

CURSO JAVA EE

WEB SERVICES CON JAX WS EN JAVA EE



Por el experto: Ing. Ubaldo Acosta



CURSO JAVA EE

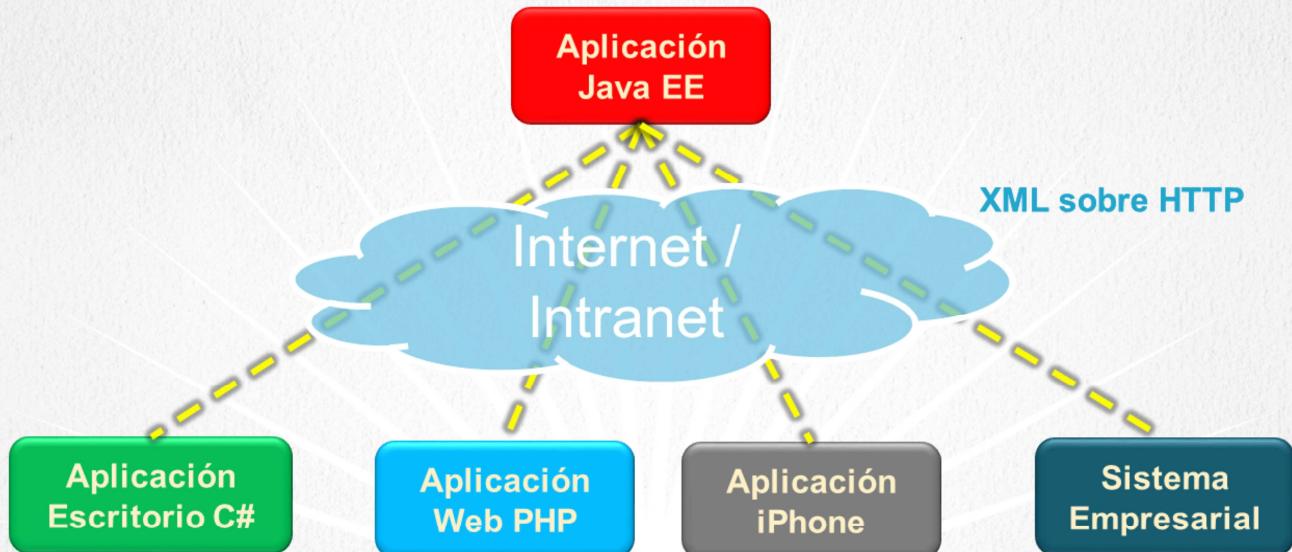
www.globalmentoring.com.mx

Hola, te saluda nuevamente Ubaldo Acosta. Espero que estés listo para comenzar con esta lección..

Vamos a estudiar el tema de Web Services utilizando la tecnología de JAX-WS de Java EE.

¿Estás listo? ¡Vamos!

¿QUÉ SON LOS WEB SERVICES?



