WebServices

Web services refers to the standardized way of application-to-application interaction using the XML, SOAP, WSDL and UDDI open standards over internet. It represents an application component to exchange the information between two applications over the network.

A web service works on client server model where client applications can access web services over the network. To access a method or any other resource, client applications use endpoint URIs provided by web services.

For instance, Java applications can be connected to .NET applications. Such compatibility is possible due to open standards of web services, while the communication happens via the data exchange formats XML/JSON and the protocols of web communication HTTP/HTTPS.

Components of Web Services:

XML:
Stands for Extensible Markup Language. It is used to tag the data.
SOAP
Stands for Simple Object Access Protocol. It is used to transfer the data.
UDDI:
Stands for Universal Description, Discovery and Integration. It is used for listing what services are available?
WSDL:
Stands for Web Services Description Language. It is used for describing the services available.

Advantages of web services:

Interoperability:
Interoperability is the ability of a system or a product to work with other systems or products without special effort. It is the most important benefit of Web Services. For example a web service created by java can be used by VB or .NET and vice versa.

Reusability:

As we discussed in previous topic a web service is a unit of managed code that can be remotely invoked over internet using HTTP requests i.e. Web services allows us to expose the functionality of our existing code over the network.

Low Cost of Communication:
As we discussed a web service can be invoked over internet using HTTP requests so we can use our existing low-cost internet for implementing web services.

Deployability: Web Services are deployed over standard Internet technologies. This makes it possible to deploy Web Services even over the fire wall to servers running on the Internet on the other side of the globe. Also thanks to the use of proven community standards, underlying security (such as SSL) is already built-in.

Loose Coupling: Web service code and server code are independent to each other which provides loose coupling.

Types of Web Services

- Soap
- Rest

SOAP Service:

SOAP Service stands for Simple Object Access Protocol. SOAP services are stateful services that use XML language to form an envelope. A SOAP envelope can be described in two portions i.e. one is a **SOAP header and body**, and the other one is the **protocol** used to send SOAP messages.

A SOAP message contains:

- An Envelope that indicates the start and end of the message
- A Header that includes attributes used to process the message and is an optional element
- A Body that holds the XML data that is to be sent and it cannot be left out
- A Fault which provides error messages when processing and it is an optional element.

This SOAP header consists of Authentication and Authorization which grants access. The body comes under the payload section of the request which uses WSDL to describe the Web Service and the protocol is mainly HTTP (HyperText Transmission Protocol).

Web Services Security
SOAP services have an SSL layer (Secure Socket Layer) which is responsible for avoiding any data leakage during transmission, and thus provides encryption and decryption.

Meanwhile, the SOAP services are more secure as it also has WSS (Web Services Security) which guarantees no divulging during communication between the service and the application.

As we all know, every Web Service (unlike Web API), needs a network to carry out its operation. Thus, it becomes necessary for Web Services to provide security when connected to a network.

Hence, Web Services has three important entities to cover the security factor during message transfer.

- **Authentication and Authorization** (Already explained above).
- **Confidentiality:** This is entirely dependent on the SSL which provides encryption and decryption of the SOAP envelope.
- **Network Security:** This means extracting all the SOAP and XML – RPC responses that you get from the server. **For Example,** If you take any Web Service tool like POSTMAN or PARASOFT, you will find that under the HTTP header manager, there is an option to set the value of the Content-Type. The value can be set to the Application/JSON so that it will extract all REST (Since SOAP services do not support HTTP Header manager options).
- Thus, you can pass the content-type: Application/XML in a **payload** itself in the form of XML. This would also extract SOAP and XML-RPC.

These three factors constitute Web Services Security to cope with the external attacks.

SOAP Communication Protocols:

