Revista Educación en Ingeniería
ISSN 1900-8260
Diciembre de 2010 • Nº. 10 • Pp 137-148 • Publicada en línea por la Asociación Colombiana de Facultades de Ingeniería -ACOFI- www.acofi.edu.co
Enviado: 07/10/2010 • Aprobado: 26/11/2010

UNA EXPERIENCIA DE APLICACIÓN PARA LA GESTIÓN DE INFORMACIÓN DE CONTENIDO EDUCATIVO CON TECNOLOGÍAS XML

Ingrid Durley Torres, Jaime A. Guzmán Luna, Jovani Alberto Jiménez Builes
Universidad Nacional de Colombia, Facultad de Minas, Medellín (Colombia)

Resumen

A fin de aprovechar al máximo el potencial de los contenidos digitales, en el ámbito educativo han surgido un conjunto de formatos abiertos para contenido general, como los representados por las tecnologías XML. Codificar los contenidos educativos en tales formatos, puede proporcionar la base para crear contenidos estructurados y reutilizables que se ajusten dinámicamente a las necesidades de información específicas de un usuario. Este articulo, aporta un ejemplo de aplicación basado en las tecnologías XML *(XMLSchema y XQuery)* para la gestión dinámica de información de contenidos educativos estructurados y reutilizables.

Palabras clave: contenidos educativos reutilizables, gestión dinámica de información, XML, XMLSchema y XQuery

Abstract

To maximize the potential of digital content in education have been a series of open formats for general content, as represented by XML technologies. Encode the educational content in such formats, can provide it the basis for creating reusable content dynamically to meet specific information needs of a user. This paper, provides an example of implementation based on XML (XMLSchema and XQuery) for dynamic management of reusable educational content information.

Keywords: educational content, dynamic management of information, XML, XMLSchema and XQuery

Introducción

En el contexto de la educación con medios electrónicos (Hernández, 2007), pueden distinguirse dos vertientes de investigación y desarrollo que han contribuido a conformar las bases tecnológicas para la integración de los nuevos sistemas de educación en línea. Una vertiente

corresponde a lo que se conoce como instrucción basada en computadora (Sarromona, 2008), que ha producido sistemas cada vez más sofisticados en el uso de tecnologías computacionales y de telecomunicaciones pero que, sin embargo, suelen ser más bien simples, rígidos y estáticos en la generación de experiencias educativas.

La otra vertiente de investigación se enfoca sobre los llamados sistemas tutores inteligentes (Morales y Agüera, 2006), desarrollados bajo la hipótesis de que los computadores son capaces de modelar el aprendizaje en diversos dominios de conocimiento e inferir, a partir de la interacción con cada estudiante, la estrategia más apropiada de enseñanza en cada caso.

En este contexto, el internet y la web (Carvin, 2006), han impulsado una revolución que ha impactado fuertemente ambas corrientes de investigación y desarrollo. La posibilidad de tener miles de usuarios conectados a un curso, pero cada uno con necesidades y preferencias de aprendizaje particulares, ha llevado a ambas corrientes a converger en el objetivo de definir las bases teóricas y pragmáticas que sustenten una nueva generación de sistemas educativos y de capacitación con medios electrónicos; es decir, sistemas que se beneficien de los logros obtenidos en ambas direcciones.

La arquitectura de los nuevos sistemas educativos puede ser adaptada de varias maneras a la web. Las alternativas van desde codificar el sistema completo como un applet que se descarga y se ejecuta en el navegador *(browser)* de Web hasta dejar al sistema completo en un servidor y realizar la interacción con el usuario a través de páginas en HTML. Entre estas dos opciones extremas se encuentra un amplio rango de arquitecturas distribuidas más o menos complejas e interesantes. En lo que respecta a los contenidos educativos digitales, éstos pueden ser codificados, dependiendo de su modalidad, en cualquiera de los formatos asociados con la web, estandarizados o propietarios: *(X)HTML, XML, GIF, PNG, JPEG, Flash, MPEG, MP3*, entre otros. También es posible generar los contenidos dinámicamente a partir de información contenida en bases de datos y de conocimiento.

En este sentido, se origina la presente propuesta, la cual profundiza a través de un ejemplo de aplicación práctico, los fundamentos para poder aprovechar los principios de las tecnologías *XML (XML 2004), (XMLSchema y XQuery)* en el formateo y construcción dinámica de contenido educativo digital. Las ventajas de la implementación de estas tecnologías dentro de los contenidos educativos, radica en las posibilidades de reutilización e intercambio (Martínez y Méndez, 2002), que les otorga a los propios contenidos, soportándose en la cualidad de separar la capa de presentación de la propia capa de contenido, lo cual permite ajustarse dinámicamente a cualquier cambio, además de facilitar el desarrollo de sus aplicaciones, ya que los entornos de construcción de software más comunes (como Java o .Net) disponen de utilidades para la manipulación de archivos XML. Además, el desarrollo propuesto, es flexible, permitiendo adaptarse a cualquier dominio de aplicación cuyo formato original coincida con presentaciones *power point* publicadas en documentos PDF.

