

## 1. INTRODUCCIÓN.

Al igual que hicimos en la UT8, recordaremos ciertos conceptos vistos en la UT1:

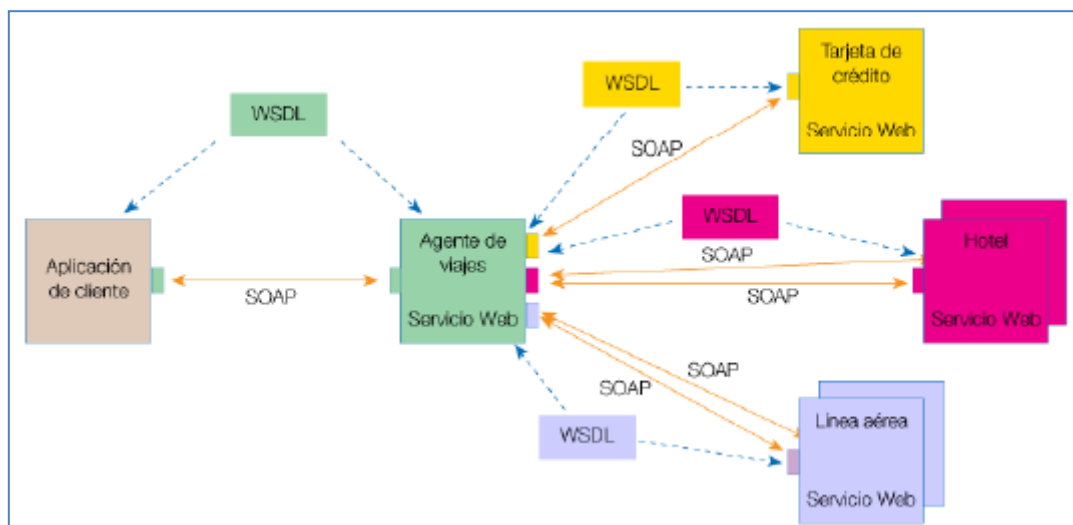
**SOA (Arquitectura Orientada a Servicios, Service-Oriented Architecture), es una arquitectura web que permite la creación de sistemas de información altamente escalables que reflejan el negocio de la organización y la exposición e invocación de servicios ([servicios web](#)), lo cual facilita la interacción entre diferentes sistemas propios de una misma organización o de otras. La tecnología más comúnmente utilizada es SOAP y WSDL.**

En las primeras aplicaciones web, su desarrollo iba muy ligado al desde dónde iban a ser accedidas (de ahí las disputas entre los navegadores, Netscape y Microsoft a mediados de los noventa).

En la actualidad, el desarrollo de aplicaciones se ha centrado mucho más en el contenido que se hará disponible en vez de la aplicación que se va a utilizar para hacerlo visible. Este contenido se pone a disposición del resto del mundo utilizando diferentes versiones de servicios Web.

Los **servicios Web son fuentes de datos que se ponen a disposición de cualquiera utilizando XML**; un ejemplo sencillo de un servicio Web es RSS (*Sindicación Realmente Sencilla*) y que utiliza XML para proporcionar fuentes de información actualizada con mucha frecuencia como pueden ser noticias o información meteorológica.

**La utilización de este tipo de arquitecturas es la base del desarrollo de aplicaciones web “orientadas a servicios”.**



(fuente imagen: <http://webservicecofu.blogspot.com/2007/06/como-functiona-el-web-service.html>)

La utilización de SOA en el desarrollo web implica que una aplicación web utilice los servicios web proporcionados por un servidor de servicios. De esta forma, una aplicación web, se convierte en el cliente de otras aplicaciones que ofrecen sus servicios a través de un servidor.

Los servicios web, permiten el intercambio de información entre diferentes aplicaciones, utilizando el protocolo HTTP.

Hoy en día, son muchas las empresas que ofrecen servicios web para ser solicitados desde una aplicación web. Una vez más, con ello se favorece la reutilización del código y la programación modular.

