SOAP en PHP con fichero WSDL

El fichero WSDL es un archivo escrito en lenguaje de marcas (etiquetas) que define el esqueleto de las funciones ofrecidas por el servidor SOAP y consumidas por el cliente SOAP.

Como todo elemento que estandariza tanto las peticiones como los servicios es un tanto "farragoso" de construir. De hecho, hay numerosas herramientas que permiten generarlo de una manera más o menos automática (batallita de guerra: hace unos años se recomendaba elaborarlo a mano en un editor de texto plano e ir corrigiendo cada una de sus secciones)

Básicamente lo que hay que definir en este fichero es el nombre y tipo de las variables que van a ser empleadas y las funciones o métodos que se ofrecen por el servidor. La implementación del procesamiento de las funciones se hará en el servidor SOAP y el cliente solicitará una de estas funciones proporcionando los datos de entrada y recibiendo los datos de salida. En resumen, el fichero WSDL es una especie de "caja negra" que marca qué entra, qué sale y cómo se llaman los procedimientos SOAP.

Así pues, debemos planificar qué tipo de variables vamos a usar y cómo se llaman los procedimientos.

Vamos a emplear una herramienta on-line para generar el fichero WSDL que encontramos en

http://marin.jb.free.fr/wsdl/

Debemos proporcionar una información trivial como nombre de la compañía, proveedor… PERO ES MUY IMPORTANTE PONER LOS SIGUIENTES DATOS:

HTTP service endpoints
URL of each service server - one per line.

```
http://localhost/soap2/soap_con_servidor.php
```

Se refiere a la localización del servidor SOAP que va a implementar las funciones SOAP. Ojo a la carpeta del directorio de Apache donde está localizado

Types definitions
<int|string|float> typename;

```
float e1;
float e2;
float s1;
```

Tipo de dato y nombres de las variables que usamos para definir las funciones. En el ejemplo defino 2 float para las entradas y 1 float para la salida de los métodos

Functions prototypes
<type> functionname(void | [type [, type...]]);

```
s1 sumar(e1,e2);
s1 restar(e1,e2);
```

Por último definimos el prototipo de cada función. En mi ejemplo la función sumar recibe e1 y e2 (que anteriormente definimos como float) y devuelve s1 (que anteriormente definimos como float)

Guardamos el fichero generado (wsdl.xml en mi ejemplo) y tenemos nuestro WSDL que, no olvidemos, deberá ser al que hagan referencia tanto el cliente como el servidor

Servidor SOAP

```
<?php
// Indico dónde está el wsdl al crear el servidor
$soap_servidor = new SoapServer("http://localhost/soap2/wsdl.xml");
// Defino el funcionamiento de las funciones
function sumar($operando1,$operando2){
    return $operando1+$operando2;
}
function restar($operando1,$operando2){
    return $operando1-$operando2;
}
// Añado las funciones al servidor soap
$soap_servidor->AddFunction("sumar");
$soap_servidor->AddFunction("restar");
// Pongo el servidor activo
$soap_servidor->handle();
?>
```

Cliente SOAP

```
<?php
try{
        // A) Se puede "preguntar" al fichero WSDL para pedirle la sintaxis de las funciones
        echo "<h1>Las funciones ofrecidas por el servidor SOAP son</h1>";
        $cliente1 = new SoapClient('http://localhost/soap2/wsdl.xml', array("trace" => 1, "exception" => 0));
  var_dump($cliente1->__getFunctions());

        // B) Normalmente se crea un cliente SOAP y se usan las funciones del WSDL
        $cliente2 = new SoapClient('http://localhost/soap2/wsdl.xml');
        $resultado_suma = $cliente2->sumar(2.4 , 3.2);
        $resultado_resta = $cliente2->restar(2.7 , 3.5);
        echo "<h1>Usando las funciones del servidor SOAP</h1>";
        echo "La suma es: ".$resultado_suma." y la resta es: ".$resultado_resta;

        // C) Se puede ver la respuesta del servidor SOAP después de utilizar una función
        $cliente3 = new SoapClient('http://localhost/soap2/wsdl.xml',array('trace' => 1));
        $resultado = $cliente3->restar(4, 5);
        echo "<h1>Viendo las respuestas del servidor SOAP</h1>";
        echo "<p>Visualizar el codigo fuente de la pagina generada</p>";
        echo $cliente3->__getLastRequestHeaders(). "<br>";    // Encabezados SOAP de la última petición
        echo $cliente3->__getLastRequest() . "<br>";                    // Ultima petición SOAP
        echo $cliente3->__getLastResponseHeaders(). "<br>";  // Encabezados SOAP de la última respuesta
        echo $cliente3->__getLastResponse(). "<br>";                    // Devuelve la última respuesta SOAP

} catch (Exception $e) {
        echo 'Error --> '. $e->getMessage();
        echo "<br>".$e->getTraceAsString();
        echo "<br>".$e->getCode();
        echo "<br>".$e->getLine();
}
?>
```