Con el fin de dar una visión más amplia de la experiencia de aplicación de *XML* en el dominio educativo, las secciones siguientes del presente artículo, se han estructurado de la siguiente manera: en el siguiente capítulo se presenta una visión general del marco teórico, capitulo tres, ilustra el diseño implementada en el caso práctico de aplicación; ocupándose de detallar cada una de las respectivas etapas desde la conversión de los documentos PDF a XML, hasta la generación de los resultados de las consultas XQuery, que permiten gestionar la información. Finalmente se presenta el capítulo cuatro, donde se recopilan las conclusiones del trabajo realizado.

Marco teórico

En la presente sección se brinda un breve recorrido por los principales conceptos que encierran la experiencia de aplicación de trasladar los tradicionales contenidos educativos de presentaciones *power point* (publicados en formato PDF) a formatos XML que posibiliten la gestión dinámica de información. Nuestra visión del problema de gestión de información, se basa en cuatro conceptos fundamentales, los cuales son abordados brevemente a continuación.

Contenido educativo digital

En la Dirección de Tecnología del Instituto Politécnico Nacional de México, definen a los contenidos educativos digitales como: "materiales multimedia digitalizados que invitan al alumno a explorar y manipular la información en forma creativa, atractiva y colaborativa" (IPNM, 2009). José Luis Rodríguez Illera (Rodríguez, *et. al*., 2005) de la Universidad de Barcelona dice al respecto que los contenidos educativos digitales deben "ser capaces de funcionar de manera autoinstructiva (como un curso de repaso) y reconfigurables o adaptables por el profesor para sus propósitos".

XML

XML *(Extensible Markup Language, 2004)*. Al no tener un conjunto predefinido de marcas, cada archivo XML indica qué gramática sigue (tags y atributos válidos, y de qué manera se pueden anidar) en un archivo aparte. En las primeras ediciones de XML se usaron "definiciones de tipo de documento" (DTD de sus siglas en inglés *Document Type Definition*), (DTD, 2001), pero fueron desplazadas por las "definiciones de esquemas XML" *(XMLScheme ó XSD ó también XMLS)*, (XMLS, 2006) por su mayor expresividad gramatical y porque estaban escritas en XML. Sin embargo, conceptualmente, el papel es el mismo. etc. Un documento XML estará "bien-formado" si cumple con la especificación de XML, pero sólo será válido si cumple con su DTD/ XMLS. El logro no es tener mezclados en el mismo documento datos y metadatos (los tags) como hacen los lenguajes predecesores (PostScript o TeX). El verdadero poder está en la capacidad de definir de forma sencilla la gramática que se nos antoje con las DTD/ XMLS y ser capaz de comprobar rápidamente si un documento XML es conforme con la gramática que dice que usa. Por tanto, el desarrollo de XML estaba condicionado por la creación de parseadores eficientes para todas las plataformas y sistemas operativos, y de dominio público. En este sentido, la existencia de Java (independiente de hardware y sistema operativo) y computadores con potencia de cálculo suficiente, fueron factores decisivos para el uso de XML.

Para que todos los modelos de contenido educativo se basen en la misma lista de etiquetas, se debe crear lo que se conoce como una definición de estructura o esquema XML Schema, que es un archivo con extensión .xsd al que deben hacer referencia todos los documentos XML que representen modelos de procesos basados en las mismas etiquetas.

XMLS

El XMLS (*Extensible Markup Language Schema*, 2006), es una descripción de un tipo de documento XML, generalmente expresada en términos de restricciones sobre la estructura y contenido de los documentos de ese tipo, por encima y más allá de las limitaciones sintácticas básicas impuestas por XML en sí mismo. Estas limitaciones se expresan generalmente mediante una combinación de reglas gramaticales que rigen el orden de los elementos, los predicados booleano que el contenido debe cumplir, tipos de datos que regulan el contenido de elementos, atributos y normas más especializadas, como la singularidad e integridad restricciones referenciales. El proceso de control para ver si un

documento XML se ajusta a un esquema se llama validación y es independiente del concepto de XML bien formados, el cual verifica la sintáctica del documento. Todos los documentos XML deben estar bien formados, pero no es necesario que un documento sea válido.

Los documentos sólo se consideran válidos si se cumplen los requisitos del esquema con el que se han asociado. Estos requisitos generalmente incluyen limitaciones como: (i) los elementos y atributos que deben o pueden ser incluidos, y su estructura permitida. (ii) la estructura según lo especificado por una expresión regular sintaxis. Y finalmente, (iii) ¿cómo los datos de caracteres, debe interpretarse, por ejemplo, como un número, una fecha, una dirección, un booleano, entre otros.

La validación de una instancia del documento con un esquema puede considerarse como una operación conceptualmente separada de análisis XML. En la práctica, sin embargo, muchos validadores de esquema se integran con un analizador XML.

XQuery

Tradicionalmente las búsquedas sobre texto se han venido aplicando a documentos como un todo, mientras que las consultas de datos se han aplicado a registros compuestos por varios campos. Dado que XML presenta datos semiestructurados es necesario que el lenguaje de consultas aplicado sobre XML pueda acceder a datos de la parte estructurada del documento, pero que además pueda realizar búsquedas sobre el texto y permita realizar consultas mixtas de contenido y estructura.

En cuanto a los resultados, tradicionalmente las consultas sobre texto devuelven una lista de documentos con cierta información acerca de ellos ordenados según un determinado criterio, mientras que una consulta sobre datos devuelve generalmente un determinado campo, registro o conjunto de registros que pueden verse afectados por un proceso de cálculo, transformación o combinación.