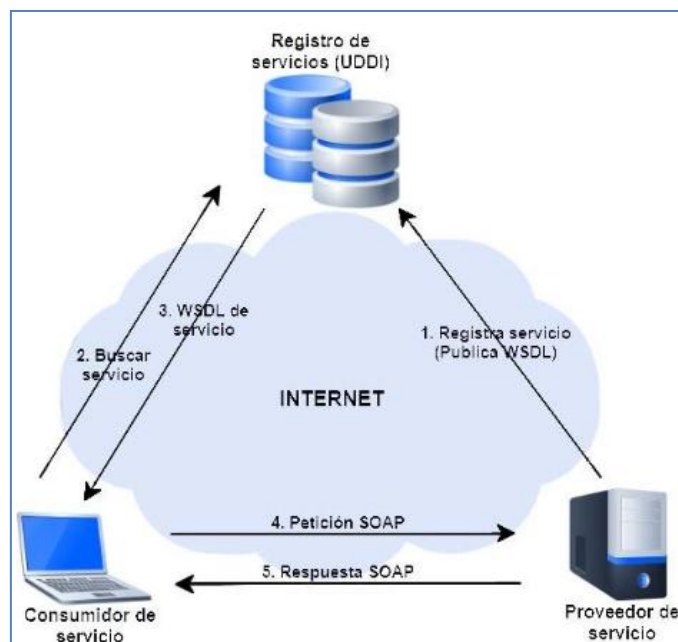
Un **servicio web (Web Service)** es un aplicación alojada en un servidor web que a través de una serie de estándares o recomendaciones W3C (SOAP, WSDL,...) ofrece información a otras aplicaciones independientemente de la tecnología de desarrollo utilizada en ellas y de la plataforma en la cual se ejecuten.

Existen varias formas de implementar el modelo SOA dentro del desarrollo web. En cualquiera de ellas (Web Services, API-REST, API-WEB,...) es necesario:

- Establecer el mecanismo de intercambio de información; es decir, establecer cómo serán los mensajes para solicitar el cliente la información que precisa y cómo serán los mensajes del servidor para proporcionar la información solicitada.
- Establecer el mecanismo de descripción de las funciones o métodos que un cliente puede utilizar.

La arquitectura *Web Services*, es un estándar definido por W3C y es el que utilizaremos nosotros. Teniendo en cuenta lo anterior, *Web Services* establece una serie de protocolos:

Arquitectura Web Services	Protocolos
Mensajes cliente-servidor	<b>SOAP</b> (basado en XML)
Descripción de los servicios	<b>WSDL</b> (basado en XML)
Descubrimiento de los servicios	UDDI (basado en XML). No es muy utilizado
Protocolo para la comunicación	HTTP (también es posible a través de otros protocolos de comunicación típicos de una red: FTP, SMTP,...)



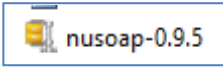
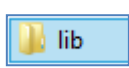
Fuente de la imagen: [https://www.researchgate.net/figure/Figura-212-Arquitectura-tipica-de-un-Servicio-Web\\_fig7\\_305403120](https://www.researchgate.net/figure/Figura-212-Arquitectura-tipica-de-un-Servicio-Web_fig7_305403120)

## 2. IMPLEMENTAR UN SERVICIO WEB UTILIZANDO PHP.

Aunque PHP tiene una extensión propia para trabajar con servicios web (soap.php), **utilizaremos NUSOAP**, un kit de herramientas para desarrollar servicios web en PHP. Cuenta con las clases necesarias para ser consumidor de un servicio web como para ser servidor de ellos, siendo la generación automática del documento WSDL que describe el servicio (la extensión nativa de PHP soap 5, no permite dicha generación automática).

