XML Y SU PAPEL EN LA INTEGRACION DE SERVICIOS

DIANA CAROLINA BENEDETTI ALJURE T00014515

SANDRA MILENA CORTES MONTALVO T00014521

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR

PROGRAMA DE INGENIERA DE SISTEMAS

MINOR EN APLICACIONES DISTRIBUIDAS

CARTAGENA DE INDIAS, D. T Y C

2010

XML Y SU PAPEL EN LA INTEGRACION DE SERVICIOS

DIANA CAROLINA BENEDETTI ALJURE T00014512

SANDRA MILENA CORTES MONTALVO T00014521

Monografía presentada como requisito para optar el título de Ingeniera de Sistemas.

Asesor

Giovany Vásquez Mendoza

Ingeniero Sistemas

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR

PROGRAMA DE INGENIERÍA DE SISTEMAS

MINOR DE APLICACIONES DISTRIBUIDAS

CARTAGENA DE INDIAS, D. T Y C

2010

Nota de Aceptación

_____

_____

_____

_____

_____

Firma del Presidente del Jurado

_____

Jurado

_____

Jurado

Cartagena de Indias D.T. y C., 10 de Mayo de 2010

Cartagena de Indias D.T. y C., 10 de Mayo de 2010

AUTORIZACIÓN

Yo, **DIANA CAROLINA BENEDETTI** identificada con cedula de ciudadanía 1.047.380.879 de Cartagena, autorizo a la UNIVERSIDAD TECNOLÓGICA DE BOLIVAR para hacer uso del trabajo de grado titulado **"XML Y SU PAPEL EN LA INTEGRACION DE SERVICIOS"** y publicarlo en el catálogo On Line de la Biblioteca.

_____

DIANA CAROLINA BENEDETTI ALJURE

C.C. 1.047.380.879 de Cartagena

Cartagena de Indias D.T. y C., 10 de Mayo de 2010

AUTORIZACIÓN

Yo, **SANDRA MILENA CORTES MONTALVO** identificada con cedula de ciudadanía 1.128.056.523 de Cartagena, autorizo a la UNIVERSIDAD TECNOLÓGICA DE BOLIVAR para hacer uso del trabajo de grado titulado **"XML Y SU PAPEL EN LA INTEGRACION DE SERVICIOS"** y publicarlo en el catálogo On Line de la Biblioteca.

_____

SANDRA MILENA CORTES MONTALVO

C.C. 1.128.056.523 de Cartagena

Cartagena de Indias D.T. y C., 10 de Mayo de 2010

Señores,

COMITÉ EVALUADOR DE TRABAJOS DE GRADO

PROGRAMA DE INGENIERÍA DE SISTEMAS

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR

La ciudad

En mi calidad de asesor de la monografía titulada **"XML Y SU PAPEL EN LA INTEGRACION DE SERVICIOS",** elaborada por DIANA BENEDETTI ALJURE Y SANDRA MILENA CORTES MONTALVO, manifiesto que he participado en la orientación desarrollo del proyecto en toda y cada una de sus etapas y por consiguiente estoy totalmente de acuerdo con los resultados obtenidos.

_____

GIOVANNY RAFAEL VASQUEZ MENDOZA

Asesor del Proyecto

Cartagena de Indias D.T. y C., 10 de Mayo de 2010

Señores,

COMITÉ EVALUADOR DE TRABAJOS DE GRADO

PROGRAMA DE INGENIERÍA DE SISTEMAS

UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR

La ciudad

Presentamos a consideración la monografía titulada **"XML Y SU PAPEL EN LA INTEGRACION DE SERVICIOS",** como requisito para optar por el título de ingeniero de sistemas.

Cordialmente,

_____ _____

DIANA BENEDETTI ALJURE SANDRA MILENA CORTES MONTALVO

AGRADECIMIENTOS

A Giovany Vásquez como asesor de este proyecto.

A Rafael González, Edwin Puerta y Fernando Alcalá por ofrecernos su experiencia en el desarrollo Web, para la construcción de esta monografía.

En especial queremos agradecerles a nuestros compañeros (Vichu, Felix, Dex, Leo, Jhon, José, Moise) que durante esta etapa de nuestras vidas que hoy dejamos atrás, fueron unas de las razones para estar seguras que estudiar Ingeniería de Sistemas fue una de las mejores decisiones que tomamos para nuestras vidas, porque no solo fueron compañeros si no verdaderos amigos, donde esperamos seguir contando siempre, como amigos y colegas. LOS QUEREMOS.

DEDICATORIA

Espere tanto tiempo para escribir esta dedicatoria, que ahora que **POR FIN** no tengo ni la más mínima idea que decir, pero si tengo claro las personas a quienes quiero dedicar esta meta:

A Dios por ser el motor de mi vida y corazón, porque cada mañana le pido su bendición y hoy estoy aquí culminando una etapa de mi vida, con la frente en alto y con Fe en EL que va a seguir guiándome en cada una de las etapas siguientes.

A mis papás por ser mi camino, porque sé que sin ustedes este nuevo triunfo no hubiese sido posible, disculpen la demora; pero aquí está con el amor más grande del mundo para ustedes.

A mis hermanitos por ser el ejemplo de mi vida, porque aunque gran parte de esta etapa estuve lejos de ustedes, sabía que tenía un apoyo incondicional por parte de mis hermanos.

A Sandra por ser mi compañera, amiga y colega; creo que las palabras para ti en estos momentos quedan cortas, fueron muchos los buenos y malos momentos que pasamos durante todo este tiempo y traerlos aquí seria nunca terminar; espero y seria todo un placer seguir compartiendo etapas de nuestras vidas juntas.

A Vichu, Felix, Dex, Leo, Jhon, José, Moise por ser el motivo de ir a estudiar cada día y no arrepentirme de carrera, saben que sin ustedes no hubiese sido igual, a todos los llevo en mi corazón como un bonito recuerdo.

Menos mal que no tenía idea de que decir, a todas la personas que pasaron por esta etapa, **MUCHAS GRACIAS**.

DIANA CAROLINA BENEDETTI ALJURE

DEDICATORIA

Primero que todo quiero agradecer a Dios por darme la oportunidad de estar aquí culminando este ciclo tan importante como lo es mi formación profesional.

A mi mamá por estar siempre a mi lado, por ser padre y madre a la vez, por no desfallecer, por sacarme adelante, por ser mi todo.

A mi Tía Jaque por sus enseñanzas, por sembrar en mi la semilla de la responsabilidad, con habito de estudios tan excelentes que me han permitido ser hoy por hoy la estudiante y profesional que soy.

A mi familia, porque cada uno de ustedes a aportado un granito de arena en mi formación profesional y personal.

A Diana, porque hemos recorrido este largo camino juntas, por tu apoyo, por el tiempo que hemos compartido, porque este logro es de la dos. TE LO MERECES!!

A Cristian, porque eres mi complemento, porque me ayudas a salir adelante cuando siento que desfallezco, porque has estado ahí durante este proceso.

A mis amigos de la UTB, porque gracias a Ustedes este camino ha sido posible, divertido y enriquecedor.

Finalmente, gracias a todas aquellas personas que me apoyaron y que no menciono porque no me alcanzaría la hoja de este trabajo para mencionarlos.

GRACIAS y este TRIUNFO ES DE USTEDES.

SANDRA MILENA CORTES MONTALVO

Artículo 105: La Universidad Tecnológica de Bolívar se reserva el derecho de propiedad de todos los trabajos de grado aprobados y no pueden ser explotados comercialmente sin su autorización.

CONTENIDO

LISTADO DE FIGURAS

INTRODUCCION

Como es sabido HTML (Hypertex Markup Language) se ha convertido en un lenguaje de inmensa popularidad durante los últimos años. También se puede notar que la comunidad ha encontrado sus propias limitaciones, que algunas de las cuales se han querido corregir con scripts, java scripts, Active X, HTML dinámico, etc.; pero en la realidad todas esas herramientas no aportan una solución global a las limitaciones del HTML. XML es una tecnología sencilla que tiene a su alrededor otras tecnologías que la complementan y la hacen mucho más grande, con posibilidades y utilidades mucho mayores a las de HTML. Con todas las tecnologías relacionadas, representa una manera distinta de hacer las cosas, más avanzada, cuya principal novedad consiste en permitir compartir los datos con los que se trabaja a todos los niveles, por todas las aplicaciones y soportes. Así pues, XML juega un papel importantísimo en este mundo actual, que tiende a la globalización y la compatibilidad entre los sistemas, ya que es la tecnología que permitirá compartir la información de una manera segura, fiable, fácil. Además, XML permite al programador y los soportes dedicar sus esfuerzos a las tareas importantes cuando trabaja con los datos, ya que algunas tareas tediosas como la validación de estos o el recorrido de las estructuras corre a cargo del lenguaje y está especificado por el estándar, de modo que el programador no tiene que preocuparse por ello.

Con esta monografía se analizará la utilización de XML como formato estándar para el intercambio de datos y la forma cómo este lenguaje de marcas proporciona una manera para describir datos estructurados, facilitando realizar declaraciones más precisas de contenido y permitiendo obtener resultados de búsquedas con más significado. Además la funcionalidad de XML en la integración de Servicios WEB, habilitando la manipulación y visualización de datos a través de la WEB.

OBJETIVOS

OBJETIVO GENERAL

Desarrollar un análisis para identificar las ventajas y características de XML dentro de la programación, como lenguaje estándar que sustituya todo conjunto de tecnologías que permitan acceder a la información a través de la WEB.

OBJETIVOS ESPECÍFICOS

- Recopilar, seleccionar, organizar y analizar la información, para así establecer los puntos y características principales de la implementación de XML
- Comprender las características de XML y como estas pueden contribuir a la integración de servicios a través de la WEB.
- Explicar mediante ejemplos ilustrativos, como las características pueden utilizarse para la integración de servicios, datos y aplicaciones.

1. PROPUESTA DE INVESTIGACIÓN

1.1 PLANTEAMIENTO DEL PROBLEMA

1.1.1 Antecedentes

Los Servicios Web son aplicaciones modulares auto descriptivas. El aspecto más interesante de los Servicios Web es como se puede acceder a ellos independientemente de la plataforma, lenguaje o modelo de objetos que utilice, apareciendo entonces el tópico de Integración de Servicios.

Entre los principales beneficios que se exponen al hablar de los servicios web, generalmente se encuentran aquellos que tienen que ver con su granularidad e interoperabilidad, es decir, con la posibilidad de desarrollar componentes de software totalmente independientes que tienen funcionalidad propia, pero que son capaces de exponer tal funcionalidad y compartirla con otros servicios y aplicaciones para lograr crear sistemas más complejos.

Desde el punto de vista de los negocios, los servicios web permiten que las organizaciones integren sus diferentes aplicaciones de una manera eficiente, sin preocuparse por cómo fueron construidas, donde residen, sobre qué sistema operativo se ejecutan o cómo acceder a ellas. Precisamente por esta razón es que los servicios web se consideran integradores, porque permiten crear una interfaz de acceso a las aplicaciones, sin importar las características de implementación de éstas. El problema surge cuando los servicios web dentro de la organización se multiplican. En este punto se hace necesario desarrollar conectores que asilen los servicios para su mayor integración.

Una solución es la Utilización de XML posibilitando la comunicación entre sistemas sobre cualquier protocolo. Es decir, se convierte en una pasarela, que se encarga de traducir de un lenguaje a otro, facilitando el intercambio entre plataformas de datos.

1.1.2 Justificación

Las organizaciones actuales tienen un gran inventario de aplicaciones que soportan sus operaciones de negocio. Muchas de esas aplicaciones están desarrolladas en lenguajes de programación distintos y corren en ambientes diferentes se convierten en grandes silos. Enfoques modernos de desarrollo de software sostienen que dividir las aplicaciones en servicios, permite con el tiempo bajos tiempos y costos de desarrollo y, por consiguiente, respuestas más agiles a las demandas del mercado.

Una de las tecnologías más aceptadas para el desarrollo de aplicaciones orientadas al servicio son los servicios web, está soportada en XML.

Los sistemas empresariales requieren interoperar para ser útiles a las exigencias de las organizaciones y XML resulta ser adecuado para integrar aplicaciones.

Estos temas no son cubiertos en cursos normales de ingeniería de software a nivel de pregrado y la comunidad estudiantil requiere de recursos que le ayuden a comprender y poner en práctica estos conceptos. Este trabajo pretender ser uno de esos recursos que exige la comunidad para comprender y aplicar XML en la integración de servicios, aplicaciones y datos.

1.1.3 Metodología

El tipo de investigación a aplicar para el desarrollo de la monografía es Revisión de Tema. Ya que se explorará acerca del Lenguaje XML y sus características, permitiendo comprobar e ilustrar la forma en permite la Integración de Servicios, datos y aplicaciones.