En definitiva, se observa que en las consultas de datos se pone especial énfasis en los grandes almacenes de datos, la integración de datos heterogéneos y la transformación de datos en formatos comunes de intercambio; en cambio, las consultas de texto ponen el énfasis en las búsquedas de texto, la manipulación de conjuntos de resultados, las relaciones de inclusión y la ordenación de los documentos obtenidos como resultado. Así, el lenguaje de consultas sobre XML, debería reunir lo mejor de los dos tipos de consultas comentadas; debería ser la unión de las ventajas obtenidas por un lenguaje de consultas sobre documentos y por un lenguaje de consultas de datos.

XQuery (Boag, *et al.,* 2003) es un lenguaje, nacido para consultar fuentes de datos heterogéneas, el cual a su vez se inspira en el diseño de los siguientes lenguajes: (i) XPath y XQL (lenguajes orientados al documento (Draper, *et al*., 2003)). De ellos adopta la sintaxis para navegar en documentos jerárquicos, ya que para consultar documentos se requiere preservar el orden y la jerarquía. (ii) SQL (lenguaje relacional). De SQL coge la idea del esquema de cláusulas SELECT-FROMWHERE para reestructurar los datos y operaciones tales como uniones (join) y agrupaciones. (iii) OQL (lenguaje orientado a objetos). De éste toma la noción de lenguaje funcional, compuesto por diferentes clases de expresiones que puede anidarse. (iv) XML-QL (lenguaje orientado a información semi-estructurada). De este lenguaje adopta la idea de asignar variables que después se utilizarán para crear nuevas estructuras, especificadas en la recomendación del W3C XML *Schema: structures.*

Descripción general del diseño

Teniendo en cuenta la multiplicidad de oferta y la amplia disponibilidad de contenidos educativos, representada en los diversos repositorios de

la Universidad Nacional de Colombia sede Medellín. La definición del problema se enfocó, en especificar un dominio particular, en este caso se optó por el material del curso de Programación Orientada a Objetos (POO), de la Facultad de Minas de la Escuela de Ingeniería Sistemas. El objetivo, consistía en implementar un sitio web, con su sistema gestor de información respectivo que permitiera generar dinámicamente contenidos educativos, haciendo uso de las tecnologías XML (XMLS y XQuery).

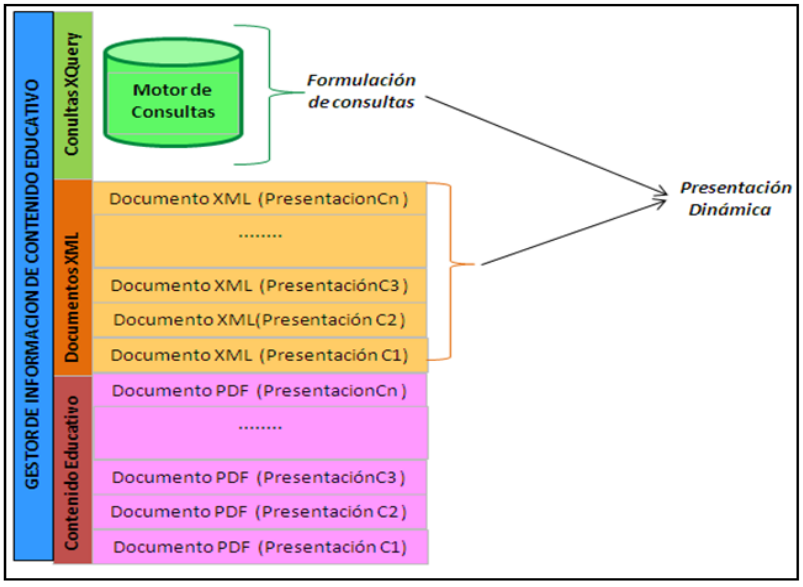
Trasladar el contenido educativo a un escenario de información web, demandaba la representación de todo el material del curso (documentos PDF) en documentos XML. Adicionalmente, para poder lograr una gestión eficiente de información, se implementaron un conjunto de consultas sobre los datos XML(XQuery), las cuales implementadas sobre un motor de razonamiento, permitirían dinámicamente representar, recuperar y presentar la información contenida en el escenario web.

La tarea siguió el diseño señalado en la gráfica1.

Tal diseño estaría representado por un sistema de tres etapas:

- La primera etapa, está representada por el contenido del curso, el cual como es tradicional se encuentra registrado en documentos digitales de formato pdf, que albergan cada una de las presentaciones de las respectivas clases.
- La segunda etapa consistió en generar un Esquema XML (XMLS), que representaría el vocabulario, las restricciones y normas que debería cumplir cada documento XML, que simbolizaría cada uno de los documentos originales (PDF).
- La tercera etapa, permitirá la administración de la información registrada en todos los documentos de manera tal que permita la gestión eficiente y precisa de la información.

