SOAP

Simple Object Access Protocol

SOAP in a nutshell

"SOAP is a protocol specification for invoking methods on servers, services, components and objects. SOAP codifies the existing practice of using XML and HTTP as a method invocation mechanism. The SOAP specification mandates a small number of HTTP headers that facilitate firewall/proxy filtering. The SOAP specification also mandates an XML vocabulary that is used for representing method parameters, return values, and exceptions." [DevelopMentor]

- Like XML-RPC, SOAP was developed to be a platform and language independent method of passing information between services, objects, and servers.
- Provides a simple standardized mechanism for moving structured information.

Where did it come from?

- Microsoft, UserLand, DevelopMentor, IBM, Ariba, Commerce One, Compaq, HP, IONA, Lotus, SAP
  ~ Submitted to W3C May 2000.
- XML based protocol
- Intended to be a simple wire-protocol for exchange of info. Largest use may be over Internet HTTP.

W3C

- July 9, 2001
  W3C Releases First Public Working Draft for SOAP Version 1.2
- Working draft produced by the XML Protocol Working Group.
- Specification can be found:
  http://www.w3.org/TR/soap12/

Motivation

- Many Distributed applications communicate using remote procedure calls (RPC) between distributed objects like DCOM and CORBA.
- HTTP isn’t designed for those objects, so RPC calls aren’t easily adapted to the Internet.
- Security problems exist for those methods of RPC, so most firewalls and proxy servers are set to block this traffic.
- HTTP is supported by all Internet browsers and servers, so SOAP presents a nice protocol for doing RPC.

SOAP

SOAP IS:

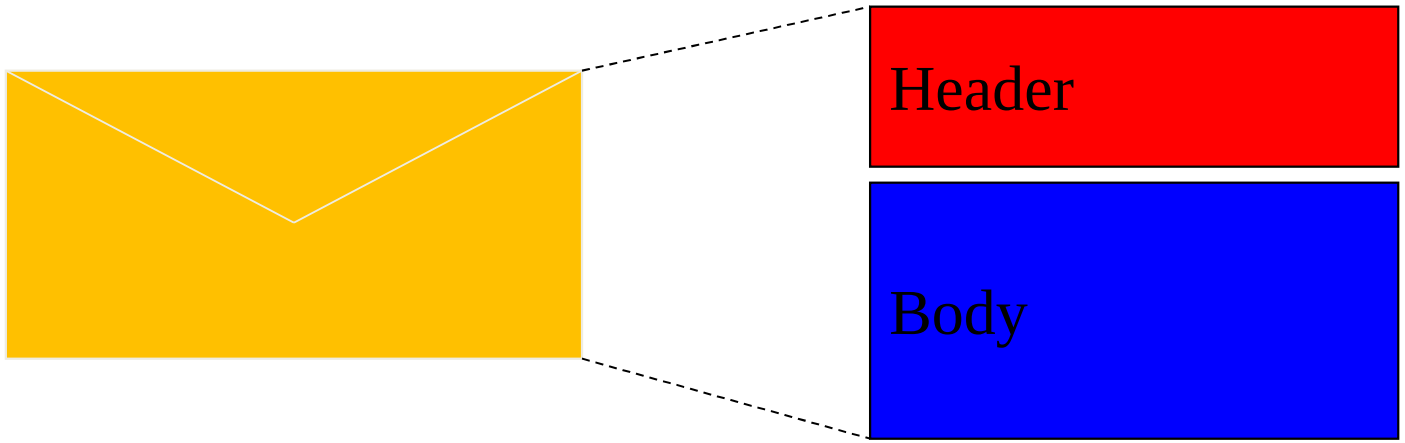
- Lightweight communication protocol
- For communication between applications - MEPs
  one-way, request/response, solicit response, notification, multicast, etc..
- Designed to communicate via HTTP
- Not tied to any component technology
- Not tied to any programming language
- Based on XML
- Simple and extensible

General (Basic) Structure SOAP Message

- Envelope
  Defines the content of the message
- Header (optional)
  contains header information
- Body
  contains call and response information

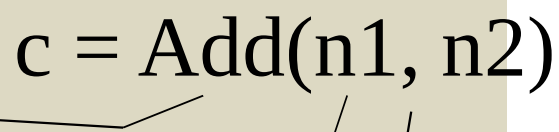
SOAP Elements

- SOAP Encoding
- Envelope package
- Header/Body pattern
  - Similar to how HTTP works



