CAPITULO V

Protocolos subyacentes a los Servicios Web XML

CONTENIDO

5.1 Infraestructura de servicios Web XML

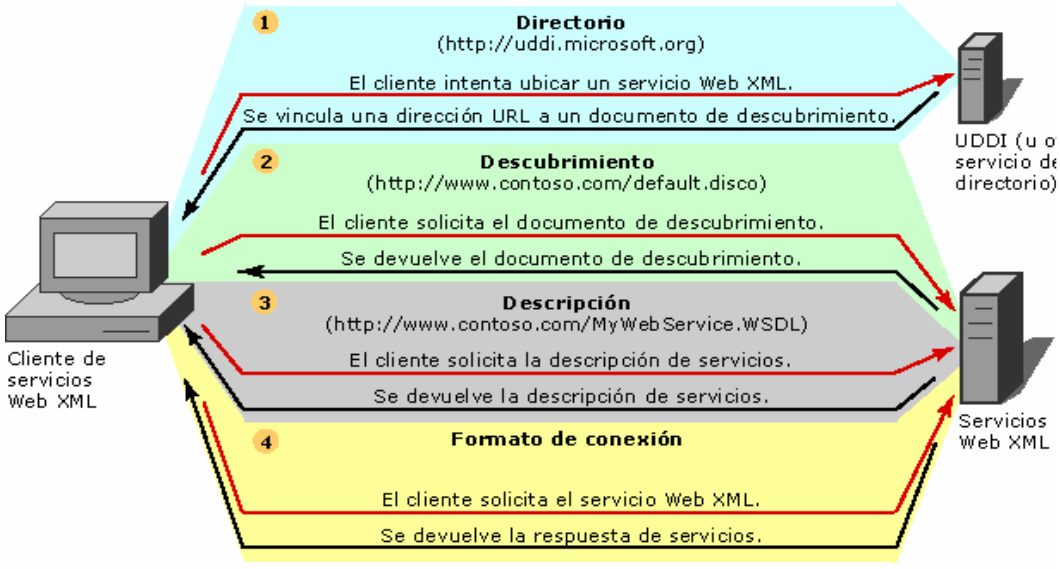

Figura 5.1 ***Infraestructura de los servicios Web [www.msdn/microsoft.com]***

La infraestructura sobre la que se construyen los servicios Web está formada por los siguientes elementos: WWW [0015]

- ***Servicio de directorio.*** Proporcionan una localización centralizada de un servicio Web. Mediante la especificación Universal de Descubrimiento, Descripción e Integración o **UDDI** (Universal Description, Discovery and Integration), aunque puede utilizarse otro tipo de servicio de directorio como **DISCO**

- ***Servicio de localización.*** Localizan uno o más documentos que describen un servicio Web mediante el Lenguaje de Descripción de Servicios Web o **WSDL** (Web Services Description Language).

□ ***Descripción de los servicios Web.*** Describe las interacciones soportadas por un servicio Web. Utiliza el lenguaje WSDL, que permite especificar las interfaces de los servicios Web.

□ ***Formatos de transmisión.*** Para permitir una comunicación lo más universal posible, se utilizan protocolos estándares tales como HTTP-GET, HTTP-POST y SOAP.

□

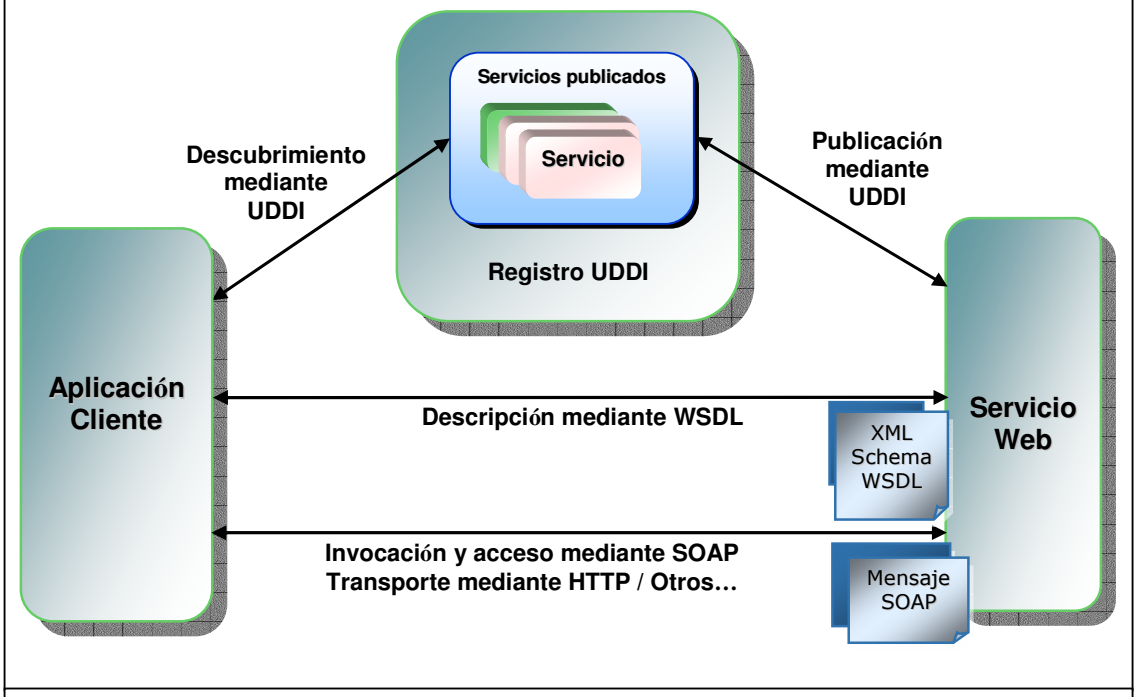

Figura 5.2 ***Proceso recreación de un Servicio Web***

La pila de protocolos de servicios Web está formado por un ***directorio*** para la búsqueda y ubicación del servicio Web; un mecanismo de ***descubrimiento*** para localizar servicios Web XML, una ***descripción de servicio*** para definir cómo se deben utilizar esos servicios y ***formatos de conexión*** estándar para la comunicación.

