GURU99   ☰

SOAP vs REST API: Difference Between Web Services

By Alyssa Walker   🕐 Updated March 11, 2023

Key Difference Between SOAP and REST API

- SOAP stands for Simple Object Access Protocol whereas REST stands for Representational State Transfer.
- SOAP is a protocol whereas REST is an architectural pattern.
- SOAP uses service interfaces to expose its functionality to client applications while REST uses Uniform Service locators to access to the components on the hardware device.
- SOAP needs more bandwidth for its usage whereas REST doesn't need much bandwidth.
- Comparing SOAP vs REST API, SOAP only works with XML formats whereas REST work with plain text, XML, HTML and JSON.
- SOAP cannot make use of REST whereas REST can make use of SOAP.

In this SOAP vs REST API tutorial, you will learn: ⌄

What is SOAP?

SOAP is a protocol which was designed before REST and came into the picture. The main idea behind designing SOAP was to ensure that programs built on different platforms and programming languages could exchange data in an easy manner. SOAP stands for Simple Object Access Protocol.

What is REST?

REST was designed specifically for working with components such as media components, files, or even objects on a particular hardware device. Any web service that is defined on the principles of REST can be called a RestFul web service. A Restful service would use the   ⬆

normal HTTP verbs of GET, POST, PUT and DELETE for working with the required components. REST stands for Representational State Transfer.

Difference Between SOAP and REST

Each technique has its own advantages and disadvantages. Hence, it's always good to understand in which situations each design should be used. This REST and SOAP API difference tutorial will go into some of the key difference between REST and SOAP API as well as what challenges you might encounter while using them.
Below is the main difference between SOAP and REST API

SOAP	REST
SOAP stands for Simple Object Access Protocol	REST stands for Representational State Transfe
SOAP is a protocol. SOAP was designed with a specification. It includes a WSDL file which has the required information on what the web service does in addition to the location of the web service.	REST is an Architectural style in which a web se can only be treated as a RESTful service if it foll the constraints of being 1. Client Server 2. Stateless 3. Cacheable 4. Layered System 5. Uniform Interface
SOAP cannot make use of REST since SOAP is a protocol and REST is an architectural pattern.	REST can make use of SOAP as the underlying protocol for web services, because in the end it just an architectural pattern.
SOAP uses service interfaces to expose its functionality to client applications. In SOAP, the WSDL file provides the client with the necessary information which can be used to understand what services the web service can offer.	REST use Uniform Service locators to access to components on the hardware device. For exam if there is an object which represents the data c employee hosted on a URL as http://demo.guru the below are some of URI that can exist to acce them.

http://demo.guru99.com/Employee

http://demo.guru99.com/Employee/1

SOAP requires more bandwidth for its usage. Since SOAP Messages contain a lot of information inside of it, the amount of data transfer using SOAP is generally a lot.

```
<?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:Body>
 <Demo.guru99WebService
 xmlns="http://tempuri.org/">
   <EmployeeID>int</EmployeeID>
   </Demo.guru99WebService>
 </soap:Body>
</SOAP-ENV:Envelope>
```

REST does not need much bandwidth when requests are sent to the server. REST messages mostly just consist of JSON messages. Below is example of a JSON message passed to a web server. You can see that the size of the message comparatively smaller to SOAP.

```
{"city":"Mumbai","state":"Maharastra'
```

SOAP can only work with XML format. As seen from SOAP messages, all data passed is in XML format.

REST permits different data format such as Plai text, HTML, XML, JSON, etc. But the most prefer format for transferring data is JSON.

When to use REST?

One of the most highly debatable topics is when REST should be used or when to use SOAP while designing web services. Below are some of the key factors that determine when REST and SOAP API technology should be used for web services REST services should be used the following instances

- **Limited resources and bandwidth** – Since SOAP messages are heavier in content and consume a far greater bandwidth, REST should be used in instances where network bandwidth is a constraint.
- **Statelessness** – If there is no need to maintain a state of information from one request to another then REST should be used. If you need a proper information flow wherein some information from one request needs to flow into another then SOAP is more suited for that purpose. We can take the example of any online purchasing site. These sites normally need the user first to add items which need to be purchased to a cart. All of the cart items are then transferred to the payment page in order to complete the purchase. This is an example of an application which needs the state feature. The state of the cart items needs to be transferred to the payment page for further processing.
- **Caching** – If there is a need to cache a lot of requests then REST is the perfect solution. At times, clients could request for the same resource multiple times. This can increase the number of requests which are sent to the server. By implementing a cache, the most frequent queries results can be stored in an intermediate location. So whenever the client requests for a resource, it will first check the cache. If the resources exist then, it will not proceed to the server. So caching can help in minimizing the amount of trips which are made to the web server.
- **Ease of coding** – Coding REST Services and subsequent implementation is far easier than SOAP. So if a quick win solution is required for web services, then REST is the way to go.

Next in this SOAP and REST difference tutorial, we will learn when to use SOAP API.

When to use SOAP?

SOAP should be used in the following instances

1. **Asynchronous processing and subsequent invocation** – if there is a requirement that the client needs a guaranteed level of reliability and security then the new SOAP standard of SOAP 1.2 provides a lot of additional features, especially when it comes to security.
2. **A Formal means of communication** – if both the client and server have an agreement on the exchange format then SOAP 1.2 gives the rigid specifications for this type of interaction. An example is an online purchasing site in which users add items to a cart before the payment is made. Let's assume we have a web service that does the final

payment. There can be a firm agreement that the web service will only accept the cart item name, unit price, and quantity. If such a scenario exists then, it's always better to use the SOAP protocol.

