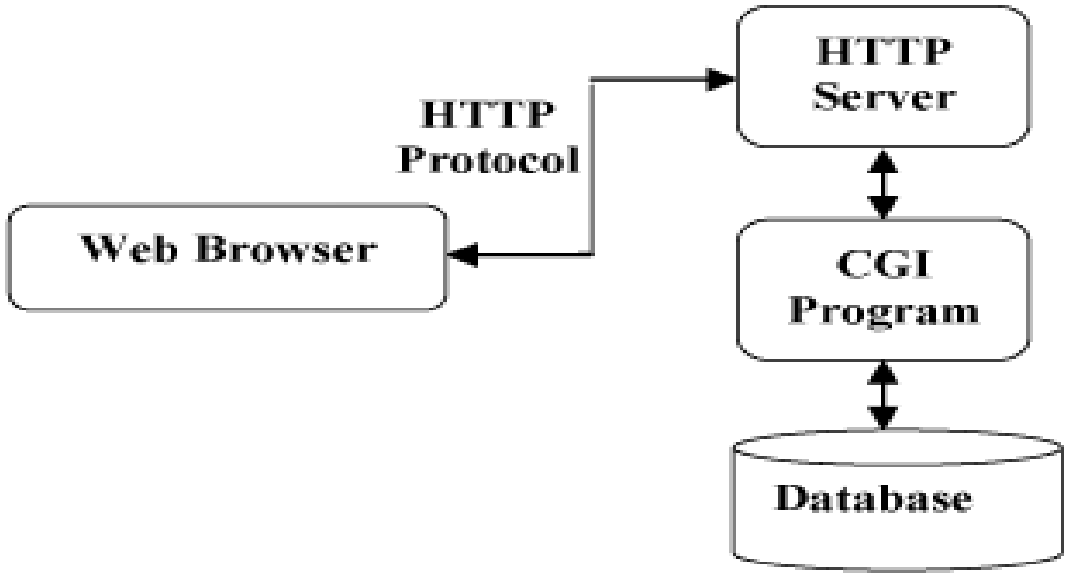
Webonise Lab
Induction Program

Web Concepts

HTTP Protocol

HTTP stands for Hypertext Transfer Protocol. It is an TCP/IP based communication protocol which is used to deliver virtually all files and other data, collectively called resources, on the World Wide Web.

- HTTP is connectionless
- HTTP is media independent
- HTTP is stateless



HTTP

- It is based on a request/response paradigm. A client establishes a connection with a server and sends a request to the server in the form of a request method
- It's a network protocol used to deliver virtually all files and other data (collectively called resources) on the World Wide Web, whether they're HTML files, image files, query results, or anything else. Usually, HTTP takes place through TCP/IP sockets.
- A browser is an HTTP client because it sends requests to an HTTP server (Web server), which then sends responses back to the client. The standard (and default) port for HTTP servers to listen on is 80, though they can use any port.
- HTTP is used to transmit resources, not just files. A resource is some chunk of information that can be identified by a URL (Uniform Resource Locator)
- It ia a stateless protocol i.e It does not maintain any connection information between transactions.

HTTP

HTTP methods -

GET method -

The Get is one the simplest Http method. Its main job is to ask the server for the resource. If the resource is available then then it will given back to the user on your browser. That resource may be a HTML page, a sound file, a picture file (JPEG) etc. We can say that get method is for getting something from the server. It doesn't mean that you can't send parameters to the server.

POST method -

The Post method is more powerful request. By using Post we can request as well as send some data to the server. We use post method when we have to send a big chunk of data to the server, like when we have to send a long enquiry form then we can send it by using the post method.

HEAD method -

Asks for the response identical to the one that would correspond to a GET request, but without the response body

HTTP

TCP -

- TCP: Transmission Control Protocol
- Layer on top of IP
- Data is transmitted in streams
- Reliability ensured by retransmitting lost datagrams, reordering, etc.
- Connection-oriented
- establish connection between client and server
- data streaming in both directions
- close connection
- Socket: end point of connection, associated a pair of (IP address, port number)

HTTP

HTTP methods -

GET method -

The Get is one the simplest Http method. Its main job is to ask the server for the resource. If the resource is available then then it will given back to the user on your browser. That resource may be a HTML page, a sound file, a picture file (JPEG) etc. We can say that get method is for getting something from the server. It doesn't mean that you can't send parameters to the server.