Gráfica 1. Metodología básica de diseño



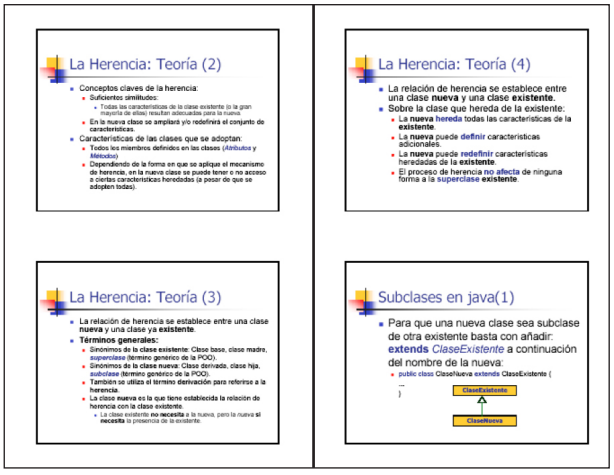
Una vez alcanzadas las tres etapas, será posible generar de manera dinámica un sitio web que recopilará la información que se requiere visualizar de los contenidos digitales.

A continuación se dará una visión más detallada de cada etapa de representación del sistema.

Representación de documentos PDF

La primera capa del sistema, como se ha mencionado corresponde a un documento PDF, construido como resultado de la impresión de las presentaciones power point de las respectivas clases de la cátedra de POO. En la gráfica 2, se puede observar apartes de los documentos en mención.

Gráfica 2. Contenido educativo original (documento PDF)



Construcción de documentos XML

El primer paso, de esta etapa, consistió en implementar el XMLS, que representaría la estructura del documento PDF, sobre el cual se validarían cada uno de los documentos XML. Debido a que el principal objetivo, de esta fase consistía en tratar de implementar una metáfora semejante a la utilizada en el *power point*. El primer paso, consistió en revisar e identificar los elementos básicos de cada documento PDF, dentro de los que se distinguieron tres elementos principales, que deberán ser incluidos en el Esquema XML (ver gráfica 3):

A. *El contenido textual:* para este tipo de información se distinguen varias categorías de texto:

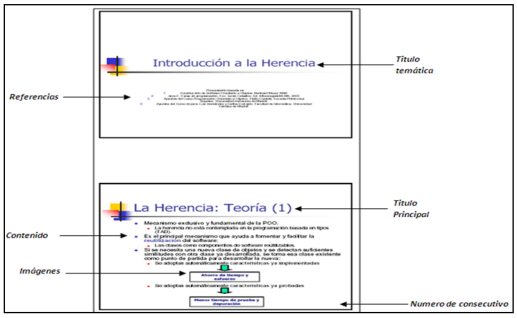
- Titulo de la temática, (es el título del tema de la clase que se muestra al inicio de las diapositivas)
- Titulo principal (Todas las diapositivas llevan título). Cada título deberá llevar un consecutivo
- Subtítulos (No todas las diapositivas llevan subtítulo). Cada subtítulo deberá llevar un consecutivo representado por el consecutivo de su título y el de él mismo.
- Contenido (No todas las diapositivas llevan contenido)
- Referencia bibliográfica

B. *Contenido de imágenes:* en el esquema a generar, incluir los elementos XML necesarios que permitan referenciar las imágenes a utilizar en la presentación. Asociado a lo anterior, y con el fin de estandarizar los formatos de la información, se dispone que las imágenes a implementar deberán de ser de tipo *.jpg

C. *Pie de página:* en este punto se hace necesario que cada documento XML lleve pie de página. Por esta razón este elemento se debe tener en cuenta dentro del Esquema XML. Su caracterización deberá tener en cuenta los siguientes elementos de información:

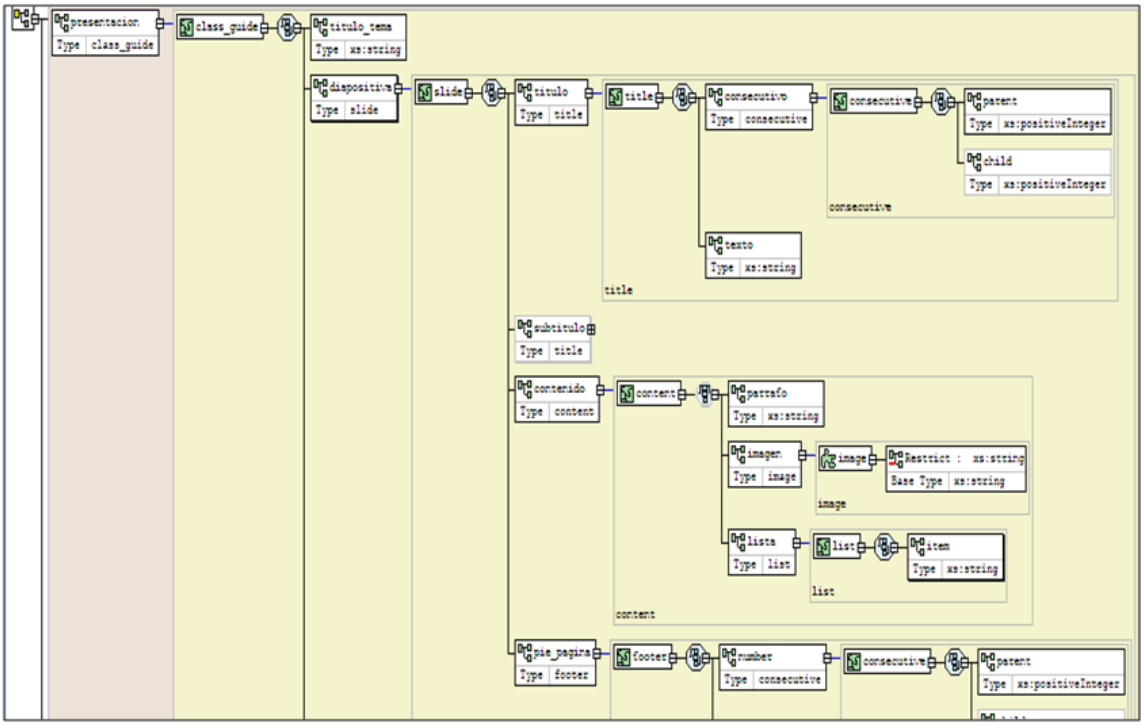
- Un número de consecutivo. Este número deberá relacionar: el número de la unidad de conocimiento que representa (tomado del programa del curso) y el número consecutivo de la diapositiva, el cual indica la posición de la diapositiva dentro de la secuencia. Ejemplo: 2.2.
- Una fecha fija de creación de la diapositiva (indicada por el formato: día/mes/año).
- Una descripción: que corresponde al título de la Unidad del curso imágenes.

