Insertar nodos

En primer lugar crearemos el nodo que deseamos insertar.

```
private XmlNode CrearNodoXml(string id, string nom, string ape, string nss, string fijo, string mvl)
        {
    //Creamos el nodo que deseamos insertar.
    XmlElement empleado = documento.CreateElement("empleado");

    //Creamos el elemento idEmpleado.
    XmlElement idEmpleado = documento.CreateElement("idEmpleado");
    idEmpleado.InnerText = id;
    empleado.AppendChild(idEmpleado);

    //Creamos el elemento nombre.
    XmlElement nombre = documento.CreateElement("nombre");
    nombre.InnerText = nom;
    empleado.AppendChild(nombre);

    //Creamos el elemento apellidos.
    XmlElement apellidos = documento.CreateElement("apellidos");
    apellidos.InnerText = ape;
    empleado.AppendChild(apellidos);

    //Creamos el elemento numeroSS.
    XmlElement numeroSS = documento.CreateElement("numeroSS");
    numeroSS.InnerText = nss;
    empleado.AppendChild(numeroSS);

    //Creamos el elemento telefonos.
    XmlElement telefonos = documento.CreateElement("telefonos");
    empleado.AppendChild(telefonos);

        //Creamos el elemento fijo.
        XmlElement nodoFijo = documento.CreateElement("fijo");
        nodoFijo.InnerText = fijo;
        telefonos.AppendChild(nodoFijo);
        //Creamos el elemento movil.
        XmlElement movil = documento.CreateElement("movil");
        movil.InnerText = mvl;
        telefonos.AppendChild(movil);

    return empleado;
        }
```

Una vez que tenemos el nodo creado procedemos a **insertar un registro al final del documento XML**.

```
private void InsertarXml()
{
    //Cargamos el documento XML.
    documento = new XmlDocument();
    documento.Load(ruta);

    //Creamos el nodo que deseamos insertar.
    XmlNode empleado = this.CrearNodoXml("4", "Gerardo", "Montaz Soaz", "444-444-444-
444",  "893434220",  "609334209");

    //Obtenemos el nodo raiz del documento.
    XmlNode nodoRaiz = documento.DocumentElement;

    //Insertamos el nodo empleado al final del archivo
    nodoRaiz.InsertAfter(empleado, nodoRaiz.LastChild);   //***

    documento.Save(ruta);
  }
```

Ahora nos fijamos en la función anterior justo en la linea que esta marcada con //***. Esta linea es la que inserta el nodo precisamente acontinuación del último nodo del documento XML.

Otras opciones son:

- **Insertar antes del último nodo del documento**.
  `nodoRaiz.InsertBefore(empleado, nodoRaiz.LastChild);`
- **Insertar después del primer nodo del documento**.
  `nodoRaiz.InsertAfter(empleado, nodoRaiz.FirstChild);`
- **Insertar antes del primer nodo del documento**.
  `nodoRaiz.InsertBefore(empleado, nodoRaiz.FirstChild);`
- **Insertar en un lugar determinado del documento**.
  `nodoRaiz.InsertAfter(empleado, documento.SelectNodes("empleados/empleado").Item(2));`

En este último caso hay que tener en cuenta que el método Item() accede a la colección de nodos del documento, y esta colección tiene un índice inicial igual a cero.

Modificar y eliminar Nodos

Los objetos XmlDocument, XmlElement y XmlNode disponen de dos métodos que nos permitirán realizar la modificación y la eliminación del nodo deseado.

- **ReplaceChild**(nodoNuevo, nodoViejo). Reemplaza el nodo existente por el nodo con los nuevos datos.
- **RemoveChild**(nodo). Elimina un nodo permanentemente.

Estos métodos actuan sobre los nodos secundarios del objeto que realiza la llamada. Por lo tanto, el "truco" consiste en posicionarse en el árbol del XML sobre el nodo padre del nodo que pretendemos actualizar, reemplazar o borrar.