CURSO JAVA EE

www.globalmentoring.com.mx

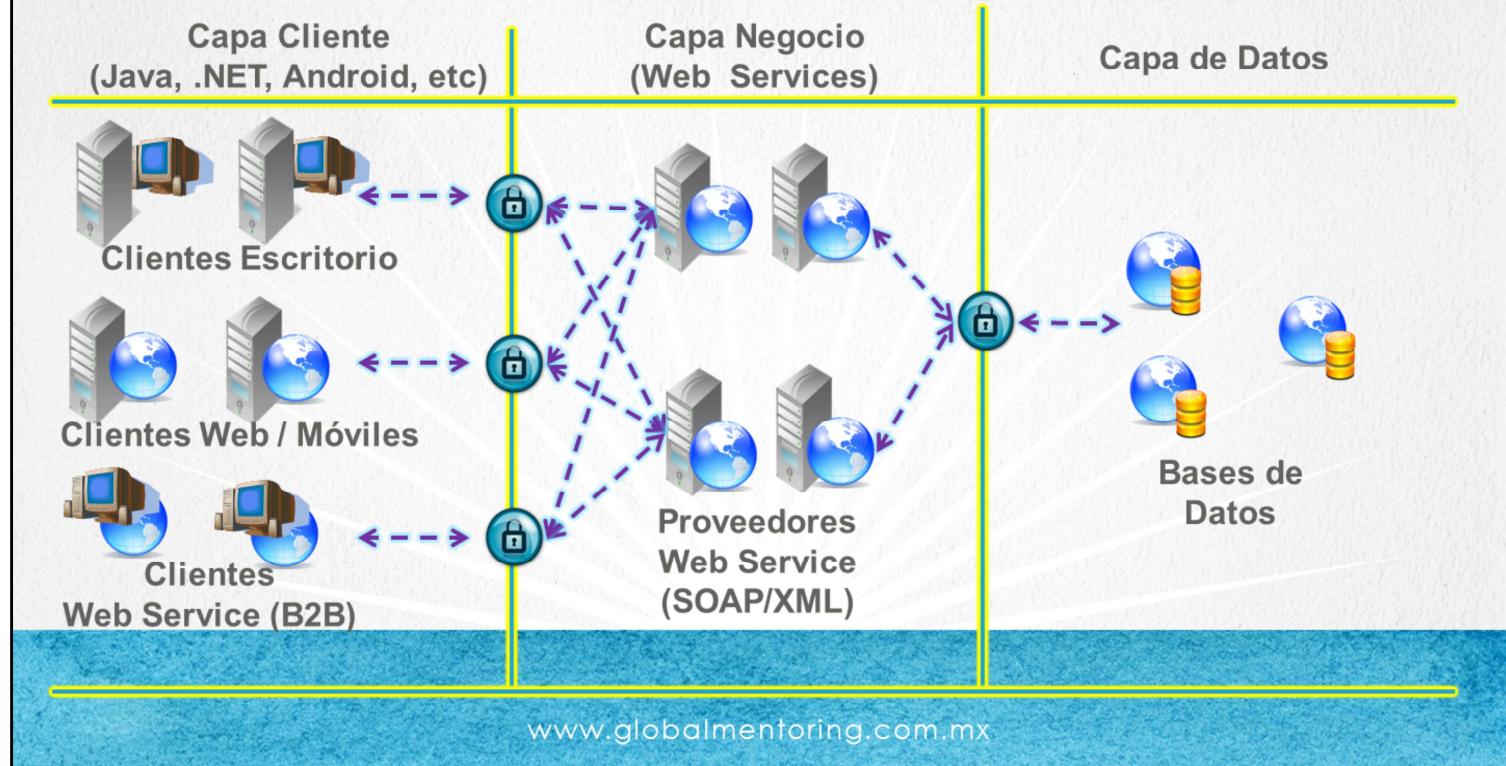
El tema de Web Services ha sido utilizado ya por varios años, a lo largo de estos años se han aprendido técnicas y nuevas formas de establecer una comunicación más eficiente entre sistemas creados en distintas plataformas o tecnologías.

Los Web Services son una tecnología orientada a la intercomunicación de sistemas. Algunas de sus características son:

- ✓ **Interoperabilidad y Portabilidad:** Los sistemas que utilizan Servicios Web permiten intercambiar información entre distintas plataformas y lenguajes de programación utilizando la Web (Intranet/Internet) como base para la comunicación. Una de las formas es utilizar el protocolo en SOAP permitiendo la comunicación vía XML, logrando la independencia de plataforma. Todo esto apoyándose del protocolo HTTP.
- ✓ **Reusabilidad:** Permite reutilizar mucha de la lógica de negocio proveniente de sistemas legados o de nuestros sistemas empresariales actuales.
- ✓ **Disponibilidad:** El objetivo de los servicios web es que se encuentran disponibles en cualquier momento y en cualquier lugar, para cualquier sistema y/o persona que necesite utilizarlos. Además, no requieren de intervención humana incluso en transacciones muy complejas.

Como podemos observar en la figura, en muchas ocasiones necesitamos intercomunicar sistemas de información, los cuales pueden haber sido desarrollados en la misma tecnología o no. Java EE cuenta con dos APIs principales para el desarrollo de Web Services: Java API for XML Web Services (JAX-WS) y Java API for RESTful Web Services (JAX-RS), las cuales estudiaremos.

ARQUITECTURA DE WEB SERVICES



En una aplicación Java Empresarial, los Web Services se pueden ver como una alternativa para exponer los EJBs sin necesidad de utilizar RMI o el uso de Java.

Los EJBs se pueden exponer como Servicios Web, con ello reutilizamos su lógica de negocio. Lo anterior permite a *El Cliente* ser escrito en cualquier lenguaje como Objective-C, .Net, PHP, Android, etc, sin ninguna dependencia con el lenguaje Java.

Si eres nuevo en el tema de Web Services, los RESTful Web Services son más sencillos de aprender. Este tipo de Servicios Web los estudiaremos más adelante.

En esta primera parte estudiaremos los SOAP Web Services, los cuales requieren de un conocimiento básico de temas tales como XML, XSD (esquemas XML) y JAXB. Si no se tiene este conocimiento, mostraremos brevemente en qué consisten estas tecnologías, cómo las aplicaremos, así como una referencia para su estudio más profundo.

Cabe aclarar que debido a que Java EE ha simplificado enormemente el proceso de creación de Web Services, muchas de estas tecnologías estarán únicamente tras bambalinas, y cada que trabajemos en un nuevo Servicio Web se irá adquiriendo poco a poco más de experiencia en cada uno de estos temas y/o tecnologías.