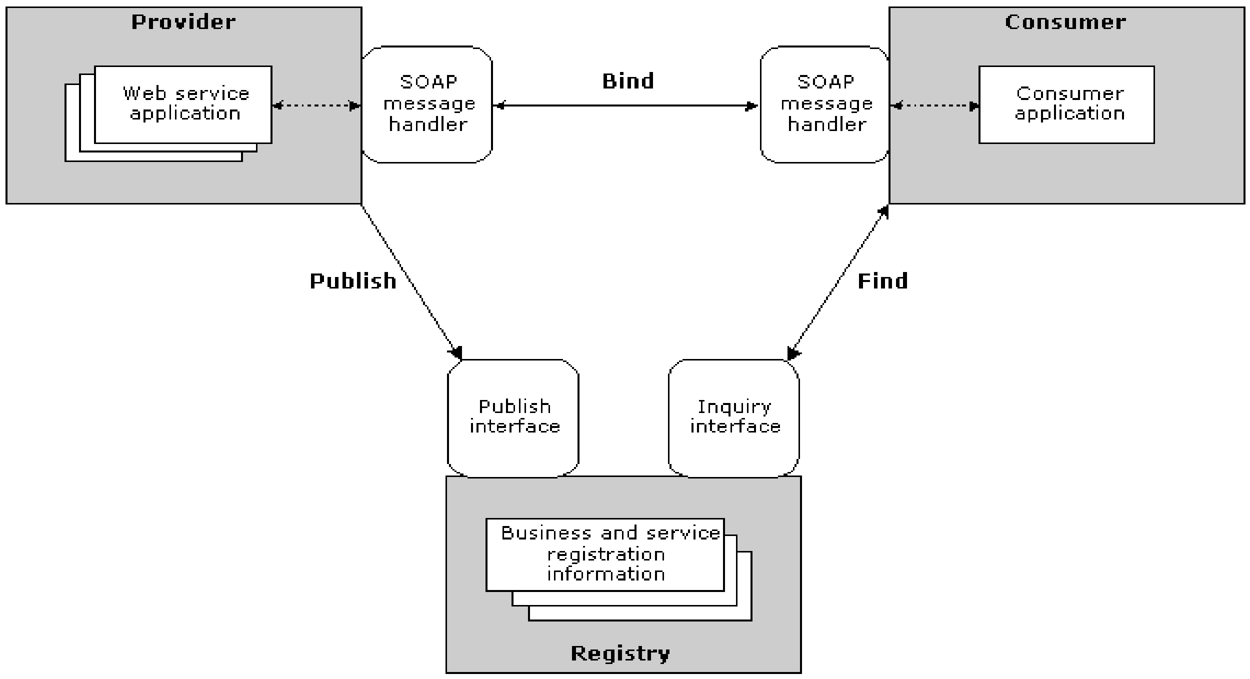
Remote Procedure Call (RPC): RPC is one of the simplest protocols but has many limitations. And it is a service-specific interface between a client and an RPC-style web service. In RPC, the client will call the methods in the service request and send the needed parameters to the web server. Finally, the server will return the response to the client accordingly. Know that existing RPC-based systems are DCOM, EJB, Java RMI, CORBA, etc.

Marshalling and Demarshalling: Marshalling is the process of encapsulating information in a SOAP message. Here, information is formatted before embedding it into the SOAP message, and then the information is sent to the service-side server through the HTTPS request. On the other hand, Demarshalling is the process of opening up the information from the formatted message by the server and replying with an appropriate response to the client.

The Architecture Of SOAP Service

Every SOAP service depends on three entities which ultimately form the architecture of SOAP Service.

- **Service Provider:** All software systems or application which is a part of or provides Web Service.
- **Service Requester:** All software systems or applications which are a part of requests Web Service from Service Provider.
- **Service Registry:** A registry or repository where all the information about Web Service is provided by the Service Provider. (Already discussed in UDDI)



A Web Service provider is an organization that creates and hosts Web Services. Typically, a provider publishes information about their organization and the services they offer in a Web Service registry that can be queried by members of the organization or possibly by other businesses.

A Web Service consumer finds a Web Service (typically by querying a Web Service registry) then runs the service by establishing a connection to the provider. This is called binding to a Web Service.

A Web Service registry is a collection of business and service information that is readily accessible to providers and consumers, through programmatic publishing and querying interfaces.

Advantages of SOAP

- It is recommended by the W3C consortium for the web services and application programming interface to communicate with the client application.
- It is a lightweight communication protocol used for exchanging data between two machines over the network.
- It is a platform-independent operating system that can run on Linux, Windows and macOS.
- SOAP uses a default protocol to send the message over a network, and all web applications also support it.
- It has an XML format that includes an envelope, header, body, and fault element as a SOAP message during the message's processing

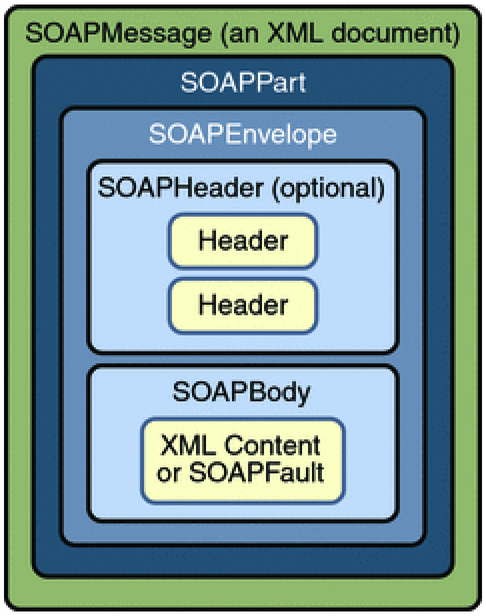
Disadvantages of SOAP

- SOAP is used only XML format data in web service, whereas JSON and other lightweight formats are not supported by it.
- It is slow because it uses XML format, whereas the payload for a simple string message is large.
- There are no security features in the SOAP specification.
- There is no state reference for the remote object in the SOAP client.

SOAP Message Format:

A SOAP message contains:

- An Envelope that indicates the start and end of the message
- A Header that includes attributes used to process the message and is an optional element
- A Body that holds the XML data that is to be sent and it cannot be left out
- A Fault which provides error messages when processing and it is an optional element.