Gráfica 3. Elementos básicos del documento PDF



Una vez identificados cada uno de los elementos básicos, definidos su frecuencia de aparición (opcional u obligatoria) y establecido la estructura organizacional de los mismos; se procedió a generar el XMLS, el cual, debe corresponder a los requisitos necesarios que debe cumplir todo documento XML. Una visión parcial del diagrama que representa el XMLS, se encuentra referenciado en la gráfica 4. Es importante aclarar que existen algunas herramientas (ALTOVA, Stylus Studio) que permiten generar en diagrama la especificación lograda en la definición del XMLS.

Gráfica 4. Vista parcial del diagrama del XMLS construido



Como se puede observar el nodo raíz del esquema se encuentra representado por el concepto presentación, el cual en su especificación interna contiene como elemento obligado a *class_guide*, este a su vez se compone de los elementos opcionales titulo del tema y diapositiva. Siendo este último elemento el que contiene todos los subelementos que describen el contenido completo de una diapositiva (denominada en el diagrama como nodo *slide*). Es de aclarar que todos los elementos de slide, son del tipo opcional y su contenido no siempre sigue el mismo orden de secuencia de presentación; salvo el titulo y la información de pie de página que aparece en cada diapositiva de primero y de últimas respectivamente. De esta manera, es posible indicar que una vez expresado el titulo en la diapositiva (esto para los casos en los que se defina), sobre él, se ha de conservar una secuencia consecutiva que nos sirve para indicar de que número de diapositiva procede *(parent)* y a que numero de diapositiva da origen *(child)*; a la vez que nos permite definir con el mismo principio de secuenciación (pera esta vez dependiente del título) un subtitulo, cuya aparición es opcional. En el recorrido de presentación, también se debe abordar el contenido *(content)*, el cual es declarado como un tipo de dato complejo *(complextype)*, sobre el que algunas veces puede aparecer como una sumatoria de imagen *(image)* y luego un párrafo *(parrafo)*, para finalmente terminar con una lista identada de elementos

(list) o jugar con las combinaciones de estos componentes en cualquier orden o simplemente elegir solo uno de ellos para presentarlo como único contenido. Para terminar y con el fin de entregar alguna información específica de la fecha de creación de la diapositiva y el numero de clase a la que pertenece además del número que le corresponde como consecutivo dentro de la secuencia de diapositivas de esa clase, se genera el elemento pie de página.

Aunque la visión parcial señala da en la gráfica 4, no logra mostrar la totalidad del esquema, es importante resaltar que junto con los elementos citados se incorpora uno que relaciona las referencias bibliográficas el cual corresponde a la especificación de uno o más publicaciones, cada una de las cuales se acompaña de una descripción *(description)*, un tipo de publicación *(tipo)* y la especificación de uno o más autores.

Una vez construido el XMLS, se procedió a generar los XML de cada clase perteneciente al curso de programación orientada a objetos, inicialmente construidos como presentaciones power point, condensados en documentos PDF. Cada documento XML generado debía ser evaluado como un documento bien formado (revisa que cumpla correctamente con la estructura sintáctica de XML) y validado contra el XMLS construido. Se debe aclarar que los XML que representaban cada clase, son instanciaciones del XMLS, es decir se toma un XMLS y se generan tantas instancias de slide, como diapositivas contenga la clase. Y en título, subtitulo, contenido etc…se ingresan los datos concretos correspondientes a esa clase (previamente descritos en los documentos PDF). Para el manejo de las imágenes se procedió a generar una carpeta con ese nombre, que implementara todas las imágenes correspondientes a las usadas en las presentaciones. Estas imágenes se definieron en el formato *.jpg y los nombres de sus correspondientes archivos, indicaban el numero del tema (numero de la clase), seguido de un número que indicaba el consecutivo de la gráfica, dentro de ese grupo (por ejemplo *"Tema7Gráfica1"*).

En la gráfica 5, se señalan apartes de la clase número 5, correspondiente al tema de "librerías fundamentales".