Los Web Services son una parte primordial para el desarrollo de arquitecturas SOA (Service-Oriented Architecture) con el objetivo de integrar aplicaciones creadas en la misma o distintas plataformas.

En los siguientes ejercicios veremos cómo exponer EJB's de tipo Stateless SessionBeans como Servicios Web.

TIPOS DE WEB SERVICES

En Java EE, JAX-WS y JAX-RS son los estándares para crear Web Services:

- ✓ **JAX-WS (Java API for XML Web Services)** es un API que permite abordar requerimientos empresariales más complejos al momento de crear Web Services. También se conocen como **SOAP Web Services**.
- ✓ **JAX-RS (Java API for RESTful Web Services)** es un API más simple de utilizar y de implementar al momento de crear de Web Services. También se conocen como Restful Web Services.
- ✓ Se recomienda utilizar JAX-WS para integrar sistemas empresariales.
- ✓ Se recomienda utilizar JAX-RS para exponer funcionalidad a aplicaciones Web, por ejemplo un cliente IPhone o Android.

CURSO JAVA EE

www.globalmentoring.com.mx

En Java EE, JAX-WS y JAX-RS son los dos estándares para crear Web Services:

- ✓ **JAX-WS: Java API for XML Web Services** permite definir Web Services también conocidos como **SOAP Web Services**. JAX-WS reemplaza el uso de JAX-RPC. La creación de Web Services utilizando JAX-RPC (Remote Procedure Call) fue el estándar en la versión 1.4, sin embargo en este curso no estudiaremos este tema. Para más información en la creación de este tipo de Web Services:

<http://docs.oracle.com/javaee/1.4/tutorial/doc/JAXRPC.html>

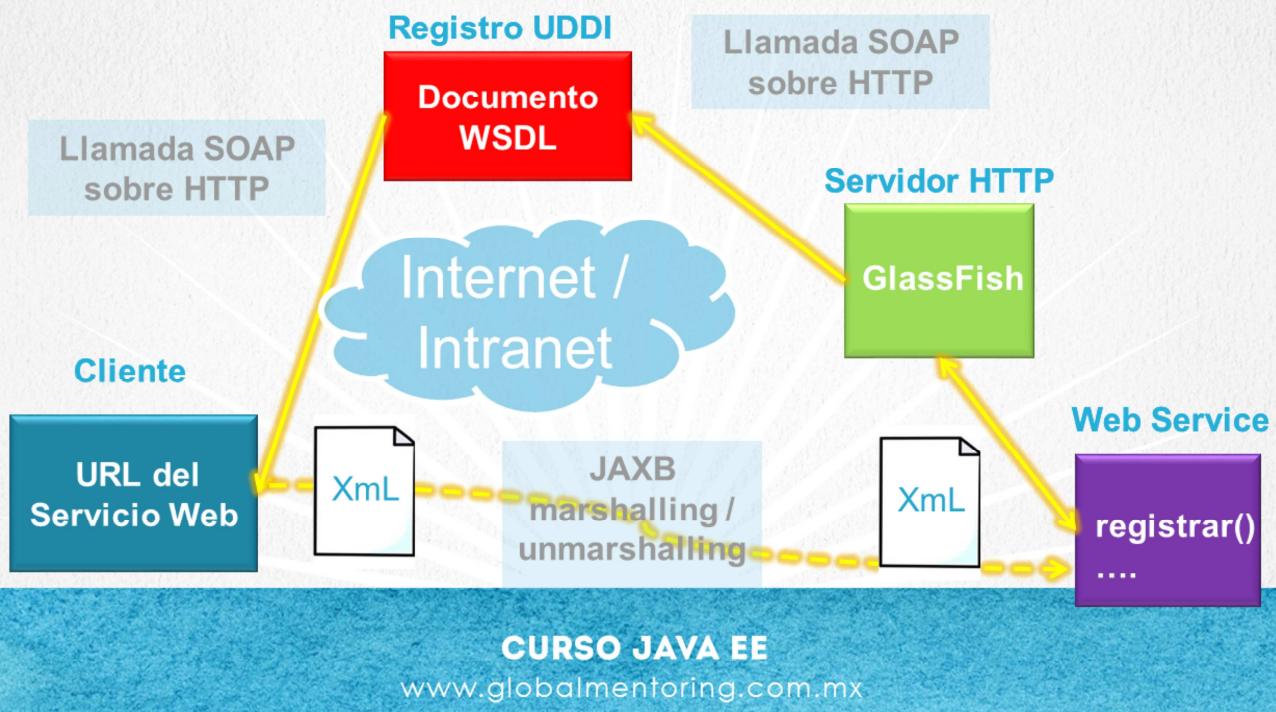
- ✓ **JAX-RS: Java API for RESTful Web Services** permite crear Web Services de acuerdo a la terminología Representational State Transfer (REST) bajo el protocolo HTTP. JAX-RS no es el estándar, pero ha ganado gran popularidad en los últimos años.