POST method -

The Post method is more powerful request. By using Post we can request as well as send some data to the server. We use post method when we have to send a big chunk of data to the server, like when we have to send a long enquiry form then we can send it by using the post method.

HEAD method -

Asks for the response identical to the one that would correspond to a GET request, but without the response body

SoA

Why SOA?

- SOA architecture enables seamless Enterprise Information Integration. Here are some of the Benefits of the Service Oriented Architecture:
- Due to its platform independence, it allows companies to use the software and hardware of their choice .
- There is no threat of vendor lock-in
- SOA enables incremental development, deployment, and maintenance.
- Companies can use the existing software (investments) and use SOA to build applications without replacing existing applications
- The training costs are low, so the available labor pool can be used for running the applications

SOA

Service Oriented Architecture or SOA for short is a new architecture for the development of loosely coupled distributed applications.

Broadly SOA can be classified into two terms: **Services** and **Connections**.

Services:

A service is a function or some processing logic or business processing that is well-defined, self-contained, and does not depend on the context or state of other services. Example of Services are Loan Processing Services, which can be self-contained unit for process the Loan Applications.

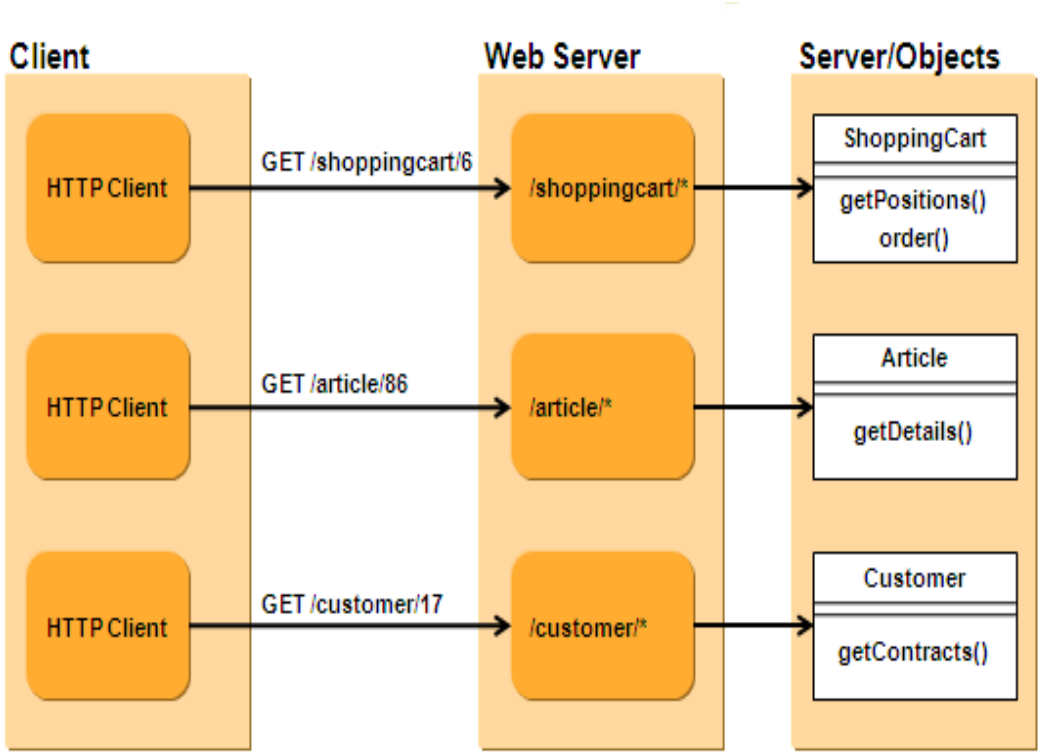
Connections:

Connections means the link connecting these self-contained distributed services with each other, it enable client to Services communications. In case of Web services SOAP over HTTP is used to communicate the between services.

