Diseño de un Sistema Clasificador de Libros de Texto con XLST.

Luis Ángel Guendolay Villalobos, Ángel Martínez Velasco, Raymundo Ulises Reyes Quijano,

Unidad Profesional Interdisciplinaria de Ingeniería y Ciencias Sociales y Administrativas, Instituto Politécnico Nacional, Av Té #950 Col Granjas México, Iztacalco, 08400, CDMX  
amartinezv1511@alumno.ipn.mx

1 INTRODUCCIÓN

El presente proyecto se enfoca al diseño de un clasificador cuyas tareas serán la selección y ordenación de los datos. Esto apoyado del inmenso poder de HTML, ayudándonos de etiquetas simples como tr, td, y styles.

Como aspecto influyente en el proyecto se diseñó un apartado XLST, el cual consiste en una serie de decisiones como: el resaltado de determinadas palabras de los títulos; que el número de edición sea incluido entre paréntesis detrás de aquellos en la forma "(2a. ed.)"; que a continuación sea incluida una indicación de la novedad de la obra a base de asteriscos; y que cuando hay más de un autor sea incluido "et al." en cursiva detrás del primero.

1.1 MARCO TEÓRICO

El patrón web que utilizaremos es XML que son las siglas o abreviatura de eXtensible Markup Language (Lenguaje de Marcas Extensible). Su característica principal es que es un lenguaje que se utiliza para decir algo acerca de otro, similar al programa extensible de etiquetas. Haciendo un poco de historia, podemos decir que los creadores de este lenguaje fueron Word Wide Web Consortium (W3C), quienes son conocidos a nivel mundial como una comunidad internacional que básicamente desarrolla programas que aseguran el crecimiento de la Web a largo plazo.

El XML es considerado como una adaptación de otro programa llamado SGML (Standard Generalized Markup Language), un lenguaje que permite la organización y el etiquetado de documentos. Por ello se dice también que el XML no es un lenguaje, sino más bien un sistema que permite definir lenguajes de acuerdo con las necesidades. El XHTML, el MathML y el SVG (en cuyo detalle no entraremos) son algunos de los lenguajes que el XML tiene la capacidad de definir.

La semántica que se utilizaría sería XSLT (eXtensible Stylesheet Language for Transformations) que es un lenguaje que permite aplicar una transformación a un documento XML para obtener otro documento XML, un documento HTML o un documento de texto plano.

La hoja de estilos XSLT con las reglas de transformación es también un documento de texto XML en sí, generalmente con extensión .xsl, por lo que se podrá comprobar si está bien formado o no.

A un documento XML se le pueden aplicar una o varias transformaciones XSLT e incluso una transformación CSS. Las hojas de estilos XSLT son más útiles que las hojas de estilos CSS porque:

Permiten cambiar el orden los elementos.

Permiten realizar operaciones con sus valores.

Permiten agrupar elementos.

​De ahí que se suelan utilizar en combinación más que decantarse por una u otra hoja de estilos.

XSLT es un estándar del W3C:

* XSLT 1.0
* XSLT 2.0
* XSLT 3.0

Para diseñar hojas de estilos XSLT podemos utilizar el software libre XML Copy Editor o cualquier otro software que comentamos aquí.

Para probarlo sólo tenemos que abrir el documento XML en el navegador Mozilla Firefox o Internet Explorer para ver el resultado de la transformación. El navegador Google Chrome por defecto no aplica las transformaciones.

En el navegador podemos utilizar la opción del botón derecho "Inspeccionar elemento" para comprobar los resultados de la transformación, ya que si elegimos "Ver código fuente" se nos abrirá el documento XML original sin ninguna transformación.

Las hojas de estilo XSLT - aunque el término de hojas de estilo no se aplica sobre la función directa del XSLT - realizan la transformación del documento utilizando una o varias reglas de plantilla. Estas reglas de plantilla unidas al documento fuente a transformar alimentan un procesador de XSLT, el que realiza las transformaciones deseadas poniendo el resultado en un archivo de salida, o, como en el caso de una página web, las hace directamente en un dispositivo de presentación tal como el monitor del usuario.

