**Web services**: To interact one application to other application of different domain.

1. SOAP based web services: JAX-RPC and JAX-WS
2. RESTful web services: JAX-RS

WS is a specification. They are providing only guidelines and rules not API.

Architecture:

WS suggested some components to share information between interoperable applications.

1. WSDL (web service description language)
2. UDDI (optional)
3. SKELETON
4. STUB
5. SOAP Protocol
6. HTTP Protocol

WS need all these components to share the information.

1. WSDL is an XML file which contains information about the service class that can help client application to invoke. This XML file contains method information and end point URL.
2. WSDL generation tools takes this service class and endpoint URL as an input and generates WSDL file.
3. Service provider will store this WSDL file in UDDI registry software with unique name. This unique name and location of UDDI registry will be sent to client.
4. Client will interact with UDDI and receives this WSDL file.
5. Using this XML file client will generate STUBS (class of client application) or Proxies with the help of STUB generation tools. STUB will have different implementation from service provider method implementation.
6. Now client will create an object for the STUB and invoke the method, then STUB will prepare XML document and store this method calling information. This XML document is called SOAP request. STUB uses some predefined SOAP tags to generate SOAP request.
7. This SOAP request needs to send to service provider to get the required information.
8. HTTP protocol will carry this SOAP request from client to server location and handovers to SKELETON.
9. SKELETON is a predefined class written in server application language. This SKELETON will take information from SOAP request and invokes server application method.
10. Now server-side application will return the result to SKELETON and this SKELETON will prepare SOAP response. HTTP will carry this SOAP response to client. STUB will read this SOAP response and gives the result to application.

JAVA Web Service API’s:

1. JAX-RPC (Synchronous) – **SOAP** Based
2. JAX-M (JAVA API for XML – **Messaging**, Asynchronous)
3. JAX-WS (Synchronous) - **SOAP** Based
4. JAX-RS (Synchronous) – **REST** Based

* Any one of these webservice API’s is necessary to develop both web service and web service client.
* Synchronous – client gets immediate response from web service.
* Asynchronous – client gets late response from web service.

**JAX-RPC: (Remote Procedure Call) - SOAP**

* It is a specification given by Sun micro systems.
* Can develop SOAP based web service and SOAP based web service client.
* Implementations:
  + JAX-RPC-SI given by sun microsystems.
  + **AXIS1** given by Apache foundation. ([web service implementation](https://www.youtube.com/watch?v=fgwc9FL5uMM&list=PLd3UqWTnYXOkofdmEC1VB42cX8ZUwNEB7&index=7)) ([web service client](https://www.youtube.com/watch?v=nrn_5udpTOY&index=9&list=PLd3UqWTnYXOkofdmEC1VB42cX8ZUwNEB7))
  + WebLogic implementation.
  + WebSphere Implementation
  + JBoss Implementation.

**JAX-WS: SOAP**

* Implementations:
  + JAX-WS-RI 🡪 sun micro systems.
  + Metro 🡪 sun micro systems.
  + Axis2 🡪 Apache
  + Apache CXF 🡪 Apache (supports Spring Integration)
  + WebLogic implementation.
  + WebSphere Implementation
  + JBoss Implementation 🡪 Red hat
  + Glassfish 🡪 sun micro systems

**JAX-RS: (RESTful Services)**

* Implementations:
  + JERSEY 🡪 Sun micro systems
  + REST EASY 🡪 Red Hat
  + REST LET
  + Apache CXF
  + Apache wink

**JAX-M:** usually we will not develop Asynchronous web services, even if we need, we go for **JMS** (java Messaging services).

There are 2 types of web service clients:

* Proxy based client
  + If client generates STUB from WSDL file.
* DII client
  + If client uses STUB generated by third party vendor API’s.