2. SERVICIOS WEB

En la literatura sobre el tema, ver [1], [2], se proponen definiciones conceptuales sobre los servicios web. En este documento, se propone la definición presentada por la organización que estandariza W3C, la cual señala que "Un Servicio Web es un conjunto de protocolos y estándares que permiten intercambiar datos entre aplicaciones de software desarrolladas en distintos lenguajes de programación y ejecutadas sobre cualquier plataforma.

Para proporcionar interoperabilidad y extensibilidad entre estas aplicaciones, y que al mismo tiempo sea posible su combinación para realizar operaciones complejas, es necesaria una arquitectura de referencia estándar[3]."

Los siguientes son algunos de los estándares utilizados:

- Web Services Protocol Stack
- XML (Extensible Markup Language)
- SOAP (Simple Object Access Protocol) o XML-RPC (XML Remote Procedure Call
- WSDL (Web Services Description Language
- UDDI (Universal Description, Discovery and Integration)
- WS-Security (Web Service Security)

2.1 RAZONES PARA USAR SERVICIOS WEB

Diversos autores como [4], [5], plantean razones bien justificadas para usar los servicios web las cuales se resumen así:

---

[1]XML Practico. Bases esenciales, conceptos y casos prácticos. Sebastián Lecompte. Eni Ediciones
[2] Servicios WEB XML de Microsoft .NET. Prentice HALL- Robert Tabor.
[3] http://www.w3c.es/Divulgacion/Guiasbreves/ServiciosWeb
[4]Estado del Arte: Servicios Web. Universidad Nacional de Colombia. Carlos Andres Machuca
[5]Servicios WEB: Distribución e integración. Universidad Icesi. Liliana M. Arboleda C.

- Se basan en HTTP sobre TCP (Transmission Control Protocol) en el puerto 80. Dado que las organizaciones protegen sus redes mediante firewalls -que filtran y bloquean gran parte del tráfico de Internet-, cierran casi todos los puertos TCP salvo el 80, que es, precisamente, el que usan los navegadores.
- Antes de que existiera SOAP, no había buenas interfaces para acceder a las funcionalidades de otros ordenadores en red. Las que había eran ad hoc y poco conocidas, tales como EDI (Electronic Data Interchange), RPC (Remote Procedure Call), u otras APIs.
- Pueden aportar gran independencia entre la aplicación que usa el servicio Web y el propio servicio. De esta forma, los cambios a lo largo del tiempo en uno no deben afectar al otro. Esta flexibilidad será cada vez más importante, dado que la tendencia a construir grandes aplicaciones a partir de componentes distribuidos más pequeños es cada día más utilizada.

  El autor [6], propone el siguiente esquema de funcionamiento para los servicios web:

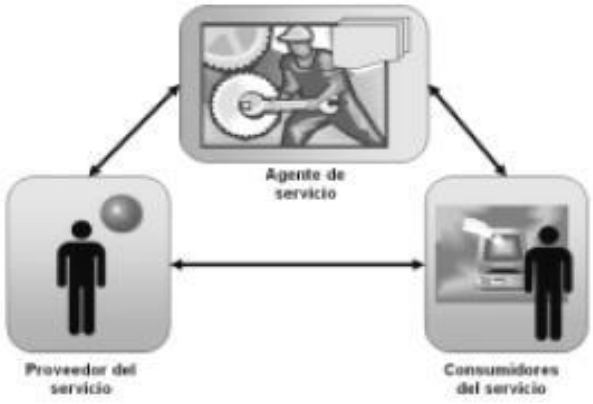

FIGURA 1. ELEMENTOS QUE INTERACTÚAN EN LOS SERVICIOS WEB[6]

[6] Servicios WEB: Distribución e integración. Universidad Icesi. Liliana M. Arboleda C.

Requiere de tres elementos fundamentales:

1. Un **proveedor del servicio web**, que es quien lo diseña, desarrolla e implementa y lo pone disponible para su uso, ya sea dentro de la misma organización o en público.
2. Un **consumidor del servicio**, que es quien accede al componente para utilizar los servicios que éste presta.
3. Un **agente de servicio**, que sirve como enlace entre proveedor y consumidor para efectos de publicación, búsqueda y localización del servicio.

2.2 VENTAJAS Y DESVENTAJAS

Luego de realizar un análisis de las ventajas y desventajas propuestas por diversos autores, se resumen a continuación los puntos más relevantes:

2.2.1 Ventajas

- Aumenta la interoperabilidad entre programas independientemente de la plataforma en donde están instalados.
- Fomentan los estándares y protocolos basados en texto, haciendo más fácil acceder y entender su contenido y funcionamiento (pero, en general, produciendo una baja en su rendimiento).
- Al emplear HTTP, pueden utilizar un sistema firewall sin cambiar las reglas de filtrado.
- Los servicios Web fomentan los estándares y protocolos basados en texto, que hacen más fácil acceder a su contenido y entender su funcionamiento.
- Permiten que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan ser combinados fácilmente para proveer servicios integrados.

2.2.2 Desventajas

- No son tan desarrollados para realizar transacciones comparadas a otros sistemas como CORBA (Common Object Request Broker Architecture).
- Su rendimiento es bajo comparado con otros sistemas como CORBA, DCOM o RMI, especialmente por el uso de protocolos y estándares basados en texto.
- Para realizar transacciones no pueden compararse en su grado de desarrollo con los estándares abiertos de computación distribuida como CORBA (Common Object Request Broker Architecture).
- Al apoyarse en HTTP, pueden esquivar medidas de seguridad basadas en firewall cuyas reglas tratan de bloquear o auditar la comunicación entre programas a ambos lados de la barrera.

2.3 EJEMPLO DE SERVICIO WEB

La organización W3C propone el siguiente modelo, que puede ser considerado como un servicio WEB.

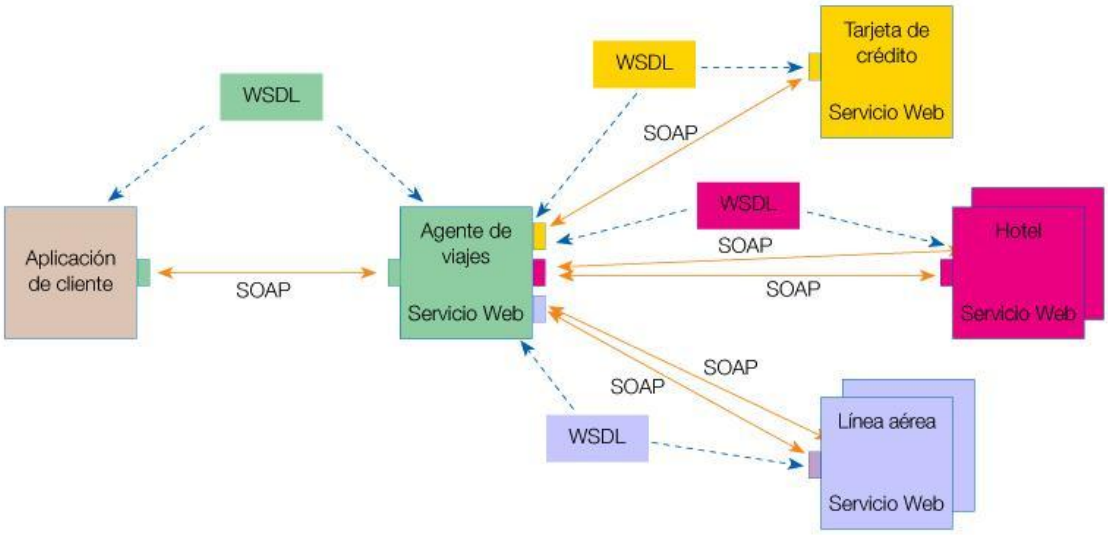

FIGURA 2. LOS SERVICIOS WEB EN FUNCIONAMIENTO

Según el gráfico, un usuario (que juega el papel de cliente dentro de los Servicios Web), a través de una aplicación, solicita información sobre un viaje que desea realizar haciendo una petición a una agencia de viajes que ofrece sus servicios a través de Internet. La agencia de viajes ofrecerá a su cliente (usuario) la información requerida. Para proporcionar al cliente la información que necesita, esta agencia de viajes solicita a su vez información a otros recursos (otros Servicios Web) en relación con el hotel y la compañía aérea. La agencia de viajes obtendrá información de estos recursos, lo que la convierte a su vez en cliente de esos otros Servicios Web que le van a proporcionar la información solicitada sobre el hotel y la línea aérea. Por último, el usuario realizará el pago del viaje a través de la agencia de viajes que servirá de intermediario entre el usuario y el servicio Web que gestionará el pago [7].

[7] http://www.w3c.es/Divulgacion/Guiasbreves/ServiciosWeb

3. LENGUAJE XML

3.1 HISTORIA

El XML proviene de un lenguaje que inventó IBM en los años 70, llamado GML (General Markup Language) y surgió por la necesidad que tenían en la empresa de almacenar grandes cantidades de información de temas diversos.

En el año 1989, dentro del ámbito de la red Internet, un usuario que había conocido el lenguaje de etiquetas (Markup) y los hiperenlaces creó un nuevo lenguaje llamado HTML, que fue utilizado para un nuevo servicio de Internet, la Web. Este lenguaje fue adoptado rápidamente por la comunidad y varias organizaciones comerciales crearon sus propios visores de HTML y lucharon entre ellos para hacer el visor más avanzado, inventándose sus propias etiquetas.

Desde el 1996 hasta hoy una entidad llamada W3C (World Wide Web Consortium) ha tratado de poner orden en el HTML y establecer sus reglas y etiquetas para que sea un estándar. Sin embargo el HTML creció de una manera descontrolada y no cumplió todos los requisitos que planteaba la sociedad global de Internet.

Fue entonces en el año 1998, donde W3C empezó con el desarrollo de XML, un lenguaje donde se ha pensado mucho, donde personas con grandes conocimientos en la materia están trabajando todavía en su gestación.

Se pretendía solucionar las carencias del HTML en lo que se respecta al tratamiento de la información, tales como:

- El contenido se mezcla con los estilos que se le quieren aplicar.
- No permite compartir información con todos los dispositivos, como pueden ser ordenadores o teléfonos móviles.
- La presentación en pantalla depende del visor que se utilice

3.1.1 Versiones

Actualmente existen dos versiones de XML:

- La primera, XML 1.0, fue definida inicialmente en 1998. Ha sido objeto de revisiones menores desde entonces y se encuentra actualmente en su quinta edición, publicada el 26 de noviembre de 2008. Es ampliamente aplicado y todavía se recomienda para uso general.
- El segundo, XML 1.1, se publicó inicialmente el 4 de febrero de 2004, el mismo día de XML 1.0 Tercera edición, y se encuentra actualmente en su segunda edición, publicada el 16 de agosto de 2006. Contiene características (algunas polémicas) que están destinadas a hacer más fácil de usar XML en algunos casos, sobre todo permitiendo el uso de caracteres de final de línea utilizado en las plataformas de EBCDIC, y el uso de guiones y caracteres ausentes de Unicode 3.2. XML 1.1 no es una muy amplia aplicación y se recomienda para uso exclusivo de quienes necesitan de sus características únicas.

  Antes del lanzamiento XML 1.0 quinta edición, difería de XML 1.1 en tener requisitos más estrictos para los caracteres disponibles para su uso en elementos y atributos e identificadores únicos: en las primeras cuatro ediciones de XML 1.0 los caracteres eran exclusivamente enumerados usando una versión específica de la estándar Unicode (Unicode 2.0 para Unicode 3.2.).

  El enfoque adoptado en la quinta edición de XML 1.0 y en todas las ediciones de XML 1.1 es que sólo ciertos caracteres están prohibidos en los nombres y todo lo demás está permitido, con el fin de acomodar el uso de caracteres del nombre adecuado en futuras versiones de Unicode.

Casi cualquier punto de código Unicode puede ser usado en los datos de carácter y valores de los atributos de un documento XML 1.0 o 1.1, aunque el carácter que corresponde al punto de código no esté definido en la versión actual de Unicode. En los datos de carácter y los valores de atributo, XML 1.1 permite el uso de los caracteres de control de XML 1.0, pero, por "robustez", la mayoría de los caracteres de control establecido en XML 1.1 deberán expresarse como referencias de caracteres numéricos (y # x7F al # x9F, que se habían permitido en XML 1.0, se encuentran en XML 1.1, incluso para ser expresado como referencias de caracteres numéricos). Entre los caracteres de control apoyado en XML 1.1, son dos códigos de salto de línea que deben ser tratados como espacios en blanco (espacios en blanco son los códigos de control único que se puede escribir directamente).