Cuando utilizamos Java EE, debemos decidir si utilizamos el API de JAX-WS o JAX-RS. Para tomar este decisión podemos tomar en cuenta los siguientes puntos:

- ✓ JAX-WS: Esta API permite abordar requerimientos más complejos y lleva más años en el mercado. Brinda soporte para los protocolos que son estándar en la industria de software al crear Web Services. Estos estándares proveen una manera muy robusta para agregar seguridad, transaccionalidad, interoperabilidad, entre varias características entre el cliente y el servidor al utilizar Web Services.
- ✓ JAX-RS: Esta API permite crear de manera más simple Web Services, solo que está restringido por el estilo REST, el cual utiliza el protocolo HTTP y sus métodos soportados y códigos de estado para establecer las operaciones básicas de un Servicio Web (GET, POST, PUT, DELETE, etc).

Estos tipos de Web Services son los que estudiaremos.

SOAP WEB SERVICES



En términos simples un Web Service es una plataforma estándar que provee interoperabilidad entre distintas aplicaciones. Para la mayoría de los programadores esto significa envío y recepción de mensajes XML los cuales son transmitidos vía HTTP/HTTPS.

Tanto el lenguaje XML como el protocolo HTTP son un estándar y son ampliamente aceptados para el envío y recepción de información, así, esta información puede ser procesada por múltiples clientes con distintas tecnologías.

Como se observa en la figura, un Web Services se publica en un tipo directorio conocido como UDDI, el cual significa Universal Description, Discovery and Integration. UDDI es un estándar y tiene como objetivo publicar y permitir el acceso a los Web Services a través de mensajes SOAP.

SOAP (Simple Object Access Protocol) es un protocolo estándar el cual define la forma en que se intercambia datos de tipo XML. Este protocolo lo analizaremos en la siguiente lámina.

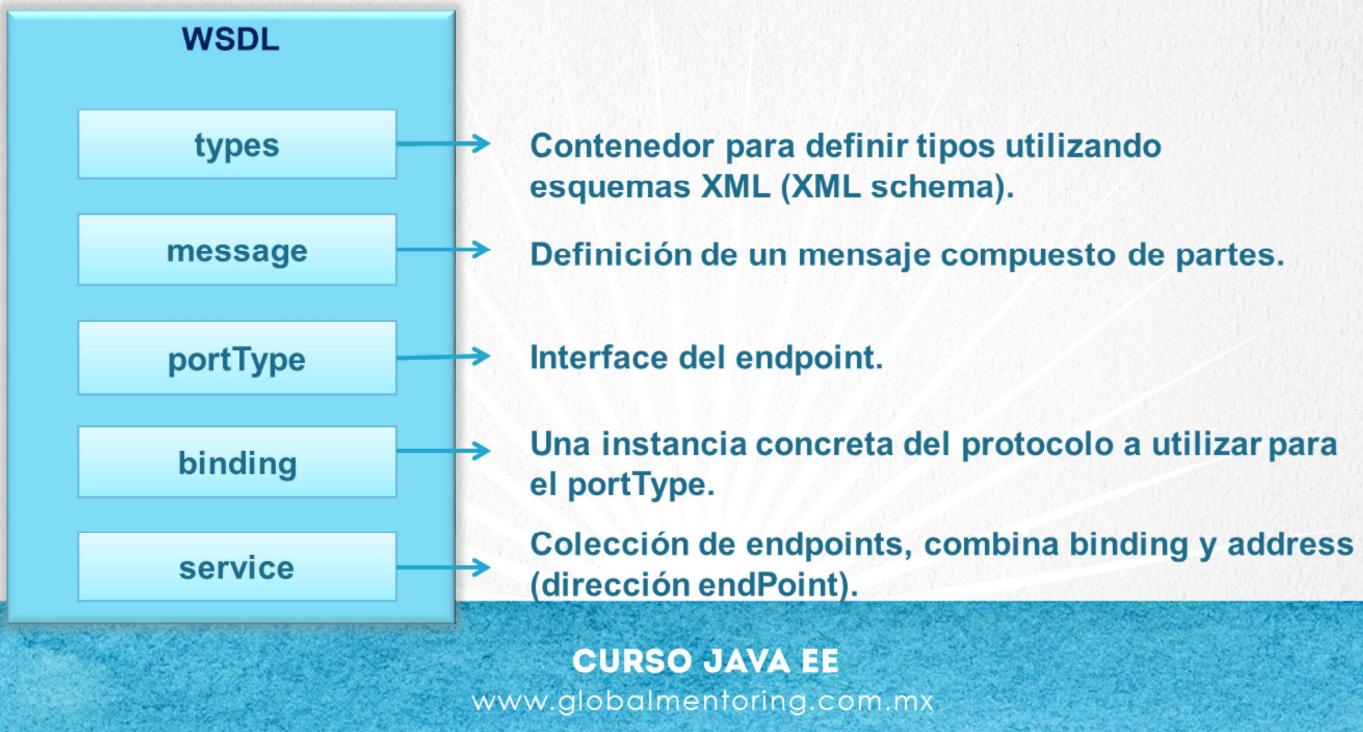
WSDL (Web Services Description Language) nos permite describir los Web Services. WSDL describe la interface pública de nuestros Web Services y se utiliza XML para su descripción. Este descriptor lo analizaremos posteriormente.