Simple Example

```
<Envelope>
   <Header>
      <transId>345</transId>
   </Header>
   <Body>
      <Add>
         <n1>3</n1>
         <n2>4</n2>
      </Add>
   </Body>
</Envelope>
```



Envelope

- MUST be associated with SOAP envelope namespace: http://www.w3.org/2001/06/soap-envelope
- SOAP serialization namespace: http://www.w3.org/2001/06/soap-encoding
  Encoding Style attributes can contain a URI describing how the data should be serialized.
- SOAP message MUST NOT contain a DTD or Processing Instructions.

SOAP Namespaces

- The SOAP envelope has the namespace identifier “http://www.w3.org/2001/06/soap-envelope”
- The SOAP serialization has the namespace identifier "http://www.w3.org/2001/06/soap-encoding"
- The SOAP mustUnderstand fault namespace identifier "http://www.w3.org/2001/06/soap-faults"
- The SOAP upgrade namespace identifier "http://www.w3.org/2001/06/soap-upgrade"

Header

- Optional. If present, must immediately follow the SOAP Envelope XML element followed by any header entries. Often contains meta-information regarding the method call.
- Actor attribute ←who should process message
- mustUnderstand attribute ←how to process(default is “0” if not present)

```
<env:Header>
 <t:Transaction
   xmlns:t=“some-URI” env:mustUnderstand=“1”>
    5
 </t:Transaction>
</env:Header>
```

Body

- Message to exchange. Most often for RPC calls and error reporting.
- Immediate child element of SOAP Envelope XML element (follows Header, if present).
- Contains serialized method arguments.
- Remote method name is used to name the method call's XML element and must immediately follow the SOAP body opening XML tag.

SOAP Example in HTTP

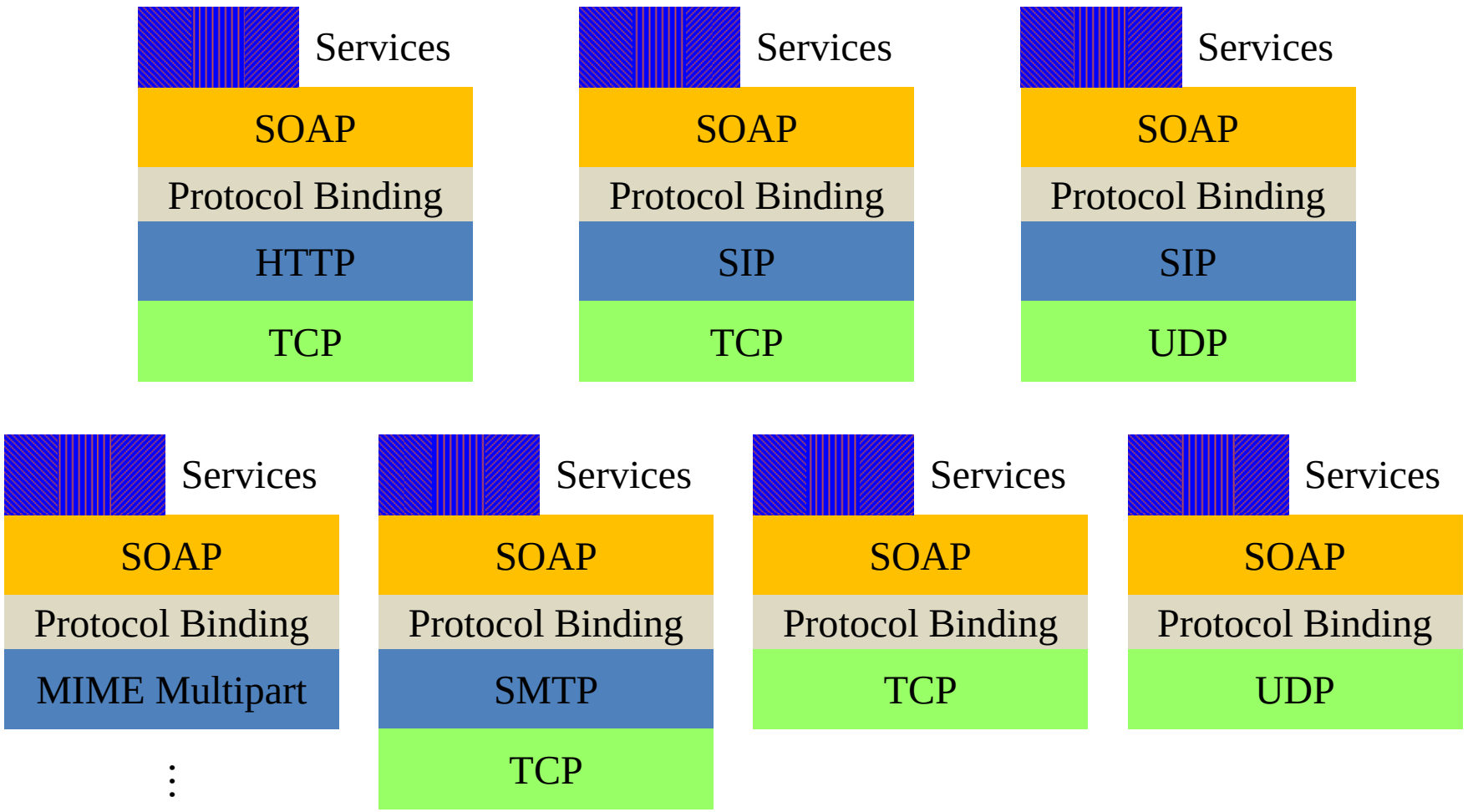
```
POST /Accounts/Henrik HTTP/1.1
Host: www.webservicebank.com
Content-Length: nnnn
Content-Type: text/xml; charset="utf-8"
SOAPAction: "Some-URI"

<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP:Header>
      <t:Transaction xmlns:t="some-URI" SOAP:mustUnderstand="1">
            5
      </t:Transaction>
   </SOAP:Header>
   <SOAP:Body>
      <m:Deposit xmlns:m="Some-URI">
         <m:amount>200</m:amount>
      </m:Deposit>
   </SOAP:Body>
</SOAP:Envelope>
```