No ha habido discusión de un XML 2.0, aunque ninguna organización ha anunciado planes para trabajar en ese proyecto.

- XML-SW (SW para Skunk Works), escrito por uno de los principales desarrolladores de XML, contiene algunas propuestas para lo que es un XML 2.0 podría ser como: la eliminación de las DTDs de la sintaxis, la integración de los espacios de nombres, XML Base, y XML Information Set (conjunto de información) en el nivel de base.

El Consorcio World Wide Web también tiene un XML Binary Characterization Working Group que hace la investigación preliminar en los casos de uso y propiedades de un código binario del conjunto de información XML.

3.2 DEFINICION DE LENGUAJE

Muchas son las definiciones que existe sobre XML, aquí se muestra la definición desarrollada por Word Wide Web Consortium (W3C)

"XML es un Lenguaje de Etiquetado Extensible muy simple, pero estricto que juega un papel fundamental en el intercambio de una gran variedad de datos. Es un lenguaje muy similar a HTML pero su función principal es describir datos y no mostrarlos como es el caso de HTML. XML es un formato que permite la lectura de datos a través de diferentes aplicaciones."[8]

La edición de documentos XML persigue los siguientes objetivos:

- Distinguir el contenido y la estructura de los documentos de su presentación en papel o en pantalla.

- Hacer explícita su estructura y sus contenidos informativos.

- Crear documentos que puedan intercambiarse y procesarse con facilidad en sistemas informáticos heterogéneos.

Para alcanzar estos objetivos XML propone un formato en el que se intercalan marcas en el texto de los documentos con el objeto de distinguir las partes o elementos estructurales del mismo.

[8] http://www.w3c.es/divulgacion/guiasbreves/tecnologiasxml

3.3 VENTAJAS Y DESVENTAJAS DEL LENGUAJE

3.3.1 Ventajas

El Lenguaje XML se puede usar para infinidad de trabajos y aporta muchas ventajas en amplios escenarios. Veamos algunas ventajas del XML en algunos campos prácticos.

- Comunicación de datos. Si la información se transfiere en XML, cualquier aplicación podría escribir un documento de texto plano con los datos que estaba manejando en formato XML y otra aplicación recibir esta información y trabajar con ella. Esta ventaja es de gran importancia ya que permite la integración de servicios, datos y aplicaciones.
- Migración de datos. Si tenemos que mover los datos de una base de datos a otra sería muy sencillo si las dos trabajan en formato XML.
- Aplicaciones web. Hasta ahora cada navegador interpreta la información a su manera y los programadores del web tienen que hacer unas cosas u otras en función del navegador del usuario. Con XML se tiene una sola aplicación que maneja los datos y para cada navegador o soporte, teniendo una hoja de estilo para aplicarle el estilo adecuado. Si mañana nuestra aplicación debe correr en WAP solo tenemos que crear una nueva hoja de estilo.
- Los autores y proveedores pueden diseñar sus propios tipos de documentos usando XML.
- La información contenida puede ser más 'rica' y fácil de usar, porque las habilidades hipertextuales de XML son mayores que las de HTML.
- XML puede dar más y mejores facilidades para la representación en los visualizadores.

- La información será más accesible y reutilizable, porque la flexibilidad de las etiquetas de XML pueden utilizarse sin tener que amoldarse a reglas específicas de un fabricante.
- Elimina muchas de las complejidades de SGML, en favor de la flexibilidad del modelo, con lo que la escritura de programas para manejar XML será más sencilla que haciendo el mismo trabajo en SGML.
- Los archivos XML válidos son válidos también en SGML, luego pueden utilizarse también fuera de la Web, en un entorno SGML (una vez la especificación sea estable y el software SGML la adopte).
- XML impone una sintaxis más rígida para las etiquetas, que permite su proceso de forma más eficiente.
- XML permite definir lenguajes de etiquetas para cualquier fin y tiene capacidades de validación de datos
- Las aplicaciones de XML son fácilmente extensibles mediante definiciones de nuevos tipos de documento (DTD).

Luego de una exploración de las ventajas de XML se muestran a continuación la superioridad de XML sobre HTML.

HTML	**XML**
Se preocupa por formatear datos y para ello son las etiquetas que tiene el lenguaje, para formatear la información que se desea mostrar.	Se preocupa por estructurar la información que pretende almacenar. La estructura la marca la lógica propia de la información.
El desarrollo estuvo marcado la competencia entre los distintos visores del mercado. Cada uno quería ser el mejor e inventaba etiquetas nuevas que	El desarrollo está siendo llevado a cabo con rigor, siempre ajustado a lo que marca el estándar que desarrolla el W3C, entidad que está desarrollando el XML

a la larga entraban a formar parte del estándar del W3C, como la etiqueta <FRAME>.	con más diligencia que las empresas con intereses particulares.
Procesar la información es inviable, por estar mezclada con los estilos y las etiquetas que formatean la información.	Se puede procesar la información con mucha facilidad, porque todo está ordenado de una manera lógica, así mismo el formateo de la información para que se pueda entender bien por el usuario es viable a través de un pequeño procesamiento, a través de hojas de estilos o similares.
Requiere DTD (Document Type Definition)	Tiene punteros a la estructura de los datos, lo que ahorra tiempo y simplifica el software de aplicación.

FIGURA 3. CUADRO COMPARATIVO VENTAJAS XML Y HTML

3.3.2 Desventajas

Es posible que con el tiempo las mayores ventajas del XML se vuelvan sus desventajas. La posibilidad de construir sistemas acordes a las necesidades para el intercambio de datos podría llegar a la proliferación de versiones incompatibles y si esto llegase a suceder, entonces la solución que plantea el XML ante la búsqueda de intercambio universal de información, lo llevaría a su opuesto; en vez de unificar todo un lenguaje, se encontraría con lenguajes muy específicos y cada vez más alejados de la “universalidad”.

3.4 ESTRUCTURA DEL LENGUAJE

El metalenguaje XML consta de tres especificaciones:

3.4.1 DTD (Document Type Definition)

Definición del tipo de documento. Es, en general, un archivo/s que encierra una definición formal de un tipo de documento y, a la vez, especifica la estructura lógica de cada documento. Define tanto los elementos de una página como sus atributos.[9] Su función básica es la descripción del formato de datos, para usar un formato común y mantener la consistencia entre todos los documentos que utilicen la misma DTD. De esta forma, dichos documentos, pueden ser validados, conocen la estructura de los elementos y la descripción de los datos que trae consigo cada documento, y pueden además compartir la misma descripción y forma de validación dentro de un grupo de trabajo que usa el mismo tipo de información. La DTD se puede incluir dentro del archivo del documento, pero normalmente se almacena en un fichero ASCII de texto separado. La sintaxis de las DTD para SGML y XML es similar pero no idéntica.

Las DTD se emplean generalmente para determinar la estructura de un documento mediante etiquetas (en inglés). Una DTD describe:

- Elementos: indican qué etiquetas son permitidas y el contenido de dichas etiquetas.
- Estructura: indica el orden en que van las etiquetas en el documento.
- Anidamiento: indica qué etiquetas van dentro de otras.

Ejemplo:

[9] http://geneura.ugr.es/~maribel/xml/introduccion/index.shtml#13

```
<! DOCTYPE etiqueta[
<!ELEMENT etiqueta (nombre, calle, ciudad, pais, codigo)>
<!ELEMENT nombre (#PCDATA)>
<!ELEMENT calle (#PCDATA)>
<!ELEMENT ciudad (#PCDATA)>
<!ELEMENT pais (#PCDATA)>
<!ELEMENT codigo (#PCDATA)>
]>
<etiqueta>
<nombre>Pedro Pérez</nombre>
<calle>Cll. 35 #14-24</calle>
<ciudad>Manizales</ciudad>
<pais>Colombia</pais>
<codigo>140319</codigo>
</etiqueta>
```

La declaración del tipo de documento empieza en la primera línea y termina con "]>". Las declaraciones DTD son las líneas que empiezan con "<!ELEMENT" y se denominan declaraciones de tipo elemento. También se pueden declarar atributos, entidades y anotaciones para una DTD.

En el ejemplo anterior, todas las declaraciones DTD se definen "etiquetas" residen dentro del documento. Sin embargo, la DTD se puede definir parcial o completamente. Por ejemplo:

```
<?xml version="1.0"?>
<!DOCTYPE coche SYSTEM "http://www.sitio.com/dtd/coche.dtd">
<coche>
<modelo>...</modelo>
...
</coche>
```

3.4.2 XSL (eXtensible Stylesheet Language)

XML se diseñó con la finalidad de almacenar datos. Sin embargo, la mayoría de las aplicaciones Web se diseñan pensando en el usuario final. Por tanto, la información

debería ser presentada en un formato legible. En este punto es donde XSL entra en juego: toma los datos de los árboles XML y los procesa para generar una salida legible.

XSL permite modificar el aspecto de un documento. XSL nos permite describir la forma en que serán formateados o transformados los archivos codificados en XML para ser mostrados. Este estándar está basado en el lenguaje de semántica y especificación de estilo de documento (DSSSL, Document Style Semantics and Specification Language, ISO/IEC 10179) y, por otro lado, se considera más potente que las hojas de estilo en cascada (CSS, Cascading Style Sheets), usado en un principio con el lenguaje DHTML.

Se pueden usar diferentes hojas de estilo, o incluso la misma, para presentar la información de diferentes maneras dependiendo de los deseos o de las condiciones del usuario.

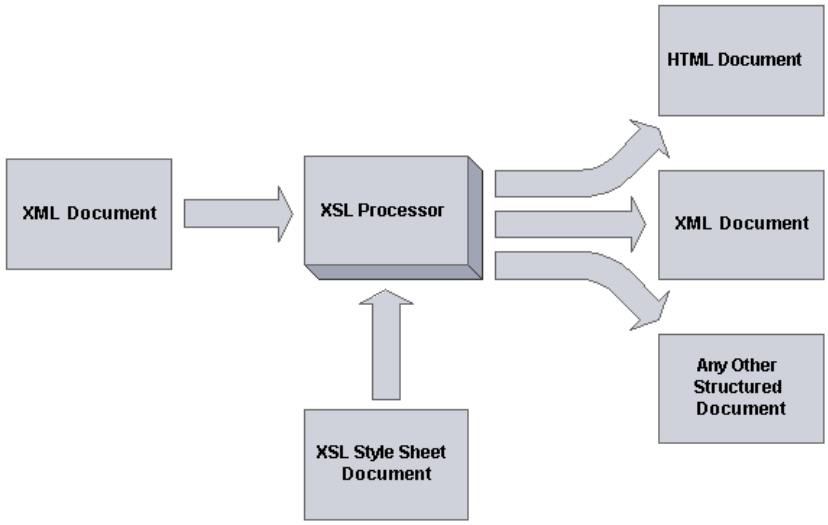

FIGURA 4. COMPOSICIÓN DE UN DOCUMENTO XML

Ejemplo.

Tenemos el siguiente tenemos el siguiente archivo xml con información de alumnos:

```
<?xml version="1.0" enconding=""iso-8859-1?>
```

```
<ALUMNOS>
        <ALUMNO>
                <NOMBRE>Diana Benedetti </NOMBRE>
                <BARRIO>MANGA</BARRIO>
        </ALUMNO>

        <ALUMNO>
                <NOMBRE>Sandra Cortes</NOMBRE>
                <BARRIO>TARACARIGUA</BARRIO>
        </ALUMNO>

        <ALUMNO>
                <NOMBRE>Victor Cortes</NOMBRE>
                <BARRIO>PIE DE LA POPA</BARRIO>
        </ALUMNO>

                <ALUMNO>
                <NOMBRE>Felix Rodriguez</NOMBRE>
                <BARRIO>GAVIOTAS</BARRIO>
        </ALUMNO>

</ALUMNOS>
```

Para mostrar la información en forma detallada y ordenada por listado ALUMNO – NOMBRE, se debe escribir un documento XSLT y utilizar la etiqueta xsl:for-each con la cual se recorren todos los alumnos dado un path (en este caso ALUMNOS/ALUMNO) gracias al atributo select de la etiqueta.

