1. **Introdução**

Compreendemos nas aulas anteriores que é possível criar serviços capazes de serem consumidos por aplicações. Nessa aula iremos discutir um pouco mais sobre *Web Services*, porém, iremos fazer uma distinção entre eles: com manutenção de estado e sem manutenção de estado.

O objetivo dessa aula é explicar e exemplificar como funcionam os WS com manutenção de estado. Para isso iremos detalhar componentes desse tipo de serviço e ao final veremos um exemplo de como construir um *Web Service* usando SOAP/XML.

1. **SOAP**

O SOAP (*Simple Object Access Protocol*) baseia-se numa invocação remota de um método e para tal necessita especificar o endereço do componente, o nome do método e os argumentos para esse método. Estes dados são formatados em XML com determinadas regras e enviados normalmente por HTTP para esse componente. Não define ou impõe qualquer semântica, quer seja o modelo de programação, quer seja a semântica específica da implementação. Este aspecto é extremamente importante, pois permite que quer o serviço, quer o cliente que invoca o serviço, sejam aplicações desenvolvidas sobre diferentes linguagens de programação. Por esta razão, o SOAP tornou-se uma norma aceita para se utilizar com *Web Services,* uma tecnologia construída com base em XML e HTTP.

Desta forma, pretende-se garantir a interoperabilidade e intercomunicação entre diferentes sistemas, através da utilização da linguagem XML e do mecanismo de transporte HTTP ou outro como, por exemplo, SMTP. O SOAP permite que os documentos XML de envio e de recepção sobre a *Web* suportem um protocolo comum de transferência de dados para uma comunicação de rede eficaz, ou seja, o SOAP providencia o transporte de dados para os *Web Services*.

Em relação a Web, o SOAP é um protocolo de RPC que funciona sobre HTTP (ou SMTP, ou outro) de forma a ultrapassar as restrições de segurança*/firewalls* normalmente impostas aos sistemas clássicos de RPC (RMI, DCOM, CORBA/IIOP) suportando mensagens XML. Em vez de usar HTTP para pedir uma página HTML para ser visualizada num *browser,* o SOAP envia uma mensagem de XML através do pedido HTTP e recebe uma resposta, se existir, através da resposta do HTTP. Para assegurar corretamente a transmissão da mensagem de XML, o servidor de HTTP (*Apache* ou *Microsoft Internet Information Server*), recebe mensagens SOAP e deve validar e compreender o formato do documento XML definido na especificação SOAP v1.1.

1. **WSDL**

É a sigla de *Web Services Description Language*, padrão baseado em XML para descrever o serviço como no COM, onde ele traz os métodos do *Web Service*. Funciona como uma espécie de "*TypeLibrary*" do *Web Service*, além de ser usado para a validação das chamadas dos métodos.

O WSDL é uma especificação desenvolvida pelo W3C e também é extensível para permitir a descrição dos serviços e suas mensagens, independentemente dos formatos de mensagem e dos protocolos de rede que sejam usados. No entanto, é comum usar-se o MIME (*Multipurpose Internet Mail Extensions*) e o HTTP://SOAP.

O WSDL descreve os serviços disponibilizados à rede através de uma semântica XML, este providencia a documentação necessária para se chamar um sistema distribuído e o procedimento necessário para que esta comunicação se estabeleça. Enquanto que o SOAP especifica a comunicação entre um cliente e um servidor, o WSDL descreve os serviços oferecidos.

A versão atual é 2.0; a versão 1.1 não foi endossada pelo W3C. A WSDL 1.2 foi renomeada para 2.0 e aceita todos os métodos de requisição HTTP (não apenas *GET* e *POST*).

