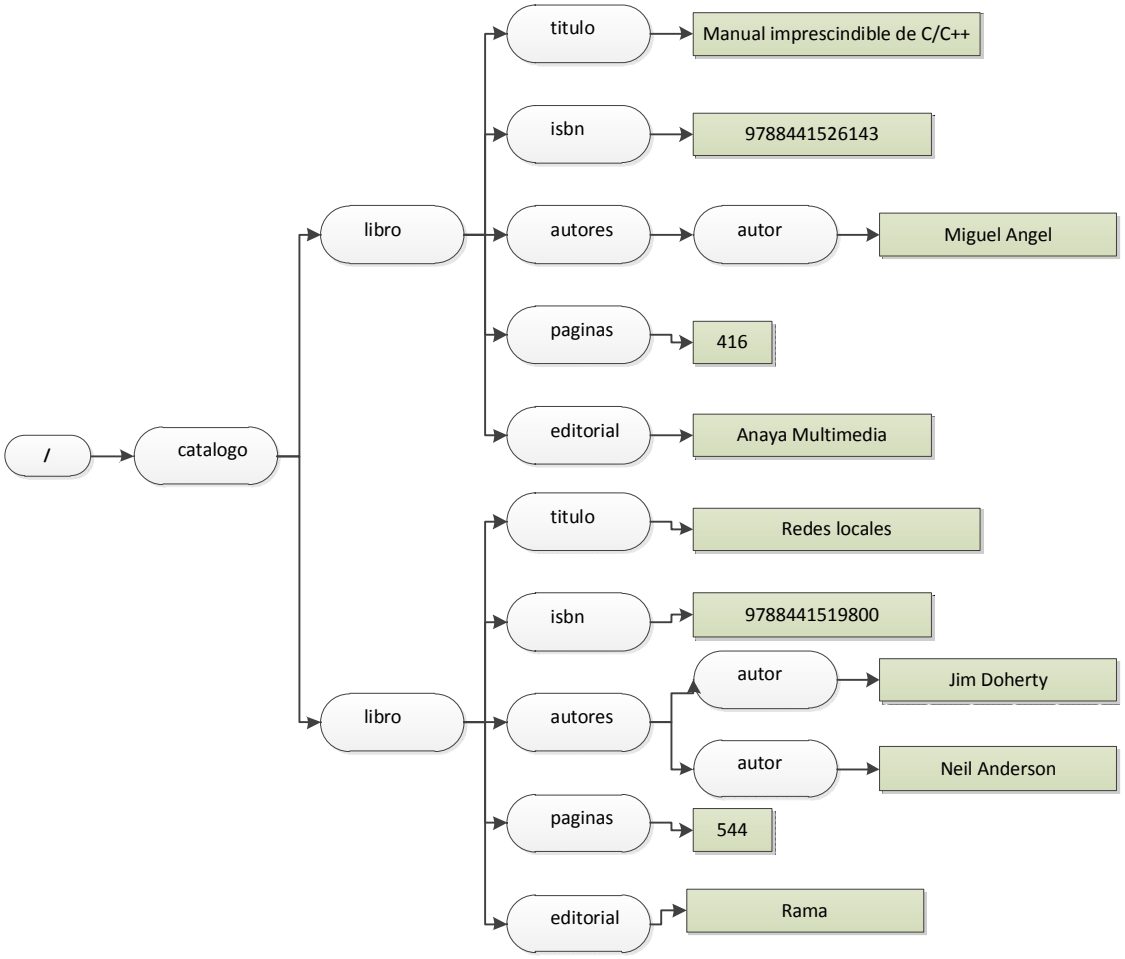
Isbn: /catalogo/libro/isbn

Autores: /catalogo/libro/autores

Editorial: /catalogo/libro/editorial

Autor: /catalogo/libro/autores/autor

Para completar la representación gráfica, nos falta incluir los datos o valores:



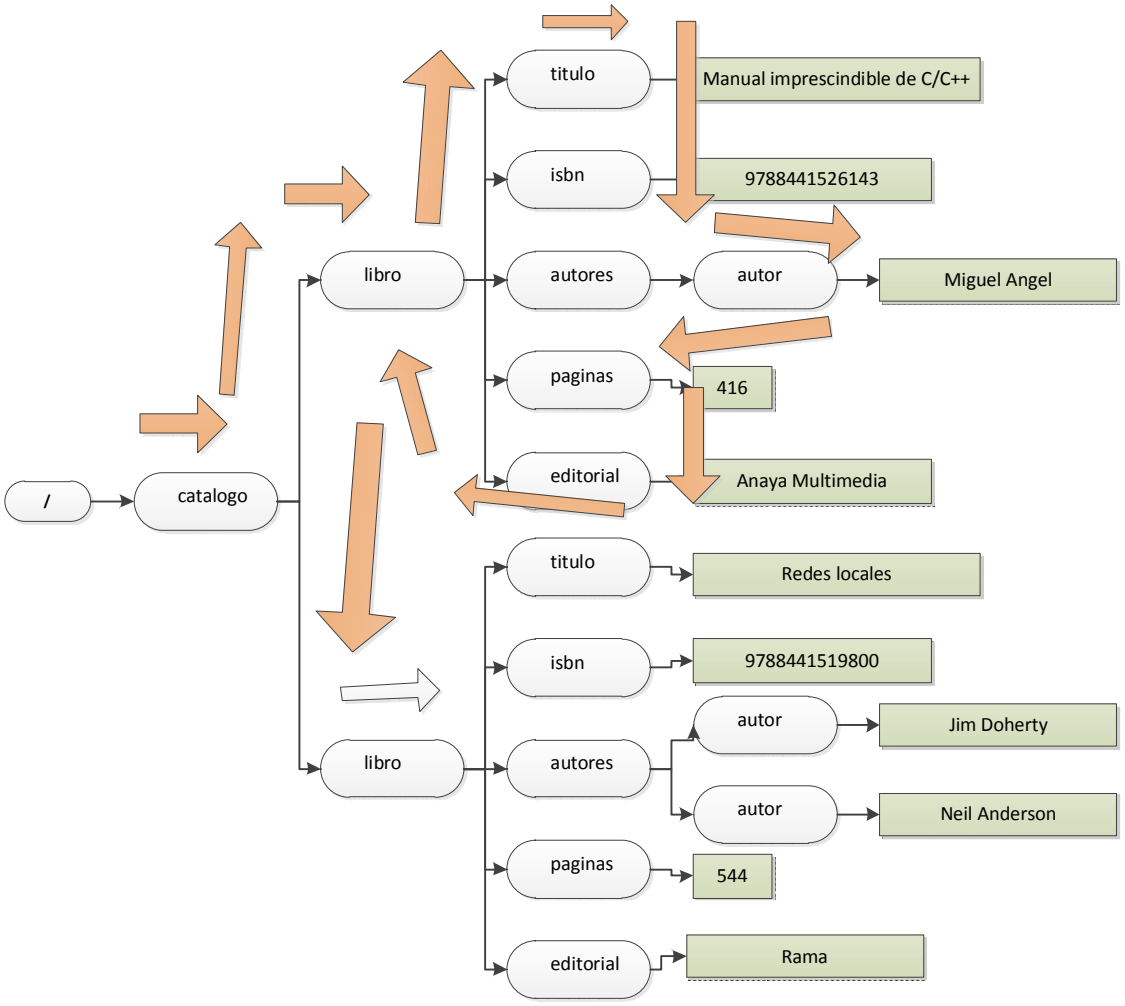
Nodo:

Cada elemento del árbol es un nodo, en nuestro ejemplo, son nodos:

Catalogo-libro-titulo-isbn-autores-autor-paginas-editorial

Orden:

A la hora de conocer el orden, es decir cómo se recorre el documento XML, tenemos que fijarnos en el árbol. Comenzar por el elemento raíz y recorrer el árbol en el sentido de las agujas del reloj:



2.2. Expresiones y funciones XPath

¿Qué obtenemos al evaluar una expresión XPath?