El código que a continuación veremos hace uso del método secundario **CrearNodoXml()** correspondiente al método "Insertar datos XML". Recordar que este método devuelve un objeto XmlNode de tipo "empleado" debidamente construido.

```
public void ModificarDatosXml(string id, string nom, string ape, string nss, string fijo, string mvl)
    {
      //Cargamos el documento XML.
      documento = new XmlDocument();
      documento.Load(Ruta);

      //Obtenemos el nodo raiz del documento.
      XmlElement empleados = documento.DocumentElement;

      //Obtenemos la lista de todos los empleados.
      XmlNodeList listaEmpleados = documento.SelectNodes("empleados/empleado");

      foreach (XmlNode item in listaEmpleados)
      {
        //Determinamos el nodo a modificar por medio del id de empleado.
        if (item.FirstChild.InnerText == id)
        {
          //Nodo sustituido.
          XmlNode nodoOld = item;
          //Nodo nuevo.
          XmlNode nodoNew = CrearNodoXml(id, nom, ape, nss, fijo, mvl);
          //Remplazamos el nodo.
          empleados.ReplaceChild(nodoNew, nodoOld);
          //Borrar un nodo.
          //empleados.RemoveChild(nodoOld);
        }
  }
   documento.Save(Ruta);          //Salvamos el documento.

  }
```

MODELO DOM

XmlDocument	XmlDocument xmlDoc = new XmlDocument(); //documento vacío en memoria	
Load	xmlDoc.load(fichero) // carga el fichero.xml en memoria	
XmlNode tipo nodo CreateElement(tag) método para crear un elemento InnerText propiedad para dar valor al elemento	XmlNode Nodo = xmlDoc.CreateElement(tag) Nodo.InnerText = Valor XmlAttribute atributo = xmlDoc.CreateAttribute(tag) atributo.Value = Valor Nodo. Attributes.Append(atributo); Nodo. AppendChild(nodoAañadir)	nodo.SetAttribute(tag, valor);
XmlAttribute Tipo atributo CreateAttribute(tag) método para crear un atributo Value propiedad para dar valor al atributo		
Attributes.Append(atributo); Método para añadir un atributo a un nodo		
AppendChild(nodoAañdir) Método para añadir un nodo un nodo padre		
Insertar elementos		
InsertAfter (Inserta después del nodo referencial)	xmlDoc.DocumentElement.InsertAfter(nodo, nodoReferencial); nodoPadre.InsertAfter(nodo, nodoRef)	nodoReferencial

InsertBefore (Inserta antes del nodo referencial)	xmlDoc.DocumentElement.InsertBefore(nodo, nodoReferencial); nodoPadre.InsertBefore(nodo, nodoRef)	Ultimo→raiz.LastChild Primero→raiz.FirstChild NodoCualquiera→Requiere una búsqueda previa
RemoveChild RemoveNode	nodo.RemoveChild(node.SelectSingleNode("Genre")) xmlDocument.RemoveNode(node);	
Save Metodo que guardar todo el documento xml	xmlDoc.Save(fichero)	

Recuperar la información de la estructura XML

XmlNodeList	Declara objeto para contener una lista de nodos
XmlElement	Declara un objeto para contener un nodo

Situase en el nodo Raiz → `XmlElement root = xmlDoc.DocumentElement;`

Buscar Nodo

SelectSingleNode	`XmlNode nodo = xmlDoc.SelectSingleNode("//Producto[@Id=\"A1\"]");` Expresión XPath
GetElementsByTagName	`XmlNodeList lista = xmlDoc.GetElementsByTagName("Producto");` Devuelve una lista de nodos (todos los que tengan la misma etiqueta) Puede accederse a cualquier elemento de la lista por el índice Lista[1]→estaríamos en el segundo nodo

`GetElementsByTagNameID`	Busca por el atributo ID , necesita el DTD o Scheme que valida el documento XML

Para tener acceso a datos, normalmente, se selecciona o vaya a uno o más nodos necesarios y utilizar uno o más nodos resultante para manipular el contenido XML. La siguiente es una lista de métodos y propiedades de **System.Xml** pertinentes:

- **InnerText** y **InnerXml**. Devuelve valores concatenados de todos los nodos secundarios
- **Value**. Proporciona acceso al valor de un nodo.
- **Attributes**. Encuentra la colección de atributos del nodo actual.
- **FirstChild**, **LastChild**y **ChildNodes**. Proporcionan acceso al primer nodo, el último nodo o todo del elemento secundario en los nodos del nodo actual.
- **NextSibling** y **PreviousSibling**. Proporcionar acceso a los nodos inmediatamente delante y detrás.
- **DocumentElement**. Proporciona acceso a la raíz **XmlElement** del documento.
- **Element**. Busca el elemento secundario especificado.
- **NodeType** (de tipo **XmlNodeType**). Indica el tipo del nodo actual; Por ejemplo, atributo, elemento o texto.
- **XmlNodeType**. Enumeración contiene todos los tipos de nodo posibles.

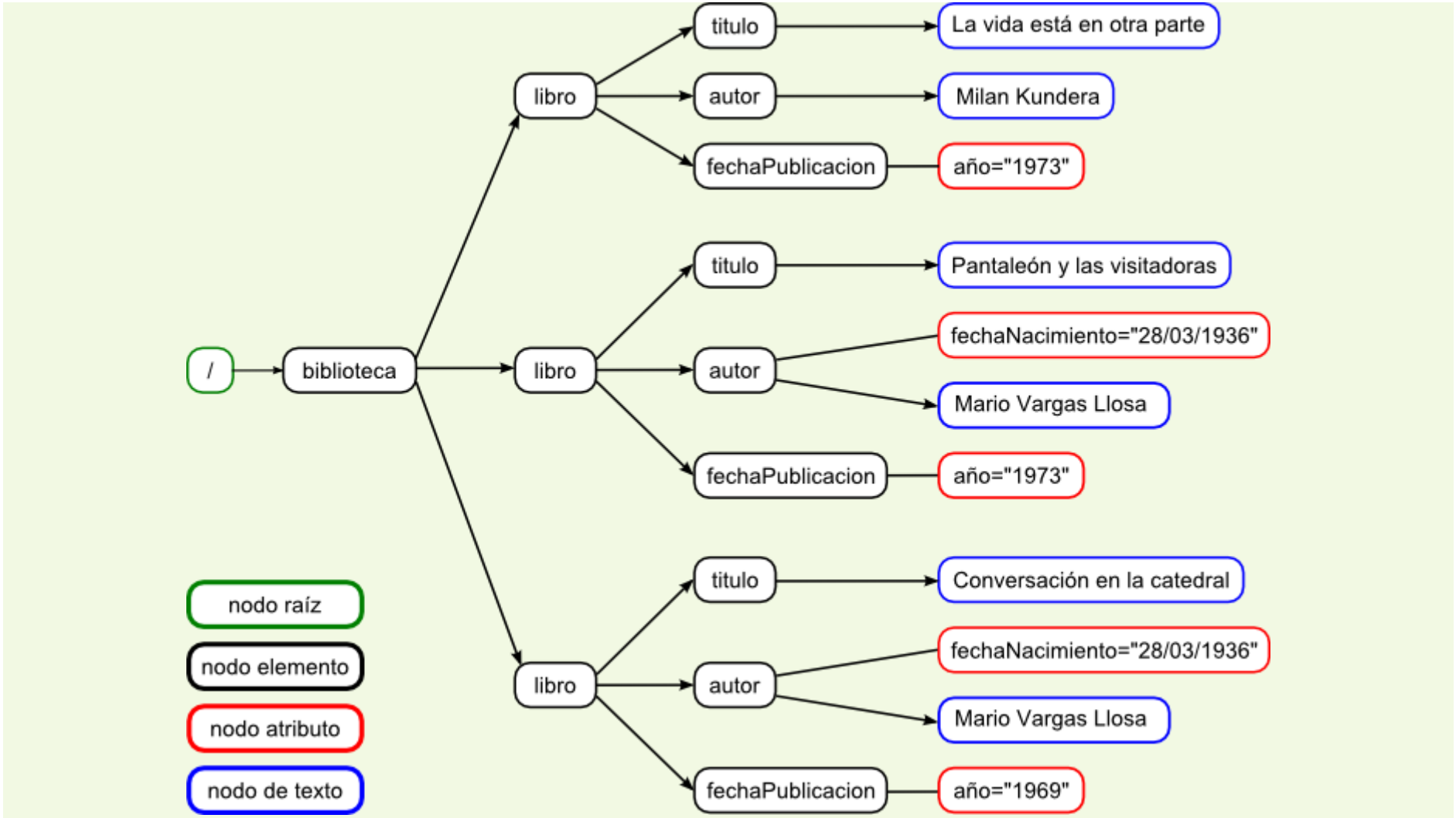
Expresiones XPath

Un documento XML puede representarse como un árbol dirigido, considerando por ejemplo los elementos como nodos y que un elemento es padre de los elementos que contiene. Pero en XPath no sólo los elementos son nodos, en realidad hay siete tipos de nodos:

- Raíz
- Elemento
- Atributo
- Texto
- Comentario
- Instrucción de procesamiento
- Espacio de nombres

```
<?xml version="1.0" encoding="UTF-8"?>
<biblioteca>
  <libro>
    <titulo>La vida está en otra parte</titulo>
    <autor>Milan Kundera</autor>
    <fechaPublicacion año="1973"/>
  </libro>
  <libro>
    <titulo>Pantaleón y las visitadoras</titulo>
    <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>
    <fechaPublicacion año="1973"/>
  </libro>
  <libro>
    <titulo>Conversación en la catedral</titulo>
    <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>
    <fechaPublicacion año="1969"/>
  </libro>
</biblioteca>
```

El documento XML tiene el siguiente grafo:



Los nodos atributos y de texto no son como los nodos elemento. Por ejemplo, los nodos atributo y de texto no pueden tener descendientes. En realidad el nodo atributo ni siquiera se considera como hijo, sino como una etiqueta adosada al elemento. El texto contenido por una etiqueta sí que se considera hijo del elemento, aunque las expresiones XPath suelen trabajar con nodos elementos y para referirse a los atributos o al texto se utilizan notaciones especiales.