Actualmente, XSLT es muy usado en la edición web, generando páginas HTML o XHTML. La unión de XML y XSLT permite separar contenido y presentación, aumentando así la productividad.

El método estándar para transformar datos XML a otros formatos XML es XSLT. Puede utilizar la función incorporada XSLTRANSFORM para convertir documentos XML a HTML, texto sin formato o diferentes esquemas XML.

XSLT utiliza hojas de estilo para convertir XML a otros formatos de datos. Puede convertir una parte de un documento XML o todo el documento y seleccionar o reorganizar los datos utilizando el lenguaje de consultas XPath y las funciones incorporadas de XSLT. XSLT se utiliza habitualmente para convertir XML a HTML, pero también puede utilizarse para transformar documentos XML que cumplan con un esquema XML en documentos que cumplan con otro esquema. XSLT también puede utilizarse para convertir datos XML a formatos relacionados, como texto delimitado por comas o lenguajes de formateo como troff. XSLT tiene dos áreas de aplicabilidad principales:

Formateo (conversión de XML a HTML)

Intercambio de datos (consulta, reorganización y conversión de datos de un esquema XML a otro, o a un formato de intercambio de datos como SOAP)

Ambos casos pueden requerir la transformación de un documento XML completo o sólo de partes seleccionadas del mismo. XSLT incorpora la especificación XPath, que permite la consulta y recuperación de datos arbitrarios del documento XML de origen. Una plantilla de XSLT también puede contener o crear información adicional, como por ejemplo cabeceras de archivo y bloques de instrucciones, que se añadirá al archivo de salida.

**1.1.1 ELEMENTOS Y RESTRICCIONES**

**El elemento <xsl:value-of>.** Se usa para seleccionar el valor de un elemento XML y añadirlo al documento resultado de la transformación. El elemento se selecciona por medio del atributo obligatorio select, que toma como valor una expresión de XPath. En el ejemplo,

select="$hoy - 5" resta 5 a la variable hoy (que fue definida por medio de <xsl:variable name="hoy">2005</xsl:variable>), dando como resultado "2000";

select="edicion/@numedic" incorpora al documento de salida el valor del atributo numedic del elemento <edicion> (hijo del nodo actual, que es "/catalogo/libro" según se estableció con <xsl:for-each>);

select="autor[1]" extrae el contenido del primer elemento <autor> que sea hijo del nodo actual;

select="." extrae el contenido del nodo actual (que es el nodo <enfasis>).

**El elemento <xsl:for-each>.** Permite realizar bucles en XSLT. Puede usarse para seleccionar secuencialmente cada elemento XML de un determinado conjunto de nodos. El valor del atributo requerido select contiene una localización XPath.

En el ejemplo se ha usado, además de para seleccionar los elementos adecuados, esto es, cada <libro> del <catalogo>, con select="catalogo/libro" (la ruta es relativa al nodo actual, la raíz del documento tal como se estableció con el atributo match="/" de la plantilla en uso), también para filtrar determinados elementos, con el predicado añadido entre corchetes, select="catalogo/libro[edicion/anyo &gt;= $hoy - 5]", que contiene una expresión XPath relacional para seleccionar sólo aquellos libros cuyo año de publicación (<anyo>, hijo del elemento <edicion>) sea mayor o igual que 2000 (2005, valor de la variable hoy, menos 5).

**El elemento <xsl:for-each select="child::\*">.** Se usa cuando se busca las etiquetas hijos del XML, para el fin de identificar las etiquetas seleccionadas con el fin de buscar y seleccionar los datos de ello, osea que aplica en una plantilla a cada nodo, que, en este caso, se busca los nodos principales que usa la biblioteca.