Gráfica 5. Vista parcial documento XML clase 5: "Librerías de clases fundamentales"

```
<?xml version="1.0"?>
<p1:presentacion xmlns:p1="http://www.ieb.com.co/joomla" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http:
    <p1:titulo_tema>Librerias de clases fundamentales</p1:titulo_tema>
    <p1:diapositiva>
        <p1:titulo>
            <p1:consecutivo>
                <p1:parent>1</p1:parent>
            </p1:consecutivo>
            <p1:texto>Cadena de Caracteres la clase java.lang.string</p1:texto>
        </p1:titulo>
        <p1:contenido>
            <p1:lista>
                <p1:item>String encapsula la cadena de caracteres y su manipulación</p1:item>
                <p1:item>string =/= de char[]</p1:item>
                <p1:item>Los caracteres de String no se pueden modificar</p1:item>
                <p1:item>Constructores:</p1:item>
                <p1:item>String(char[])</p1:item>
                <p1:item>string(string)</p1:item>
                <p1:item>Creación de strings:</p1:item>
                <p1:item>Char[] char={'a', 'b', 'c'};</p1:item>
                <p1:item>String s=new string (chars);</p1:item>
                <p1:item>String s1="hello";</p1:item>
                <p1:item>String s2=new String(S1)</p1:item>
                <p1:item>Literales:java crea objetos de tipo string para literales</p1:item>
            </p1:lista>
        </p1:contenido>
        <p1:pie_pagina>
            <p1:number>
                <p1:parent>5</p1:parent>
                <p1:child>2</p1:child>
            </p1:number>
            <p1:fecha>2010-08-27</p1:fecha>
            <p1:descripcion>Cadena de Caracteres la clase java.lang.string</p1:descripcion>
        </p1:pie_pagina>
    </p1:diapositiva>

    <p1:diapositiva>
        <p1:titulo>
            <p1:consecutivo>
                <p1:parent>2</p1:parent>
            </p1:consecutivo>
```

Una vez construidos todos los documentos XML que representan el curso de programación orientada a Objetos, se analizaran las paginas XML del sitio, a través de un conjunto de consultas XQuery, tendientes a presentar información específica de las mismas.

Generación de documentos XHTML mediante consultas con XQuery

Con el fin de gestionar la información del curso de programación orientada a objetos, se han creado una serie limitada de consultas, que permiten explotar los contenidos educativos (ahora expresados como documentos XML). Una vez construida la consulta, esta es ejecutada en un motor que soporta el lenguaje XQuery (Altova, Stylus Studio, eXists…). Para el caso particular, se ha desarrollado e implementado un "presentador", es decir una aplicación que actúa como índice de las consultas. Cada consulta es construida como un archivo de extensión .xql que internamente el presentador comunica al motor de ejecución *(eXist)* e inmediatamente almacena el resultado en un documento XHTML, el cual es construido y presentado dinámicamente.

El detalle de cada una de las consultas será abordado a continuación.

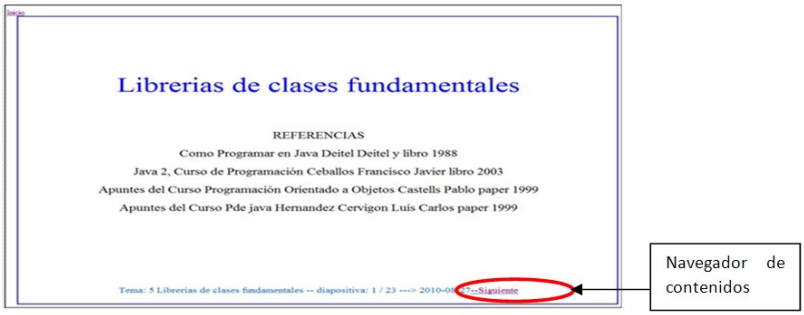
Consulta1. La primera consulta señalada en la gráfica 6, es de propósito general y permite generar cada una de las páginas XHTML con la implementación del contenido registrado en su respectivo documento XML. Para ello maneja una variable que le permite al inicio de la consulta indicar que página desea generar, además de implementar las rutinas respectivas para llamar la generación de la respectiva página. La consulta considera además que cada página generada debe tener su formato particular de presentación XHTML ya que al ser leídas por el navegador *Internet Explorer*, cada página deberá verse de forma adecuada, permitiendo su correcta lectura y manejo de las imágenes del respectivo material del curso de objetos.

Gráfica 6. Consulta de presentación dinámica de contenidos

La gráfica 6, corresponde a la consulta implementada en el lenguaje Xquery, en ella se construye una función que va a iterar sobre cada uno de los datos registrados en el documento, a la vez que permite ir recuperándolos. Sin embargo, para lograr la correcta presentación los resultados son separados por los tipos de contenidos, a los que se les incluye algunas instrucciones de presentación del tipo html (tamño de letra, ubicación. color, etc).

Los datos obtenidos como resultado de la consulta son mostrados en la gráfica 7.

Gráfica 7. Resultado consulta presentación dinámica de contenido



Consulta 2. Consiste en generar inicialmente un archivo XML llamado índice que relacione cada uno de los títulos y subtítulos de cada archivo XML e indique el nombre da cada archivo en el que se encuentra. Una vez construido el XML este debe ser generado como un XHTML, para su respectiva presentación. El archivo .xql de esta consulta es mostrado en la gráfica 8 y el resultado correspondiente es mostrado en la gráfica 9.