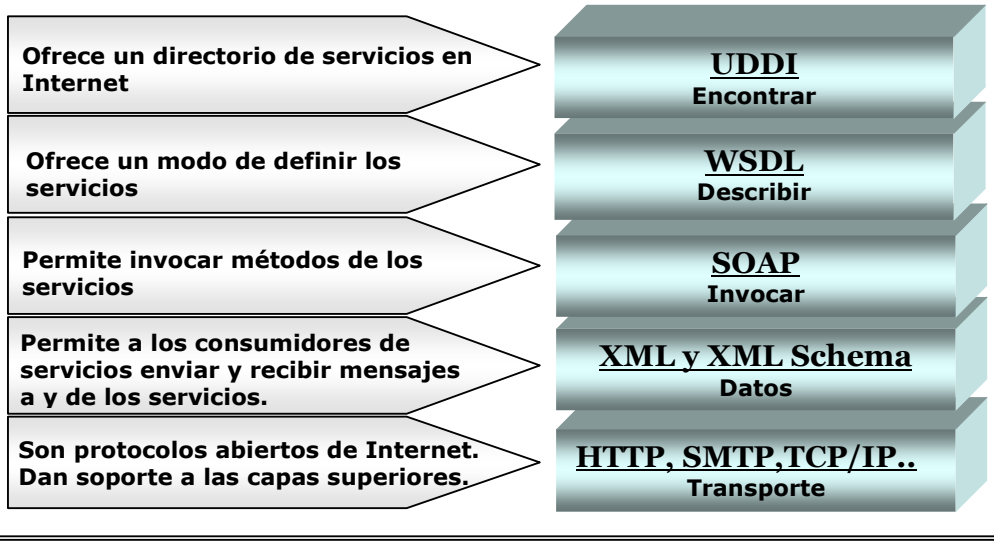

Figura 5.3 ***Pila de los Protocolos subyacentes a los Servicios Web***

5.2 SOAP como protocolo de invocación y comunicación para los servicios Web

Lo esencial en una aplicación es que utilice componentes de otras aplicaciones, incluso aunque éstos estén desarrollados en otros lenguajes. Mediante **CORBA** y **DCOM** era posible, pero un desarrollo con estos estándares no era fácil de lograr peor aún si había un cortafuego entre las máquinas a comunicar, ya que esto dificultaba las comunicaciones.

Figura 5.4 ***Si hay unos cortafuegos entre máquinas se dificultan las comunicaciones.***

SOAP RFC[3288] es la evolución del protocolo **XML-RPC** RFC[3529] que permite la comunicación usando XML como lenguaje común y HTTP como protocolo de transporte. SOAP trabaja sobre HTTP, permitiendo manejar muy bien los cortafuegos. Además permite la ejecución de métodos entre diferentes plataformas, por lo que puede tener clientes que acceden a diferentes servidores programables en cualquier lenguaje y bajo cualquier plataforma. [WWW018]

> **SOAP** Protocolo de Acceso a Objetos Simple (Simple Object Access Protocol) es un protocolo estándar creado por el **W3C** que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. SOAP es uno de los protocolos utilizados en los servicios Web.

En el **procesamiento de peticiones** cuando se hace una petición, el cliente envía ciertos datos que hay que convertir a una representación neutra antes de enviarlos por la red al servidor.

La llamada se transforma en un documento XML en el que van descritos el nombre del método invocado y los parámetros enviados. Ese XML se envía usando el protocolo HTTP hacia el receptor.

Una vez que llega el mensaje XML al receptor, se convierten los datos a la representación utilizada por el lenguaje en que está programado el servidor. El método en el servidor realiza las operaciones y devuelve el resultado realizando las conversiones anteriores en sentido inverso. [WWW019]

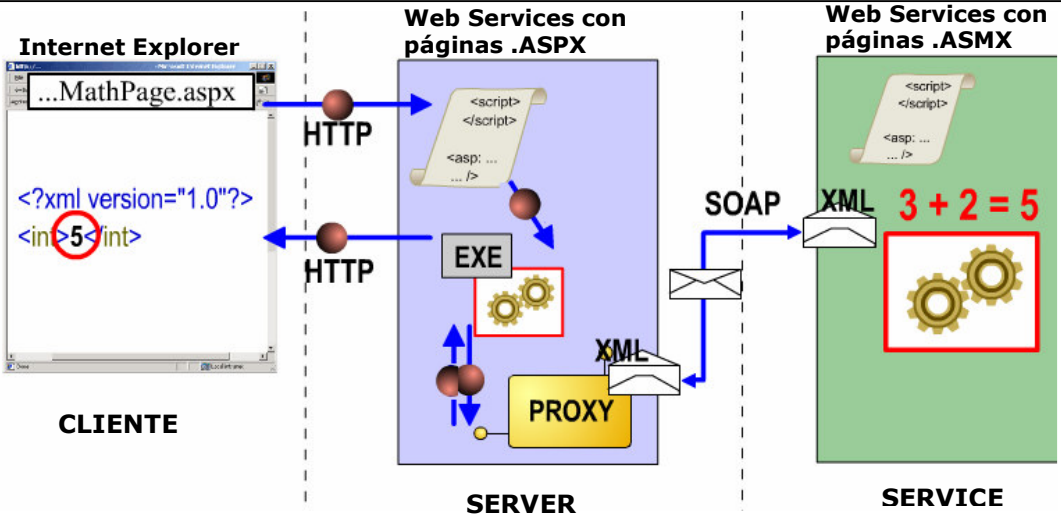

Figura 5.5 ***Arquitectura de solicitud SOAP***

5.2.1 Propiedades de SOAP

- Es un protocolo ligero, sencillo de implementar, probar y usar.
- Se usa para comunicación entre aplicaciones.
- Es un estándar de la industria, creado por un consorcio del cual Microsoft forma parte, adoptado por **W3C** y por varias otras empresas.
- No esta asociado con ningún lenguaje de programación.
- No se encuentra asociado a ningún protocolo de transporte utiliza cualquier protocolo capaz de transmitir texto.
- Está diseñado para comunicarse vía HTTP y utiliza los mismos estándares. Atraviesa "cortafuegos" y ruteadores, que "piensan" que es una comunicación HTTP.

- Está basado en XML, lo que permite que el protocolo no sólo sea más fácil de utilizar sino que también sea muy sólido.

- Se puede utilizar tanto de forma anónima como con autenticación (nombre/clave).

- Las solicitudes SOAP se pueden hacer en tres estándares: GET, POST y SOAP. Los estándares GET y POST son idénticos a las solicitudes hechas por navegadores de Internet. SOAP es un estándar similar a POST, pero las solicitudes se hacen en XML.[WWW020]

5.2.2 Estructura de un mensaje

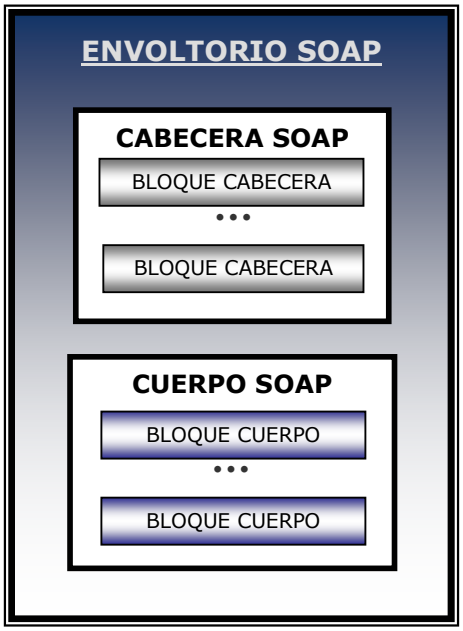

Figura 5.6 ***Estructura de un mensaje SOAP***

Un mensaje SOAP es un documento XML que consta de los siguientes elementos **[RFC3288]**:

- ***Envoltorio (envelope)*** define el contenido del mensaje y la forma de procesarlo.

- ***Cabecera (header)*** es opcional y contiene información referente a la cabecera del mensaje. Define reglas de codificación para expresar las instancias de tipos de datos definidos por la aplicación.