Sintaxis de la expresiones XPath

Una expresión XPath es una cadena de texto que representa un recorrido en el árbol del documento. Las expresiones más simples se parecen a las rutas de los archivos en el explorador de Windows o en la shell de GNU/Linux.

Evaluar una expresión XPath es buscar si hay nodos en el documento que se ajustan al recorrido definido en la expresión. El resultado de la evaluación son todos los nodos que se ajustan a la expresión. Para poder evaluar una expresión XPath, el documento debe estar bien formado.

Las expresiones XPath se pueden escribir de dos formas distintas:

- sintaxis abreviada: más compacta y fácil de leer
- sintaxis completa: más larga pero con más opciones disponibles

Las expresiones XPath se pueden dividir en pasos de búsqueda. Cada paso de búsqueda se puede a su vez dividir en tres partes:

- eje: indica el nodo o los nodos en los que se realiza la búsqueda
- nodo de comprobación: especifica el nodo o los nodos seleccionados dentro del eje
- predicado: permite restringir los nodos de comprobación

Ejes

- ***/***: si está al principio de la expresión, indica el nodo raíz, si no, indica "hijo".

/biblioteca/libro/autor	<autor>Milan Kundera</autor> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>
/autor	No devuelve nada porque "autor" no es hijo del nodo raíz.
/biblioteca/autor	No devuelve nada porque "autor" no es hijo de "biblioteca".

- ***//***: indica "descendiente" (hijos, hijos de hijos, etc.).

/biblioteca//autor	<autor>Milan Kundera</autor> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>
//autor	<autor>Milan Kundera</autor> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>
//autor//libro	No devuelve nada porque "libro" no es descendiente de "autor".

- **@atributo**: selecciona el atributo.

/biblioteca/libro/autor/@fechaNacimiento	fechaNacimiento="28/03/1936" fechaNacimiento="28/03/1936"
/biblioteca/libro/@fechaNacimiento	No devuelve nada porque "libro" no tiene el atributo fechaNacimiento.

- **..**: selecciona el elemento padre.

