5.6 SIMPLE OBJECT ACCESS PROTOCOL (SOAP)

SOAP is an XML-based protocol for exchanging information between computers. SOAP is an application of the XML specification.

Features of SOAP:

- SOAP is a communication protocol for 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 gets sent and how
- SOAP enables client applications to easily connect to remote services and invoke remote methods.

SOAPMessage Structure

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

- **Envelope:** (Mandatory) Defines the start and the end of the message.

- **Header:** (Optional) Contains any optional attributes of the message used in processing the message, either at an intermediary point or at the ultimate end point.
- **Body:** (Mandatory) Contains the XML data comprising the message being sent.
- **Fault:** (Optional) An optional Fault element that provides information about errors that occurred while processing the message

SOAP Envelope Element

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 is a packaging mechanism. Every SOAP message has a root Envelope element. 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.

```
<?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">
 ...
 Message information goes here
 ...
</SOAP-ENV:Envelope>
```

SOAP Header Element

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.

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 more than one header is defined, all immediate child elements of the SOAP header are interpreted as SOAP header blocks.

SOAP Header element can have following two attributes

a) **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.

b) **Must Understand attribute**: Indicates whether a Header element is optional or mandatory. If set to true ie. 1 the recipient must understand and process the Header attribute according to its defined semantics, or return a fault.

SOAP Header

```
<?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>
<t:Transaction
xmlns:t="http://www.tutorialspoint.com/transaction/"
SOAP-ENV:mustUnderstand="true">5</t:Transaction>
</SOAP-ENV:Header>
....
</SOAP-ENV:Envelope>
```

SOAP Body Element

The SOAP body is a mandatory element which 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. The body contains mandatory information intended for the ultimate receiver of the message.

Example: SOAP Body

```
<?xml version="1.0"?>
<SOAP-ENV:Envelope
.....
<SOAP-ENV:Body>
<m:GetQuotationResponsexmlns:m="http://www.tp.com/Quotation">
```

```
<m:Quotation>This  is Qutation</m:Quotation>
</m:GetQuotationResponse>  </SOAP-ENV:Body>  </SOAP-ENV:Envelope>
```

SOAP Fault Element

When 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, the address of the SOAP processor that generated. A SOAP Message can carry only one fault block. Fault element is an optional part of SOAP Message For the 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 Element	Description
<faultCode>	A text code used to indicate a class of errors.
<faultString>	A text message explaining the error
<faultActor>	A text string indicating who caused the fault. This 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.
<detail>	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 must be used in the faultcode element when describing faults

```
<?xml  version='1.0'  encoding='UTF-8'?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/1999/XMLSchema"> <SOAP-ENV:Body>
<SOAP-ENV:Fault>
<faultcodexsi:type="xsd:string">SOAP-ENV:Client</faultcode>
<faultstringxsi:type="xsd:string">
      Failed to locate method (ValidateCreditCard) in class
      (examplesCreditCard) at  /usr/local/ActivePerl-5.6/lib/
```

```
site_perl/5.6.0/SOAP/Lite.pm line 1555. </faultstring>
</SOAP-ENV:Fault> </SOAP-ENV:Body> </SOAP-ENV:Envelope>
```

SOAP Encoding

SOAP includes a built-in set of rules for encoding data types. This 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. Structs contain multiple values, but each element is specified with a unique accessor element.

```
<?xml version='1.0' encoding='UTF-8'?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://www.w3.org/2001/12/soap-envelope"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<SOAP-ENV:Body>
<ns1:getProductResponse
  xmlns:ns1="urn:examples:productservice"
  SOAP-ENV:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<return xmlns:ns2="urn:examples" xsi:type="ns2:product">
<name xsi:type="xsd:string">Red Hat Linux</name>
<price xsi:type="xsd:double">54.99</price>
<description xsi:type="xsd:string">
      Red Hat Linux Operating System
</description>
<SKU xsi:type="xsd:string">A358185</SKU>
</return> </ns1:getProductResponse> </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

SOAP

In this example, a GetQuotation request is sent to a SOAP Server over HTTP. The request has a QuotationName parameter, and a Quotation will be returned in the response.

The namespace for the function is defined in "http://www.xyz.org/quotation" address.

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

SOAP response:

```
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:Bodyxmlns:m="http://www.xyz.org/quotation">
<m:GetQuotationResponse>
<m:Quotation>Here is the quotation</m:Quotation>
</m:GetQuotationResponse> </SOAP-ENV:Body> </SOAP-ENV:Envelope>
```

Advantages and Disadvantages of SOAP

- **Language neutrality:** SOAP can be developed using any language.
- **Interoperability and Platform Independence:** SOAP can be implemented in any language and can be executed in any platform.

- **Simplicity:** SOAP messages are in very simple XML format.
- **Scalability:** SOAP uses HTTP protocol for transport due to which it becomes scalable.

Disadvantages of SOAP

- **Slow:** SOAP uses the XML format which needs to be parsed and is lengthier too which makes SOAP slower than CORBA, RMI or IIOP.
- **WSDL Dependence:** It depends on WSDL and does not have any standardized mechanism for dynamic discovery of the services.

Differences between SOAP and HTTP

SOAP	HTTP
It is a protocol for accessing web services and based on XML.	Http (HyperText Transfer Protocol) is a transfer used protocol, which called a stateless protocol because each command is executed independently, without any knowledge of the commands that came before it.
SOAP provides a way to communicate between applications running on different operating systems, with different technologies and programming languages.	This is the main reason that it is difficult to implement Web sites that react intelligently to user input.