



Desarrollo de aplicaciones Web con XML y Java

L. M. Dorantes, T. García, C. F. García

Instituto Tecnológico de San Juan del Río
Av. Tecnológico No. 2, Col Centro.
San Juan del Río, Qro., México

Universidad Autónoma de Querétaro
Av. Universidad S/N, Col. Las Campanas
Querétaro, Qro., México

Resumen : La evolución de la Internet y de las comunicaciones vía Web, permite nuevos escenarios en lo relacionado con el acceso a contenidos, esto ha propiciado el surgimiento de nuevas tecnologías para el desarrollo de aplicaciones web cuya filosofía se basa en la separación entre lógica, interfaz y origen de datos. Como parte de las tecnologías que dan soporte a las necesidades reales y actuales de los servicios Web y plataformas de comercio electrónico están por un lado la tecnología Java y por otro XML (eXtensible Markup Language) que juntos potencian la portabilidad de datos, de código y simplifican el desarrollo de servicios web. Las aplicaciones basadas en XML pueden procesar documentos XML, aplicar su lógica de negocio o recuperar información y generar documentos XML resultantes. Para lo anterior se requiere de herramientas de procesamiento como son SAX (Simple API for XML), DOM (Document Object Model) y XSLT (XML Style Sheet Language Transformations) e incluso usar un modelo de documento como JDOM para representar las estructura de datos principales y aplicarles la lógica de negocio. La interfaz de programación de aplicaciones Java para procesar documentos XML proporciona al desarrollador Java estas herramientas.

Abstract: The evolution of the Internet and communications through Web, allows new scenes in the related to the access to contents, this has caused new technologies for the development of Web applications whose philosophy is based on the separation among logic, interface and data origin. As part of these technologies which give support to the real and present necessities of Web services and platforms of e-commerce is on the one hand the java technology and by other XML (Extensible Markup Language) that

together harness portability of data, code and simplify the Web services development. The XML based applications can process Xml documents, applying their business logic or information recovery and generation of XML documents. In order to the previous is required processing tools as SAX(Simple for API XML), DOM (Document Object Model) and XSLT (XML Style Sheet Language Transformations) and even use a document model like JDOM, to represent main's data structure in order to be applied to the business logic. Programming java applications interface to process XML Documents, provide the tools to the java developer.

Key words: Web, XML, SAX, DOM, XSLT, JDOM, Java.

Introducción

La evolución de la Internet y de las comunicaciones vía Web, permite nuevos escenarios en lo relacionado con el acceso a contenidos. Su filosofía se basa en la separación entre lógica, interfaz y origen de datos. Es por ello que las nuevas tecnologías para el desarrollo de servicios web están enfocadas a este modelo, de tal forma que la información se formatea al momento de ser requerida por un dispositivo específico [1].

La parte más compleja en el desarrollo de servicios web es la programación de la infraestructura para el acceso a los datos como son las capacidades de seguridad y de mensajería, el manejo de transacciones distribuidas y el control de almacenes de conexiones. Otra dificultad es que estos servicios web deben ser capaces de manejar un enorme número de usuarios simultáneamente, por eso las aplicaciones deben ser altamente escalables [3].



Como parte de las tecnologías que dan soporte a las necesidades reales y actuales de los sistemas de aplicación Web y plataformas de comercio electrónico, están por un lado la tecnología Java y por otro XML que juntos potencian la portabilidad de datos, de código y simplifican el desarrollo de servicios web. Hoy en día se ofrecen al menos cinco extensiones de la plataforma Java que permiten la construcción de aplicaciones basadas en XML, como son las API's JAXP, JAXB, JAXM, JAX RPC y JAXR, las cuáles soportan los estándares de la industria, lo que asegura la interoperabilidad [1-3].