Luego, para cada uno de los valores de los elementos de ALUMNOS, con la etiqueta xsl:value-of se recupera e incluye en un documento que combina etiquetas de XHTML (por ejemplo, la etiqueta BR). [10]

Entonces el código del documento XSLT quedaría así:

```
<xsl:for-each select="ALUMNOS/ALUMNO">
        Nombre:<xsl:value-of select="NOMBRE"/><br/>
        Barrio:<xsl:value-of select="BARRIO"/>
</xsl:for-each>
```

[10] http://luauf.com/2009/11/30/xsl-lenguaje-de-hojas-de-estilo-extensible/

La visualización quedaría así:

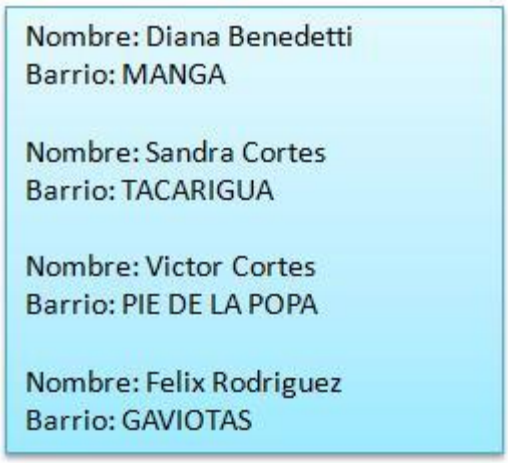

FIGURA 5. RESULTADO DE DOCUMENTO XML Y XSLT

3.4.3 XLL (eXtensible Linking Language)

Define el modo de enlace entre diferentes enlaces. Se considera que es un subconjunto de HyTime (Hipermedia/Timed-based structuring Language o Lenguaje de estructuración hipermedia/basado en el tiempo, ISO 10744) y sigue algunas especificaciones del TEI (Text Encoding Initiative o Iniciativa de codificación de texto). Desde marzo de 1998 el W3C trabajo en los enlaces y direccionamientos del XML. Provisionalmente se le renombró como Xlink y a partir de junio se le denomina XLL. Este lenguaje de enlaces extensible tiene dos importantes componentes: Xlink y el Xpointer. Va más allá de los enlaces simples que sólo soporta el HTML.

Para entender de mejor forma como interactúa el lenguaje XLL se propone el siguiente ejemplo: Una empresa quiere comprar ciertos productos de un suministrador. Los trabajadores de la empresa podrán visitar la página web del proveedor y hacer enlaces externos a comentarios sobre las especificaciones de tal

producto. Cuando otro compañero de la misma empresa visite la página, el servidor de enlaces de nuestra empresa le mostrará la página junto con los enlaces externos creados, y mostrar nuestros comentarios como si fueran parte del documento original.

Los mecanismos hipertextuales que soportará esta especificación:

- Denominación independiente de la ubicación.
- Enlaces que pueden ser también bidireccionales.
- Enlaces que pueden especificarse y gestionarse desde fuera del documento a los que se apliquen (Esto permitirá crear en un entorno intranet/extranet un banco de datos de enlaces en los que se puede gestionar y actualizar automáticamente. No habrá más errores del tipo "404 Not Found").
- Hiperenlaces múltiples (anillos, múltiples ventanas, etc.).
- Enlaces agrupados (múltiples orígenes).
- Transclusión (el documento destino al que apunta el enlace aparece como parte integrante del documento origen del enlace).
- Se pueden aplicar atributos a los enlaces (tipos de enlaces).

3.5 CARACTERISTICAS DEL LENGUAJE

3.5.1 Sintaxis

La posibilidad de marcado descriptivo, con un conjunto de marcas abierto. En HTML y XML se intercalan marcas en los documentos. La principal diferencia entre uno y otro está en la función de estas marcas. En XML las marcas tienen la función de diferenciar los contenidos informativos de los documentos, frente al uso que se hace

en HTML, donde las marcas sirven para indicar cómo se deben visualizar los contenidos. Por otra parte, mientras que HTML nos indica qué marcas podemos utilizar cuando creamos un documento, XML no especifica un conjunto válido de marcas, sino que nos ofrece las reglas que nos permiten crear nuevos vocabularios o conjuntos de marcas aplicables para la codificación de distintos tipos de documentos.

XML impone una sintaxis más rígida para las marcas, que permite su proceso de forma más eficiente. En XML, las marcas de término no pueden ser omitidas. Marcas sin contenido, como IMG o BR en HTML, terminan con un /> para indicar que allí acaban. También distingue entre minúsculas y mayúsculas. También, cualquier valor de un atributo en una marca debe ir entre comillas (es decir, no se pueden omitir). Esto significa que interpretar XML sin conocer el conjunto válido de marcas es mucho más sencillo. XML permite definir lenguajes de marcas para cualquier fin y tiene capacidades de validación de datos.

3.5.2 Extensible

Dentro de XML se pueden definir un conjunto ilimitado de etiquetas. Mientras que las etiquetas de HTML pueden utilizarse para desplegar una palabra en negrita o itálicas, el XML proporciona un marco de trabajo para etiquetado de datos estructurados. Un elemento de XML puede declarar que sus datos asociados sean el precio de venta al público, un impuesto de venta, el título de un libro o cualquier otro elemento de datos deseado. Al irse adoptando las etiquetas XML a lo largo de una intranet de alguna organización y a lo ancho de la Internet, habrá una correspondiente habilidad para buscar y manipular datos sin importar las aplicaciones dentro de las cuales se encuentre.

3.5.3 Estructurado

El XML proporciona una representación estructural de los datos que ha probado ser ampliamente implementable y fácil de distribuir. Las implementaciones industriales en la comunidad del SGML y en otros lugares han demostrado que la calidad intrínseca y la fortaleza industrial del formato de datos con estructura de árbol del XML. El XML es un subconjunto del SGML que está optimizado para su transmisión por Web; al estar definido por el Consorcio de la World Wide Web, asegura que los datos estructurados serán uniformes e independientes de aplicaciones o compañías.

En XML se establece una clara diferencia entre la estructura de un documento y su presentación. Las marcas de un documento XML no indican nada sobre cómo debe presentarse el documento. Para indicar cómo se debe presentar un documento en pantalla o en papel, será necesario crear una hoja de estilo aparte, y asociarla posteriormente al documento

El lenguaje XML proporciona un estándar de datos que puede codificar el contenido, la semántica y el esquema de una amplia variedad de casos que van desde simples a complejos.

3.5.4 Versátil

XML aísla contenido, estructura y presentación. Esta separación de datos de la presentación permite una integración de datos perfecta de fuentes diversas. La información de clientes, órdenes de compra, resultados de investigaciones, pagos de facturas, registros médicos, datos de catálogo y cualquier otra información se puede convertir a XML, permitiendo a los datos ser intercambiados en línea tan fácilmente como las páginas de HTML despliegan datos hoy. Los datos codificados en XML

pueden ser transmitidos sobre la Web hasta el escritorio. No es necesario retro ajustar información en formatos propietarios almacenados en bases de datos o documentos de mainframes y, debido a que se usa el HTTP para transmitir documentos XML sobre la red, no se necesitan cambios para esta función. Los documentos XML son fáciles de crear; si está familiarizado con el HTML, puede aprender rápidamente a crear uno.

3.5.5 Interoperabilidad

Un lenguaje de marcas hace fácil identificar segmentos específicos de información con significados especiales. Usar texto simple y mantener el contenido universalmente interpretable son dos condiciones para implementar una tecnología exitosamente entre sistemas heterogéneos. Una cadena del texto se puede transferir y se puede manejar fácilmente en cualquier plataforma destino.

3.5.6 Validable

Cada documento se puede validar frente a una DTD, o en su defecto, se puede declarar bien formado. XML es además independiente de medio ya que permite publicar contenidos en múltiples formatos.

3.6 APLICACIONES DE LENGUAJE

XML puede tener tres posibles roles:

- De contenedor de información

  Cuando el XML actúa de contenedor, ignora por completo la información que contiene. Esta información irá normalmente almacenada en un elemento del tipo CDATA, estos elementos no son analizados por el parser de XML.

Imaginemos que tenemos dos aplicaciones distintas en un proyecto, podemos establecer una gramática XML para encapsular los mensajes XML (añadiendo una cabecera al documento), y tener un programa que procese las cabeceras de los mensajes y los direccione a la aplicación correspondiente.

También nos puede servir para definir un flujo de datos en un workflow.

- Para definir el contenido de los mensajes

  Para que se puedan intercambiar información dos aplicaciones y puedan procesarla automáticamente.

- Para describir el contenido de los mensajes

  Si lo utilizamos con este rol, podemos describir el esquema de contenidos de los mensajes. Estamos definiendo recursos, lo utilizamos como metadatos. Podemos facilitar el acceso a la información a los agentes de software.

  Estos son algunos roles y posibles aplicaciones del XML, pero se nos pueden ocurrir muchas más.

  La idea del XML es tener la información (definida semánticamente: XML), la estructura de la información (DTD/Schema), formato (XSL) y procesos (Java, VB, etc.), todo ello separado.

  Es un lenguaje entendible por las máquinas.

En la actualidad existe una imperiosa necesidad de las empresas por contar con la información de manera inmediata, por ello optan por utilizar aplicaciones basadas en Web que permitan obtener datos de manera remota en un corto tiempo y sin la necesidad de desplazarse.

Uno de los sectores que más han explotado esta tecnología son las compañías enfocadas a las ventas, ya que de este modo no requieren esperar a que el personal de sus sucursales, puntos de venta o vendedores lleven la información hasta la

matriz o las oficinas principales, simplemente capturan o transmiten los datos recabados por medio de Internet, lo que permite obtenerlos de manera casi inmediata, y por ende, esto redunda en una toma de decisiones más eficiente y rápida.

Otro ejemplo de utilización de bases de datos en documentos XML es el de casi todas las aplicaciones que encontramos en Internet, como directorios, control de usuarios, inventarios en línea, catálogos de productos, etc.

Al final de cuentas y como toda tecnología exitosa lo importante es la utilidad y ventajas que ésta genera y que se traduce en ahorros significativos, así como en mejoras en la toma de decisiones de las organizaciones, aspectos en los que XML definitivamente tiene su fortaleza y que permiten visualizar un amplio desarrollo y aplicación de este lenguaje para la gestión de datos vía Web, y veremos si poco a poco comienza a afianzarse como una opción viable en la gestión en aplicaciones locales.[11]

A continuación enunciaremos algunas aplicaciones y tecnologías XML.

3.6.1 Disminuir trabajo en el Servidor

Por medio del Modelo de Objetos de Documentos (DOM), podemos evitarle trabajo al servidor, espera al cliente y no saturar tanto la red.
Por ejemplo, una tienda en Internet que vende una serie de productos, el cliente al conectarse, obtiene el catálogo de productos, y va seleccionando ítems, pero cada vez que selecciona uno, no se manda al servidor nada, sino que por medio del DOM se trata esa selección creando un nuevo nodo del árbol (en el cliente), una vez que el

[11] http://www.gestiopolis.com/administracion-estrategia/xml-alternativa-para-el-manejo-de-datos-moviles.htm

usuario termina, puede ver su cesta de la compra y verificar su pedido para mandarlo al servidor. Una vez le llegue, éste responderá al cliente del estado de su pedido.

En todo el proceso únicamente hay una primera petición al servidor para bajarse la lista de productos y una segunda donde se le envía el pedido.

Este proceso se puede aplicar a las tiendas on-line, a los bancos, etc.

3.6.2 Soporte a clientes

Gracias a los enlaces extendidos (XLL), el cliente si tiene algún tipo de problema, haciendo click o presionando sobre un enlace de este tipo le puede emerger una lista con la persona concreta de contacto para brindarle el soporte, partes de manuales concretos donde resolver esa duda ó departamentos determinados.

3.6.3 Personalización de la WEB

Gracias al XSL/XSLT, podemos transformar y dar el formato que queramos a un mismo documento XML, por lo tanto si sabemos gracias a las cookies, u, los login, quién es el usuario/cliente que accede podemos personalizar la vista de la información. El cliente/Usuario de esta forma se sentirá más a gusto, considerará que esa información es suya y que es reconocido en la Web. Muy útil para los Sitios de Comercio Electrónico y actualmente lo encontramos en la Configuración de Cuenta de ***HOTMAIL***.

3.6.4 Independencia del dispositivo de acceso a la información

Otra aplicación de las XSL/XSLT, si disponemos de un servidor WAP y otro WEB, podemos saber en función del protocolo, desde donde se nos pide la información y en función del dispositivo, sacarle la misma información de una forma u otra.

3.6.5 Gestión de la información/Conocimiento

En una empresa donde se maneje infinidad de información, normalmente el usuario al buscarla no está interesado en leerse toda, incluso cuando se abre un documento solamente le interesa los tres últimos párrafos pero para encontrarlos, debe leerse todo el documento con el correspondiente tiempo invertido en ello, pues bien: si etiquetamos la información y a cada usuario se le proporciona una serie de etiquetas de interés, se podría resaltar la información que le es interesante, frente a la que no es relevante.
Las búsquedas serían mucho más rápidas y no se perdería tanto tiempo.
Con XML se nos pueden ocurrir muchas más aplicaciones para una mejor gestión de la información.