/biblioteca/libro/autor/@fechaNacimiento/..	<autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>

- Nota: En este ejemplo se seleccionan únicamente los nodos "autor" que tienen el atributo fechaNacimiento.
- **|**: permite elegir varios recorridos.

//autor|//titulo	<titulo>La vida está en otra parte</titulo> <autor>Milan Kundera</autor> <titulo>Pantaleón y las visitadoras</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <titulo>Conversación en la catedral</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>

Nodos de comprobación

- **node()**: selecciona todos los nodos (elementos y texto).

//libro/node()	<titulo>La vida está en otra parte</titulo> <autor>Milan Kundera</autor> <fechaPublicacion año="1973"/> <titulo>Pantaleón y las visitadoras</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1973"/> <titulo>Conversación en la catedral</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1969"/>
//autor/node()	Milan Kundera Mario Vargas Llosa Mario Vargas Llosa
//libro//node()	<titulo>La vida está en otra parte</titulo> La vida está en otra parte <autor>Milan Kundera</autor> Milan Kundera <fechaPublicacion año="1973"/> <titulo>Pantaleón y las visitadoras</titulo> Pantaleón y las visitadoras <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> Mario Vargas Llosa <fechaPublicacion año="1973"/> <titulo>Conversación en la catedral</titulo> Conversación en la catedral <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> Mario Vargas Llosa <fechaPublicacion año="1969"/>

- **text()**: selecciona el contenido del elemento (texto).

//autor/text()	Milan Kundera Mario Vargas Llosa Mario Vargas Llosa

//libro/text()	No devuelve nada porque "libro" no contiene texto.

- *****: selecciona todos los elementos

/biblioteca/*	<libro> <titulo>La vida está en otra parte</titulo> <autor>Milan Kundera</autor> <fechaPublicacion año="1973"/> </libro> <libro> <titulo>Pantaleón y las visitadoras</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1973"/> </libro> <libro> <titulo>Conversación en la catedral</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1969"/> </libro>
//autor/*	No devuelve nada porque "autor" sólo contiene texto.
/biblioteca//*	<libro> <titulo>La vida está en otra parte</titulo> <autor>Milan Kundera</autor> <fechaPublicacion año="1973"/> </libro> <titulo>La vida está en otra parte</titulo> <autor>Milan Kundera</autor> <fechaPublicacion año="1973"/> <libro> <titulo>Pantaleón y las visitadoras</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1973"/> </libro> <titulo>Pantaleón y las visitadoras</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1973"/> <libro>

	<titulo>Conversación en la catedral</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1969"/> </libro> <titulo>Conversación en la catedral</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1969"/>

- @*: selecciona todos los atributos

//@*	año="1973" fechaNacimiento="28/03/1936" año="1973" fechaNacimiento="28/03/1936" año="1969"
//libro/@*	No devuelve nada porque "libro" no tiene atributos.
//autor/@*	fechaNacimiento="28/03/1936" fechaNacimiento="28/03/1936"

Predicados

Los predicados se escriben entre corchetes

- **[@atributo]: selecciona los elementos que tienen el atributo.**

//autor[@fechaNacimiento]	<autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>

[número]: si hay varios resultados selecciona uno de ellos por número de orden; last() selecciona el último de ellos

//libro[1]	<libro> <titulo>La vida está en otra parte</titulo> <autor>Milan Kundera</autor> <fechaPublicacion año="1973"/> </libro>

//libro[last()]	<libro> <titulo>Conversación en la catedral</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1969"/> </libro>
//libro[last()-1]	<libro> <titulo>Pantaleón y las visitadoras</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1973"/> </libro>

Los predicados permiten definir condiciones sobre los valores de los atributos. En las condiciones se pueden utilizar los operadores siguientes:

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

Las comparaciones se pueden hacer entre valores de nodos y atributos o con cadenas de texto o numéricas. En el caso de las cadenas de texto deben estar rodeadeas por comillas simples o dobles, en el caso de las cadenas numéricas, las comillas son optativas.

[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 (utilizando .)