- Un conjunto de nodos.Los nodos pueden ser de 7 tipos diferentes:
    - Elemento.
    - Atributo.
    - Texto.
    - Espacio de nombres.
    - Instrucción de procesamiento.
    - Comentario.
    - Raíz.
- **Booleano.**
- **Número.**
- **Cadena.**

Los tokens que podemos utilizar en una expresión XPath son los siguientes:

- Paréntesis, “()”; llaves , “{}” y corchetes, “[ ]”.

- Elemento actual, elemento padre.
- Atributo, “@”.
- Elemento, “*”.
- Separador, “::”.
- Coma, “,”.
- El nombre de un elemento.
- Tipo de nodo, que puede ser:
    - comment.
    - text.
    - procesing instruction.
    - node.
- Operadores: and, or, mod, div, *, /, //, |, +, -, =, !=, <, >, <=, >=.
- Nombres de función.
- Denominación de ejes: ancestor, ancestor-or-self-atribute, child, descendant, descendant-or-self, following, following-sibling, namespace, parent, preceding, preceding-sibling, self.
- Literales, se ponen entre comillas dobles o simples. Pueden anidarse alternando el tipo de comillas.
- Números.
- Referencias a variables, para lo que se utiliza la sintaxis: **$nombreVariable**

Ahora vamos a verlo aplicado. Para ello, vamos a cambiar de ejemplo de referencia para la explicación:

```
<?xml version="1.0" encoding="utf-8"?>
<catalogo>
	<producto>
		<calidad valor="9"/>
		<nombre>Mesa THYO</nombre>
		<referencia>3948</referencia>
		<precio>60</precio>
	</producto>
	<producto>
		<calidad valor="7"/>
		<nombre>Silla THYO</nombre>
		<referencia>5343</referencia>
```

```
            <precio>90</precio>
        </producto>
</catalogo>
```

Elementos:

Antes hemos visto como seleccionar elementos, escribiendo su ruta.

Por ejemplo, para seleccionar <nombre>, escribiríamos:

/catalogo/producto/nombre

Seleccionar todos los elementos que contiene un elemento concreto:

Si queremos seleccionar todos los elementos que están dentro de uno concreto, podemos utilizar el símbolo ***.**

Por ejemplo, si quiero seleccionar todos los elementos que contiene <producto>, lo expresaría así:

/catalogo/producto/*

Atributos:

Para seleccionar los atributos se utiliza el operador @ delante de los mismos.

Por ejemplo, si quiero seleccionar el atributo "valor" del elemento <calidad>:

/catalogo/producto/calidad/@valor

Filtros:

A la hora de seleccionar, también podemos establecer filtros, que se escriben al final de la ruta y entre corchetes.

Operadores que podemos usar: = , >= , <=, !=

Por ejemplo, si queremos seleccionar el elemento <producto> cuyo precio sea mayor o igual a 90, escribiremos:

/catalogo/producto [precio >= 90]

Con los atributos también se pueden aplicar filtros, por ejemplo, si quiero seleccionar un producto cuya calidad sea mayor que 5.

/catalogo/producto/calidad [@valor > 5]

Predicado:

En ocasiones, tendremos que seleccionar un nodo que cumple ciertas características, de entre varios nodos posibles, para conseguirlo, se usan los Predicados.

El predicado es una expresión booleana que añade un nivel de verificación al paso de localización.

En los predicados podemos incorporar funciones XPath

Los predicados se incluyen dentro de una ruta de localización utilizando los corchetes, por ejemplo:

/catalogo/producto/calidad [@valor =“1”]/nombre

En este caso se está indicando al intérprete que escoja, dentro de un fichero XML de recetas, el nombre del ingrediente cuyo código tiene el valor “1”.

Veamos que es lo que podemos incluir en un predicado.