- ***Cuerpo (body)*** contiene la información para representar las llamadas y respuestas a procedimientos remotos.

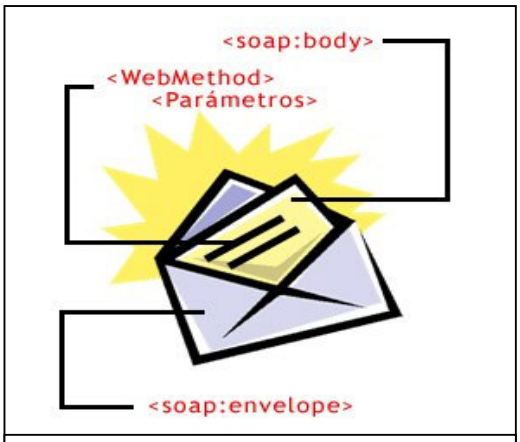

Figura 5.7 ***Elementos de un mensaje SOAP***

El **Envoltorio (Envelope)** define un sobre extensible obligatorio para encapsular los datos y como procesarlos. Constituye la unidad básica de intercambio entre procesadores de mensajes SOAP. Es la única parte obligatoria de la especificación.

En la cabecera **HEADER**, residen los metadatos que describen el cuerpo, detallan cómo se debería procesar y facilitan información adicional sobre el mensaje. El elemento Header es opcional, pero si existe, deberá ser el primer elemento secundario de **ENVELOPE.** El elemento Header está compuesto por cero o más elementos secundarios denominados **bloques de encabezado**. Cada bloque de encabezado deberá constituir un espacio de nombres completo.

El cuerpo **BODY** es donde se encuentran los principales datos del mensaje que contiene la información para representar las llamadas y respuestas a procedimientos remotos.

Las especificaciones de SOAP definen tres atributos que se pueden aplicar a los bloques de encabezado: **ENCODINGSTYLE, ACTOR y MUSTUNDERSTAND**. Estos tres atributos son opcionales y se pueden utilizar otros atributos que desee incluir.

Atributo encodingStyle Indica cómo se codificarán los datos encapsulados. SOAP incluye un mecanismo para codificar datos que contienen información de tipo de datos en atributos XML y cada vez son más usados los encabezados en el esquema XML.

Atributo actor Indica qué nodo debería procesar este bloque de encabezado en concreto. La ausencia del atributo actor implica que el bloque de encabezado se dirige al último destinatario del mensaje SOAP.

Atributo mustUnderstand Determina el modo en que un bloque de encabezado debe comprender y procesar de conformidad con las especificaciones. Si el destinatario no sabe cómo procesar el bloque de encabezado, se generará un error MustUnderstand y se interrumpirá el procesamiento del mensaje. El atributo mustUnderstand es un valor binario y se asumirá que es False cuando éste no se encuentre presente.

5.2.3 Gestión y tratamiento de errores

Cuando ocurre un error en una comunicación con SOAP, dicho error se transmite como un elemento de fallo en el cuerpo del mensaje. Por ejemplo, si se hubiera producido una división por cero, obtendríamos el siguiente documento XML: [WWW021]

```
<SOAP-ENV:Envelope>
 <SOAP-ENV:Body>
  <SOAP-ENV:Fault>
   //proceso de encapsulamiento para envio
        <faultcode>SOAP-ENV:Server</faultcode>
    <faultstring> / by zero </faultstring>
    <faultactor>/soap/servlet/rpcrouter</faultactor>
  </SOAP-ENV:Fault>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figura 5.8 ***Código devuelto de una división por cero en XML***

Una llamada SOAP puede fallar por diversos motivos:

- **Error en el transporte:** puede ocurrir si hay un corte en la comunicación entre el cliente y el servidor
- **Parámetros incorrectos:** si el número de parámetros de la llamada o sus tipos no son los adecuados
- **Error en el servidor:** si se produce un error interno en el método del servidor.

La forma que el cliente tiene de comprobar si hubo un error es utilizando el método **fault().** En caso de que se indique que sí hubo errores, a continuación una explicación e información detallada LIB[004]:

faultcode :	devuelve el código que identifica el error
faultstring :	devuelve una cadena legible explicando el tipo de error
faultdetail :	nos da detalles sobre el error (como un objeto)
<faultactor>:	causante del error
<detail> :	información específica del error

TABLA 5.1 ***Método Fault() para manejo de errores con SOAP***

✓ *Ejemplo de comunicación (sin cabeceras HTTP)*

Suponiendo que se tiene una sencilla aplicación en la que hay una clase HelloServer que tiene un método llamado sayHello() que recibe como parámetro una cadena de caracteres que será el nombre de la persona a quien saludaremos el mensaje generado por el cliente y devuelto por el servidor sería:

```
<SOAP-ENV:Envelope>
 <SOAP-ENV:Body>
  <sayHello>
   <name>Pepe</name>
  </sayHello>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figura 5.9 ***Mensaje generado por el cliente sin cabeceras HTTP***

```
<SOAP-ENV:Envelope>
 <SOAP-ENV:Body>
  <sayHelloResponse>
   <return>Hello, Pepe</return>
  </sayHelloResponse>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Figura 5.10 ***Mensaje generado por el servidor sin cabeceras HTTP***

Como vemos, la principal diferencia entre ambos mensajes (XML) es que la respuesta se encuentra en una etiqueta que se llama con el método invocado más la palabra Response (indicando que es la respuesta).

Al igual que **HTTP-GET** y **HTTP-POST**, **SOAP** es utilizado como mecanismo de paso de mensajes entre clientes y servidores, ambos intercambian mensajes en formato XML, siendo SOAP la especificación que describe el formato de estas peticiones y respuestas en XML. La diferencia de SOAP con HTTP-GET y HTTP-POST reside en que al estar los datos transmitidos codificados en XML, se permite la transmisión de información de mayor complejidad que simples pares nombre-valor. Otra diferencia es que SOAP puede ser utilizado sobre otros protocolos de transporte además de HTTP, como el protocolo de transmisión de correo.

5.3 XML y HTML en la transferencia de datos de servicios Web

5.3.1 Definición de HTML RFC [1866] Es un conjunto de etiquetas que sirven para definir la forma en la que presentar el texto y otros elementos de la página Web.

5.3.2 Definición de XML [RFC3076] (Extensible Markup Language) lenguaje de marcado ampliable o extensible ofrece un formato para la descripción de datos estructurados desarrollado por el consorcio W3C.

XML, es una versión de **SGML** [RFC1874] (Lenguaje Estructurado Generalizado de Marcado). Su objetivo principal es conseguir una página web más semántica. Una de las principales funciones con las que XML nace es suceder al HTML, separando la estructura del contenido y permitiendo el desarrollo compatible con cierta unidad y simplicidad del lenguaje.

La ventaja de XML sobre HTML es que separa la interfaz de usuario de los datos estructurados. Esta separación de datos de la presentación habilita la integración de datos desde diversos orígenes. Es posible convertir información de clientes, pedidos, resultados de investigaciones, pagos de facturas, historiales clínicos, datos de catálogos y mucha otra información a XML.

A continuación se muestra un ejemplo de como Amazón presenta en su Web información sobre los libros.

5.3.3 XML frente a HTML

Código en HTML:

```
<p>
<dt>
<b>
<a href="/exec/obidos/ASIN/0764531999/qid=919015337">
Xml : Lenguaje de Marcado extendible</a></b> ~
<NOBR><font color=#990033>Lleve el libro en 24 horas
</font></NOBR>
<dd> Elliotte Rusty Harold / Paperback / Publicado 1998
<br>
Precio: $31.99 ~
<NOBR><font color =#990033>Ahorre: $8.00
(20%)</font></NOBR>
<br>
<a href="/exec/obidos/ASIN/0764531999/qid=919015337">
<i>Leer mas sobre este libro...</i></a>
```

Código XML:

```
<?xml version="1.0"?>
<libro>
  <titulo>Xml: Lenguaje de Marcado Extendible</titulo>
  <disponible tiempo="24" unidad="horas"/>
  <autor>Elliotte Rusty Harold</autor>
  <formato>Paperback</formato>
  <publicacion>1998</publicacion>
  <precio cantidad="31.99" moneda="dólar"/>
  <descuento cantidad="20"/>
  <enlacelibro
href="/exec/obidos/ASIN/0764531999/qid=919015337"/>
</libro>
```

Figura 5.11 ***Ejemplo de una aplicación XML y HTML***

A continuación se presentan algunos puntos importantes que se deben tener en cuenta acerca de XML y HTML.

XML	HTML
El código xml permite insertar menús, tablas, imágenes o bases de datos en los documentos.	Maneja datos estáticos.
Con XML el usuario puede ordenar los datos o actualizarlos en tiempo real o realizar un pedido	Con HTML se pueden hacer accesos a información comparativa, pero nada más.
Es un lenguaje estructurado.	Es un estándar de almacenamiento estructurado.
XML está diseñado para describir datos y está centrado en lo que son los datos.	HTML está diseñado para mostrar datos y se centra en la apariencia de los datos.
Posee etiquetas propias de XML y son ilimitadas.	Las etiquetas HTML están predefinidas; los autores de HTML sólo pueden utilizar etiquetas compatibles con el estándar HTML .

TABLA 5.2 ***Diferencias entre XML y HTML***

Ventajas XML	Ventajas HTML
Los formatos XML se basan en texto, fácil de documentar y depurar.	Posee una sintaxis simple con un conjunto fijo de etiquetas.
Variedad de estructura de datos	Permite crear fácilmente documentos multimedia al incorporar imágenes y sonido.
El análisis XML está perfectamente definido y ampliamente implementado, lo que posibilita la recuperación de información en diversos entornos.	Permite enlazar varios elemento entre si.
Facilita la creación de documentos internacionalizados.	Papel importante en crecimiento de internet.

TABLA 5.3 ***Ventajas de XML Y HTML***

Desventajas XML	Desventajas HTML
Consumen más ancho de banda de red y espacio de almacenamiento, requiere más tiempo de procesador.	✓ Gestión de enlaces rotos
	✓ Intercambio de datos
El análisis XML puede ser más lento que el análisis muy optimizado de los formatos binarios y puede requerir más memoria.	✓ Extensibilidad
	✓ Estructura lógica
Distingue entre mayúscula y minúsculas	✓ Reutilización de datos
	✓ No es orientado a objetos

TABLA 5.4 ***Desventajas de XML y HTML***

5.3.4 Características de XML

- **Es una arquitectura más abierta y extensible.** No se necesita versiones para que puedan funcionar en futuros navegadores.

- **Integración de los datos.** Puede hacer el intercambio de documentos entre las aplicaciones tanto en el propio PC como en una red local o extensa.

- **Datos compuestos de múltiples aplicaciones.** Su extensibilidad y flexibilidad permitirá agrupar una variedad amplia de aplicaciones, desde páginas Web hasta bases de datos.

- **Gestión y manipulación de los datos desde el propio cliente web.**

- **Los motores de búsqueda devolverán respuestas más adecuadas y precisas,** ya que la codificación del contenido web en XML consigue que la estructura de la información resulte más accesible.

- **El concepto de "hipertexto" se desarrollará ampliamente permite denominación independiente de la ubicación,** enlaces bidireccionales, enlaces que pueden especificarse y gestionarse desde fuera del documento, hiperenlaces múltiples, enlaces agrupados, atributos para los enlaces, etc. Creado a través del Lenguaje de enlaces extensible (XLL).

- **Exportabilidad a otros formatos.** El documento maestro de la edición electrónica podría ser un documento XML que se integraría en el formato deseado de manera directa.

5.3.5 Estructura de XML

El metalenguaje XML consta de cuatro especificaciones (el propio XML sienta las bases sintácticas y el alcance de su implementación): [WWW022]

- **DTD** (Document Type Definition) [RFC3017] Definición del tipo de documento. Es 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. El DTD del XML es opcional. En tareas sencillas no es necesario su utilización.

- **XSL** (eXtensible Stylesheet Language) Define o implementa el lenguaje de estilo de los documentos escritos para XML. Permite modificar el aspecto de un documento.

- **XLL** (eXtensible Linking Language) Define el modo de enlace entre diferentes enlaces. Consta de dos especificaciones:

  XLink determina el documento que se va a enlazar.
  XPointer marca el lugar exacto de dicho documento.

5.3.6 XML y Los Servicios Web

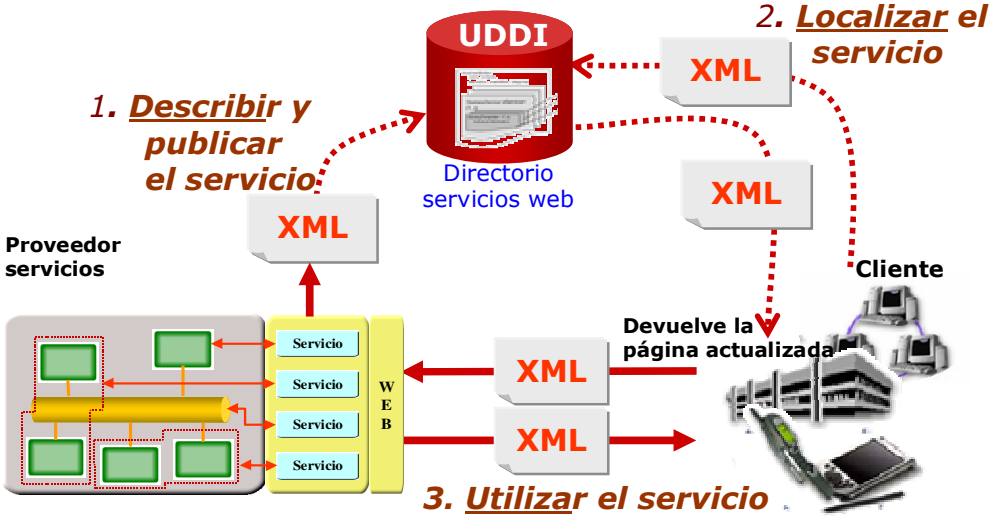

Figura 5.12 ***Arquitectura de un servicio Web mediante XML***

Xml es utilizado en los servicios Web por las siguientes razones:

- ✓ **Es un estándar abierto** es reconocido mundialmente ya que la gran mayoría de software de escritorio, sistema operativo, aplicaciones móviles permiten la compatibilidad, esto lo hace muy potente en la comunicación entre distintas plataformas de software y hardware (y si este es objetivo de los Servicios Web) [WWW022].

- ✓ **Simplicidad de sintaxis** el código XML es sencillo y la representación de los datos es entendible es flexible a la hora de representar datos y basta con contar con cualquier editor de texto para aprender unas cuantas intrusiones básicas y estar en condiciones de escribir código siendo el lenguaje ideal para utilizarlo en los servicios Web.

- ✓ **Independencia del protocolo de Transporte**, XML es un lenguaje de Marcado de Texto y no necesita ningún protocolo de trasporte especial, solo necesita de un protocolo que pueda transferir texto o documentos simples en el mercado existen protocolos con estas características como HTTP y SMTP.

5.4 Lenguaje de definición de servicios Web WSDL (Web Service Description Languaje)

Cuando una aplicación expone alguna funcionalidad, se puede acceder a ella a través de ciertas operaciones que requieren que le pasemos alguna información.

Una vez que las operaciones se completan, la aplicación devuelve los resultados al cliente. Esos intercambios de información necesitan un protocolo, lo cual implica que el desarrollador debe haber descrito las interfaces de acceso al servicio.

> WSDL nos permite **describir y localizar** servicios Web. Es un documento XML que describe un conjunto de mensajes SOAP con una estructura determinada y cómo se realiza el intercambio de mensajes.

En SOAP no hay una forma estándar de hacer especificaciones. Sin embargo, IBM y Microsoft han propuesto al W3C (por lo tanto aún no es un estándar) el lenguaje de descripción WSDL, que permite describir una aplicación (servicio) para indicar qué hace, cómo hace su función, y cómo los clientes pueden utilizarla.

Ventajas de WSDL

- **WSDL** facilita **escribir y mantener servicios** mediante una aproximación estructurada para definir interfaces Web.
- **WSDL** facilita **el acceso a esos servicios web** reduciendo el código que hay que escribir para hacer un cliente
- **WSDL** facilita **hacer cambios para ampliar los servicios**, reduciendo la posibilidad de que los clientes dejen de funcionar al llamar a esos servicios

Desventajas de WSDL

- **WSDL** no proveen versión.
- **WSDL** carece de la posibilidad de especificar una secuencia de las operaciones necesarias en un intercambio de mensajes (por ejemplo, un login previo).

5.4.1 Estructura de los documentos WSDL

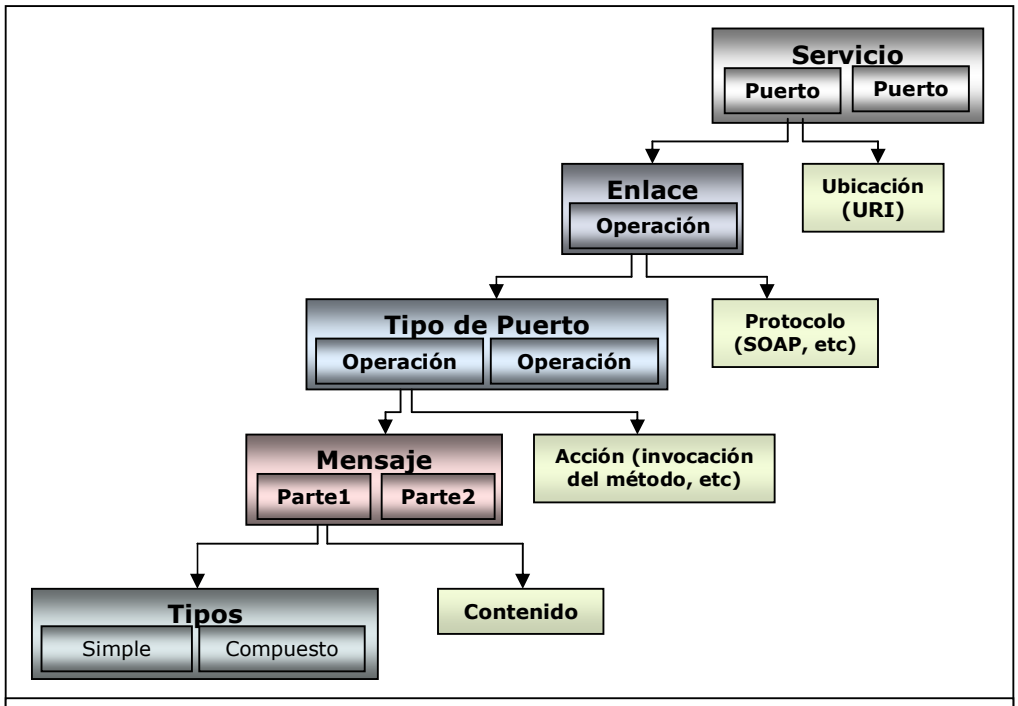

Figura 5.13 ***Elementos de la estructura WSDL***

La estructura del WSDL define seis elementos: [WWW 023]

ELEMENTO	DESCRIPCION
<types>	Proporciona definiciones del tipo de datos utilizado para describir el intercambio de mensajes.
<message>	Representa una definición abstracta de los datos que se están transmitiendo. Un mensaje se compone de partes lógicas, cada una de las cuales está asociada a una definición dentro de algún sistema de tipos.
<portType>	Definen las operaciones permitidas y los mensajes intercambiados. Cada operación hace referencia a un mensaje entrante y a mensajes salientes. Podría compararse con una librería de funciones o una clase.
<binding>	Especificación del protocolo y del formato de datos para un tipo de puerto determinado. Define los protocolos de comunicación usados
Port	Punto final único que se define como la combinación de un enlace y una dirección de red. Definen la dirección de un binding
Service	Colección de puntos finales relacionados.

Tabla 5.5 ***Elementos de la estructura WSDL***

La descripción de un servicio Web especifica la interfaz abstracta a través de la cual un cliente puede acceder al servicio (los detalles de cómo se debe utilizar). Eso se consigue definiendo cuatro elementos: datos, mensajes, interfaces y servicios.

Un **servicio** se compone de una colección **de puertos** (direcciones que implementan el servicio). Un **puerto** tiene una **definición abstracta** (tipo de puerto) y una **definición concreta** (binding, lazo). Los **tipos de puerto** funcionan como la especificación de la interfaz software, y se componen de colecciones de operaciones que definen los intercambios de mensajes. Los bindings (lazos) indican los protocolos usados por cada puerto. Por último, un mensaje es una colección de datos de algún tipo.