… or SOAP by Itself…

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope”
  SOAP:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
   <SOAP:Header>
     <m:MessageInfo xmlns:m="http://www.info.org/soap/message">
           <m:to href="mailto:you@your.com"/>
           <m:from href="mailto:me@my.com"/>
           <m:contact href="mailto:someone@my.com">
     </m:MessageInfo>
   </SOAP:Header>
   <SOAP:Body>
     <msg:Message xmlns:m="http://www.info.org/soap/message">
           <msg:subject>Your house is on fire!</msg:subject>
           <msg:feed href="ram://livenews.com/yourhouse"/>
     </msg:Message>
   </SOAP:Body>
</SOAP:Envelope>
```

SOAP Stack Examples



SYNTAX RULES

- MUST be encoded using XML
- MUST have a SOAP Envelope
- CAN have a SOAP header
- MUST have a SOAP Body
- MUST use the SOAP Envelope namespace
- MUST use the SOAP Encoding namespace
- Must NOT contain a DTD reference
- Must NOT contain XML Processing

ERRORS: SOAP Fault Element

- Error messages from a SOAP application are carried inside a Fault element.
- Must appear as an element w/in the <Body> element and can only appear once in a SOAP message.
- Fault sub elements:

<faultcode>	MUST: --code identifying the error, for use by the software
<faultstring>	MUST: --error as a string
<faultactor>	MUST(Apps not acting as ultimate destination of SOAP message) --who caused the error
<detail>	MUST(if contents of Body could not be successfully processed) --specific error information

- Fault Codes:

VersionMismatch	Invalid namespace for the SOAP Envelope Element
MustUnderstand	A child element of the Header element, with the mustUnderstand attribute set to "1", was not understood
Client	The message was incorrectly formed or contained incorrect information.
Server	There was a problem with the server so the message could not be process

- Example:

```
<env:Fault>
  <faultcode>env:MustUnderstand</faultcode>
  <faultstring>SOAP Must Understand Error</faultstring>
</env:Fault>
```

Easy Transport….HTTP

- Why HTTP?
  HTTP has a request/response model allowing SOAP to embed request parameters in HTTP request, and SOAP response parameters into HTTP responses.
- HTTP apps MUST use the media type “text/xml” when including SOAP entity bodies in HTTP messages.
  Content-Type: text/xml; charset=“utf-8” ←SOAP HTTP Header info
- Basicly…sending XML formatted messages across the network using HTTP.

Part of the HTTP payload

- Commonly used with HTTP protocol (response/request model) thereby making it very scalable.
- SOAPAction HTTP Header Field.

  SOAPAction: “Some-URI”
- Firewall access? Most firewalls/proxies are opened to port 80 for HTTP giving SOAP a widely used transport protocol across distributed systems. The SOAPAction HTTP header field allows firewalls/proxys to filter SOAP messages if desired.
- Security? No security has been implemented with SOAP, but consider HTTPS/SSL.

Stock Quote example embedded in an HTTP POST Request

- POST /StockQuote HTTP/1.1
  Host: www.stocksserver.com
  Content-Type: text/xml; charset="utf-8" ←SOAP HTTP Header info
  Content-Length: nnnn
  SOAPAction: "Some-URI" ←SOAP HTTP Header info

```
<env:Envelope
 xmlns:env="http://www.w3.org/2001/06/soap-envelope">
  <env:Body>
       <m:GetStockQuote xmlns:m="Some-URI"
            env:encodingStyle="http://www.w3.org/2001/06/soap-encoding">
         <symbol>SUNW</symbol>
       </m:GetStockQuote >
  </env:Body>
</env:Envelope>
```

- Sent from the client to the service. First few lines are HTTP headers and SOAP HTTP headers. Followed by the SOAPAction HTTP header and SOAP Envelope XML element. The SOAP Body contains application defined element(s), as defined by the service.

Stock Quote Response example embedded in HTTP Response

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset="utf-8"          ←SOAP HTTP Header info
Content-Length: nnnn

<env:Envelope    xmlns:env=http://www.w3.org/2001/06/soap-envelope">
  <env:Body>
       <m:GetStockQuoteResponse    xmlns:m="Some-URI"
            env:encodingStyle="http://www.w3.org/2001/06/soap-encoding">
         <price>5.00</price>
     </m:GetStockQuoteResponse>
  </env:Body>
</env:Envelope>
```

Client/Server…

- In order for SOAP to work, the client must have code running that is responsible for building the SOAP request.
- In response, a server must also be responsible for understanding the SOAP request, invoke the specified method, build the response message, and return it to the client.
- These details are up to you.
- There already exist SOAP implementations for languages such as Perl and Java.

SOAP box

- Inherently stateless if it uses HTTP for transport.
- Doesn’t implement security. However transport with HTTP allows for SSL and HTTPS at the application level.
- SOAPAction HTTP header field allows your firewall to filter SOAP method invocations/deny SOAP processing altogether. Firewall filters SOAP packets based on the object name, particular method, or a combo of the two.

SOAP for RPC

- Design goals of SOAP: simplicity & extensibility.
- RPC with XML allows for uniform representation of remote procedure calls and responses.
- You're not limited to HTTP protocol binding…just a natural choice for the request/response model.

Advantages of SOAP

- Uses HTTP which is widely used and scalable
- Wide remote system interoperability
- Flexible for growth because of XML properties
- It but can be used for RPC.

Disadvantages of SOAP

- No good way to describe the serialization pattern (XML schema is optional at this point)
- Parsing of SOAP packet and mapping to objects reduces performance
- Doesn’t implement security because it is a wire protocol—relies on HTTP

References…and other great links!

- W3C SOAP Version 1.2, W3C Working Draft 9 July 2001
  http://www.w3.org/TR/soap12/
- W3c SOAP notes
  http://www.w3.org/TR/SOAP/
- Index for articles and various SOAP related resources.
  http://www.perfectxml.com/soaptutor.asp
- soap primer
  http://discuss.develop.com/archives/wa.exe?A2=ind0007&L=soap&F=&S=&P=9777
- soap school
  http://www.w3schools.com/SOAP/default.asp
- Older soap specification
  SOAP: Simple Object Access Protocol 18 April 2000
  http://static.userland.com/xmlRpcCom/soap/SOAPv11.htm
- understanding soap
  http://softwaredev.earthweb.com/article/0,,10455_641321,00.html
- how soap works
  http://www-106.ibm.com/developerworks/library/ws-peer3/