Web Services

What is Web Services?

- A Web service is an interface that describes a collection of operations that are network accessible through standardized XML messaging.

- It provides platform independent communication.
  - For example, there is one application which is written in java over linux platform and it uses oracle database and there can be another application which is written in C++ over a window platform and it uses mysql as database so these two application can communicate with each other over the internet with the help of web services.

What is the difference between a Web Application and a Web Service?

- A web application is an application that is accessed through a web browser running on client’s machine.

- A web service is a system of software that allows different machines to interact with each other through a network.

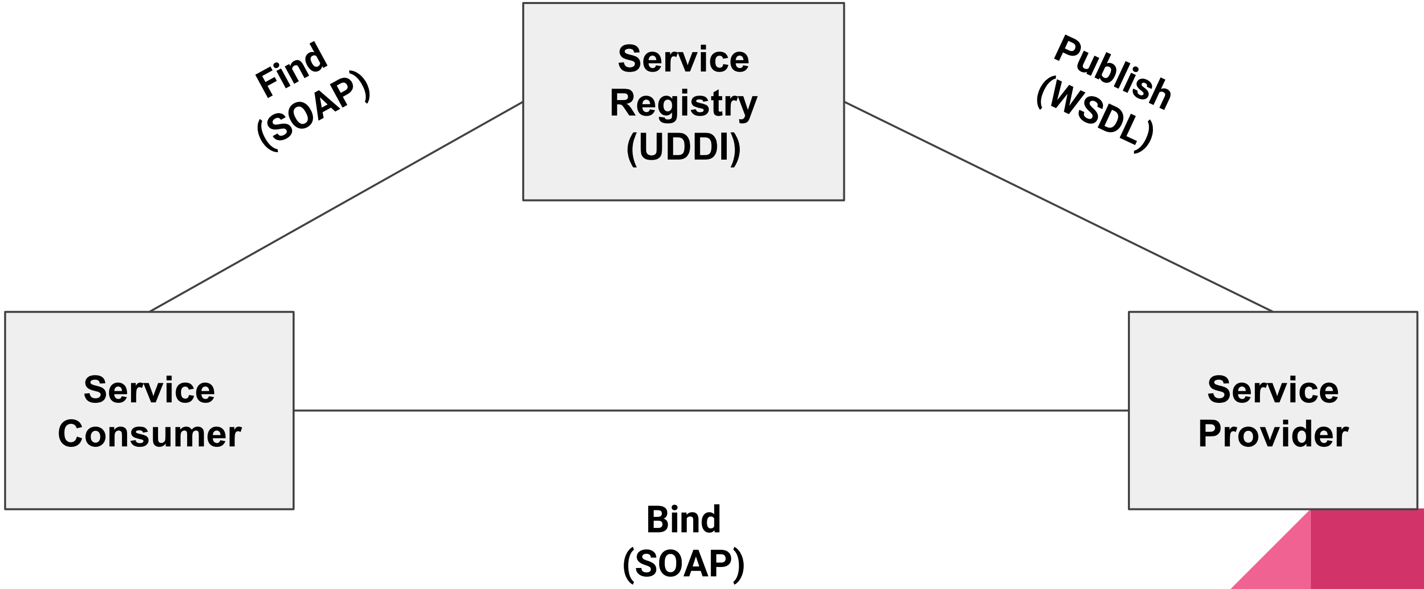
- Web services use XML, SOAP, WSDL and UDDI open standards to achieve this task.

Web Service Architecture

- Web Services involve three major roles
  - Service Provider
  - Service Registry
  - Service Consumer

- Three major operations surround web services
  - Publishing – making a service available
  - Finding – locating web services
  - Binding – using web services

Web Service Architecture



Types of Web Services

- SOAP(Simple Object Access Protocol)
  - Medium: HTTP(Post)
  - Format: XML

- REST(Representational State Transfer)
  - Medium: HTTP(Post, GET, DELETE, PUT etc)
  - Format: XML/ JSON

WSDL and UDDI

- **WSDL**
  - WSDL is an interface for the Web Services.
  - Service provider creates this interface for there web service and service consumer can get the WSDL to access the web services.