```
<?xml version="1.0" enconding="UTF-8"?>
<wsdl:definitions>
   <wsdl:types>
    ...
   </wsdl:types>
   <wsdl:message>
      <part name="parametro" type="xsd:string"/>
   </wsdl:message>
   <wsdl:portType>
      <wsdl:operation name="funcion">
       ...
      </wsdl:operation>
   </wsdl:portType>
   <wsdl:binding>
    ...
   </wsdl:binding>
   <wsdl:service name="Hola Servivios Web">
    ...
   </wsdl:service>
</wsdl:definitions>
```

Figura 5.14 ***Estructura de un documento WSDL mediante un servicio Hola Mundo***

WSDL define cuatro tipos de operaciones [WWW023]:

- ➔ ***Request-response(petición-respuesta)*** comunicación del tipo RPC en la que le cliente realiza una petición y el servidor envía la correspondiente respuesta.

- ➔ ***One-way (un-sentido)*** Comunicación del estilo documento en la que el cliente envía un mensaje pero no recibe una respuesta del servidor indicando el resultado del mensaje procesado.

- ➔ ***Solicit-response(solicitud-respuesta)*** La contraria a la operación petición-respuesta. El servidor envía una petición y el cliente le envía de vuelta una respuesta.

- ➔ ***Notification (Notificación)*** La contraria a la operación un-sentido el servidor envía una comunicación del estilo documento al cliente.

5.5 UDDI (Universal Description, Discovery and Integration) como medio de localización de servicios Web.

Es fundamental tener un medio de localizar los servicios web, lo cual es una tarea más difícil conforme crece el número de servicios disponibles.

La especificación UDDI simplifica esa tarea, permitiendo a una organización publicar información sobre los servicios que ofrece y localizar información sobre servicios web que necesita utilizar. UDDI es simplemente un repositorio de documentos XML (y un esquema) que define un mensaje SOAP para el registro y petición de información.

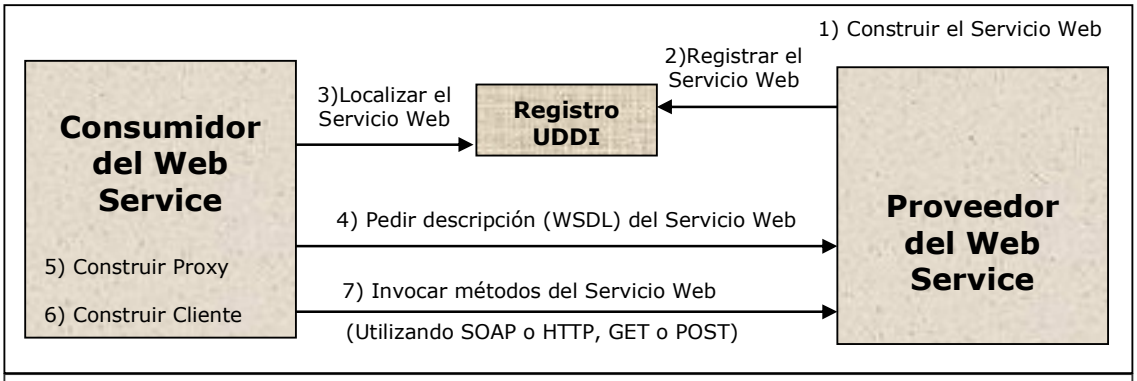

Figura 5.15 ***Descripción del registro UDDI***

Características de UDDI

UDDI es un sistema ideado para describir servicios (junto con WSDL) y localizar empresas que ofrezcan estos servicios.

- UDDI significa "Descripción, Localización e Integración Universales"

- Es un directorio para almacenar información sobre servicios web; guarda las interfaces de esos servicios descritos en WSDL

- UDDI utiliza SOAP para llevar a cabo las comunicaciones.

- Está desarrollado e integrado en la plataforma .NET de Microsoft.

- UDDI ha sido propuesto por Dell, Fujitsu, HP, Hitachi, IBM, Intel, Microsoft, Oracle, SAP y Sun (entre otros).

5.5.1 Estructura central del modelo de datos UDDI

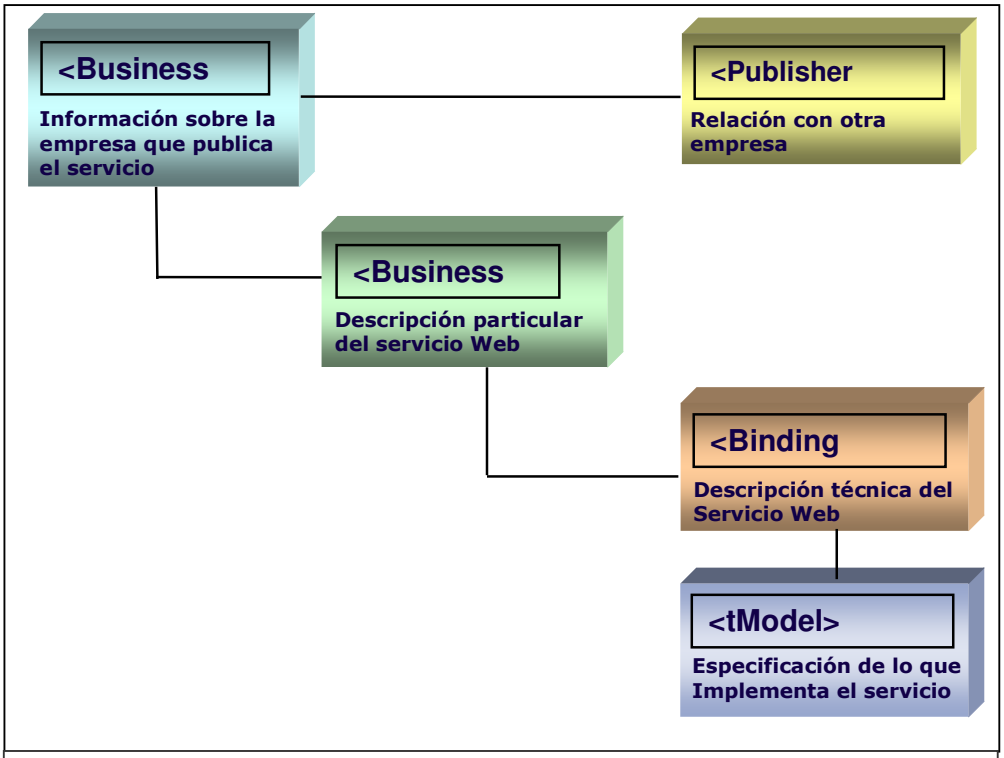

Figura 5.16 ***Modelo de datos UDDI***

La información que compone un registro UDDI consiste por el momento de cuatro tipos de estructuras de datos: [WWW024]

a. ***Estructura <businessEntity>*** contiene toda la información administrativa necesaria sobre la empresa y los servicios que ofrece (páginas blancas y páginas amarillas). Esta estructura puede considerarse la estructura raíz de la entrada de registro.

b. ***Estructura <publisherAssertion>*** Esta estructura contiene elementos que describen las relaciones de la empresa con otras empresas, puesto que muchas empresas o grupos de empresas pueden no quedar bien representados por una única businessEntity o entrada en el registro.

c. ***Estructura <businessService>*** Cada estructura businessEntity hace referencia a una o varias estructuras businessService. Una estructura businessService describe un conjunto de servicios, debidamente categorizados, que ofrece una empresa. Esta estructura no pertenece a la businessEntity, sino que puede ser compartida por varias empresas.

d. ***Estructura <bindingTemplate>*** La estructura bindingTemplate contiene una descripción técnica de uno de los servicios de la estructura businessService. Un bindingTemplate pertenece por tanto a un único businessService.