Una vez que ya conocemos la ubicación del Web Services a través del WSDL (URI) del mismo, podemos enviar la petición SOAP y así ejecutar el método expuesto del Web Service. En este procedimiento el XML de petición y respuesta suele validarse utilizando XSD (schema).

Una vez llegado el mensaje XML al servidor Java, este debe convertirse en objetos Java. Para esto es común utilizar la tecnología JAXB, el cual nos permitirá convertir objetos Java en documentos XML (marshaling) y viceversa (unmarshaling). Esta tecnología la comentaremos más adelante.

Como podemos observar son varias las tecnologías relacionadas para entender a detalle la creación de SOAP Web Services, sin embargo la buena noticia es que el estándar JAX-WS nos permitirá ocultar y automatizar muchos de los pasos necesarios para crear un Servicio Web, así como la creación del Cliente del Web Services.

WSDL – WEB SERVICES DESCRIPTION LANGUAGE



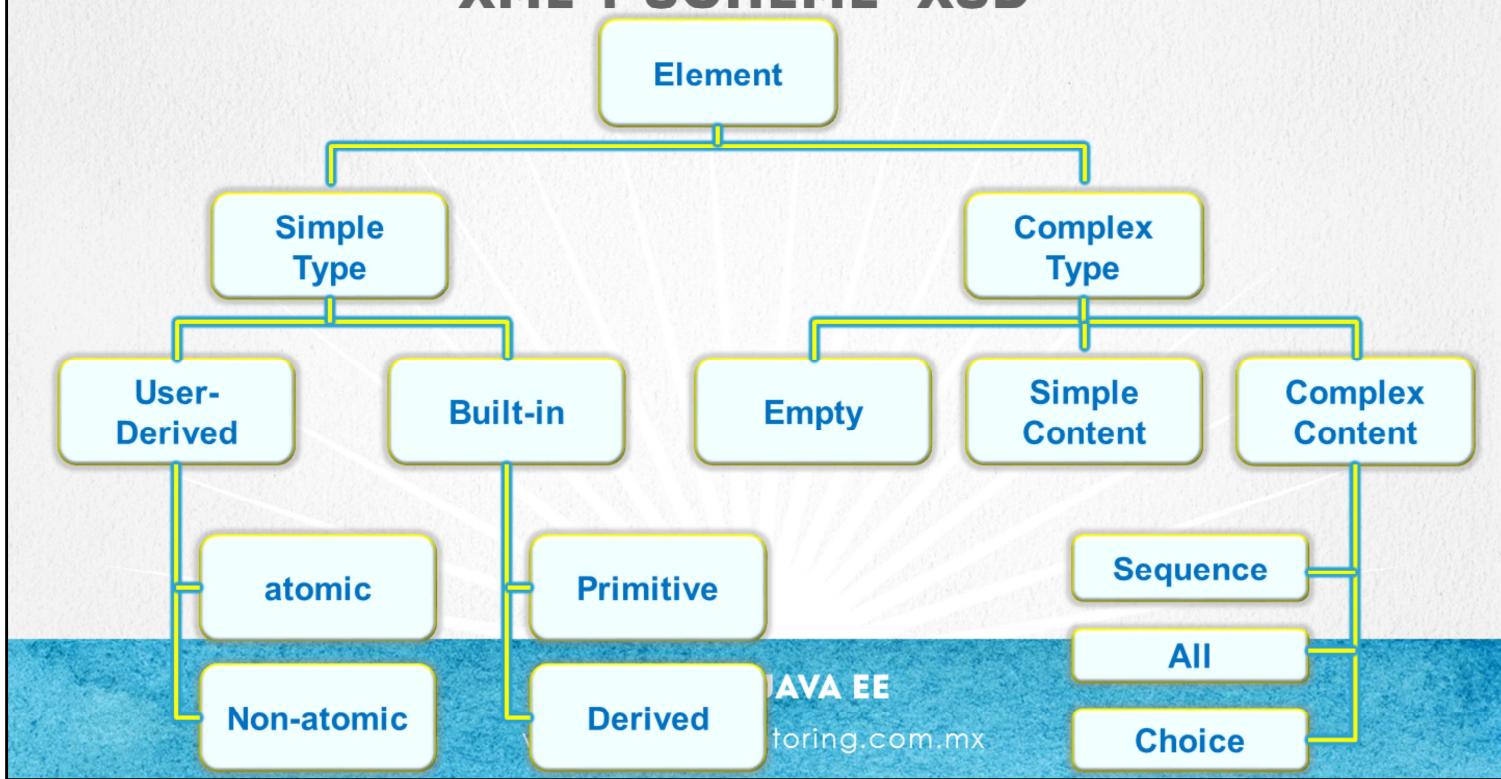
WSDL (Web Services Description Language) es un lenguaje basado en XML, el cual provee la descripción de un servicio Web. De hecho provee una descripción bastante detallada del Servicio Web y con la cual es posible generar tanto el código del Cliente como del Servicio Web asociado. El código generado manejará en automático la conversión de código Java a XML (marshaling) y viceversa (unmarshaling).