Antecedentes

Desde la aparición de la WWW, se han desarrollado varias tecnologías para la implementación y mejora de las aplicaciones Web, buscando agilizar la funcionalidad y resolver los diferentes problemas que plantean las diversas tecnologías usadas. Inicialmente, las aplicaciones estaban estructuradas alrededor de documentos HTML con una estructura de hipertexto, que permitía recorrer los diferentes documentos que componían la aplicación. La estructura basada en documentos HTML se ha mejorado añadiendo estándares para el intercambio y almacenamiento de información [2-4], códigos de lenguajes de programación y de lenguajes de scripting o de marcación mediante diferentes tecnologías, para aumentar la funcionalidad de la aplicación.

XML es un estándar industrial para representar datos, independientes del sistema. Al igual que HTML, XML encierra los datos en etiquetas, pero hay importantes diferencias entre los dos lenguajes de marcas. La primera es que, las etiquetas XML tienen relación con el significado del texto que encierran, mientras que las etiquetas HTML especifican cómo mostrar el texto encerrado. La segunda diferencia, es que las etiquetas XML son extensibles, permitiendo escribir etiquetas propias para describir el contenido de una aplicación determinada. Para la creación de etiquetas se usa un esquema de lenguaje XML, el cual describe la estructura de un conjunto de documentos XML y puede usarse para limitar los contenidos de los documentos XML[2]. Uno de los lenguajes de esquemas más usado es el DTD (Document Type Definition), en el cual se describen las reglas para procesar los datos.

Por otro lado Java es un lenguaje de programación de propósito general orientado a objetos, independiente de plataformas, con soporte incorporado a Internet para facilitar la conexión a servidores web. Permite la escritura de páginas web a través del *servlet* que proporciona los mecanismos para que las aplicaciones web interactúen con clientes externos y con otras aplicaciones web. Java facilita el acceso a bases de datos a través del JDBC (Java Database Connectivity) en distintas plataformas y la posibilidad de interactuar con otras bases de datos. Otra herramienta importante en Java son los JavaBeans, que es un modelo de componentes de software, que agiliza el proceso de desarrollo de una aplicación al combinar componentes de software existentes [5-6].

Desarrollo

Las APIs de Java para XML permiten escribir aplicaciones web completamente en el lenguaje Java. Se clasifican en dos grupos las que tratan directamente con documentos XML, como el JAXP para procesar documentos XML usando varios analizadores y el JAXB para mapear elementos XML a clases del lenguaje Java. Y los que tratan con procedimientos, como el JAXM para enviar mensajes SOAP de una forma estándar, el JAXR que proporciona una forma estándar para acceder registros de negocios que comparten información y el JAX-RPC que envía llamadas a métodos SOAP remotos sobre Internet y recibe información [5].

El API Java para Proceso de XML (JAXP) hace fácil el proceso de datos XML con aplicaciones escritas en el lenguaje Java. JAXP contiene los analizadores estándares SAX (API Simple para XML) y DOM (Modelo de Objetos del Documento) para poder analizar los datos como flujos de eventos y construir una representación de objetos con ellos. JAXP es flexible, permitiendo utilizar cualquier analizador compatible con XML desde una aplicación, esto lo hace a través de la capa de conectividad, que permite enlazar una implementación de los APIs SAX o DOM, o enlazar a un procesador XSL lo que permite controlar la forma en que se muestran los datos. También proporciona soporte para espacios de nombres, permitiendo trabajar con DTD.



Un archivo XML por sí solo no realiza ninguna función, por lo que es importante utilizar o trasladar la información a un programa Java, Perl o una aplicación de servidor a través de analizadores o parsers que son los encargados de llevar a cabo la transformación. Así, SAX es un analizador basado en eventos lo que significa que lee un documento XML desde el principio hasta el final, y cada vez que reconoce una sintaxis de construcción se lo notifica a la aplicación que lo está ejecutando. El analizador SAX puede realizar validación mientras analiza los datos XML, lo que significa que verifica si los datos siguen las reglas especificadas en el DTD de los documentos XML. Por otro lado, el analizador DOM que es un conjunto de interfaces para construir una representación de objeto, en forma de árbol, de un documento XML analizado, permite el acceso aleatorio a piezas de datos particulares de un documento XML, también permite añadir nuevos elementos o eliminar elementos existentes. DOM requiere tener el modelo del objeto del documento completo en memoria, lo que generalmente es menos eficiente para búsquedas que implican varios ítems en documentos largos.