### Pasos a seguir:

1. Será necesario descargar e instalar dicho kit (<https://sourceforge.net/projects/nusoap/files/latest/download>)

2. Descomprimir  y ubicar  en una carpeta nueva creada en DOCUMENT\_ROOT (en nuestro caso y teniendo en cuenta la plataforma de desarrollo utilizada, *www*).
3. Crearemos un servicio web (*servicio.php*) utilizando dicha librería; será un servicio sencillo, pues simplemente sumará dos números.

```
<?php
//creación del servicio web
include_once "lib/nusoap.php";

//Definición de los servicios web (en este ejemplo sólo habrá uno, el proporcionado por la función Suma)

function Suma($num1,$num2)
{
    $resultadoSuma=$num1+$num2;
    $resultado="El resultado de la suma ".$num1."+".$num2." es: ".$resultadoSuma;
    return $resultado;
}

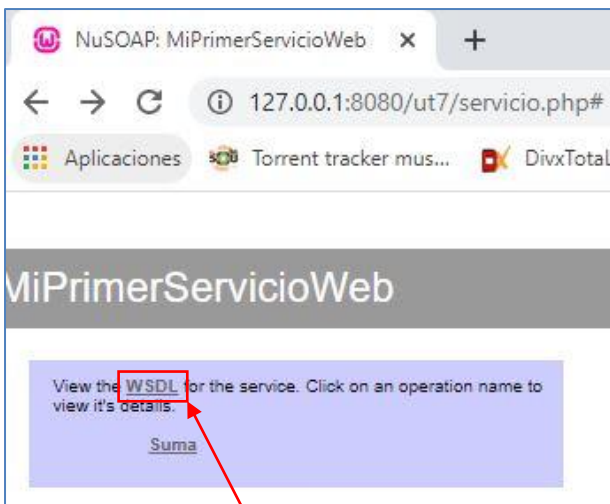
//Instanciación del objeto nusoap_server para crear el servicio web
$servicio=new nusoap_server();

$ns="urn:miserviciowsdl"; //nombre del espacio de nombres a utilizar en los servicios web
                        //recogidos en este documento (sólo hay uno, la función Suma)
$servicio->configureWSDL("MiPrimerServicioWeb",$ns); //crear el documento WSDL que describe el servicio web

$servicio->register("Suma",/*nombre del servicio que esta incluido en el documento WSDL y
                        que posteriormente podrá ser ejecutado desde un cliente o consumidor de servicios web*/
    array('num1'=>'xsd:integer','num2'=>'xsd:integer'),/*parámetros de entrada del servicio web
                        necesarios para su ejecución*/
    array('return'=>'xsd:string'),/*parámetros de salida del servicio*/
    $ns)/*nombre del espacio de nombres a utilizar en este servicio*/;

$servicio->service(file_get_contents('php://input')); //a los datos generados en este servicio web
                                                    // que son el documento WSDL y la función Mifunción
?>
```

4. Al ejecutar dicho servicio desde el navegador, obtendremos la siguiente salida:



5. Si **accedemos a WSDL**, podremos consultar el código XML que describe nuestro servicio y que utilizando la clase nusoap se ha generado a partir de los parámetros que le hemos proporcionado en *servicio.php*. (tener en cuenta que NUSOAP está basado en SOAP 1.1 y WSDL 1.1; WSDL 2.0 utiliza una estructura ligeramente diferente).