En la figura podemos observar la estructura general de un documento WSDL.

- ✓ **Types:** En la sección de Types se definen los tipos a utilizar en el mensaje XML. Estos tipos se definen utilizando esquemas XML (xsd).
- ✓ **Message:** Esta sección define los tipos de mensajes que el Servicio Web soporta. Los mensajes pueden estar compuestos de una o más partes, las cuales pueden ser de entrada (input) o de salida (output).
- ✓ **PortType:** Esta sección define un grupo de operaciones (interface del web service). Cada operación debe tener un nombre único y contiene una combinación de elementos de entrada (input) y salida (output), los cuales hacen referencia a los elementos **Message**.
- ✓ **Binding:** Esta sección relaciona el portType (interface) con el protocolo a utilizar (ej. SOAP).
- ✓ **Service:** Esta sección relaciona la sección **Binding** como servicios y define un endpoint.

Aunque esta información parece en un inicio complicada, conforme se van realizando los primeros Web Services nos iremos familiarizando con estos términos. Sin embargo, si queremos manejar temas más avanzados será necesario estar familiarizados con este terminología y conceptos.

XML Y SCHEME XSD



Existen varias formas de validar un documento XML. Esto es básico al momento de transmitir un mensaje a través de un Web Services. La validación nos permitirá automatizar los mensajes entre el Cliente y el Servidor sin necesidad de intervención humana.

Para validar un documento XML es posible realizarlo por medio de dos técnicas. La primera conocida como DTD (Document Type Definition) y la segunda como XSD (Schema XML). La validación de archivos por DTD es cada vez menos utilizada, debido a que el lenguaje que se utiliza no es XML, y por lo tanto tiene varias limitantes al momento de validar documentos XML con restricciones más avanzadas.

Por otro lado, la validación por XML Schema es cada vez más utilizado para validar la estructura y restricciones de un documento XML. Ya que permite agregar mucho más precisión al momento de validar elementos y atributos complejos de un documento XML, y debido a que está creado con el mismo lenguaje XML, favorece bastante para crear validaciones robustas y flexibles.