En el 2000 surge JDOM para manejo optimizado de datos XML con Java. Es un API que se basa en árbol para crear, analizar y manipular documentos XML. JDOM representa un documento XML como un árbol completo disponible todo el tiempo y compuesto por elementos, atributos, comentarios, instrucciones, nodos y secciones entre otros. JDOM puede acceder y modificar cualquier parte del árbol en cualquier momento y los diferentes tipos de nodos del árbol son representados por clases concretas en lugar de interfaces. JDOM depende de un parser de SAX con un manejador de contenido común para analizar documentos y construir modelos o árboles JDOM a partir de estos y así evitar la sobrecarga de construir el árbol en memoria dos veces en dos representaciones diferentes[6].

Resultados

En este artículo, se han expuesto las principales tecnologías disponibles para procesar documentos XML. Estas tecnologías se dirigen a diferentes niveles de abstracción y proporcionan diferentes niveles de utilización para el programador Java. Algunas de estas

tecnologías como SAX, DOM y JDOM podrían relacionarse unas con otras. Por ejemplo, un constructor de documentos DOM podría usar un analizador SAX para generar un árbol DOM desde un documento XML. Como se mezclan, un desarrollador de aplicaciones XML podría verse tentado a elegir una de ellas para conseguir el mismo resultado, pero hacer esto traerá dificultades y pérdidas entre el rendimiento, el uso de memoria y la flexibilidad.

Por ejemplo SAX podría traer eficiencia durante el mapeo de datos XML a objetos de negocio Java. DOM y especialmente JDOM podrían traer simplicidad y eficiencia en la edición de grandes modelos de documentos en memoria. En algunos casos, un modelo de objeto de documento podría ser elegible como la estructura de datos principal de la aplicación.

Sin embargo, todos estos APIs necesitan ser considerados cuando implementamos una aplicación XML. Además con las nuevas versiones de SAX y la intercambiabilidad (como fuente y resultado) de SAX y DOM en algunos APIs, se pueden construir complejas redes de procesamiento XML combinando implementaciones estándares y personalizadas.

En el siguiente ejemplo podemos comparar la simplicidad de JDOM frente al equivalente DOM, para crear un documento XML con un único elemento:

```
<?xml version="1.0"?>
<saludo>Hola</saludo>
```

Versión JDOM

```
Element root = new Element("saludo");
root.setText("Hola");
Document doc = new Document(root);
```

Versión DOM

```
DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
DocumentBuilder builder =
factory.newDocumentBuilder();
DOMImplementation impl =
builder.getDOMImplementation();
Document doc =
impl.createDocument(null, "saludo",
null);
```



```
Text text =  
doc.createTextNode("Hola");  
doc.appendChild(text);
```

Cómo podemos observar en el ejemplo anterior, JDOM hace el trabajo del programador más fácil.

Conclusiones

Las tecnologías XML se usan cada día más en servicios web y hoy en día con los APIs de Java para XML, se hace sencillo implementar servicios web y escribir aplicaciones que sean clientes de servicios web.

De acuerdo a las características mostradas en este documento podemos comparar DOM y JDOM ya que SAX es de sólo lectura. JDOM es más eficiente en el manejo de memoria ya que no requiere usar factorías ni otros modelos avanzados de programación para crear un documento, define clases lo que hace más fácil la programación. En cambio, DOM no tiene una manera estándar de crear documentos, cada parser tiene implementados sus métodos específicos, siendo los programas dependientes del parser utilizado y trabaja a base de interfaces, generando con esto más código (ver figura 1).