- **Ejes**, permiten seleccionar el subárbol dentro del nodo contexto que cumple un patrón. Pueden ser o no de contenido.
    - **child**, es el eje utilizado por defecto. Su forma habitual es la barra, “/”, aunque también puede ponerse: /child::
    - **attribute**, permite seleccionar los atributos que deseemos. Es el único eje que veremos que no es de contenido.
    - **descendant**, permite seleccionar todos los nodos que descienden del conjunto de nodos contextos. Se corresponde con la doble barra, //, aunque se puede usar: descendant::
    - **self**, se refiere al nodo contexto y se corresponde con el punto ”.”.
    - **parent**, selecciona los nodos padre, para referirnos a él usamos los dos puntos, “..”.
    - **ancestor**, devuelve todos los nodos de los que el nodo contexto es descendiente.
- Nodos test, permiten restringir lo que devuelve una expresión Xpath. Podemos agruparlos en función de los ejes a los que se puede aplicar.
    - Aplicable a cualquier eje:
        - **“*”**, solo devuelve elementos, atributos o espacios de nombres pero no permite obtener nodos de texto, o comentarios de cualquier tipo.
        - **nod()**, devuelve los nodos de todos los tipos.
    - Solo aplicables a ejes de contenido:
        - **text()**, devuelve cualquier nodo de tipo texto.
        - **comment()**, devuelve cualquier nodo de tipo comentario.
        - **processing-instruction()**, devuelven cualquier tipo de instrucción de proceso.

Varios predicados pueden unirse mediante los operadores lógicos **and**, **or** o **not**.

Funciones:

Las funciones principales que podemos utilizar son las siguientes:

- **boolean()**, al aplicarla sobre un conjunto de nodos devuelve true si no es vacío.
- **not()**, al aplicarla sobre un predicado devuelve true si el predicado es falso , y falso si el predicado es true.
- **true()**, devuelve el valor true.
- **false()**, devuelve el valor false.
- **count()**, devuelve el número de nodos que forman un conjunto de nodos.
- **name()**, devuelve un nombre de un nodo.
- **local-name()**, devuelve el nombre del nodo actual o del primer nodo de un conjunto de nodos.
- **namespace-uri()**, devuelve el URI del nodo actual o del primer nodo de un conjunto dado.
- **position()**, devuelve la posición de un nodo en su contexto comenzando en 1. Por ejemplo, para seleccionar los dos primeros elementos de tipo elemento de un fichero XML pondremos: **//elemento[position()<=2**]
- **last()**, Devuelve el último elemento del conjunto dado.
- **normalize-space()**, permite normalizar los espacios de una cadena de texto, es decir, si se introduce una cadena donde hay varios espacios consecutivos, esta función lo sustituye por uno solo.
- **string()**, es una función que convierte un objeto en una cadena. Los valores numéricos se convierten en la cadena que los representa teniendo en cuenta que los positivos pierden el signo. Los valores booleanos se convierten en la cadena que representa su valor, esto es “true” o “false”.
- **concat()**, devuelve dos cadenas de texto concatenadas. El ejemplo siguiente devuelve “Xpath permite obtener datos de un fichero XML”.

  concat('XPath', 'permite obtener datos de un fichero XML')

- **string-length()**, devuelve la cantidad de caracteres que forman una cadena de caracteres.
- **sum()**, devuelve la suma de los valores numéricos de cada nodo en un conjunto de nodos determinado.

2.3. Acceso a elementos de otros documentos.

Además de acceder a los datos del fichero XML con el que se está trabajando directamente, es útil poder acceder a los datos de otros ficheros.

Para ello utilizaremos la función **document()**, pero dicha función **NO** es del lenguaje XPath, sino que pertenece a XSLT.

Esta función puede admitir dos argumentos diferentes:

document(URI)

En este caso la función devuelve el elemento raíz del documento XML que se localiza en el URI especificado.

document(nodo)

En esta ocasión lo que devuelve la función, es el conjunto de nodos cuya raíz es el nodo dado.