- **UDDI**
  - UDDI is a directory where all the service provider will put their WSDL document so that the consumers query and get hold of the WSDL and once the consumer has the WSDL they can make use of the web services.

Components of Web Services

WSDL

UDDI

where does this WSDL and UDDI fits into the architecture?

Continue...

- Service consumer needs to know all the details about the web services.
  - What are the web services?
  - What are the various components?
  - What are the various functionalities?
  - What are the parameters?
  - How exactly to use the web services.

- To enable this every service provider publishes description about the web services and this is kind of an interface to the web service which is written in **XML** language where all the attributes and functionalities described and this is called as **WSDL(Web Service definition language)**

Continue...

- There are two ways a service consumer can get hold of this WSDL document
  - If the service consumer and the service provider already know each other then service provider can directly handover this WSDL document or the WSDL URL to the service consumer and then service consumer uses the web services.

  - When the service consumer and service provider don’t know each other how a service consumer get hold of a WSDL document. So to enable this all the web services provider publishes their description of their web services on an online registry or a directory where consumer can query and search the web services. This online registry/directory is called UDDI.

SOAP Web Service

- The Web Services that follows basic set of specification includes {SOAP, WSDL and UDDI} are known as SOAP Web Services. These guidelines given by central body(W3C).

- Extended set of guidelines includes security and other things as well and it is used to create a web services for enterprise.

SOAP(Simple Object Access Protocol)

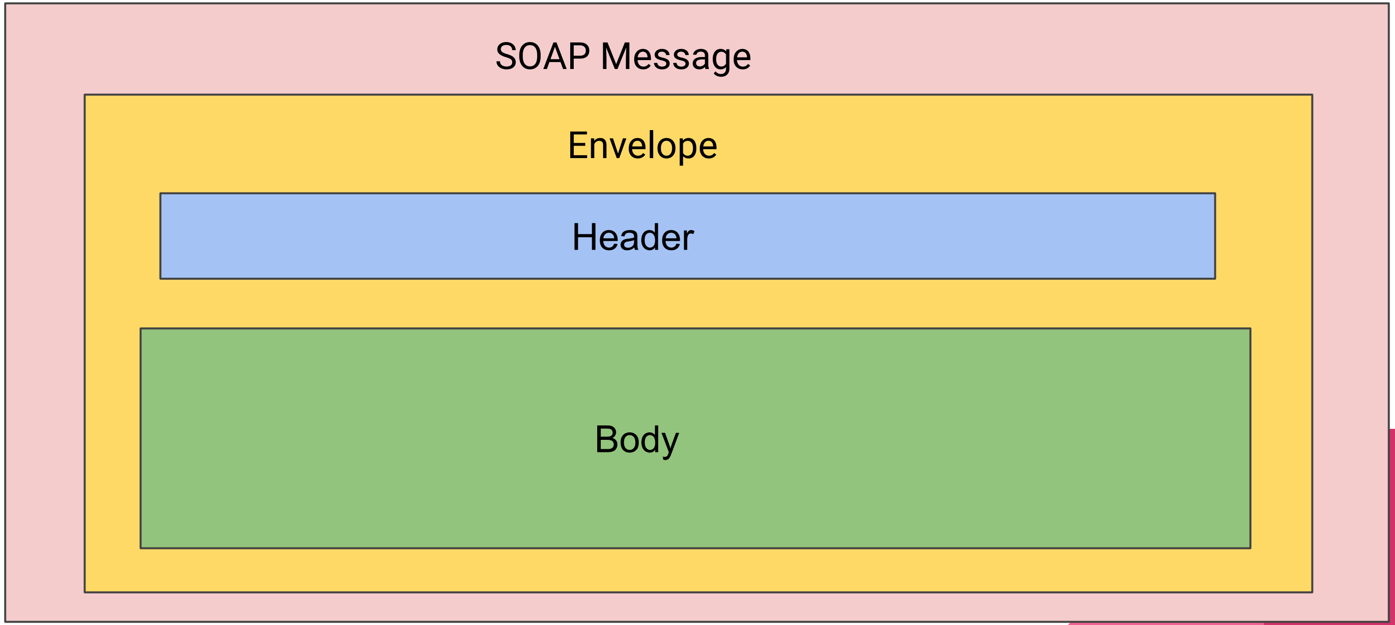
- It is a Protocol/Rules/Definition on how to two application will communicate with each other.