**1.1.2 ETL**

Es el proceso que permite a las organizaciones mover datos desde múltiples fuentes, reformatearlos y limpiarlos, y cargarlos en otra base de datos, data mart, o data warehouse (DW) para analizar, o en otro sistema operacional.

En este caso, se realiza el ETL usando el XSLT para clasificar los datos de la biblioteca, incluyendo, determinar los datos por sus atributos, que en esta ocasión se usa texto (text) y entero (int). Para llevarse a cabo en usar los queries en SQL para cargarlos en el DataWarehouse.

Gracias a ello, se puede crear el Data mart ya sea porque menores cantidades de datos implican que se procesan antes, tanto las cargas de datos como las consultas y se adopta a la arquitectura Top-Down.

2 DISEÑO DE LA APLICACIÓN

**2.1 XML**

<?xml version="1.0" encoding="ISO-8859-1"?>

<?xml-stylesheet type="text/xsl" href="catalogo.xsl"?>

<catalogo>

<operaciones>

<disenio>

<libro>

<autor>texto</autor>

<titulo>texto</titulo>

<numedicion>entero</numedicion>

<anyo>entero</anyo>

</libro>

</disenio>

<datos>

<libro>

<autor>Matteo Pagani.</autor>

<titulo> Windows Phone 8 <enfasis> Development </enfasis> </titulo>

<numedicion>3</numedicion>

<edicion numedic="3">

<anyo>2002</anyo>

</edicion>

</libro>

<libro>

<autor>Chris Rose.</autor>

<titulo> Direct 3D </titulo>

<numedicion>2</numedicion>

<edicion numedic="2">

<anyo>2001</anyo>

</edicion>

</libro>

<libro>

<autor>Stacia Misner.</autor>

<titulo> SQL Server Analysis Services </titulo>

<numedicion>4</numedicion>

<edicion numedic="4">

<anyo>2000</anyo>

</edicion>

</libro>

<libro>

<autor>Chris Rose.</autor>

<titulo> Assembly Language </titulo>

<numedicion>2</numedicion>

<edicion numedic="2">

<anyo>2005</anyo>

</edicion>

</libro>

<libro>

<autor>Emanuele DelBono.</autor>

<titulo> <enfasis>ASP.NET</enfasis> Web API </titulo>

<numedicion>3</numedicion>

<edicion numedic="3">

<anyo>2002</anyo>

</edicion>

</libro>

<libro>

<autor>Joe Booth.</autor>

<titulo> Visual Studio Add-Ins </titulo>

<numedicion>2</numedicion>

<edicion numedic="2">

<anyo>2004</anyo>

</edicion>

</libro>

<libro>

<autor>Peter Shaw.</autor>

<titulo> Postgres </titulo>

<numedicion>1</numedicion>

<edicion numedic="1">

<anyo>1999</anyo>

</edicion>

</libro>

<libro>

<autor>Chris Rose.</autor>

<titulo> Direct2D </titulo>

<numedicion>1</numedicion>

<edicion numedic="1">

<anyo>2001</anyo>

</edicion>

</libro>

<libro>

<autor>Stacia Misner.</autor>

<titulo> Windows Azure SQL Reporting </titulo>

<numedicion>3</numedicion>

<edicion numedic="3">

<anyo>2006</anyo>

</edicion>

</libro>

<libro>

<autor>Marc Clifton.</autor>

<titulo> Unit Testing </titulo>

<numedicion>2</numedicion>

<edicion numedic="2">

<anyo>2003</anyo>

</edicion>

</libro>

<libro>

<autor>Steve Fenton.</autor>

<titulo> TypeScript </titulo>

<numedicion>2</numedicion>

<edicion numedic="2">

<anyo>2004</anyo>

</edicion>

</libro>

<libro>

<autor>Buddy James.</autor>

<titulo> WPF </titulo>

<numedicion>2</numedicion>

<edicion numedic="2">