e. ***Estructura <tModel>*** Uno de los elementos clave de UDDI es la estructura tModel. Un tModel describe la especificación, el comportamiento, el concepto, o incluso el diseño compartido de un servicio. Proporciona información específica sobre como interactuar con el servicio. Una estructura tModel incluye su nombre, un identificador de referencia, una descripción (opcional), y un URL (opcional) que apunta a una localización donde puede encontrarse más información sobre el tModel.

5.5.2 Almacén de información UDDI

En la especificación UDDI el servicio almacena información de tres tipos que son:[WWW024]

a. ***La Sección Blanca,*** especifica la dirección, contactos, e identificadores de la empresa es muy similar a la información que aparece en el directorio telefónico que incluye nombre, teléfono, dirección, etc.

b. ***La Sección Amarilla*** es muy similar a su equivalente telefónico. Estos pueden tener información sobre la categoría de la empresa y los servicios que ofrece, e incluyen categorías de catalogación industrial tradicionales, ubicación geográfica, etc. Mediante el uso de códigos y claves predeterminadas, los negocios se pueden registrar y así facilitar a otros servicios la búsqueda usando estos índices de clasificación.

c. ***La Sección Verde*** contiene la información técnica acerca de los servicios ofrecidos por los negocios. Se incluyen referencias de especificaciones de servicios Web así como también información complementaria para los mecanismos diversos de búsqueda basados en URL. Como por ejemplo: Información sobre los servicios (Web) que ofrece una empresa.

UDDI provee dos categorías de API: ***El API de publicación y el API de consulta*** *[LIB004]*.

El ***API de publicación***, provee el mecanismo para que los proveedores de servicios se registren ellos mismos y sus servicios en el Registro UDDI. El ***API de consulta*** permite a los subscriptores de servicios buscar los servicios disponibles. El API de consulta provee dos tipos de llamados, *un mecanismo de búsqueda y un mecanismo de obtención,* cuando ya se tiene disponible toda la información referente a la disponibilidad de un servicio.

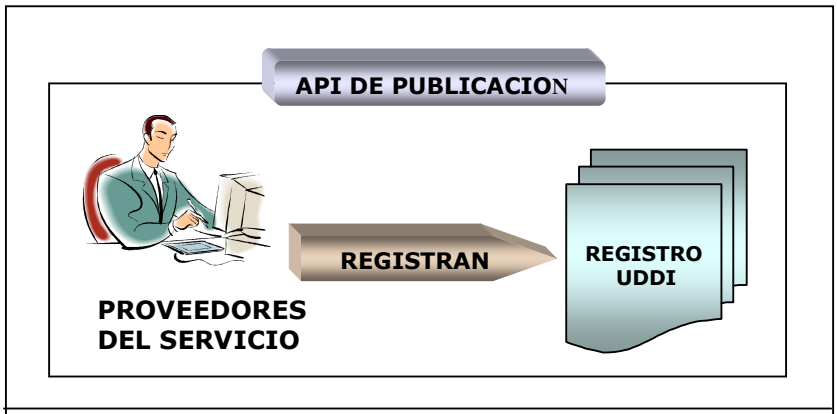

Figura 5.17 ***Descripción API de publicación***

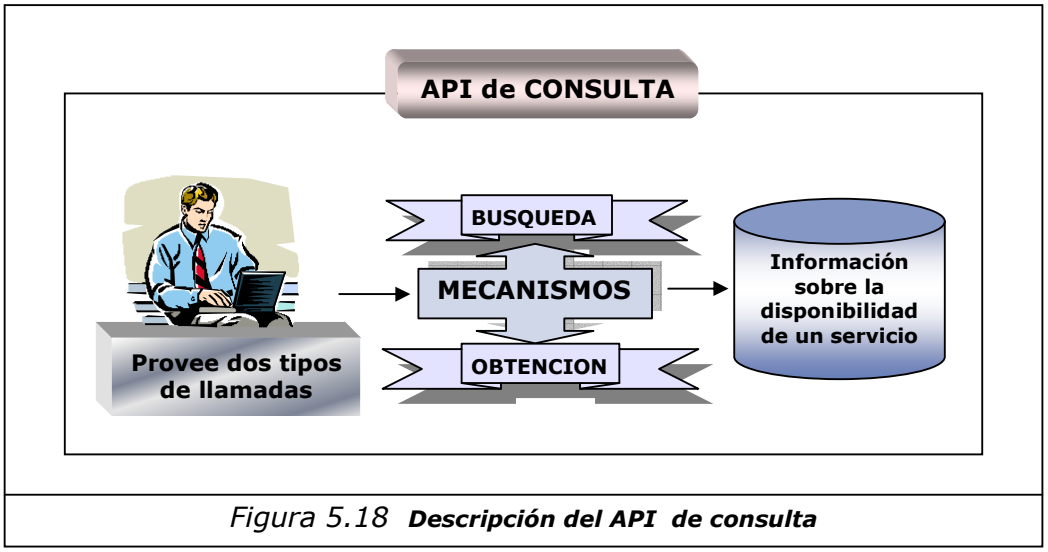

Figura 5.18 ***Descripción del API de consulta***

La especificación UDDI de los servicios se almacena en un ***UBR (UDDI Business Registry)*** al que los clientes pueden acceder.

El acceso al **UBR** se realiza utilizando una API estándar y SOAP. El acceso puede ser tanto para registrar servicios como para realizar consultas o actualizar los datos de un servicio concreto.

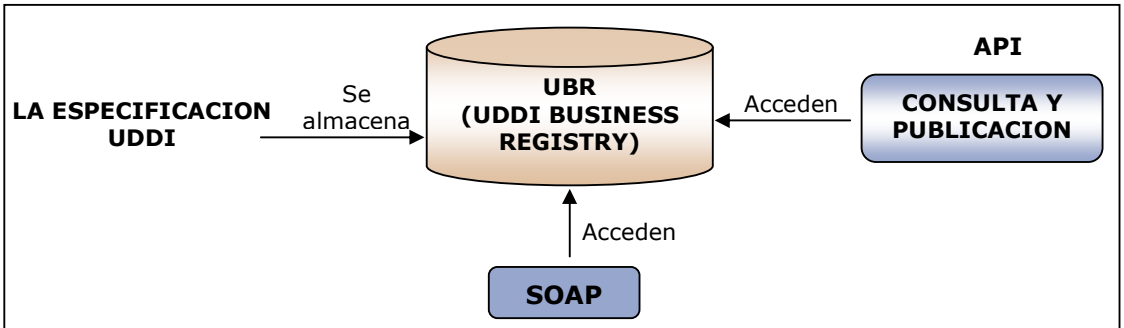

Figura 5.19 ***Descripción del almacenamiento UDDI***

5.5.3 Pasos para el registro y publicación UDDI

a. Modelar la compañía y sus servicios en UDDI. Determinar tModels (modelos tecnológicos) o ficheros WSDL para nuestro servicio, que representan las interfaces, abstracciones técnicas y metadatos del servicio.

b. Si el servicio está basado en un fichero WSDL ya registrado, se necesita saber la URL hasta ese fichero y el identificador que le asignó UDDI al registrarlo. Si el servicio está basado en un fichero WSDL no registrado, hay que registrarlo previamente.

c. Durante el proceso de registro hay que saber las categorías e identificadores de la empresa.

d. Decidir el nivel de seguridad

e. Una vez modelada la empresa y los servicios, obtendremos una cuenta en el sistema de registro UDDI a través del sistema web.

Los pasos para realizar una publicación UDDI son:

a. Conseguir una cuenta en el proveedor del registro UDDI

b. Añadir una nueva empresa.