Gráfica 8.1 Consulta generación dinámica de un índice

```
declare namespace p1="http://xue.unalmed.edu.co/~jaguzman/SRIW";
declare function local:buildList($s){
<ol>
{
for $i in $s
return <li>{data($s/p1:texto)}</li>
}
</ol>
};

<html><head><title>Indice</title></head><font face="verdana"><h1>Indice</h1><ol>{
for $k in //p1:presentacion
return for $diap in $k//p1:diapositiva
return <li>{data($diap/p1:titulo/p1:texto)} <font size="1" color="006699"> <i>[{data(($k)/p1:titulo_tema)}]</i></font>
{for $s in $diap/p1:subtitulo
return local:buildList($s)}
</li>
}</ol></font></html>
```

Gráfica 8.2 Resultado consulta generación de índice

Indice

1. Contenido[Constructores y Sobrecarga de Métodos]
2. Los Constructores(1)[Constructores y Sobrecarga de Métodos]
3. Los Constructores(2)[Constructores y Sobrecarga de Métodos]
   1. Características
4. Los Constructores(3)[Constructores y Sobrecarga de Métodos]
   1. Sintaxis de los Constructores
5. Los Constructores(4)[Constructores y Sobrecarga de Métodos]
6. Los Constructores(5)[Constructores y Sobrecarga de Métodos]
   1. Creación de Objetos con Constructores
7. Los Constructores (6)[Constructores y Sobrecarga de Métodos]
   1. Constructores por Defecto
8. Variable final en constructor[Constructores y Sobrecarga de Métodos]
9. La variable This en los constructores[Constructores y Sobrecarga de Métodos]
10. Sobrecarga de métodos y de constructores[Constructores y Sobrecarga de Métodos]
11. Sobrecarga de métodos[Constructores y Sobrecarga de Métodos]
12. Sobrecarga de Constructores[Constructores y Sobrecarga de Métodos]
13. Llamada a métodos sobrecargados[Constructores y Sobrecarga de Métodos]
14. Otro ejercicio sobrecarga (1/4)[Constructores y Sobrecarga de Métodos]
15. Otro ejercicio sobrecarga (2/4)[Constructores y Sobrecarga de Métodos]
16. Otro ejercicio sobrecarga (3/4)[Constructores y Sobrecarga de Métodos]
17. Otro ejercicio sobrecarga (4/4)[Constructores y Sobrecarga de Métodos]
18. Ambigüedad en la sobrecarga de métodos[Constructores y Sobrecarga de Métodos]
19. Destrucción de objetos (I)[Destrucción de Objetos y Control de Acceso]
20. Destrucción de objetos (II)[Destrucción de Objetos y Control de Acceso]
21. Los Destructores[Destrucción de Objetos y Control de Acceso]
    1. Características
22. Ejemplo (1)[Destrucción de Objetos y Control de Acceso]
23. Ejemplo (2)[Destrucción de Objetos y Control de Acceso]
24. Ejemplo (3)[Destrucción de Objetos y Control de Acceso]

Consulta 3. Genera una página XHTML que implementa una tabla tipo Excel (con bordes) que contiene el título de la gráfica (incluyendo la palabra gráfica, su consecutivo y su título), una versión en miniatura de la imagen de la propia gráfica y el nombre del archivo XML que la contiene. Para ello itera entre las imágenes de cada presentación y retorna la imagen de la diapositiva (si existe) además de entregar el pie de página con el número de la diapositiva. La formulación de la consulta se detalla en la gráfica 9.

Gráfica 9. Consulta generación dinámica de índice de figuras con su respectiva imagen

```
declare namespace p1="http://xue.unalmed.edu.co/~jaguzman/SRIW";
<html><head><title>Indice de Imagenes</title></head>
<center><font face="verdana" color="006699"><h2>Indice de Imagenes</h2></font></center>
<table border="1" align="center" width="85%" cellpadding="20"><tr><td><font face="verdana"><b>Pag</b></font></td><td><font face="verdana"><b
{
for $pres in //p1:presentacion
return for $diap in $pres//p1:diapositiva
return for $img in $diap//p1:imagen
let $n:=concat($diap/p1:pie_pagina/p1:number/p1:parent,".",$diap/p1:pie_pagina/p1:number/p1:child)
return <tr><td><font face="verdana">{$n}</font></td><td><font face="verdana">Figura {data($diap/p1:titulo/p1:texto)}</font></td><td><font fa
}
</table></html>
```

El correspondiente documento XHTML, es mostrado en la gráfica 10.

Gráfica 10. Documento XHTML con el índice de figuras con su respectiva imagen

Inicio

Indice de Imagenes

Archivo	Pag	Figura	Titulo Presentacion	Vista
Presentacion6.xml	6.2	Figura 1. La Herencia: Teoría (1)	Introducción a la Herencia	
Presentacion6.xml	6.2	Figura 2. La Herencia: Teoría (1)	Introducción a la Herencia	
Presentacion6.xml	6.6	Figura 3. Subclases en Java(1)	Introducción a la Herencia	
Presentacion6.xml	6.6	Figura 4. Subclases en Java(2)	Introducción a la Herencia	
Presentacion6.xml	6.7	Figura 5. Conversión de tipos (1)	Introducción a la Herencia	

Aunque se desarrollaron e implementaron algunas otras consultas por cuestiones de espacio no han sido incluidas en el presente artículo.