<anyo>2006</anyo>

</edicion>

</libro>

<libro>

<autor>Ryan Hodson.</autor>

<titulo> iOS </titulo>

<numedicion>3</numedicion>

<edicion numedic="3">

<anyo>2005</anyo>

</edicion>

</libro>

<libro>

<autor>John Garland.</autor>

<titulo> Windows Store Apps </titulo>

<numedicion>1</numedicion>

<edicion numedic="1">

<anyo>2005</anyo>

</edicion>

</libro>

<libro>

<autor>Robert Horvick.</autor>

<titulo> Data Structures Part 1 </titulo>

<numedicion>1</numedicion>

<edicion numedic="1">

<anyo>2002</anyo>

</edicion>

</libro>

<libro>

<autor>Peter Shaw.</autor>

<titulo> GIS </titulo>

<numedicion>2</numedicion>

<edicion numedic="2">

<anyo>1998</anyo>

</edicion>

</libro>

<libro>

<autor>Robert Horvick.</autor>

<titulo> Data Structures Part 2. </titulo>

<numedicion>2</numedicion>

<edicion numedic="2">

<anyo>2003</anyo>

</edicion>

</libro>

<libro>

<autor>Ryan Hodson.</autor>

<titulo> Objective-C </titulo>

<numedicion>3</numedicion>

<edicion numedic="3">

<anyo>2004</anyo>

</edicion>

</libro>

<libro>

<autor>Robert Pickering.</autor>

<titulo> F# </titulo>

<numedicion>2</numedicion>

<edicion numedic="2">

<anyo>1999</anyo>

</edicion>

</libro>

<libro>

<autor>Ryan Hodson.</autor>

<titulo> Knockout.js </titulo>

<numedicion>1</numedicion>

<edicion numedic="1">

<anyo>2005</anyo>

</edicion>

</libro>

<libro>

<autor>Michael McLaughlin.</autor>

<titulo> <enfasis>C++</enfasis> </titulo>

<numedicion>2</numedicion>

<edicion numedic="2">

<anyo>2004</anyo>

</edicion>

</libro>

<libro>

<autor>Lyle Luppes.</autor>

<titulo> ASP.NET MVC 4 Mobile Websites </titulo>

<numedicion>1</numedicion>

<edicion numedic="1">

<anyo>2005</anyo>

</edicion>

</libro>

<libro>

<autor>Jan Van der Haegen.</autor>

<titulo> LightSwitch </titulo>

<numedicion>2</numedicion>

<edicion numedic="2">

<anyo>1998</anyo>

</edicion>

</libro>

<libro>

<autor>Cody Lindley.</autor>

<titulo> JavaScript </titulo>

<numedicion>3</numedicion>

<edicion numedic="3">

<anyo>2007</anyo>

</edicion>

</libro>

<libro>

<autor>Scott Allen.</autor>

<titulo> <enfasis>HTTP</enfasis> </titulo>

<numedicion>5</numedicion>

<edicion numedic="5">

<anyo>2005</anyo>

</edicion>

</libro>

<libro>

<autor>Cody Lindley.</autor>

<titulo> jQuery </titulo>

<numedicion>4</numedicion>

<edicion numedic="4">

<anyo>2004</anyo>

</edicion>

</libro>

</datos>

</operaciones>

</catalogo>

**2.3 XSLT**

<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:variable name="hoy">

2005

</xsl:variable>

<xsl:template match="//catalogo">

<html lang="es-ES">

<head>

<meta content="text/html; charset=utf-8" http-equiv="Content-Type"/>

<title>

Ingeniería del Conocimiento

</title>

</head>

<body style="font-family : sans-serif;">

<h1>

Ingeniería del Conocimiento - "Clasificador Equipo 6"

</h1>

<h2>

Mejores libros informaticos de

<xsl:value-of select="$hoy - 5">

</xsl:value-of>

o posteriores, ordenadas por autor

</h2>

<table style="border : 2px solid black; border-collapse : collapse" summary="Listado tabulado de las obras">