//fechaPublicacion[@año<1970]	<fechaPublicacion año="1969"/>
//fechaPublicacion[@año<1970]/..	<libro> <titulo>Conversación en la catedral</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1969"/> </libro>
//libro[autor='Mario Vargas Llosa']	<libro> <titulo>Pantaleón y las visitadoras</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1973"/> </libro> <libro> <titulo>Conversación en la catedral</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>

	<fechaPublicacion año="1969"/> </libro>
//autor[.="Mario Vargas Llosa"]/..	<libro> <titulo>Pantaleón y las visitadoras</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1973"/> </libro> <libro> <titulo>Conversación en la catedral</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1969"/> </libro>

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.

//libro[autor='Mario Vargas Llosa'][fechaPublicacion/@año="1969"]	<libro> <titulo>Conversación en la catedral</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1969"/> </libro>
//libro[autor='Mario Vargas Llosa' and fechaPublicacion/@año="1969"]	<libro> <titulo>Conversación en la catedral</titulo> <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor> <fechaPublicacion año="1969"/> </libro>

Expresiones anidadas

Las expresiones XPath pueden anidarse, lo que permite definir expresiones más complicadas. Por ejemplo, en el documento utilizado anteriormente:

```
<?xml version="1.0" encoding="UTF-8"?>
<biblioteca>
  <libro>
    <titulo>La vida está en otra parte</titulo>
    <autor>Milan Kundera</autor>
    <fechaPublicacion año="1973"/>
  </libro>
  <libro>
    <titulo>Pantaleón y las visitadoras</titulo>
    <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>
    <fechaPublicacion año="1973"/>
  </libro>
  <libro>
    <titulo>Conversación en la catedral</titulo>
    <autor fechaNacimiento="28/03/1936">Mario Vargas Llosa</autor>
    <fechaPublicacion año="1969"/>
  </libro>
</biblioteca>
```

Un ejemplo de expresión anidada sería, por ejemplo, obtener los títulos de los libros publicados el mismo año que la novela "La vida está en otra parte". Esta información no está directamente almacenada en el documento, pero se puede obtener la respuesta en dos pasos:

obtener primero el año en que se publicó la novela "La vida está en otra parte":

//titulo[.="La vida está en otra parte"]/../fechaPublicacion/@año	año="1973"

y obtener después los títulos de los libros publicados en 1973:

//fechaPublicacion[@año=1973]/../titulo	<titulo>La vida está en otra parte</titulo> <titulo>Pantaleón y las visitadoras</titulo>

Estas dos expresiones se pueden unir en una única expresión, sustituyendo en la segunda expresión el valor 1973 por la primera expresión:

//fechaPublicacion[@año=**//titulo[.="La vida está en otra parte"]/../fechaPublicacion/@año**]/../titulo	<titulo>La vida está en otra parte</titulo> <titulo>Pantaleón y las visitadoras</titulo>

Otro ejemplo de expresión anidada sería obtener los títulos de los libros del mismo autor que la novela "Pantaleón y las visitadoras". Como en el ejemplo anterior, la respuesta puede obtenerse en dos pasos:

- obtener primero el autor de la novela "Pantaleón y las visitadoras":

//libro[titulo="Pantaleón y las visitadoras"]/autor/text()	Mario Vargas Llosa

y obtener después los títulos de los libros escritos por Mario Vargas LLosa:

//libro[autor="Mario Vargas Llosa"]/titulo	<titulo>Pantaleón y las visitadoras</titulo> <titulo>Conversación en la catedral</titulo>

Estas dos expresiones se pueden unir en una única expresión, sustituyendo en la segunda expresión el valor "Mario Vargas Llosa" por la primera expresión:

//libro[autor=**//libro[titulo="Pantaleón y las visitadoras"]/autor/text()**]/titulo	<titulo>Pantaleón y las visitadoras</titulo> <titulo>Conversación en la catedral</titulo>

Un detalle importante es que no hay que escribir la primera expresión entre comillas.

Se puede omitir el nodo de comprobación text() de la segunda expresión y escribir la expresión XPath así:

//libro[autor=**//libro[titulo="Pantaleón y las visitadoras"]/autor**]/titulo	<titulo>Pantaleón y las visitadoras</titulo> <titulo>Conversación en la catedral</titulo>