This XML file does not appear to have any style information associated with it. The document tree is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="urn:miserviciowsdl" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" targetNamespace="urn:miserviciowsdl">
  <types>
    <xsd:schema targetNamespace="urn:miserviciowsdl">
      <xsd:import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
      <xsd:import namespace="http://schemas.xmlsoap.org/wsdl/" />
    </xsd:schema>
  </types>
  <message name="SumaRequest">
    <part name="num1" type="xsd:integer"/>
    <part name="num2" type="xsd:integer"/>
  </message>
  <message name="SumaResponse">
    <part name="return" type="xsd:string"/>
  </message>
  <portType name="MiPrimerServicioWebPortType">
    <operation name="Suma">
      <input message="tns:SumaRequest"/>
      <output message="tns:SumaResponse"/>
    </operation>
  </portType>
  <binding name="MiPrimerServicioWebBinding" type="tns:MiPrimerServicioWebPortType">
    <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="Suma">
      <soap:operation soapAction="http://127.0.0.1/ut7/servicio.php/Suma" style="rpc"/>
      <input>
        <soap:body use="encoded" namespace="urn:miserviciowsdl" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
      </input>
      <output>
        <soap:body use="encoded" namespace="urn:miserviciowsdl" encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
      </output>
    </operation>
  </binding>
  <service name="MiPrimerServicioWeb">
    <port name="MiPrimerServicioWebPort" binding="tns:MiPrimerServicioWebBinding">
      <soap:address location="http://127.0.0.1:8080/ut7/servicio.php"/>
    </port>