3.6.6 Buscador WEB

Si disponemos de un Sitio donde toda la información se encuentre etiquetada en documentos XML, las búsquedas serían mucho más efectivas, ya que se conjuga la potencia de la búsqueda indexada junto la búsqueda semántica. Ejemplo de este es el buscador XML: GOXML

3.6.7 Intercambio de información

Si contratamos a una empresa un servicio de noticias, y nos facilitan la estructura de los datos que vamos a recibir (DTD/Schema), sabremos en todo momento que tipos de documentos XML estamos recibiendo, y podremos tratarlos de la forma que deseemos.

Ejemplo: Contratamos noticias internacionales, nacionales, y deportes (resultados de la jornada de liga) . Cuando recibimos los documentos XML sabemos en función de su cabecera a que clase pertenece y actuaremos en consecuencia: las noticias internacionales le aplicamos un estilo serio y ordenado, a las nacionales le aplicamos otro estilo e incluso hay campos que no nos interesa visualizar y los deportes las transformamos y las procesamos por medio de otra aplicación para terminar almacenando esos datos en una base de datos.

Podemos tratar esos documentos como queremos para aplicarnos a nuestro negocio.

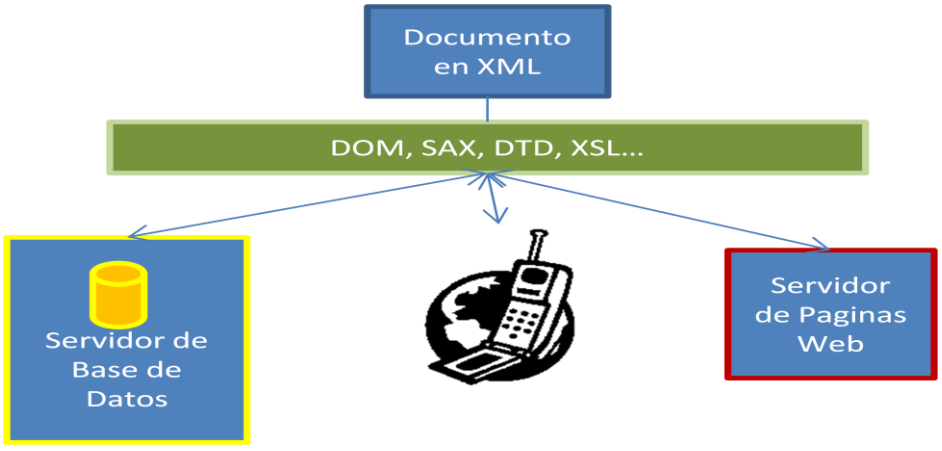

FIGURA 6. INTERCAMBIO DE INFORMACION

En la gráfica se muestra un documento XML que a través de varias herramientas desarrolladas para el lenguaje (DOM, SAX, DTD, XSL… entre otras más) puede ser utilizado para varios fines (aparatos inalámbricos, bases de datos, servidores-web), las flechas bidireccionales indican que el proceso puede ser llevado en cualquier sentido y a cualquier medio que soporte XML .Es esta facilidad de Intercambio de Información por la que ha sido aclamado XML en desarrollos B2B (Business to Business)

4. XML Y LOS SERVICIOS WEB

Las tendencias actuales del sector requieren la integración de sistemas distribuidos distintos, y aunque algunas tecnologías cumplen con los requisitos, son inadecuadas, poco realistas técnicamente (y por tanto costosas), inflexibles, o no siguen un estándar.

Para atender las necesidades empresariales actuales, los programadores requieren una tecnología que se pueda integrar a través de muchos tipos de sistemas distintos, interactuar fácilmente con sistemas heredados, y que no plantee riesgos de seguridad para la red.

Por otro lado, sería una ventaja si dicha tecnología:

- Utilizara tecnologías existentes económicas (protocolos, acceso de red, hardware, software), fuera fácil de poner en funcionamiento y de mantener y permitiera utilizar los conocimientos, y recursos existentes.
- Estuviera basada en un estándar abierto que fuera accesible para todos.
- Pudiera garantizar el envío de mensajes sin costosas soluciones de software.
- Fuera sencilla y comprensible a la hora de intentar comprender y depurar un problema.

Dado que las tecnología existentes no están presentes en todo el mercado, no son buenas candidatas sobre las que construir una nueva economía, de modo que cualquier compañía pueda ofrecer los datos y los servicios de computación, económicamente, a otra compañía independientemente de la plataforma hardware o de software. Es necesaria una manera distinta de transferir la información a través de los límites de la plataforma y de la compañía para proporcionar los servicios a cualquier otro negocio de Internet. Debido al fenómeno de la “Era de la Internet”, el

servicio se debe integrar fácilmente en los sistemas existentes de negocios de consumo.

Si todos los archivos utilizaran XML, sería posible realizar intercambio de datos entre sistemas. Esta grafica describe como utilizando XML no importa el tamaño o plataforma subyacente, es posible intercambiar datos:

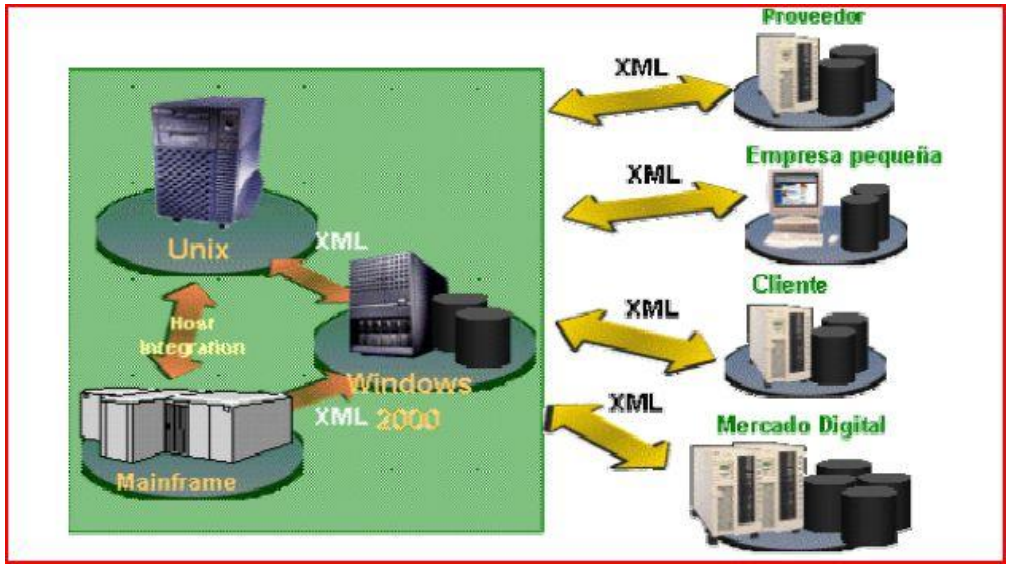

FIGURA 7. INTERACCION DE XML

XML, contiene datos estructurados y por tal motivo es fácil almacenarlos en una base de datos. En las grandes empresas actualmente, hay islas de información: los departamentos utilizan diferentes sistemas operativos, bases de datos, lenguajes de programación, cada departamento es autónomo de decidir cual utilizar en base a muchos requerimientos particulares. Bajo esta necesidad nacen dos protocolos estándares que basados en XML nos permiten la integración de servicios: XML-RPC y SOAP. A la hora de programar un servicio web, hay que decidir qué protocolo usar, porque un protocolo es incompatible con el otro. De modo que si se programa un servicio web con XML-RPC, no se puede invocar desde un lenguaje de programación que trabaje con SOAP, como por ejemplo .Net de Microsoft.

Tanto SOAP (Protocolo de acceso a objetos simple, Simple Object Access Protocol) como XML-RPC son lenguajes de mensajería basada en XML, estandarizados por el consorcio W3C, que especifican todas las reglas necesarias para ubicar servicios Web XML, integrarlos en aplicaciones y establecer la comunicación entre ellos.

4.1 SOAP: UNA SOLUCION DE INTEGRACIÓN

El Protocolo de Acceso a Objetos (SOAP) es una especificación que define como los mensajes se pueden enviar entre dos sistemas de software con el uso del LENGUAJE DE MARCAS EXTENDIDO (XML). Estos mensajes suelen seguir un Modelo de Solicitud/Respuesta: un equipo hace una llamada de método, y el otro equipo realizar cierto servicio y, después, devuelve un resultado a la aplicación que llama.

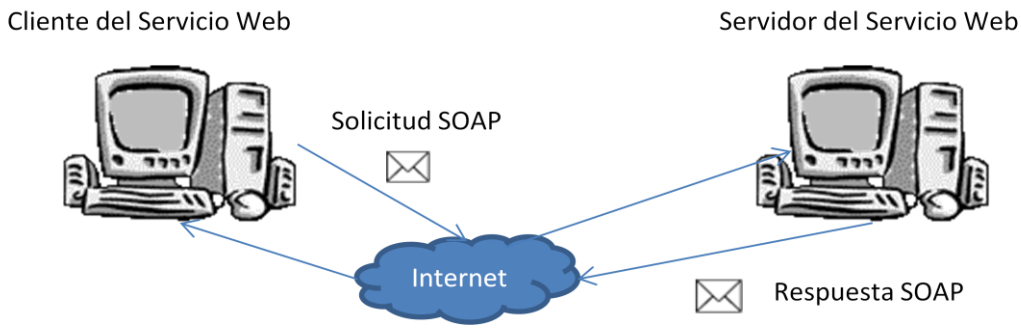

FIGURA 8. FUNCIONAMIENTO DE SOAP

SOAP ofrece muchas ventajas sobre otros protocolos, ventajas que se derivan del uso de XML:

- SOAP es independiente de la plataforma: dado que SOAP es solo XML sin formato, se puede utilizar en cualquier plataforma, es decir, se pueden eliminar las barreras típicas entre los sistemas de autorización basados en RPC.

- SOAP es fácil de integrar en sistemas distintos: dado que SOAP es solo XML, es fácil integrar un servicio que resida como un filtro o proxy de mensajes para enviar y recibir los mensajes de SOAP, y después para traducirlos en mensajes específicos para cualquier aplicación o plataforma dadas.
- SOAP plantea pocos riesgos de seguridad: SOAP es simplemente XML, así que se puede utilizar en el puerto 80, que se puede configurar para HTTP. Esta característica ofrece a los mensajes de SOAP la ventaja de poder pasar a través de los servidores de seguridad sin ningún otro puerto abierto, reduciendo los fallos de seguridad potenciales.

Varias especificaciones existentes son similares a SOAP, y todas tienen un objetivo: facilitar las llamadas de procedimiento remoto (RPC). RPC es una tecnología que permite a una aplicación llamar un procedimiento que resida físicamente en codigo en otra parte.

SOAP utiliza XML para expresar las llamadas RPC y, y por tanto, puede aprovechar la ventajas de XML, así como sus tecnologías relacionadas.

La perfección de SOAP es que al ser XML de una manera específica, absolutamente cualquier lenguaje de programación o de secuencia de comandos que pueda concatenar, validar y devolver las cadenas de los procedimientos, puede proporcionar y consumir los servicios Web de SOAP.

Obviamente, algunos entornos de desarrollo ofrecen componentes, bibliotecas de clases que facilitan el trabajo con SOAP, pero eso no significa que el desarrollo sea imposible en otros entornos. Con sus raíces en XML, SOAP es la forma perfecta para comunicarse entre aplicaciones porque no hay afinidad para un lenguaje de programación, plataforma, tipo de equipo o red.

4.1.1 Componentes de un Mensaje SOAP

Igual que una página web, un mensaje de SOAP tiene varias secciones importantes, cada una con un propósito especial. Hay tres secciones importantes en un mensaje SOAP:

- Sobre: este contenedor para el encabezado y el cuerpo puede contener declaraciones de espacios de nombres XML, atributos y demás información.
- Encabezado: esta sección contiene información opcional compresible o no para el consumidor.
- Cuerpo: esta sección contiene los datos reales de la llamada o de la respuesta del método.