En la figura podemos observar las consideraciones a tomar en cuenta para la validación de un documento XML, y se puede observar que son tan detalladas como se necesite, por ejemplo:

```

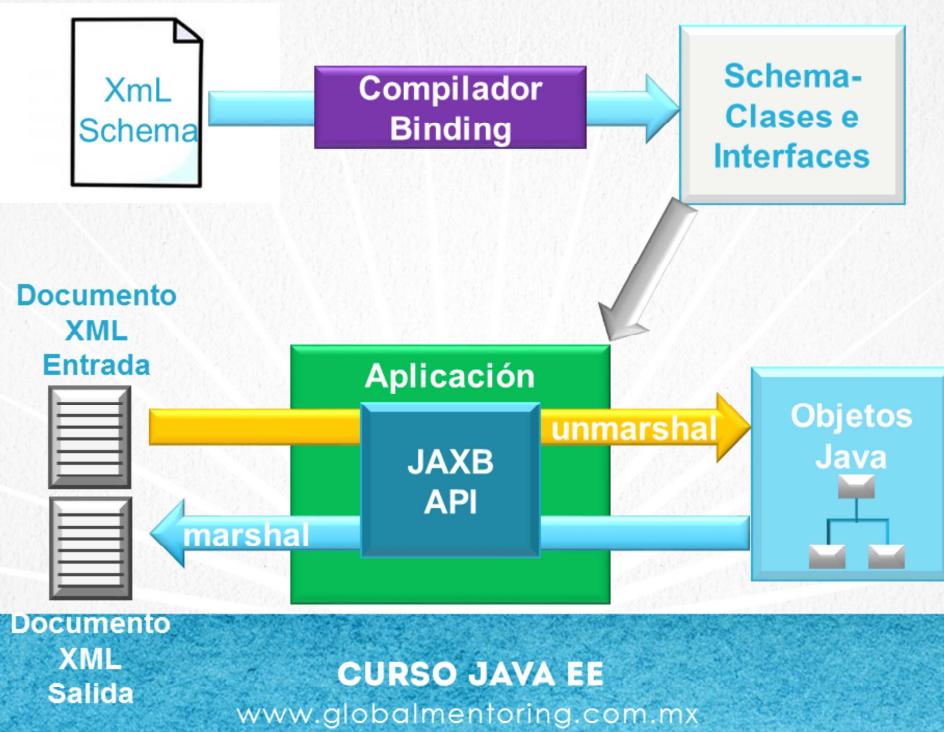
<xsschema xmlns:xss="http://www.w3.org/2001/XMLSchema">
  <xselement name="nota">
    <xsccomplexType>
      <xsequence>
        <xselement name="para" type="xs:string"/>
        <xselement name="de" type="xs:string"/>
        <xselement name="titulo" type="xs:string"/>
        <xselement name="contenido" type="xs:string"/>
      </xsequence>
    </xsccomplexType>
  </xselement>
</xsschema>
  
```

Un ejemplo de un XML validado por el esquema anterior puede ser el siguiente ejemplo:

```

<nota>
  <para>Juan</para>
  <de>Karla</de>
  <titulo>Recordatorio</titulo>
  <contenido>Nos vemos el viernes!</contenido>
</nota>
  
```

JAXB JAVA ARCHITECTURE FOR XML BINDING



CURSO JAVA EE

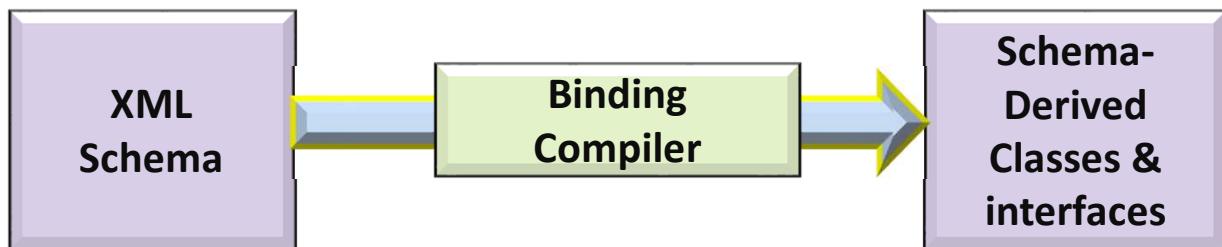
www.globalmentoring.com.mx

Al trabajar con Web Services, es necesario en algún momento convertir los mensajes XML en objetos Java y viceversa. Existen varias APIs para realizar esta labor, sin embargo el estándar en la versión Java EE es el API de JAXB (Java Architecture for XML Binding).

Como podemos observar en la figura, la tecnología JAXB provee dos principales características. La habilidad de convertir un objeto Java en un documento XML (marshal) y viceversa (unmarshaling).

JAXB permite de manera muy simple acceder y procesar documentos XML sin necesidad de conocer a detalle XML u otras API's de procesamiento.

JAXB asocia el esquema (XSD) del documento XML a procesar (binding), y posteriormente genera las clases Java que representan el documento XML (unmarshalling).