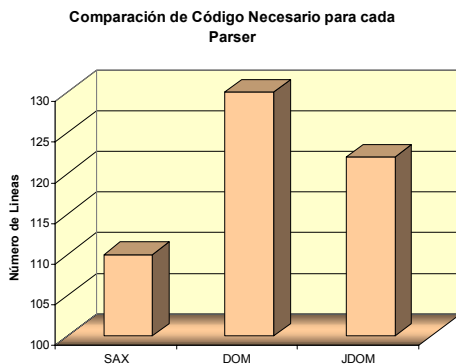


Figura 1. Comparacion de numero de líneas codificadas para cada parser.

Finalmente, de acuerdo a los resultados expuestos podemos concluir que JDOM facilita la programación requiriendo de menos líneas de código que con DOM,

por lo que puede reducir el esfuerzo del programador en aproximadamente un 10 por ciento en la primer aplicación y posteriormente se reduce más al reutilizar código a través de las clases. JDOM es más rápido cargando y manipulando documentos, y los requerimientos de memoria son menores. Por lo tanto, JDOM es ligeramente más rápido que DOM usando la misma implementación (ver figura 2). El tiempo de procesamiento de un documento XML depende del parser usado y del tamaño del documento, teniendo un comportamiento lineal conforme se incrementase el tamaño del documento, ya que entre más información se tenga el número de nodos generados en el árbol es mayor y por tanto el tiempo de procesamiento crece proporcionalmente.

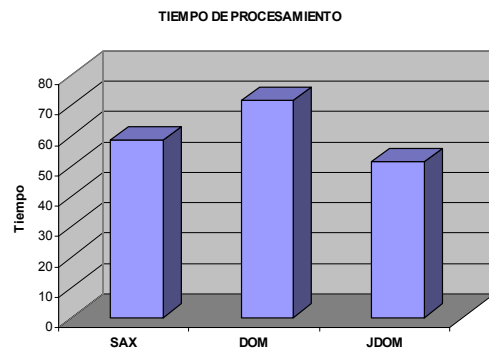


Figura 2: Tiempo de procesamiento para JDOM, SAX y DOM.

Referencias

1. Mayurama Hiroshi, Tamura Kent, Uramoto Naohiko. "Sitios Web con XML y Java". Prentice Hall, 2000.
2. Gutiérrez Abraham, Martínez Raúl. "XML a través de Ejemplos". Alfaomega RA-MA. 2001
3. McLaughlin Brett, "Java y XML", Ediciones Anaya Multimedia O'reilly (Grupo Anaya, S.A.), Madrid, 2001.
4. Marchal Benoit, "XML con Ejemplos", Ediciones Pearson Educación de Mexico, S.A. de C.V., Mexico, 2001.
5. Griffin John, "Creación de sitios WEB con XML y SQL Server 2000", Ediciones Pearson Educación, S.A., Madrid, 2002.
6. <http://java.sun.com/xml>



Luz María Dorantes Hernández

Egresada del Instituto Tecnológico de San Juan del Río, Qro. Docente del ITSJR y alumna de la maestría en Ingeniería de Software Distribuido de la Universidad Autónoma de Querétaro. Participación en concursos de creatividad asesorando a estudiantes, en proyectos académicos, en la organización de eventos académicos al interior del instituto.

Ma. Teresa García Ramírez

Maestra Tiempo Completo de la Facultad de Informática de la Universidad Autónoma de Querétaro, participación en cuerpos colegiados, en revisiones curriculares, en congresos como ponente ANIEI, CIIIEE, proyectos de investigación en el área de sistemas distribuidos y aplicaciones en Internet.

Carlo Franco García Sánchez

Egresado en 2005, actualmente estudiante de la maestría en Ingeniería de Software Distribuido, participación en congresos del ANIEI, CIIIEE. Colaborador en proyectos de investigación. Tesis para obtener el título de Ingeniero en Computación “Desarrollo de páginas dinámicas (JSP) en Internet utilizando Apache”.