- SOAP protocol says that all the exchange of data or message b/w these two application has to be in a common format and in case of SOAP it has to be a XML.

- So SOAP message is a XML document which follows some structure.

Continue...

SOAP message contains the envelope which is the root element and inside envelope it can have a header which is optional and gives information about the complex types and authentication parameter and it contains body.



SOAP Message

Envelope

Header

Body

Continue...

- **Envelope**
  - Envelope is the root element of a soap message.
  - This is the basic unit of the XML document which contains other units like header and body.

- **Header**
  - Header elements provides information about the message itself.
  - Header can include authentication, routing information etc.

- **Body**
  - Body contains the actual data of the request meant to be sent to the server.
  - All the parameters that client want to send as a request will be inside the body.

Example

```
<?xml version = “1.0”?>

<soap:Envelope xmlns:soap = "http://schemas.xmlsoap.org/soap/envelope/"
soap:encodingstyle = “http://www.w3.org/2001/12/soap-encoding”>

    <soap:body pb = “http://www.example.com/employees”>
          <pb:GetEmployee>
                   <pb:EmployeeId>123</pb:EmployeeId>
          </pb:GetEmployee>
    </soap:body>

<soap:Envelope>
```

REST Web Service

- It is an architectural style.

- REST defines a set of principles to be followed while designing a service for communication or data exchange between two application.

Principles of REST architecture

- Uniform Interface
  - Resource: everything is a resource.
  - URI: any resource/data can be accessed by a URI.
- Stateless
  - All client-server communication are stateless.
- Cacheable
  - Happens at client side.
- Layered System
  - Layers can exist between client and server.
- Code on demand
  - Ability to download and extend code on client-side.

Uniform Interface

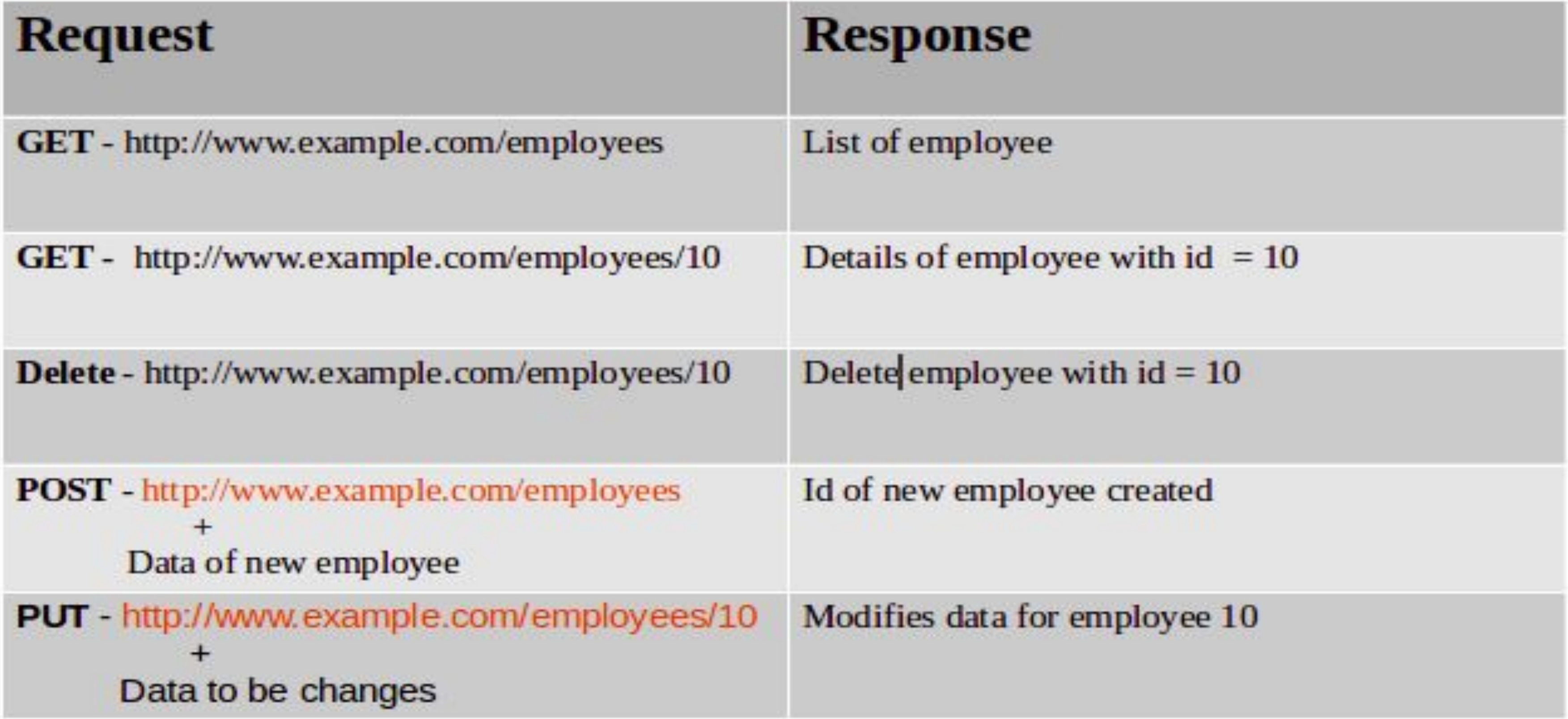
- Resource: Everything is a resource.
  - Concept of resource says we can define any information or any module as a resource.
  - For example in Employee management System application which contains modules Employee, Department, etc. So every module can be defined as resource.
- URI
  - Any data or any module can be accessed with the help of URI.
  - For example to access (Employee Management System) application which is hosted somewhere, we have to use domain name and then slash(/) to access the employees resources.
  - So this application must be having tables of departments, employee and these tables have linkages which can be one to many or one to one.
- To retrieve employee details with given employee id:
  - http://example.com/employees/id