<tr style="color : black; background-color : green;">

<th style="border : 1px solid green;">

Autor

</th>

<th style="border : 1px solid green;">

Título

</th>

</tr>

<xsl:for-each select="operaciones/datos/libro[edicion/anyo >= $hoy - 5]">

<xsl:sort select="autor[1]">

</xsl:sort>

<tr>

<td style="border : 1px solid green;">

<xsl:value-of select="autor">

</xsl:value-of>

</td>

<td style="border : 1px solid green;">

<xsl:apply-templates select="titulo" />

<xsl:if test="edicion[@numedic]">

(

<xsl:value-of select="edicion/@numedic" />

a. ed.)

</xsl:if>

<xsl:choose>

<xsl:when test="edicion/anyo >= $hoy - 1"> (\*\*\*\*) </xsl:when>

<xsl:when test="edicion/anyo >= $hoy - 2"> (\*\*\*) </xsl:when>

<xsl:when test="edicion/anyo >= $hoy - 3"> (\*\*) </xsl:when>

<xsl:when test="edicion/anyo >= $hoy - 4"> (\*) </xsl:when>

<xsl:otherwise>

</xsl:otherwise>

</xsl:choose>

</td>

<td>

<td style="border : 1px solid gray;">

<xsl:apply-templates select="anyo">

</xsl:apply-templates>

<xsl:if test="edicion/anyo = 2005">

(2005)

</xsl:if>

</td>

</td>

</tr>

</xsl:for-each>

</table>

<p>

Nota: Los asteriscos indican obras de los cuatro últimos años (\*\*\*\*: año en curso -

<xsl:value-of select="$hoy"/>

- y anterior)

</p>

<h1>

Lista de consultas SQL

</h1>

<xsl:for-each select="child::\*/\*">

<xsl:choose>

<xsl:when test="name()='disenio'">

<xsl:for-each select="child::\*">

CREATE TABLE

<xsl:value-of select="name(.)"/>

(

<xsl:for-each select="child::\*">

<xsl:value-of select="name(.)"/>

<xsl:choose>

<xsl:when test="(.)='entero'"> int </xsl:when>

<xsl:when test="(.)='texto'"> text </xsl:when>

<xsl:when test="(.)='logico'"> boolean </xsl:when>

<xsl:when test="(.)='decimal'"> double </xsl:when>

<xsl:when test="(.)='fecha'"> date </xsl:when>

</xsl:choose>

<xsl:if test="following-sibling::\*">

,

</xsl:if>

</xsl:for-each>

);

<br/>

</xsl:for-each>

</xsl:when>

<xsl:when test="name()='datos'">

<xsl:for-each select="child::\*">

INSERT INTO

<xsl:value-of select="name(.)" />

VALUES (

<xsl:for-each select="child::\*">

<xsl:value-of select="(.)" />

<xsl:if test="following-sibling::\*">

,

</xsl:if>

</xsl:for-each>

);

<br/>

</xsl:for-each>

</xsl:when>

</xsl:choose>

</xsl:for-each>

</body>

</html>

</xsl:template>

<xsl:template match="enfasis">

<span style="font-weight : bold;">

<xsl:value-of select=".">

</xsl:value-of>

</span>

</xsl:template>

</xsl:styles

REFERENCIAS

1. Sociedad del conocimiento, OEA, recuperado de <http://www.oas.org/es/temas/sociedad_conocimiento.asp>
2. Qué son y para qué sirven los archivos fiel y xml (2015), Soy conta, Recuperado de: <http://www.soyconta.mx/que-son-y-para-que-sirven-los-archivos-fiel-y-xml/>
3. Programación SQL XML, recuperado de: <https://www.ibm.com/support/knowledgecenter/es/ssw_ibm_i_72/rzasp/rzaspxml0648.htm>