3. **Stateful operations** – if the application has a requirement that state needs to be maintained from one request to another, then the SOAP 1.2 standard provides the WS* structure to support such requirements.

Next in this REST vs SOAP API difference, we will learn about challenges with SOAP API.

Challenges in SOAP API

API is known as the **Application Programming Interface** and is offered by both the client and the server. In the client world, this is offered by the browser whereas in the server world it's what is provided by the web service which can either be SOAP or REST.

Challenges with the SOAP API

1. WSDL file – One of the key challenges of the SOAP API is the WSDL document itself. The WSDL document is what tells the client of all the operations that can be performed by the web service. The WSDL document will contain all information such as the data types being used in the SOAP messages and what all operations are available via the web service. The below code snippet is just part of a sample WSDL file.

```
<?xml version="1.0"?>
<definitions name="Tutorial"
        targetNamespace=http://demo.guru99.com/Tutorial.wsdl
        xmlns:tns=http://demo.guru99.com/Tutorial.wsdl
        xmlns:xsd1=http://demo.guru99.com/Tutorial.xsd
        xmlns:soap=http://schemas.xmlsoap.org/wsdl/soap/
        xmlns="http://schemas.xmlsoap.org/wsdl/">

        <types>
                <schema
```

```
targetNamespace=http://Demo.guru99.com/Tutorial.xsd
                xmlns="http://www.w3.org/2000/10/XMLSchema">


        <element name="TutorialNameRequest">
                    <complexType>
                        <all>
                            <element name="TutorialName"
type="string"/>
                        </all>
                    </complexType>
            </element>
        <element name="TutorialIDRequest">
                <complexType>
                    <all>
                            <element name="TutorialID"
type="number"/>
                    </all>
                </complexType>
            </element>
            </schema>
</types>
```

As per the above WSDL file, we have an element called "TutorialName" which is of the type String which is part of the element TutorialNameRequest.

Now, suppose if the WSDL file were to change as per the business requirements and the TutorialName has to become TutorialDescription. This would mean that all the clients who are currently connecting to this web service would then need to make this corresponding change in their code to accommodate the change in the WSDL file.

This shows the biggest challenge of the WSDL file which is the tight contract between the client and the server and that one change could cause a large impact, on the whole, client applications.

2. Document size – The other key challenge is the size of the SOAP messages which get transferred from the client to the server. Because of the large messages, using SOAP in places where bandwidth is a constraint can be a big issue.

Next in this RESTful vs SOAP difference, we will learn about challenges with REST API.

Challenges in REST API

1. **Lack of Security** – REST does not impose any sort of security like SOAP. This is why REST is very appropriate for public available URL's, but when it comes down to confidential data being passed between the client and the server, REST is the worst mechanism to be used for web services.

2. **Lack of state** – Most web applications require a stateful mechanism. For example, if you had a purchasing site which had the mechanism of having a shopping cart, it is required to know the number of items in the shopping cart before the actual purchase is made. Unfortunately, the burden of maintaining this state lies with the client, which just makes the client application heavier and difficult to maintain.

Difference between SOAP Vs CORBA Vs DCOM Vs Java RMI

Remote access techniques such as the RPC (Remote Procedure calls) methods were in common use before SOAP and REST API came along. The various remote access techniques which were available are mentioned below.

1. **CORBA** – This was known as Common Object Request Broker Architecture. This system was put in place to ensure that applications built on various platforms could talk to each other. CORBA was based on an object-oriented architecture, but it was not necessary for the calling application to be based on this architecture. The major disadvantage of this technique was that it has to be developed in a separate language called the Interface Definition Language, and it just presented an additional language that had to be learned by developers to make use of the CORBA system.

2. **DCOM** – This is the Distributed Component Object Model, which is a proprietary Microsoft technology for clients to access remote components. The biggest issue with this mechanism was it was up to the client application to free up resources when no longer required.Secondly, when the client sent the request, it was up to the client to ensure that the request was wrapped or marshaled in a correct way so that the web service could understand the request sent. Another issue was if the client application was a Java based application which had to work DCOM (Microsoft Technology) additional coding was required to ensure that applications built in other programming languages could work with DCOM based web services.

3. **Java RMI** – Known as Java **R**emote **M**ethod **I**nvocation, this was Java implementation on how remote objects could be called through remote procedure calls. The biggest restriction of this technology was that Java RMI could only be run on a Java Virtual Machine. This meant that the calling application also has to be run on the Java framework in order to make use of Java RMI.

The main differences between SOAP and these techniques are as follows

1. **Working over HTTP** – All of the RPC techniques have one big limitation, and it is that they don't work by the HTTP protocol. Since all applications on the web had to work on this protocol, this used to be a major roadblock for clients which had to access these RPC-style web services.

2. **Working with non-standard ports** – Since the RPC style web services did not work by the HTTP protocol, separate ports had to be open for them for clients to access the functionality from these web services.

You Might Like:

- [19 BEST API Testing Tools: REST & SOAP Automation Software](#)
- [What is SOA? Service-Oriented Architecture Principles](#)
- [Microservices Tutorial: What is, Architecture and Example](#)
- [JSON Tutorial: Learn with Simple File Format EXAMPLE](#)
- [SOA vs Microservices – Difference Between Them](#)

← Prev   Report a Bug   Next →

About

About Us

Advertise with Us

Write For Us

Contact Us

Career Suggestion

SAP Career Suggestion Tool

Software Testing as a Career

Interesting

eBook

Blog

Quiz

SAP eBook

Execute online

Execute Java Online

Execute Javascript

Execute HTML

Execute Python