Continue...

- At the backend these tables have all the interlinkages.
  - To retrieve the list of all employees of particular department
  - http://example.com/departments/123/employees

- By using HTTP method along with URI any resource information can be modified.

Continue...



Request	Response
GET - http://www.example.com/employees	List of employee
GET - http://www.example.com/employees/10	Details of employee with id = 10
Delete - http://www.example.com/employees/10	Delete employee with id = 10
POST - http://www.example.com/employees + Data of new employee	Id of new employee created
PUT - http://www.example.com/employees/10 + Data to be changes	Modifies data for employee 10

Continue...

- Example of REST Web Service
  - http://www.thomas-bayer.com/sqlrest
  - http://www.thomas-bayer.com/sqlrest/CUSTOMER
  - http://www.thomas-bayer.com/sqlrest/CUSTOMER/0

```
<?xml version = “1.0”?>

<soap:Envelope xmlns:soap = "http://schemas.xmlsoap.org/soap/envelope/"
soap:encodingstyle = “http://www.w3.org/2001/12/soap-encoding”>

    <soap:body pb = “http://www.example.com/employees”>
          <pb:GetEmployee>
                   <pb:EmployeeId>123</pb:EmployeeId>
          </pb:GetEmployee>
    </soap:body>

<soap:Envelope>
```

SOAP message to get the details of the employee with given id.

```
http://www.example.com/employees/123
```

REST Web service URL to get the details of employee with given id.

Stateless

- Server does not require to store the state of a session.

- If the concept of a session is required all information should be stored at client side and sent with every subsequent request.
  - For example, if a person is adding item to his cart on a shopping website, the state of the cart should be stored at the client side only. And this state of the cart has to be sent from the client to server whenever it is required.

- Note:
  - Each request from the client to the server must contains all of the data that is necessary to handle the request.
  - There is no overload of maintaining or storing the session therefore it improves web service performance.

Caching

- Caching happens at the client side.

- Caching is used to maintain login scenario where the server has to know whether the authentication is successful or not.

- Server generates responses that indicate whether they are cacheable or not to improve the performance by reducing number of requests for duplicate requests.

- Server does this by including **Cache-control** and **Last-Modified (data value)** in HTTP Response Header.

- Note:
  - Client uses the cache-control header to determine whether to cache the resource or not.