Conclusiones

Con el trabajo descrito en este artículo se demostró que existe una forma diferente de gestionar dinámicamente la información registrada en los contenidos educativos, haciendo uso de las tecnologías XML. Convirtiendo para ello cada presentación de una clase magistral, representada como un documento digital PDF, en un contenido reutilizable representado por un documento XML (validado por su respectivo *XMLS*). Posteriormente se logra la gestión dinámica de la información registrada en tales contenidos, la formulación de consultas en el lenguaje *XQuery*. El trabajo desarrollado, puede ser trasladado a cualquier clase de documento PDF que simbolice una presentación power point, independiente del dominio. La conversión realizada, presenta además algunas otras ventajas comparativas, frente a los tradiciones documentos PDF: (i) presenta una información estructurada, (ii) separa la capa de presentación del contenido, (iii) cumpliendo fielmente con las dos características anteriores, el nuevo esquema de representación permite la fácil adaptación a cualquier cambio al interior del contenido, (iv) facilita la gestión dinámica de la información, ya sea con el uso de lenguajes como XQuery o con la interoperabilidad con entornos de desarrollo de software como JAVA, y (v) finalmente, permite generar contenidos reutilizables.

Agradecimiento

El trabajo descrito en este artículo, hace parte del trabajo realizado dentro de la clase de

recuperación de información en la web, impartida para el curso de maestría en ingeniería de sistemas, de la Universidad Nacional de Colombia sede Medellín y constituye un primer aporte del proyecto de investigación "Un Modelo de Planificación Incremental para la Composición Dinámica de Rutas de Aprendizaje en Ambientes Virtuales Inciertos y Heterogéneos" adscrito a la misma Universidad.

Referencias

Boag, S.; Chamberlin, D.; Fernández, M.; Florescu, D.; Robie, J. y Siméon, J. (2003). XQuery 1.0: An XML Query Language. World Wide Web Consortium, W3C Working Draft, 22 August.

Carvin, A. (2006). The semantic web and the online educational experience. Learning.now. Consultado en septiembre de 2010, en: http://www.pbs.org/teachersource/learning.now/2006/11/the_semantic_web_and_the_onlin.html.

Document Type Definition, DTD, (2001). Consultado en agosto de 2010, en http://www.comptechdoc.org/independent/web/dtd/.

Draper, D.; Fankhauser, P.; Fernández, M.; Malhotra, A.; Rose, M.; Rys, M.; Simeon, J.; Wadler, P. (2003). XQuery 1.0 and XPath 2.0 Formal Semantics. W3C Working Draft, 22 August.

Hernández P. (2007). Tendencias de Web 2.0 aplicadas a la educación en línea. No Solo Usabilidad journal, 6. ISSN 1886-8592.

IMS Learning Design Specification. IMS Global Learning Consortium, Burlington USA (2003). Consultado septiembre de 2010 en http://www.imsglobal.org/learningdesign.

Instituto Politécnico Nacional de México, IPNM. (2009). Secretaria de apoyo académico. Dirección de Tecnología Educativa. México. Consultado en agosto de 2010 en http://www.te.ipn.mx/webT3/dte/DocuemntosBase/pro10doc/

Martínez, A.I. y Méndez, R. (2002). Integrating process modeling and simulation through reusable models in XML. Proceedings of the Summer Computer Simulation Conference 2002. The Society for Modeling and Simulation International, pp. 452-460.

Morales, R. y Agüera, A. S. (2006). Un marco teórico para el desarrollo de sistemas inteligentes de capacitación basados en tecnología web. Memorias de la IEEE ROC&C'2001: 12ª Reunión de Otoño de Comunicaciones, Computación, Electrónica y Exposición Industrial, 2001, CP–20.

Rodriguez, I., J.; Escofet, Ana y Azzato, Mariela (2005). Un sistema abierto para la creación de contenidos educativos digitales. Revista de Educación a Distancia RED, Consultada en octubre de 2010 en http://www.um.es/ead/red/M4/rodriguez41.pdf

Sarramona, J. (2008). Teoría de la educación. 2ª Edición. ISBN: 978-84-344-2670-2. España, Editorial Book Print Digital.

Schmidt, A.; Waas, F.; Kersten, M.; Florescu, D.; Manolescu, I.; Carey, M. y Busse, R. (2002). XMark: a benchmark for XML data management. Proceedings of International Conference on Very Large Data Bases (VLDB'02), Hong Kong, pp. 974-985

XML (2004). Extensible Markup Language. World Wide Web Consortium. Consultado en septiembre de 2010 en http://www.w3.org/XML.

XMLS (2006). Extensible Markup Language Schema. Consultado en agosto de 2010, en http://www.w3.org/XML/Schema. W3C.

Sobre los autores

Ingrid Durley Torres
Miembro Grupo de Investigación SINTELWEB e Inteligencia Artificial en Educación. Universidad Nacional de Colombia Facultad de Minas, sede Medellín, Colombia.
idtorresp@unal.edu.co

Jaime A. Guzmán Luna
Director Grupo de Investigación SINTELWEB. Universidad Nacional de Colombia Facultad de Minas, sede Medellín, Colombia.
jaguzman@unal.edu.co

Jovani Alberto Jiménez Builes
Director Grupo de Investigación Inteligencia Artificial en Educación. Universidad Nacional de Colombia Facultad de Minas, sede Medellín, Colombia.
jajimen1@unal.edu.co