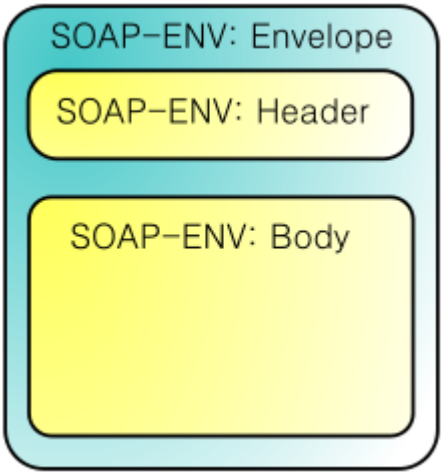

FIGURA 9. ESTRUCTURA SOAP

4.1.1.1 Sobre SOAP

Un sobre SOAP se concibe de varias maneras. En primer lugar, el sobre es el codigo XML contenido en el mensaje SOAP, que incluye el encabezado de SOAP y el cuerpo de SOAP así como los elementos secundarios adicionales que se puedan definir. Por tanto, el sobre es como la carga Útil de XML. Para enviar un mensaje mediante un

protocolo, la información de dirección del destinatario previsto se agrega para adaptarse con el estándar del protocolo. En principio, esta información asegura, que el mensaje llega al destinatario deseado.

4.1.1.2 Encabezado SOAP

El primer elemento que se puede ver en un sobre SOAP es el encabezado. Es opcional pero su propósito es incluir la información adicional que no es parte de la llamada del método pero que puede ser importante para su aplicación. Por ejemplo aquí se podría incluir la información de la cuenta (inicio de sesión, identificador del cliente, etc.) para notificar al servicio WEB que cliente utiliza tal servicio. El servidor por su parte podría usarlo para facturación o registros, autenticación etc.

Es necesario tener en cuenta varias reglas respecto a los encabezados de SOAP:

- Aunque el elemento del encabezado es opcional, si aparece debe ser el primer elemento que sigue de la etiqueta de apertura del sobre ***<SOAP-ENV:Envelope>***
- El elemento del encabezado debe seguir la especificación de SOAP a menos que se haya modificado mediante el atributo ***ENV:encodingStyle*** de SOAP.
- Los elementos secundarios del encabezado deben pertenecer al espacio de nombres cualificado.

- El encabezado puede contener el atributo ***ENV:mustUnderstand*** de SOAP, que se usa cuando se desea requerir que el consumidor del mensaje de SOAP comprenda el valor que se pasa.

Dependiendo de la naturaleza del mensaje de SOAP, el contenido del cuerpo cambia. El contenido del cuerpo también cambia cuando se produce un erro o una anomalía.

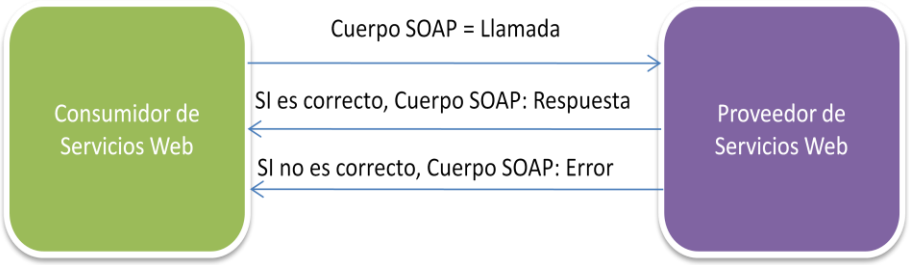

FIGURA 10. FLUJO DE LOS MENSAJES DE SOAP

4.1.1.3 Llamada de Cuerpo SOAP

La llamada es responsable de especificar el nombre del método que se ejecutara y todos los parámetros que se deben pasar al método para que se ejecute correctamente. Para hacer esto, el primer nodo secundario debajo del nodo del cuerpo es el nombre real del método. Después los parámetros del método son nodos secundarios debajo del nodo del método. Ejemplo:

```
<SOAP-ENV:Envelope
        Xmlns:xsi="http://www.w3.org/1999/XMLSchema/instance"
        Xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
        <SOAP-ENV:Body xmlns:MyMethod = "urn:some-uti-com:MyMethod">
                <MyMethod:CalculateNumbers>
                        <lFirstNumber xsi:type="long">500</lFirstNumber>
                        <lSecondNumber xsi:type="long">350</lSecondNumber>
                </MyMethod>
        </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Como se puede observar el primer nodo secundario de la etiqueta Body es el nombre del método, denominado ***CalculateNumber***. Bajo este nodo hay dos nodos adicionales, ***lFirstNumber, lSecondNumber.*** Estos son los parámetros del método.

Dentro de la etiqueta de los elementos de cada nodo de parámetros hay atributos que definen el tipo de dato de los datos pasados dentro de este método.

- Respuesta de Cuerpo SOAP: después que el proveedor del Servicio Web reciba, interprete y procese la llamada de método, envía un mensaje de respuesta o error. Si se asume que el procesamiento era correcto, las respuesta al ejemplo anterior podría ser la siguiente:

```
<SOAP-ENV:Envelope
Xmlns:SOAP-ENV":http://schemas.xmlsoap.org/soap/envelope/">
        <SOAP-ENV:Body xmlns:MyMethod=urn:some-uri-com:MyMethod>
                <MyMethod:CalculateNumbersResponse>
                        <value>850</value>
                </MyMethod:CalculateNumbersResponse>
        </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

A continuación se muestra por completo un ejemplo de mensaje SOAP, donde un cliente solicita información acerca de un producto.

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <getProductDetails xmlns="http://warehouse.example.com/ws">
       <productId>827635</productId>
     </getProductDetails>
   </soap:Body>
</soap:Envelope>
```

FIGURA 11. EJEMPLO MENSAJE SOAP

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
   <soap:Body>
     <getProductDetailsResponse xmlns="http://warehouse.example.com/ws">
       <getProductDetailsResult>
         <productName>Toptimate 3-Piece Set</productName>
         <productId>827635</productId>
         <description>3-Piece luggage set.  Black Polyester.</description>
         <price>96.50</price>
         <inStock>true</inStock>
       </getProductDetailsResult>
     </getProductDetailsResponse>
   </soap:Body>
 </soap:Envelope>
```

FIGURA 12. RESPUESTA DEL SERVIDOR WEB

4.2 XML – RPC

Elliotte Rusty Harold propone en su libro Processing XML with Java, que el protocolo XML-RPC es:

"Una aplicación XML diseñada para permitir llamadas de procedimiento remoto (RPC) a través de Internet. En la jerga de Java, una llamada a procedimiento es una invocación a un método, en lenguajes como C, podría llamarse una llamada a una función."

En general XML-RPC es un protocolo de llamada a procedimiento remoto que usa XML para codificar los datos y HTTP como protocolo de transmisión de mensajes.[12]

Es un protocolo muy simple ya que solo define unos cuantos tipos de datos y comandos útiles, además de una descripción completa de corta extensión. La simplicidad del XML-RPC está en contraste con la mayoría de protocolos RPC que tiene una documentación extensa y requiere considerable soporte de software para su uso.

[12] http://www.xmlrpc.com/spec

Fue creado por Dave Winer de la empresa UserLand Software en asociación con Microsoft en el año 1998. Al considerar Microsoft que era muy simple decidió añadirle funcionalidades, tras las cuales, después de varias etapas de desarrollo, el estándar dejó de ser sencillo y se convirtió en lo que es actualmente conocido como SOAP. Una diferencia fundamental es que en los procedimientos en SOAP los parámetros tienen nombre y no interesan su orden, no siendo así en XML-RPC.

4.2.1 Objetivos de XML-RPC

UserLand Software plantea en su artículo [13] que XML_RPC es muy humilde en sus metas. No fue creado pretendiendo ser la solución a cada problema sino como un medio simple y eficaz para solicitar y obtener información.

"Queríamos un formato limpio, extensible que es muy simple. Debería ser posible para un programador de HTML para poder mirar en un archivo que contiene una llamada a procedimiento XML-RPC, entender lo que está haciendo, y ser capaces de modificarlo y hacer que trabajo en el primer intento o segundo... También quería que fuera fácil de implementar un protocolo que podría rápidamente adaptados para funcionar en otros ambientes o en otros sistemas operativos. " – www.Xmlrpc.com

En la siguiente Figura se indica el proceso de interacción que utiliza XML-RPC en la comunicación de datos, utilizando HTTP como protocolo de transmisión de mensajes.

[13] http://www.xmlrpc.com/spec

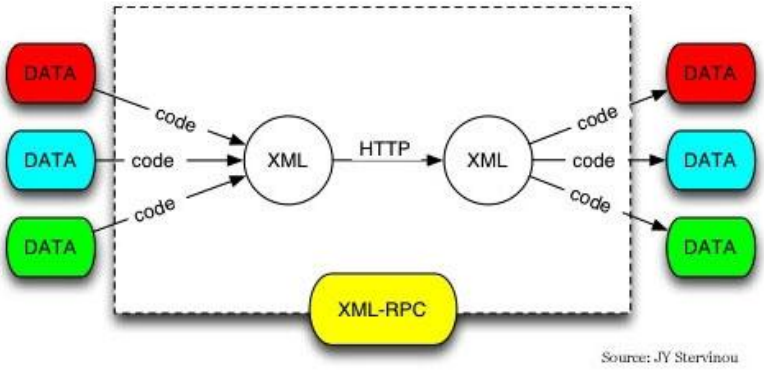

FIGURA 13. COMUNICACIÓN DE DATOS CON XML-RPC[14]

4.2.2 Arquitectura de XML-RPC

En XMLRPC siempre se habla en términos de cliente/servidor, existe un sistema que realiza la solicitud ("el cliente") y otro que la atiende ("el servidor"), y como es de imaginarse un elemento clave en ambos puntos es: **el parser XML**.

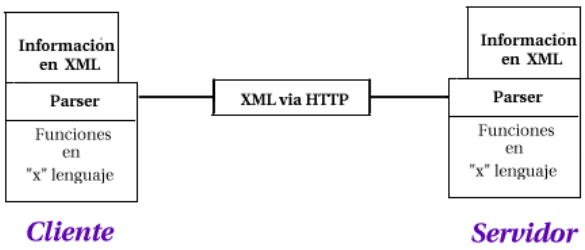

FIGURA 14. ARQUITECTURA XML-RPC15[15]

[14] http://www.xmlrpc.com/#theXmlrpcCommunity
[15] http://tldp.org/HOWTO/XML-RPC-HOWTO/xmlrpc-howto-interfaces.html

Aunque es posible diseñar desde cero cualquier tipo de cliente y/o servidor para emplear XML, hoy en día ya existen diversas implementaciones para diversos ambientes y lenguajes, es a través de estas implementaciones que se logran ahorrar diversas labores como configuraciones de parser, cuestiones de seguridad, integración a servidores de paginas y otros detalles secundarios, utilizando estas estructuras ("Frameworks") se concentra en los procedimientos específicos y no en detalles comunes o secundarios que siempre son utilizados en XMLRPC.

Algunas implementaciones:

- XMLRPC Apache (http://ws.apache.org/xmlrpc/) para el lenguaje "Java" .
- Frontier::RPC (http://www.cpan.org) para el lenguaje "Perl" .
- XMLRPC-PHP (http://sourceforge.net/projects/phpxmlrpc/) para el lenguaje
- "PHP" .
- XMLRPC-Python (http://www.pythonware.com/products/xmlrpc/) para el
- lenguaje "Python" .
- XMLRPC-C (http://xmlrpc-c.sourceforge.net/) para los lenguajes "C" y "C++".
- XMLRPC-ASP (http://aspxmlrpc.sourceforge.net) para el lenguaje
- "COM/VBasic" .[16]

4.2.3 Estructura de Datos de XML-RPC

El modelo implícito de cómo se implementan los servicios es mucho más cercano a C que a Java. Esto no quiere decir que no se puede utilizar Java para escribir XML-RPC clientes o servidores - que ciertamente puede - sólo que una llamada de

[16] http://xml.osmosislatina.com/curso/xmlrpc.htm

procedimiento RPC-XML es más como una llamada a la función C que en una llamada a un método Java y que XML-RPC tipos de datos se asemejan más a tipos de datos de C que en los tipos de objetos de Java. Por casualidad, XML-RPC no tiene ningún concepto de los punteros (la raíz de todos los males de C), pero sí tiene estructuras y matrices para la manipulación de las combinaciones de valores y listas de valores. Estas construcciones se pueden anidar. Es decir, una estructura puede ser miembro de otra estructura o una matriz, y un elemento de la matriz puede ser una estructura o matriz de otro. Esto le permite pasar datos de estructuras complejas de manera justa a procedimientos remotos.[17]

Una estructura de XML-RPC está representado por un struct elemento. Cada miembro de la estructura está representado por un miembro elemento. Cada miembro elemento tiene un nombre y un valor.

[17] http://www.cafeconleche.org/books/xmljava/chapters/ch02s05.html

5. XML COMO INTEGRADOR

5.1 DATOS

Los servicios Web utilizan la tecnología Web para realizar a solicitud de integración de datos. Ayudan a integrar las aplicaciones a un costo menor que cualquier otra tecnología de manera significativa .Estos servicios pueden ser vistos como una nueva forma de "middleware" que se basa en XML (Lenguaje de marcado extensible) y las tecnologías Web, como HTTP (Hyper transporte texto Protocolo).