REST

REST stands for Representational State Transfer. (It is sometimes spelled "ReST".) It relies on a stateless, client-server, cacheable communications protocol -- and in virtually all cases, the HTTP protocol is used.

REST uses the existing features o the HTTP protocol, and thus allows existing layered proxy and gateway components to perform additional functions on the network such as HTTP caching and security enforcement.



REST

The query will probably look like this:

http://www.acme.com/phonebook/UserDetails/12345

Note that this isn't the request body -- it's just a URL. This URL is sent to the server using a simpler GET request, and the HTTP reply is the raw result data -- not embedded inside anything, just the data you need in a way you can directly use.

Key goals of REST include:

- Scalability of component interactions
- Generality of interfaces
- Independent deployment of components
- Intermediary components to reduce latency, enforce security and encapsulate legacy systems

SOAP

What is Soap ?

SOAP is a protocol for building Web Services

SOAP is a “protocol framework” for building Web Services

SOAP

SOAP Processing Model

- SOAP sender
- SOAP receiver
- SOAP message path
- Initial SOAP sender (Originator)
- SOAP intermediary
- Ultimate SOAP receiver

SOAP

Packaging – Soap

SOAP RPC:

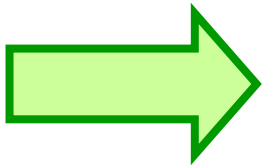
- encode and bind data structures into xml.
- encode an RPC call

```
<service name="MessageService">

<port name="MessageServicePort"
binding="tns:MessageServiceBinding">

<soap:address
location="http://localhost:8080/setMessage/"/>

</port>

</service>
```

SOAP

Serialization

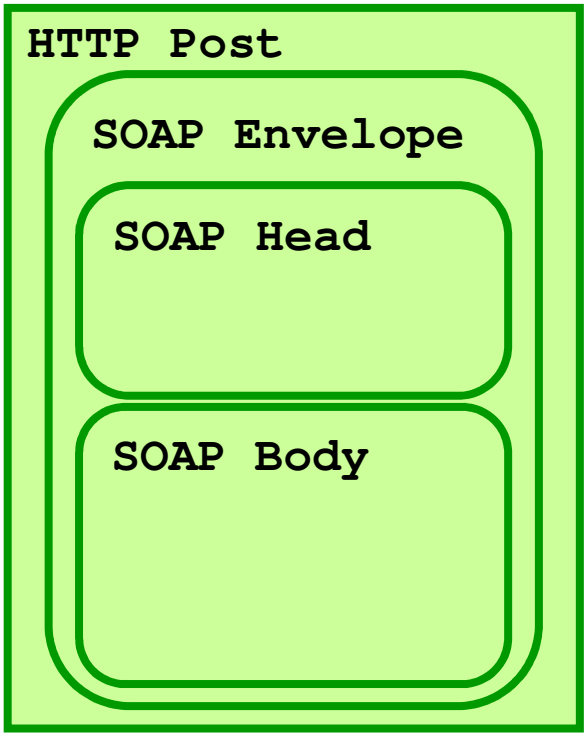
```
class PurchaseOrder {
String item = "socks";
int amount = 1;
}
```



```
<PurchaseOrder>
<item type="xsd:string">
socks
</item>
<amount type="xsd:int">
1
</amount>
</PurchaseOrder>
```

SOAP

Packaging – Soap




```
<s:Envelope xmlns:s="URN">
        <s:header>
                <s:transaction
xmlns:m="soap-
transaction">
                        <m:transactionID>
                                1234

        </m:transactionID >
                </s:transaction>
        </s:header>

 <s:Body>
                <n:purchaseOrder xmlns:n="URN">
                        <n:item>socks</n:item>
                        <n:amount>1</n:amount>
                </n:purchaseOrder>
         </s:Body>
 </s:Envelope>
```

