8 SOAP & REST

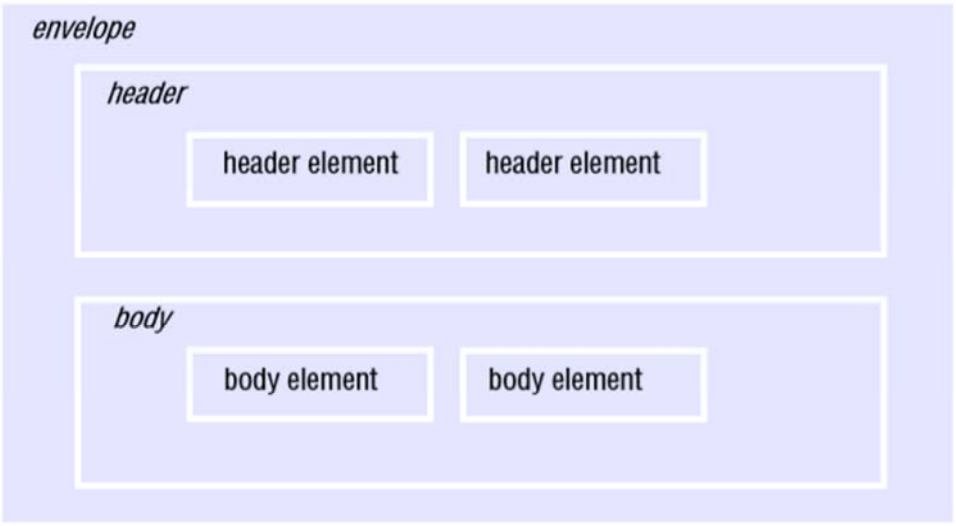
Saturday, January 12, 2019 2:04 PM

Compare SOAP web services and RESTful web services. Describe the methods used in RESTful webservices and the common data formats.

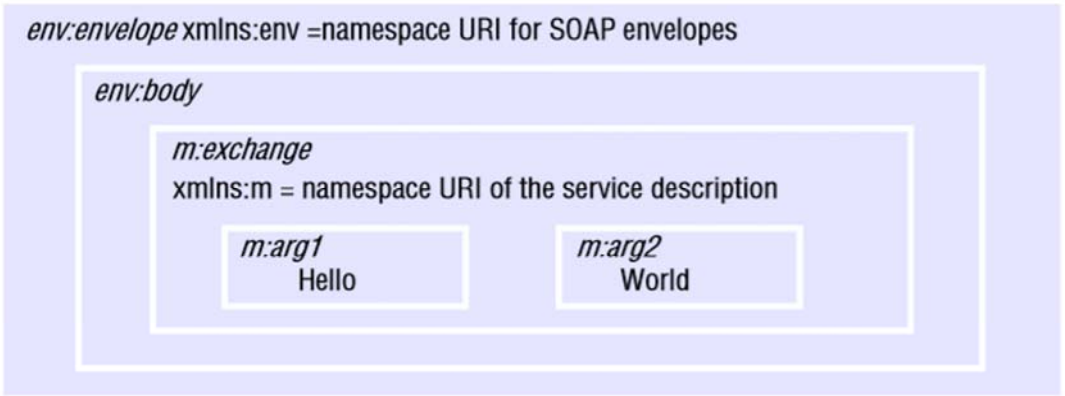
Messages represented using
- SOAP: XML
- REST: (GET PUT POST DELETE)
- Compare the two

- Service references: URI, usually a URL in this context = service endpoint

SOAP
- Defines a scheme for using XML to carry messages
- Initially based on HTTP, extended to UDP, TCP, SMTP



- SOAP spec states
  - How XML represents the contents of messages
  - How a pair of messages can be combined to form a request-reply pattern
  - How the XML elements should be processed
  - How HTTP/SMTP should be used to carry them
- Header is optional



In this figure and the next, each XML element is represented by a shaded box with its name in italics,at the top left corner,followed by any attributes and its content

- Replies have m:exchangeResponse and m:res1

```
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>
```

```
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>
```

REST > Representational State Transfer

- Representational: A resource (object) can be represented in different ways (typically XML and/or json) [the resource is "something" that is different from both the xml and the json]
- State: behind every webservice there is a state = the collection of resources (managed by the service) However, the service itself is stateless (= you don't have any session data)
- State Transfer: using HTTP commands to access / update resources

HTTP Commands

- GET - safe (=doesn't change / destroy any resource or information)
- PUT - idempotent (= only modifies once)
- DELETE - idempotent
- POST - none of the above
- PATCH - none of the above

Command	Response		
	/users	/users/27	/users/27 (user doesn't exist)
GET	200 - all users	200 - user27	404 - not found
PUT	403	204 - no content	201
DELETE	403	204 - no content	204 - no content
POST	201 - Create user	403	403 - forbidden

PATCH	403	200 - patch user	404

Difference between PUT and PATCH
if you use PUT, you send the entire object. If you use PATCH, you just send the changed parts; e.g. changing a password

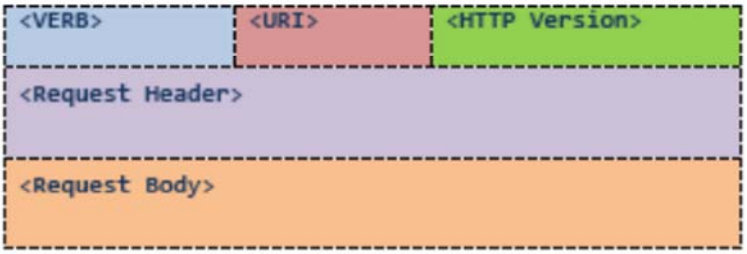


Figure 1: HTTP request format.

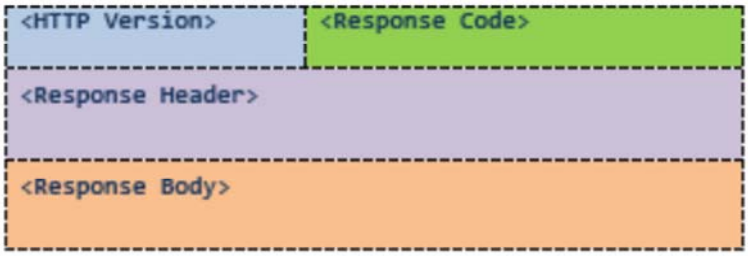


Figure 2: HTTP response format.

- Sample url: http://MyService/Persons?id=1