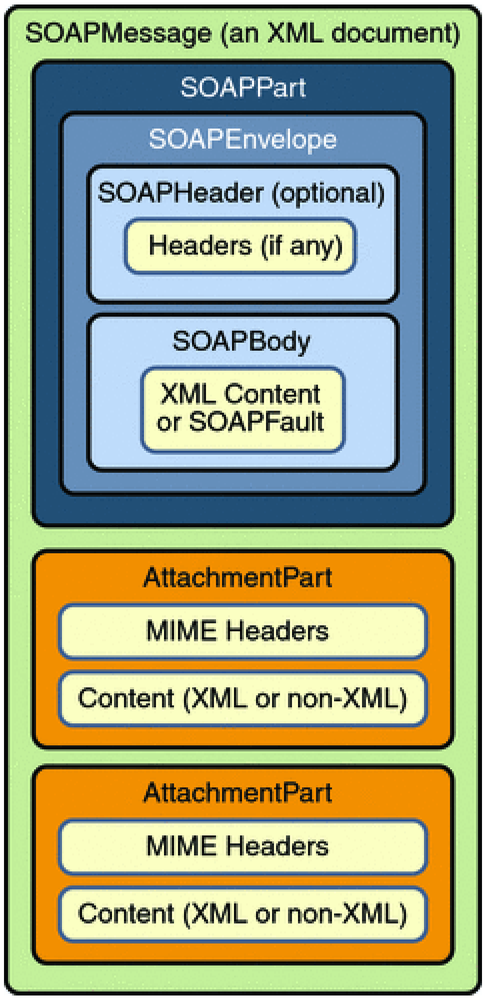
Envelope:
It specifies that the XML message is a SOAP message. A SOAP message can be defined as an XML document containing header and body encapsulated in the envelope. The fault is within the body of the message.

Header:
This part is not mandatory. But when it is present it can provide crucial information about the applications.

Body:
It contains the actual message that is being transmitted. Fault is contained within the body tags.

Fault:
This section contains the status of the application and also contains errors in the

application. This section is also optional. It should not appear more than once in a SOAP message.



SOAP Message Structure

The following block depicts the general structure of a SOAP message –

```
<?xml version = "1.0"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV = "http://www.w3.org/2001/12/soap-envelope"
  SOAP-ENV:encodingStyle = "http://www.w3.org/2001/12/soap-encoding">

  <SOAP-ENV:Header>
    ...
    ...
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    ...
    ...
    <SOAP-ENV:Fault>
      ...
      ...
    </SOAP-ENV:Fault>
    ...
  </SOAP-ENV:Body>
</SOAP_ENV:Envelope>
```

Here is the SOAP request –

```
POST /Quotation HTTP/1.0
Host: www.xyz.org
Content-Type: text/xml; charset = utf-8
Content-Length: nnn

<?xml version = "1.0"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV = "http://www.w3.org/2001/12/soap-envelope"
  SOAP-ENV:encodingStyle = "http://www.w3.org/2001/12/soap-encoding">

  <SOAP-ENV:Body xmlns:m = "http://www.xyz.org/quotations">
    <m:GetQuotation>
      <m:QuotationsName>MiscroSoft</m:QuotationsName>
    </m:GetQuotation>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

A corresponding SOAP response looks like –

```
HTTP/1.0 200 OK
Content-Type: text/xml; charset = utf-8
Content-Length: nnn

<?xml version = "1.0"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV = "http://www.w3.org/2001/12/soap-envelope"
  SOAP-ENV:encodingStyle = "http://www.w3.org/2001/12/soap-encoding">
  <SOAP-ENV:Body xmlns:m = "http://www.xyz.org/quotation">
    <m:GetQuotationResponse>
      <m:Quotation>Here is the quotation</m:Quotation>
    </m:GetQuotationResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Working with WSDL:

WSDL, or Web Service Description Language, is an XML based definition language. It's used for describing the functionality of a SOAP based web service.

Features of WSDL

- WSDL is an XML-based protocol for information exchange in decentralized and distributed environments.
- WSDL definitions describe how to access a web service and what operations it will perform.
- WSDL is a language for describing how to interface with XML-based services.
- WSDL is an integral part of Universal Description, Discovery, and Integration (UDDI), an XML-based worldwide business registry.
- WSDL is the language that UDDI uses.

WSDL Document Elements

- **Definitions**: It is the basic element of the WSDL document that contains the definition of one or more services.
- **Types**: The Type element is used to give information about the complicated data types used within the WSDL document.
- **Message**: It contains abstract data that is being used in communication between client and web server. It also defines the data elements for each operation.
- **portType**: It contains the collection of abstract operation supported by one or more endpoints.
- **Port**: It is used to define the single endpoint as an address for the binding.
- **Services**: It is a collection of endpoint networks that specify the port address for the binding.

- **Binding**: It specifies how operations are implemented by concrete protocols and data format features for operation and messaging.

WSDL Structure:

```
<definitions>
       <types>
               Definition of types goes here.
      </types>

      <message>
              Definition of a message goes here.
      </message>

     <portType>
        <operation>
            Definition of an operation goes here.
       </operation>
    </portType>

    <binding>
          Definition of a binding goes here.
    </binding>

   <service>
         Definition of service goes here.
      </service>
</definition>
```