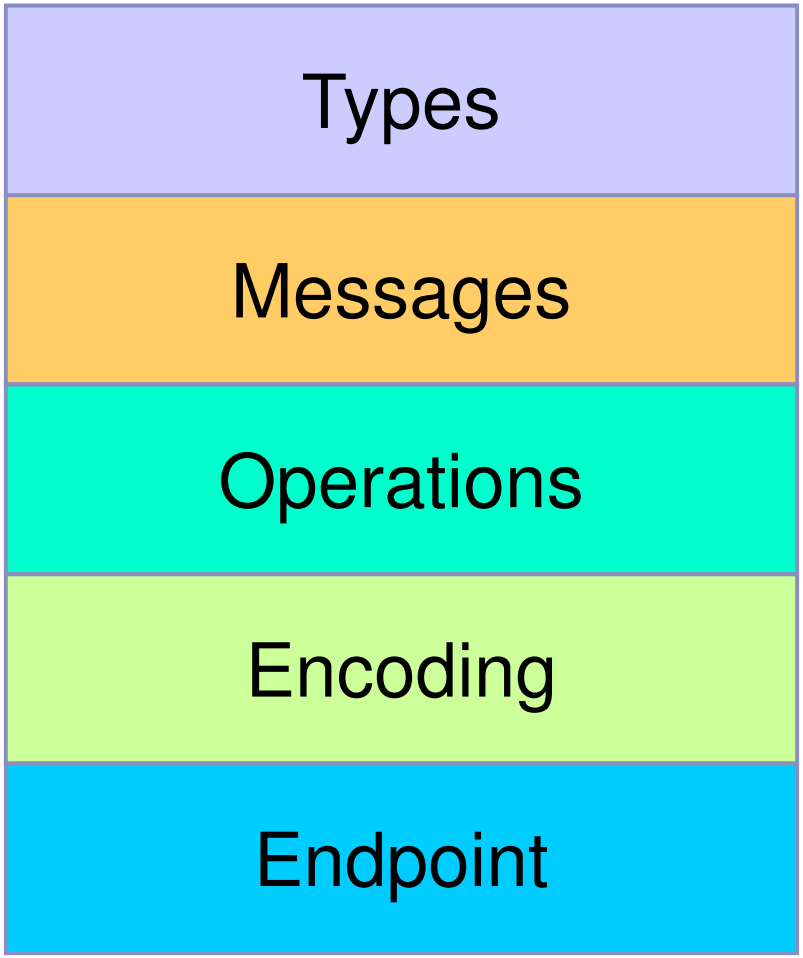
SOAP

Description – WSDL

- Web Services Description Language
- “Web Services Description Language (WSDL)

provides a model and an XML format for describing

Web services.” w3c.org



SOAP

Types

```
<types>
<schema targetNamespace=" IMessageService.xsd"
xmlns="…/XMLSchema" xmlns:SOAPENC="…/soap/encoding/"/>
</types>
```

Messages

```
<message name=“purchase">
<part name=“item" type="xsd:string"/>
<part name=“quantity" type="xsd:integer"/>
</message>
```

SOAP

Operations

```
<operation name="setMessage">

<input name="setMessageRequest“
message="tns:setMessageRequest"/>

<output name="setMessageResponse“
message="tns:setMessageResponse"/>

</operation>
```

SOAP

EndPoints

```
<service name="MessageService">

<port name="MessageServicePort"
binding="tns:MessageServiceBinding">

<soap:address location="http://localhost:8080/setMessage/"/>

</port>

</service>
```

CURL

cURL is a command line tool for getting or sending files using URL syntax.

cURL supports FTP, FTPS, Gopher, HTTP, HTTPS, SCP, SFTP, TFTP, Telnet, and many other protocols.

Examples :

To send http get request:

curl "http://www.example.com?email=coach@coach.com"

To send http post request:

curl --data "username=coach@coach.com" http://www.example.com

Assignment

Web service call with data in JSON

Thank You