1. **UDDI**

Protocolo desenvolvido para a organização e registro de *Web Services*. O UDDI (*Universal Description Discovery and Integration*) é uma iniciativa em desenvolvimento no âmbito do consórcio industrial UDDI promovido originalmente pela IBM, *Microsoft* e *Arriba*, com objetivo de acelerar a interoperabilidade e utilização dos *Web Services*, pela proposta de um serviço de registro de nomes de organizações e de descrição do serviço. UDDI nada mais é do que um serviço de diretório onde empresas podem registrar (publicar) e buscar (descobrir) por serviços *Web* (*Web Services*).

Um registro UDDI contém três tipos de informação:

* Informações gerais de cada organização, tais como o nome, endereço e contatos;
* Informações de organizações e serviços por categorias de negócios;
* Informações técnicas sobre os serviços providenciados pelas organizações.

O UDDI providencia três funções principais, conhecidas como publicação, descoberta e ligação:

1) publicação: permite que uma organização divulgue o(s) seu(s) serviço(s);

2) descoberta: permite que o cliente do serviço procure e encontre um determinado serviço;

3) ligação (*bind*): permite que o cliente do serviço possa estabelecer a ligação e interagir com o serviço.

Um serviço de registro UDDI é um *Web Service* que gerencia informação sobre provedores, implementações e metadados de serviços. Provedores de serviços podem utilizar UDDI para publicar os serviços que eles oferecem. Usuários de serviços podem usar UDDI para descobrir serviços que lhes interessem e obter os metadados necessários para utilizar esses serviços, que podem ter três partes:

* "páginas brancas" descrevem a companhia: nome, endereço, contatos, etc.
* "páginas amarelas" incluem as categorias, baseada em taxonomias padrões.
* "páginas verdes" descrevem a interface para o serviço em nível de detalhe suficiente para se escrever uma aplicação que use o *Web Service.*

A especificação UDDI define:

* APIs SOAP utilizadas para publicar e obter informações de um registro UDDI;
* Esquemas XML do modelo de dados do registro e do formato das mensagens SOAP;
* Definições WSDL das APIs SOAP;
* Definições de registro UDDI (modelos técnicos - *tModels*) de diversos sistemas de identificação e categorização, que podem ser utilizados para identificar e categorizar registros UDDI.

1. **Vantagens e desvantagens**

Vantagens:

* Integração entre aplicações construídas em diferentes tecnologias;
* Inteligível para o ser humano, o que facilita o desenvolvimento de novos aplicativos utilizando esta tecnologia;
* Intuitiva, pois é descrita em linguagem natural com termos próximos aos utilizadas pela aplicação;
* Precisa, pois a WSDL e o *Schema* garantem conformidade com os padrões estabelecidos entre provedores e requisitantes.

Desvantagens:

* Extremamente verbosa, o que a torna menos produtiva que outras propostas como aqueles que utilizam, por exemplo, JSON.
* Performance de uma aplicação que consome muitos *Web Services* é tipicamente inferior;
* Os custos de integração e construção de *Web Services* nem sempre são baixos.

1. **Exemplo**

Para publicar um serviço capaz de fornecer o WSDL completo e também consumir esse serviço de forma eficiente, é muito comum utilizarmos bibliotecas prontas. Existem diversas bibliotecas implementadas para realizar essa tarefa em diversas linguagens.

Para o PHP temos uma biblioteca já escrita chamada NUsoap. Faça download dessa biblioteca aqui: <https://sourceforge.net/projects/nusoap/>

1. Adicione a biblioteca via *require*:

require 'lib/nusoap.php';

1. Crie uma função que torne os dados disponíveis

function get\_price($name){

$products = [

"book"=>20,

"pen"=>10,

"pencil"=>5

];

foreach($products as $product=>$price){

if($product==$name){

return $price;

break;

}

}

}

1. Instancie o servidor, configure e execute:

$server = new nusoap\_server(); // Instancia o NUSOAP

$server->configureWSDL("Exemplo de SOAP","urn:exemplosoap"); // Configure WSDL file

$server->register(

"get\_price", // name of function

array("name"=>"xsd:string"), // inputs

array("return"=>"xsd:integer") // outputs

);

$server->service(file\_get\_contents("php://input"));