Web service is a technology to communicate one programming language with another. For example, java programming language can interact with PHP and .Net by using web services. In other words, web service provides a way to achieve interoperability.

There are a number of communication protocols for web services that use the XML format such as Web Services Flow Language (WSFL), Blocks Extensible Exchange Protocol (BEEP) among others. Simple Object Access Protocol (SOAP) and Representational State Transfer (REST) are by far the most used options for accessing web services; however they are not directly comparable as they vary in the sense that SOAP is a communications protocol while REST is a set of architectural principles for data transmission.

# What is SOAP?

SOAP is an acronym for Simple Object Access Protocol. It is an XML-based messaging protocol for exchanging information among computers. SOAP is an application of the XML specification.

## Points to Note

* SOAP is a communication protocol designed to communicate via Internet.
* SOAP can extend HTTP for XML messaging.
* SOAP provides data transport for Web services.
* SOAP can exchange complete documents or call a remote procedure.
* SOAP can be used for broadcasting a message.
* SOAP is platform- and language-independent.
* SOAP is the XML way of defining what information is sent and how.
* SOAP enables client applications to easily connect to remote services and invoke remote methods.

Although SOAP can be used in a variety of messaging systems and can be delivered via a variety of transport protocols, the initial focus of SOAP is remote procedure calls transported via HTTP.

Other frameworks including CORBA, DCOM, and Java RMI provide similar functionality to SOAP, but SOAP messages are written entirely in XML and are therefore uniquely platform- and language-independent.

A SOAP message is an ordinary XML document containing the following elements −

* **Envelope** − Defines the start and the end of the message. It is a mandatory element.
* **Header** − Contains any optional attributes of the message used in processing the message, either at an intermediary point or at the ultimate end-point. It is an optional element.
* **Body** − Contains the XML data comprising the message being sent. It is a mandatory element.
* **Fault** − An optional Fault element that provides information about errors that occur while processing the message.

The SOAP envelope indicates the start and the end of the message so that the receiver knows when an entire message has been received. The SOAP envelope solves the problem of knowing when you are done receiving a message and are ready to process it. The SOAP envelope is therefore basically a packaging mechanism.

Points to Note

* Every SOAP message has a root Envelope element.
* Envelope is a mandatory part of SOAP message.
* Every Envelope element must contain exactly one Body element.
* If an Envelope contains a Header element, it must contain no more than one, and it must appear as the first child of the Envelope, before the Body.
* The envelope changes when SOAP versions change.
* The SOAP envelope is specified using the *ENV* namespace prefix and the Envelope element.
* The optional SOAP encoding is also specified using a namespace name and the optional *encoding Style* element, which could also point to an encoding style other than the SOAP one.
* A v1.1-compliant SOAP processor generates a fault upon receiving a message containing the v1.2 envelope namespace.
* A v1.2-compliant SOAP processor generates a *Version Mismatch* fault if it receives a message that does not include the v1.2 envelope namespace.

The optional Header element offers a flexible framework for specifying additional application-level requirements. For example, the Header element can be used to specify a digital signature for password-protected services. Likewise, it can be used to specify an account number for pay-per-use SOAP services.

## Points to Note

* It is an optional part of a SOAP message.
* Header elements can occur multiple times.
* Headers are intended to add new features and functionality.
* The SOAP header contains header entries defined in a namespace.
* The header is encoded as the first immediate child element of the SOAP envelope.
* When multiple headers are defined, all immediate child elements of the SOAP header are interpreted as SOAP header blocks.

## SOAP Header Attributes

A SOAP Header can have the following two attributes −

### Actor attribute

The SOAP protocol defines a message path as a list of SOAP service nodes. Each of these intermediate nodes can perform some processing and then forward the message to the next node in the chain. By setting the Actor attribute, the client can specify the recipient of the SOAP header.

### MustUnderstand attribute

It indicates whether a Header element is optional or mandatory. If set to true, the recipient must understand and process the Header attribute according to its defined semantics, or return a fault.

The SOAP body is a mandatory element that contains the application-defined XML data being exchanged in the SOAP message. The body must be contained within the envelope and must follow any headers that might be defined for the message.

The body is defined as a child element of the envelope, and the semantics for the body are defined in the associated SOAP schema.

If an error occurs during processing, the response to a SOAP message is a SOAP fault element in the body of the message, and the fault is returned to the sender of the SOAP message.

The SOAP fault mechanism returns specific information about the error, including a predefined code, a description, and the address of the SOAP processor that generated the fault.

Points to Note

* A SOAP message can carry only one fault block.
* Fault is an optional part of a SOAP message.
* For HTTP binding, a successful response is linked to the 200 to 299 range of status codes.
* SOAP Fault is linked to the 500 to 599 range of status codes.

Sub-elements of Fault

The SOAP Fault has the following sub elements −

|  |  |
| --- | --- |
| **Sr.No** | **Sub-element & Description** |
| 1 | **<faultCode>**  It is a text code used to indicate a class of errors. See the next Table for a listing of predefined fault codes. |
| 2 | **<faultString>**  It is a text message explaining the error. |
| 3 | **<faultActor>**  It is a text string indicating who caused the fault. It is useful if the SOAP message travels through several nodes in the SOAP message path, and the client needs to know which node caused the error. A node that does not act as the ultimate destination must include a *faultActor* element. |
| 4 | **<detail>**  It is an element used to carry application-specific error messages. The detail element can contain child elements called detail entries. |

SOAP Fault Codes

The faultCode values defined below must be used in the *faultcode* element while describing faults.

|  |  |
| --- | --- |
| **Sr.No** | **Error & Description** |
| 1 | **SOAP-ENV:VersionMismatch**  Found an invalid namespace for the SOAP Envelope element. |
| 2 | **SOAP-ENV:MustUnderstand**  An immediate child element of the Header element, with the mustUnderstand attribute set to "1", was not understood. |
| 3 | **SOAP-ENV:Client**  The message was incorrectly formed or contained incorrect information. |
| 4 | **SOAP-ENV:Server**  There was a problem with the server, so the message could not proceed. |

SOAP includes a built-in set of rules for encoding data types. It enables the SOAP message to indicate specific data types, such as integers, floats, doubles, or arrays.

* SOAP data types are divided into two broad categories − scalar types and compound types.
* Scalar types contain exactly one value such as a last name, price, or product description.
* Compound types contain multiple values such as a purchase order or a list of stock quotes.
* Compound types are further subdivided into arrays and structs.
* The encoding style for a SOAP message is set via the *SOAP-ENV:encodingStyle* attribute.