  </service>
</definitions>
```

**WDSL** es un lenguaje basado en XML y es utilizado para describir un servicio web. Consta de diferentes secciones:

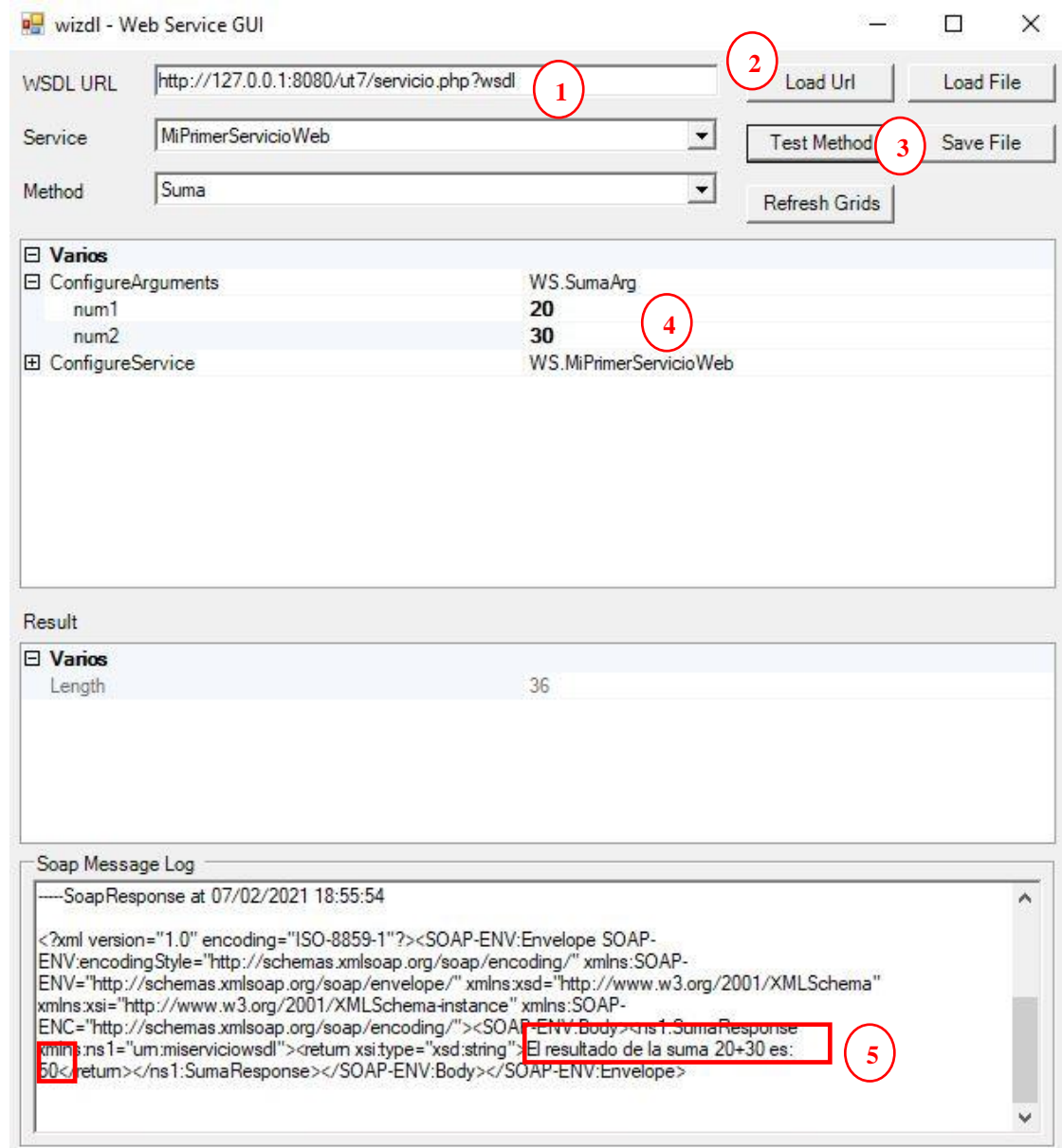
- **Descripción**, donde básicamente se detallan los espacios de nombres utilizados a lo largo del documento.
- **Types**, se definen los tipos de datos que se utilizan en el servicio web.
- **Message**, define los parámetros y/o valores de resultados de las funciones que ofrecerán un servicio.
- **Porttype**, incluye las funciones que configuran el servicio web; cada función es definida como una operación.
- **Binding**, se define la forma de transmitir la información de cada porttype.
- **Service**, se incluye un listado de url desde las que se puede acceder al servicio web.

6. Para comprobar que el servicio funciona, podemos utilizar un simulador de funcionamiento como es wizdl (descargar de <http://99-developer-tools.com/simple-webservice-test-tools/>)

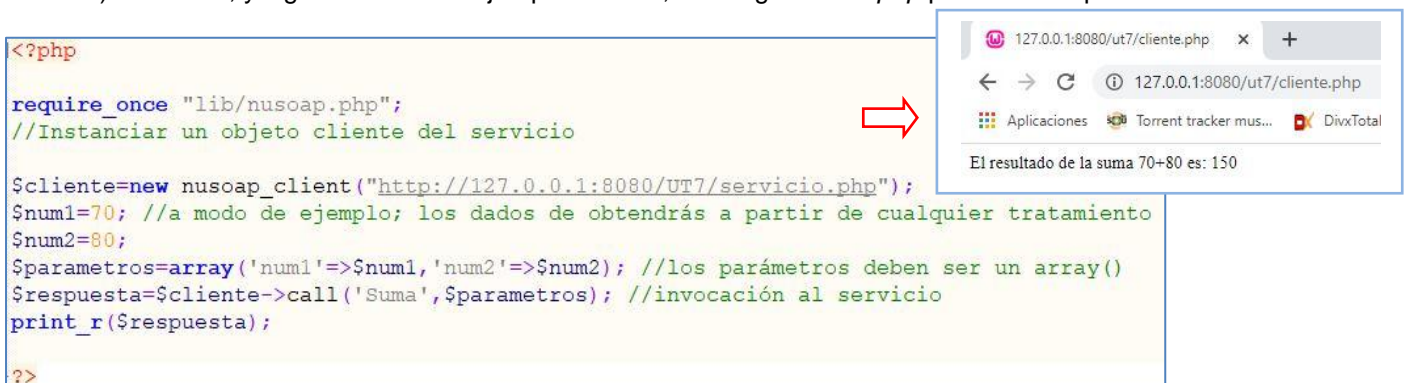
7. La herramienta requiere:

1. especificar la url para acceder el documento asociado al servicio web
2. cargar el documento
3. proporcionar los parámetros necesarios para que se ejecute el servicio web
4. ejecutar la función que ofrece un servicio
5. y por último, comprobar el resultado obtenido.





8. Una vez comprobado que funciona, lo interesante es probarlo desde otra aplicación web (cliente del servicio web). Para ello, y siguiendo con un ejemplo sencillo, el código *cliente.php* podría servir para ello:



**ACTIVIDAD:** Crea tu propio servicio web y también tu propio cliente. A continuación, intenta que tu compañero/a acceda a tu servicio a través de un cliente web suyo.