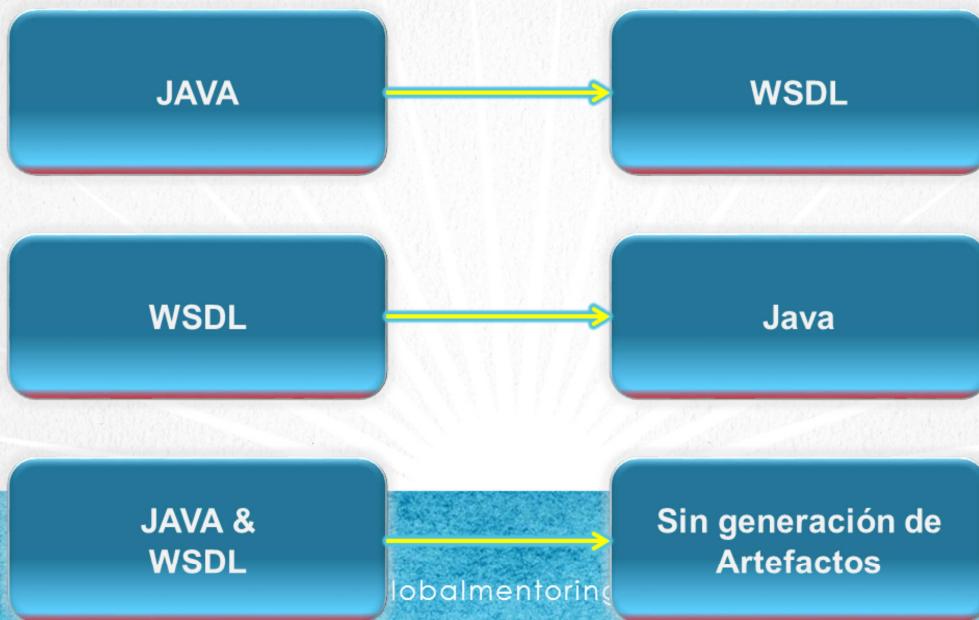
Después de generado el código, se puede acceder y desplegar los datos del documento XML asociado, simplemente utilizando las clases y objetos Java generados. No hay necesidad de utilizar un parser XML o incluso, no hay necesidad de conocer el documento XML original.

También es posible el proceso inverso (marshalling), en el cual a partir de código Java se generan el documento XSD que representa la estructura de los objetos Java y su relación.

ESTRATEGIA GENERACIÓN DE WEB SERVICES

Punto de Inicio

Generación de Artefactos



Cuando creamos un nuevo Servicio Web, hay dos artefactos que deben generarse: El documento WSDL y las clases Java que implementan el Servicio Web. Además, se debe generar el documento XSD (esquema XML) asociado al mensaje XML a intercambiar por el Servicio Web.

Con el documento WSDL y el esquema XSD, un cliente de un Web Services puede ser autogenerado con ciertas herramientas, por el comando `wsimport` de Java. Este comando permite generar el código Java necesario para invocar un Web Service a partir de la URL del WSDL. Existen herramientas similares en otros lenguajes para crear los clientes de los SOAP Web Services de manera muy similar.

Según podemos observar en la figura y enfocándonos en los dos principales artefactos, WSDL y el código Java, podemos tomar diferentes caminos. La estrategia a seleccionar puede ser por cada Web Service de manera independiente, es decir, no es una decisión única.

Cuando ya tenemos listo el código Java, delegamos la generación del Servicio Web al API de JAX-WS, lo que generará de manera automática tanto el documento WSDL y el archivo XSD asociado al mensaje XML a intercambiar, todo esto basado en el código Java y la relación existente en el método Web Service a exponer.

JAX-WS puede exponer métodos de clases Java POJO's o de EJB's. Seleccionar uno u otro dependerá de si queremos aprovechar los beneficios de tener un EJB, o utilizar un POJO para exponer métodos Java como un Servicio Web.

Para exponer un Servicio Web desde una clase Java solamente debemos anotar la clase con `@WebService` y agregar la anotación `@WebMethod` al método a exponer como un Servicio Web. Esto en automático generará el WSDL y el esquema XSD asociado.

Esta es la estrategia que utilizaremos en nuestros ejercicios, ya que expondremos algunos métodos Java de nuestros EJBs como Servicios Web.

EJERCICIOS CURSO JAVA EMPRESARIAL

- **ABRIR LOS ARCHIVOS DE EJERCICIOS EN PDF.**
- **EJERCICIO:** HolaMundo con Web Services con JAX-WS.
- **EJERCICIO:** Creación proyecto SGA con Web Services.



CURSO JAVA EE
www.globalmentoring.com.mx

CURSO ONLINE

JAVA EMPRESARIAL

JAVA EE

Por: Ing. Ubaldo Acosta

**CURSO JAVA EE**www.globalmentoring.com.mx

En Global Mentoring promovemos la Pasión por la Tecnología Java. Te invitamos a visitar nuestro sitio Web donde encontrarás cursos Java Online desde Niveles Básicos, Intermedios y Avanzados, y así te conviertas en un experto programador Java.

A continuación te presentamos nuestro listado de cursos:

- | | |
|---|--|
| <ul style="list-style-type: none">✓ Lógica de Programación✓ Fundamentos de Java✓ Programación con Java✓ Java con JDBC✓ HTML, CSS y JavaScript✓ Servlets y JSP's✓ Struts Framework | <ul style="list-style-type: none">✓ Hibernate Framework & JPA✓ Spring Framework✓ JavaServer Faces✓ Java EE (EJB, JPA y Web Services)✓ JBoss Administration✓ Android con Java✓ HTML5 y CSS3 |
|---|--|

Datos de Contacto:Sitio Web: www.globalmentoring.com.mxEmail: informes@globalmentoring.com.mx