**WSDL (Web Services Description Language)**

Is an XML-based language for describing Web services and how to access them.  
Specifies the location of the service and the operations (or methods) that privides the services

**SOAP**XML-based protocol to let applications exchange information over HTTP. used to encode the information in Web service request and response messages before  
sending them over a network

**JAXB**Maps Java classes to XML representations

The ability to marshal Java objects into XML and the inverse, unmarshal XML back into Java objects.  
In other words, JAXB allows storing and retrieving data in memory in any XML format,   
without the need to implement a specific set of XML loading and saving routines for the program's class structure.  
JAXB is particularly useful when the specification is complex and changing.   
In such a case, regularly changing the XML Schema definitions to keep them synchronised with the Java definitions can be time consuming and error prone.

**XSLT:**XML style sheet and Transformation : --Convert XML in to plain HTML

TransformerFactory factory = TransformerFactory.newInstance();  
StreamSource xslStream = new StreamSource(inputXSL);  
Transformer transformer = factory.newTransformer(xslStream);  
StreamSource in = new StreamSource(dataXML);  
StreamResult out = new StreamResult(outputHTML);  
transformer.transform(in, out);

**JAXP**The Java API for XML Processing, or JAXP (pronounced jaks-p), validating and parsing XML documents.

SAX --> Simple API for a larg XML document

DOM: --> Document Object Model --> Read xml file to the memory and parse it to tree of Object

JAXB

**JAX-RPC :**Allows a Java application to invoke a Java-based Web Service with a known description

**JAX-WS 2.0**The JAX-RPC name is misleading, developers assume that all JAX-RPC is about is just RPC,

not Web Services. By renaming JAX-RPC to JAX-WS we can eliminate this confusion.

UDDI

- EJB remote web services :

1- Create @Remote API interface --> Name of WS

@Remote  
 public interface CRMRetrieveWS extends Serializable{

@WebMethod  
 StatusFormResponse status(@WebParam String enrolmentId);

2- Create Remote EJB Bean @Stateless, @lacal ,,@Webservice

@Stateless  
 @Local  
 @WebService(name = "CRMRetrieveService")  
 public class CRMRetrieveWSBean implements CRMRetrieveWS {

Dojo is a [JavaScript](http://en.wikipedia.org/wiki/JavaScript) framework targeting the many needs of large-scale client-side web development. For example, Dojo abstracts the differences among diverse browsers to provide APIs that will work on all of them (it can even run on the server under [Node.js](http://en.wikipedia.org/wiki/Node.js)); it establishes a framework for defining modules of code and managing their interdependencies; it provides build tools for optimizing JavaScript and CSS, generating documentation, and unit testing; it supports [internationalization, localization](http://en.wikipedia.org/wiki/Internationalization_and_localization), and [accessibility](http://en.wikipedia.org/wiki/Web_accessibility); and it provides a rich suite of commonly needed utility classes and user-interface widgets.