Un servicio Web se puede desarrollar utilizando cualquier lenguaje, y pueden ser desplegados en cualquier plataforma, se puede acceder cualquier otra aplicación, sin importar el idioma o la plataforma. Los Servicios Web simplifican el proceso de hacer varias aplicaciones comunicarse entre sí y esto trae consigo la reducción de costos de desarrollo, rapidez al mercado, fácil el mantenimiento, y reducción total del costo de propiedad.

Permitir el acceso a los datos corporativos a los empleados de una empresa podría llegar a ser una tarea desalentadora, Especialmente, si los datos deben ser accesibles en cualquier momento y desde cualquier lugar. . En tales situaciones, el uso de los servicios Web pueden simplificar enormemente el proceso de integración.

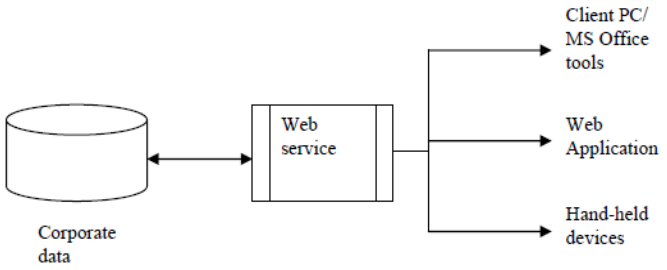

FIGURA 15. SERVICIO WEB PARA ACCEDER A LOS DATOS CORPORATIVOS DE FORMA REMOTA

Por ejemplo, si los datos residen en un servidor de base de datos, a través de un servicio Web que se pueden presentar en la Web o al alcance de un cliente de Microsoft Office (como Excel, Word, InfoPath, etc.) La figura muestra las interacciones potenciales para la recepción de los datos requeridos forma de servidor de base de datos corporativa de la aplicación cliente de Excel. Para desarrollar esta aplicación en el cliente, un código corto o macro es requerida para procesar la petición del cliente y la respuesta del servidor[18].

A continuación se plantea un ejemplo propuesto por la Organización W3c, donde se puede describe una orden de compra generada por los productos ordenados y programando la demanda:

[18] Using Web Services for Internal Data Integration, Mirza Murtaza, - Jaymeen R. Shah

```
<?xml version="1.0"?>
<purchaseOrder orderDate="1999-10-20">
   <shipTo country="US">
      <name>Alice Smith</name>
      <street>123 Maple Street</street>
      <city>Mill Valley</city>
      <state>CA</state>
      <zip>90952</zip>
   </shipTo>
   <billTo country="US">
      <name>Robert Smith</name>
      <street>8 Oak Avenue</street>
      <city>Old Town</city>
      <state>PA</state>
      <zip>95819</zip>
   </billTo>
   <comment>Hurry, my lawn is going wild<!/comment>
   <items>
      <item partNum="872-AA">
         <productName>Lawnmower</productName>
         <quantity>1</quantity>
         <USPrice>148.95</USPrice>
         <comment>Confirm this is electric</comment>
      </item>
      <item partNum="926-AA">
         <productName>Baby Monitor</productName>
         <quantity>1</quantity>
         <USPrice>39.98</USPrice>
         <shipDate>1999-05-21</shipDate>
      </item>
   </items>
</purchaseOrder>
```
[19]

De acuerdo a lo visto en el capítulo 3, este ejemplo muestra claramente la estructura del lenguaje XML. El pedido consiste en un elemento principal, purchaseOrder, y los subelementos shipTo, billTo, comment y ítem. Estos subelementos (excepto comment) a su vez contienen otros subelementos, y así sucesivamente. Los elementos que contienen subelementos o llevan atributos se denominan complejos, mientras que los elementos que contienen números (y

[19] http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/

cadenas, y las fechas, etc.), pero no contienen subelementos se dice que son simples.

Recordemos que un esquema XML es un lenguaje para expresar restricciones sobre los documentos XML. Existen varios esquemas diferentes de uso generalizado, pero las principales son de tipo Definiciones de documentos (DTD), Relax-NG, Schematron y W3C XSD (esquema XML Definiciones). A continuación se muestra el esquema XML para el documento XML, Orden de compra:

```
xmlns:xsd="http://www.w3.org/2001/XMLSchema"> <xsd:schema

  <xsd:annotation>
    xml:lang="es"> <xsd:documentation
     Compra de esquema para que Example.com.
     Copyright 2000 Example.com.  Todos los derechos reservados.
    </ Xsd:> documentación
  </ Xsd:> anotación

  <xsd:element name="purchaseOrder" type="PurchaseOrderType"/>

  <xsd:element name="comment" type="xsd:string"/>

  name="PurchaseOrderType"> <xsd:complexType
    <xsd:sequence>
      <xsd:element name="shipTo" type="USAddress"/>
      <xsd:element name="billTo" type="USAddress"/>
      <xsd:element ref="comment" minOccurs="0"/>
      <xsd:element name="items" type="Items"/>
    </ Xsd: sequence>
    <xsd:attribute name="orderDate" type="xsd:date"/>
  > </ Xsd: complexType

  name="USAddress"> <xsd:complexType
    <xsd:sequence>
      <xsd:element name="nombre" type="xsd:string"/>
      <xsd:element name="street" type="xsd:string"/>
      <xsd:element name="city" type="xsd:string"/>
      <xsd:element name="state" type="xsd:string"/>
      <xsd:element name="zip" type="xsd:decimal"/>
    </ Xsd: sequence>
    <Xsd: atributo name = "país" type = "xsd: NMTOKEN"
                   fijo = "EE.UU." />
  > </ Xsd: complexType
```

```
  name="Items"> <xsd:complexType
    <xsd:sequence>
      name="item" <xsd:element minOccurs="0" maxOccurs="unbounded">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="productName" type="xsd:string"/>
            name="quantity"> <xsd:element
              <xsd:simpleType>
                base="xsd:positiveInteger"> <xsd:restriction
                  value="100"/> <xsd:maxExclusive
                </ Xsd:> restricción
              > </ Xsd: simpleType
            </ Xsd: element>
            <xsd:element name="USPrice" type="xsd:decimal"/>
            <xsd:element ref="comment" minOccurs="0"/>
            name="shipDate" <xsd:element type="xsd:date" minOccurs="0"/>
          </ Xsd: sequence>
          name="partNum" <xsd:attribute type="SKU" use="required"/>
        > </ Xsd: complexType
      </ Xsd: element>
    </ Xsd: sequence>
  > </ Xsd: complexType

  <! - Stock Keeping Unit, un código para la identificación de los productos ->
  name="SKU"> <xsd:simpleType
    base="xsd:string"> <xsd:restriction
      value="\d{3}-[AZ]{2}"/> <xsd:pattern
    </ Xsd:> restricción
  > </ Xsd: simpleType

</ Xsd:> esquema
```

El esquema de orden de compra consiste en un schema de elementos y una variedad de subelementos, sobre todo element , complexType y simpleType que determinan la aparición de elementos y contenidos en sus documentos de instancia.

Cada uno de los elementos en el esquema tiene un prefijo xsd: que está asociada con el esquema de espacio de nombres XML a través de la declaración, xmlns:xsd="http://www.w3.org/2001/XMLSchema", que aparece en el schema elemento . El prefijo xsd: es usado por convención para denotar el esquema de espacio de nombres XML. El mismo prefijo también aparece en los nombres de los tipos integrados simple, por ejemplo, xsd: string. El objetivo de la asociación es identificar los elementos y tipos simples como pertenecientes al vocabulario del lenguaje de esquemas XML en lugar de en el vocabulario del autor del esquema.

5.2 PROTOCOLOS

Dependiendo de la aplicación de restricciones para el intercambio de datos en la Web, los desarrolladores pueden elegir entre una serie de protocolos como HTTP, SOAP y Web Services.

Los desarrolladores de aplicaciones hoy en día, pueden utilizar la infraestructura de correo electrónico de Internet para transmitir mensajes SOAP ya sean como mensajes de correo electrónico de texto o como adjuntos. El ejemplo que se muestra a continuación plasma un modo de transmitir mensajes.

El Ejemplo muestra el mensaje de petición de reserva de viaje transportado como un mensaje de correo electrónico entre un agente de usuario remitente y un agente de usuario destinatario. Está implícito que el nodo destinatario tiene capacidad para entender SOAP, por lo que se envía el mensaje de correo electrónico para su procesamiento. (Se asume que también el nodo remitente puede manejar errores SOAP que pudiera recibir en la respuesta o correlacionar cualesquiera mensajes SOAP recibidos en respuesta a éste)[20].

[20] http://www.w3.org/TR/2007/REC-soap12-part0-20070427/

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
 <env:Header>
  <m:reservation xmlns:m="http://travelcompany.example.org/reservation"
          env:role="http://www.w3.org/2003/05/soap-envelope/role/next"
           env:mustUnderstand="true">
   <m:reference>uuid:093a2dal-q345-739r-ba5d-pqff98fe8j7d</m:reference>
   <m:dateAndTime>2001-11-29T13:20:00.000-05:00</m:dateAndTime>
  </m:reservation>
  <n:passenger xmlns:n="http://mycompany.example.com/employees"
          env:role="http://www.w3.org/2003/05/soap-envelope/role/next"
           env:mustUnderstand="true">
   <n:name>Åke Jógvan Øyvind</n:name>
  </n:passenger>
 </env:Header>
 <env:Body>
  <p:itinerary
    xmlns:p="http://travelcompany.example.org/reservation/travel">
   <p:departure>
     <p:departing>New York</p:departing>
     <p:arriving>Los Angeles</p:arriving>
     <p:departureDate>2001-12-14</p:departureDate>
     <p:departureTime>late afternoon</p:departureTime>
     <p:seatPreference>aisle</p:seatPreference>
   </p:departure>
   <p:return>
     <p:departing>Los Angeles</p:departing>
     <p:arriving>New York</p:arriving>
     <p:departureDate>2001-12-20</p:departureDate>
     <p:departureTime>mid-morning</p:departureTime>
     <p:seatPreference/>
   </p:return>
  </p:itinerary>
  <q:lodging
   xmlns:q="http://travelcompany.example.org/reservation/hotels">
   <q:preference>none</q:preference>
  </q:lodging>
 </env:Body>
</env:Envelope>
```

El mensaje SOAP en el ejemplo contiene dos subelementos específicos de SOAP dentro del env:Envelope, es decir, un env:Header y un env:Body.

En el ejemplo, el encabezado contiene dos bloques de encabezado, cada uno de los cuales se define en su propio espacio de nombres XML y representan algún aspecto con el tratamiento global del cuerpo del mensaje SOAP. Para esta aplicación de reservas de viajes, como información relativa esta el bloque reservation que proporciona una referencia a la solicitud general de viaje y la identidad del viajero que se encuentra en el bloque passenger.

Los bloques de encabezado reservation y passenger deben ser procesados por los próximos intermediario SOAP se encuentran en la ruta del mensaje o, si no hay un intermediario, por el destinatario final del mensaje. El hecho de que se dirige a la siguiente nodo SOAP es encontrado en el camino indicado por la presencia del atributo env:role con el valor "http://www.w3.org/2003/05/soap-envelope/role/next" (en adelante simplemente "al lado"), que es un papel que todos los nodos SOAP debe estar dispuesto a jugar.
La presencia de un atributo env:mustUnderstand con el valor "true" indica que el nodo (s) de procesamiento de la cabecera es absolutamente necesario para estos bloques de encabezado.

El **env:Body** y sus elementos asociados elementos secundarios, **itinerary** y **lodging** , están dirigidos al intercambio de información entre el remitente SOAP inicial y el nodo SOAP que asume el papel del receptor final SOAP en la ruta del mensaje, que es la aplicación de servicios de viajes . Por lo tanto, el **env:Body** y sus contenidos están dirigidos implícitamente y se espera que sea entendido por el receptor final. Los medios por los que un nodo SOAP asume tal papel no están definidos por la especificación SOAP, y se determina como una parte de la semántica de la aplicación global y del flujo de mensajes.

Un mensaje SOAP como el Ejemplo puede ser transferido por diferentes protocolos subyacentes y se puede utilizar en una variedad de patrones de intercambio de mensajes Por ejemplo, para un Servicio Web que acceda a una aplicación de servicios de viaje, ya que podría ser colocado en el cuerpo de una solicitud POST HTTP. En otro protocolo vinculante, podría ser enviado en un mensaje de correo electrónico.

5.3 DESCRIPCION DE SERVICIOS

La infraestructura de servicios Web XML se fundamenta en la comunicación por medio de mensajes basados en XML que cumplen con una descripción de servicio publicada. La descripción del servicio es un documento XML escrito en una gramática XML denominada WSDL (Lenguaje de descripción de servicios Web) que define el formato de mensaje comprensible para el servicio Web XML. La descripción del servicio sirve como acuerdo que define el comportamiento de un servicio Web XML e indica a los clientes potenciales cómo interactuar con él. El comportamiento de un servicio Web XML está determinado por modelos de mensajería que el servicio define y admite. Estos modelos dictan conceptualmente lo que el consumidor del servicio puede esperar que ocurra cuando se envía un mensaje con formato correcto al servicio Web XML.

Además de las definiciones de formato de mensaje y los modelos de mensajería, la descripción del servicio puede contener la dirección asociada a cada punto de entrada del servicio Web XML. El formato de esta dirección se ajusta al protocolo utilizado para el acceso al servicio, como una dirección URL para HTTP o una dirección de correo electrónico para SMTP.[21]

Veamos un ejemplo de un documento WSDL:

```
<message name="obtTerminoDePet">
<part name="param" type="xs:string"/>
</message>
<message name="obtTerminoDeResp">
<part name="valor" type="xs:string"/>
```

[21] http://msdn.microsoft.com/es-es/library/77axffs8(VS.80).aspx

```
</message>
<portType name="terminosDeDiccionario">
<operation name="obtTermino">
<input message="obtTerminoDePet"/>
<output message="optTerminoDeResp"/>
</operation>
</portType>
```

Vemos como el portType define terminosDeDiccionario como el nombre de un puerto y define obtTermino como el nombre de una operación. Esta operación tiene un mensaje de entrada (es decir, un parámetro) denominado obtTerminoDePet y un mensaje de salida (esto es, un resultado) denominado obtTerminoDeResp. Los elementos message definen los tipos de datos que están asociados a los mensajes.[22]

5.4 SEGURIDAD

La necesidad de garantizar la integridad, la confidencialidad y la autenticidad de los datos que fluyen a través de la Web se ha convertido en un requisito esencial. Por este motivo el área de seguridad crece rápidamente, pero hay muchas dificultades a la hora de manejar datos con estructuras jerárquicas y con subgrupos de datos con diferentes requisitos en lo que se refiere a confidencialidad, derechos de acceso o integridad.

Para hacer frente a estos problemas se han desarrollado estándares como XML Encryption (encriptación) y XML Signature (firma digital), preparados para manejar situaciones en las que partes de un mismo documento necesitan un tratamiento

[22] http://www.cibernetia.com/manuales/servicios_web/4_wsdl.php

diferente, como ocurren en documentos con diferentes secciones cuyo contenido puede ser visto por unos usuarios pero no por otros. En estos casos la encriptación juega un papel muy importante ya que es lo que va a confirma la integridad del texto. Por otro lado, las firmas digitales permiten la autenticación del remitente. Otro problema añadido surge cuando diferentes personas firman digitalmente un mismo documento XML o cuando es necesario hacerlo conjuntamente codificando ciertas partes de ese documento.

El siguiente caso muestra la forma en la que interviene cada parte en un proceso de transacción de datos:

LIBROS S.L. tiene un formulario para comprar libros que será rellenado por María, un usuario que desea comprar un libro.

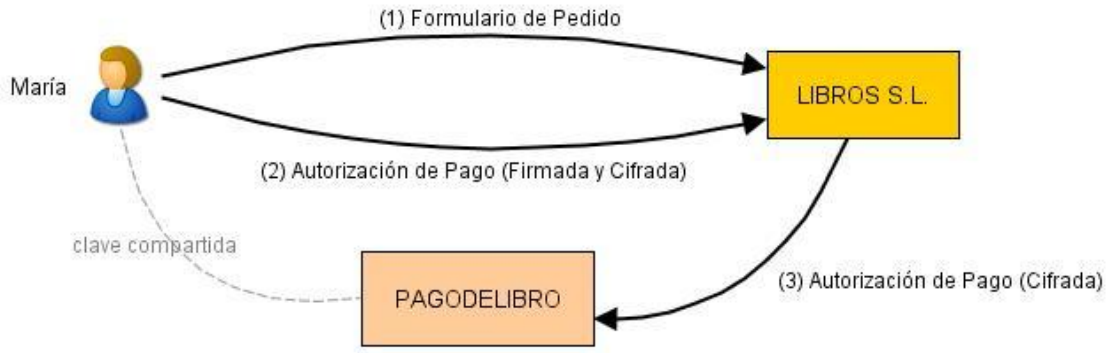

FIGURA 16. Ilustración del ejemplo de compra de un libro[23]

1. María realiza un pedido a LIBROS S.L a través de un formulario emitido por LIBROS S.L.

[23] http://www.w3c.es/divulgacion/guiasbreves/Seguridad

2. María rellena el formulario para comprar el libro. La autorización de pago realizada por María es encriptada con una clave compartida con PAGODELIBRO; la firma y la envía a LIBROS S.L.
3. LIBROS S.L procesa el formulario y confirma la exactitud de la orden (el título del libro y el precio) y pasa la información de la tarjeta de crédito cifrada a PAGODELIBRO.

6. TENDENCIAS Y TRABAJOS FUTUROS

- Profundizar en como las grandes ventajas que se plantean de XML, pueden convertirse en desventajas a largo plazo. La posibilidad de construir sistemas acordes a las necesidades para el intercambio de datos podría llegar a la proliferación de versiones incompatibles y si esto llegase a suceder, entonces la solución que plantea el XML ante la búsqueda de intercambio universal de información, lo llevaría a su opuesto; en vez de unificar todo un lenguaje, se encontraría con lenguajes muy específicos y cada vez más alejados de la “universalidad”.
- La importancia de SOAP como protocolo de comunicación entre los Servicios WEB.

7. CONCLUSIONES

Finalmente ahora que ya conocemos algo más sobre XML no queda responde ¿porque XML es utilizado en los servicios Web? Y diremos porque:

Es un estándar abierto es decir que es reconocido mundialmente ya que muchas compañías tecnológicas integran en su software compatibilidad con dicho lenguaje. Esto quiere decir que la gran mayoría de software de escritorio de sistema operativo, aplicaciones móviles permiten la compatibilidad con XML esto lo hace muy potente a la hora de permite la comunicación entre distintas plataformas de software y hardware (y si bien recordamos este es el sentido final de los Servicios Web).

Simplicidad de sintaxis esto quiere decir que es muy fácil de escribir código en XML y la representación de los datos es casi entendible por cualquier ser humano. Esto lo hace muy flexible a la hora de querer reprensar datos de cualquier especie, bastara con contar con cualquier editor de texto y aprende unas cuantas intrusiones básicas y ya está en condiciones de escribir código XML el cual será soportado o entendido por cualquier aplicación que pueda leer documentos XML. El hecho de que XML sea tan fácil de codificar y de entender lo hace el lenguaje ideal para utilizarlo en los servicios Web.

Independencia del protocolo de Transporte, el hecho de que XML es un lenguaje de Marcado de Texto, no necesita de ningún protocolo de trasporte especial, solo necesita de un protocolo que pueda trasferir texto o documentos simples. Esto nos trae a la memoria que en mercado existen muchos protocolos con estas característica como lo son los más conocidos en HTTP y SMTP por nombrar algunos. Volviendo al tema de los servicios Web una de las características de estos es la independencia del protocolo de trasporte.

8. GLOSARIO

- **DTD:** (*Document Type Definition*) es una descripción de estructura y sintaxis de un documento XML o SGML. Su función básica es la descripción del formato de datos, para usar un formato común y mantener la consistencia entre todos los documentos que utilicen la misma DTD. De esta forma, dichos documentos, pueden ser validados, conocen la estructura de los elementos y la descripción de los datos que trae consigo cada documento, y pueden además compartir la misma descripción y forma de validación dentro de un grupo de trabajo que usa el mismo tipo de información.

- **EBCDIC:** (*Extended Binary Coded Decimal Interchange Code*) es un código estándar de 8 bits usado por computadoras mainframe IBM. IBM adaptó el EBCDIC del código de tarjetas perforadas en los años 1960 y lo promulgó como una táctica customer-control cambiando el código estándar ASCII.

- **GML:** (*Generalized Markup Language*) simplifica la descripción de un documento en términos de su formato, estructura de organización, piezas contentas y su relación, y otras características. El margen de beneficio de GML (o las etiquetas) describe las piezas tales como los capítulos, las secciones importantes, y las secciones menos importantes (especificando niveles del título), párrafos, listas, tablas, y así sucesivamente.

- **ISO:** (*International Organization for Standardization*) es el organismo encargado de promover el desarrollo de normas internacionales de fabricación, comercio y comunicación para todas las ramas industriales a excepción de la eléctrica y la electrónica. Su función principal es la de buscar

la estandarización de normas de productos y seguridad para las empresas u organizaciones a nivel internacional.

- **ITU-T:** (*Sector de Normalización de las Telecomunicaciones de la UIT*) es el órgano permanente de la Unión Internacional de Telecomunicaciones (UIT) que estudia los aspectos técnicos, de explotación y tarifarios y publica normativa sobre los mismos, con vista a la normalización de las telecomunicaciones a nivel mundial. Con sede en Ginebra (Suiza) fue conocido hasta 1992 como Comité Consultivo Telefónico y Telegráfico (CCITT).
- **James Clark**
- **Jon Bosak de Sun**
- **Michael Sperberg-McQueen**
- **SGML:** (*Standard Generalized Markup Language*) consiste en un sistema para la organización y etiquetado de documentos. La Organización Internacional de Estándares (ISO) normalizó este lenguaje en 1986. El lenguaje SGML sirve para especificar las reglas de etiquetado de documentos y no impone en sí ningún conjunto de etiquetas en especial.
- **Tim Bray**
- **UIT-T Rec.. X.891 | ISO / IEC 24824-1**

- **Unicode:** es un estándar de codificación de caracteres diseñado para facilitar el tratamiento informático, transmisión y visualización de textos de múltiples lenguajes y disciplinas técnicas además de textos clásicos de lenguas muertas. El término Unicode proviene de los tres objetivos perseguidos: universalidad, uniformidad y unicidad.

- **W3C:** (*World Wide Web Consortium*) es un consorcio internacional que produce recomendaciones para la World Wide Web. Está dirigida por Tim Berners-Lee, el creador original de URL (Uniform Resource Locator, Localizador Uniforme de Recursos), HTTP (HyperText Transfer Protocol, Protocolo de Transferencia de HiperTexto) y HTML (Lenguaje de Marcado de HiperTexto) que son las principales tecnologías sobre las que se basa la Web.

9. BIBLIOGRAFIA

- Arquitectura SOA. Cesar de la Torre,Roberto Gonzalez.
- Processing XML with Java. Elliotte Rusty Harold. Copyright 2001, 2002 Elliotte Rusty Harold
- XML Práctico. Bases esenciales, conceptos y casos prácticos. Sebastián Lecompte. Eni Ediciones.
- Servicios WEB XML de Microsoft .NET. Prentice HALL- Robert Tabor.
- Implementing SOA Using Java™ EE. B.V. Kumar, Prakash Narayan, Tony NG.
- Alfreso Reino. Lenguaje de Enlace XML (Xlink). http://www.webtaller.com/construccion/lenguajes/xml/lecciones/lenguaje-enlace-xml-xlink.php
- W3C. Guia Breve de Tecnología XML. http://www.w3c.es/divulgacion/guiasbreves/tecnologiasxml Citado: 2008/09/01
- W3C. All Standards and Drafts. http://www.w3.org/TR/
- XML. http://www.dcc.uchile.cl/~rbaeza/inf/xml.html
- XML y la gestión de contenidos. http://www.hipertext.net/web/pag256.htm
- XML, RPC Y SOAP. http://www.osmosislatina.com/xml/xmlrpc.htm Citado: 2005/10/20
- http://www.w3.org/XML/
- W3C. Servicios Web. http://www.w3c.es/Presentaciones/2006/0420-servicios-JA/
- María Jesús Lamarca Lapuente. XML

http://www.hipertexto.info/documentos/xml.htm

- Introducción a XML.
  http://www.adobe.com/es/devnet/dreamweaver/articles/xml_overview_03.html
- Miguel Angel Alvarez. Historia del XML.
  http://www.desarrolloweb.com/articulos/450.php
  Citado: 2001/06/14
- Gunjan Samtani. Using Web services for application integration
  http://articles.techrepublic.com.com/5100-10878_11-1045211.html
  Citado: 2002/05/13
- Benjamín González C. XML: el lenguaje de los Servicios Web
  http://www.desarrolloweb.com/articulos/1574.php
  Citado: 2004/07/19