c. Categorización de la empresa, según la taxonomía que utilice el registro

d. Publicación de los servicios ofrecidos.

UDDI puede ayudarnos a resolver los siguientes problemas:

- ✓ Descubrir la empresa más adecuada de entre las muchas presentes en Internet
- ✓ Obtener información sobre cómo contactar con esa empresa
- ✓ Conseguir nuevos clientes y facilitar el acceso a los actuales
- ✓ Incrementando los servicios ofertados y extendiendo el mercado al que se puede acceder.
- ✓ Describir servicios y procesos empresariales en un entorno seguro y fácil de usar.

5.6 HTTP como protocolo de transporte de los servicios web

HTTP-GET y HTTP-POST son protocolos estándar que utilizan ***HTTP (Hypertext Transfer Protocol, Protocolo de transferencia de hipertexto)*** para la codificación y el análisis de parámetros de nombre y valor, junto con la semántica de solicitudes asociada. Cada uno se compone de una serie de encabezados de solicitud HTTP que, entre otras características, definen lo que el cliente solicita al servidor, el cual responde con una serie de encabezados de respuesta HTTP y los datos solicitados.

HTTP maneja cuatro tipos de métodos que son: [RFC2616]

- ✓ **GET:** recupera una página Web
- ✓ **POST:** recupera una página web dinámica
- ✓ **PUT:** escribe un fichero en el servidor
- ✓ **DELETE**: borra un fichero en el servidor

HTTP se ayuda con el manejo de ***proxies.*** Un ***Proxy*** es un intermediario entre un cliente y el servidor el cual es importante para reducir la carga en los servidores y mejorar la escalabilidad en las aplicaciones así como también solucionar problemas de consistencia. WWW[020]

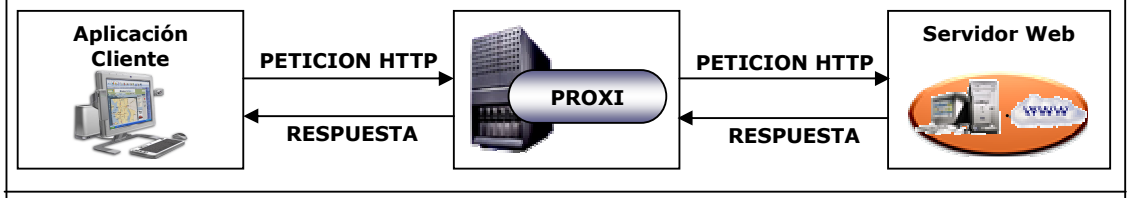

Figura 5.20 ***Peticiones HTTP entre un cliente y el servidor mediante un proxy***

5.6.1 Anatomía del período de duración de un servicio Web XML

El proceso que se produce cuando se realiza una llamada a un servicio Web XML es similar al que tiene lugar cuando se llama a un método regular. La diferencia principal es que, en vez de llamar a un método que se encuentra en la aplicación cliente, se genera un mensaje de solicitud sobre el transporte especificado, como HTTP. Dado que el método del servicio Web XML se puede encontrar en un equipo diferente, la información que tiene que procesar el servicio Web XML debe pasar por la red al servidor que aloja el servicio Web XML. El servicio Web XML procesa la información y devuelve el resultado, a través de la red, a la aplicación cliente.

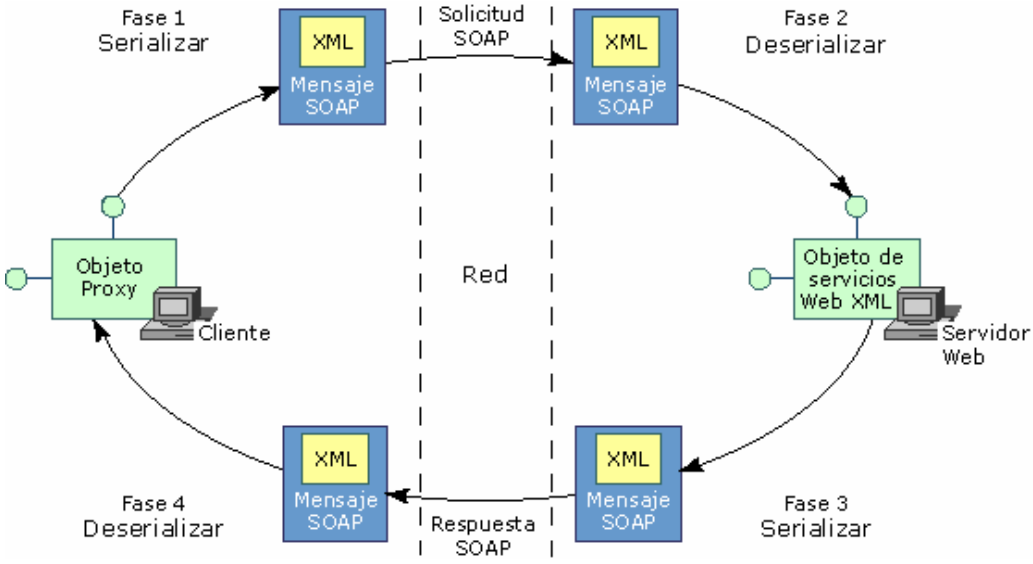

Figura 5.21 ***Proceso de comunicación entre un cliente y un servicio Web XML***.

A continuación se describe la secuencia de eventos que se producen cuando se llama a un servicio Web XML:

a. El cliente crea una nueva instancia de una clase de proxy de servicio Web XML. Este objeto reside en el mismo equipo que el cliente.

b. El cliente invoca un método en la clase de proxy.

c. La infraestructura del equipo cliente serializa los argumentos del método del servicio Web XML en un mensaje SOAP y lo envía a través de la red al servicio Web XML.

d. La infraestructura recibe el mensaje SOAP y deserializa el XML. Crea una instancia de la clase que implementa el servicio Web XML e invoca el método de servicio Web XML; el XML deserializado se pasa como argumentos.

e. El método de servicio Web XML ejecuta el código y establece finalmente el valor devuelto y los parámetros out.

f. La infraestructura del servidor Web serializa en un mensaje SOAP el valor devuelto y los parámetros de salida, y lo devuelve al cliente a través de la red.

g. La infraestructura de servicios Web XML, en el equipo cliente, recibe el mensaje SOAP, deserializa el XML en el valor devuelto y los parámetros out, y los pasa a la instancia de la clase de proxy.

h. El cliente recibe el valor devuelto y los parámetros out.