**SOAP** – Simple Object Access Protocol – Is a web-service that transfers SOAP objects in XML format over HTTP. Was designed to ship data from one place to another over the internet. Each soap message has a defined structure that defines a very strongly typed messaging framework that relies heavily on XML and schemas. Both parties must use this structure to communicate data with one another. The structure for communication happens inside of an Envelope element. The body element is required and is where you put the data. Within <body> you have another optional element called <fault> which is where you put instructions of what to do when things go wrong. The <header> normally has security standards about how to authenticate users etc…

The SOAP itself is a protocol (over HTTP) for developing SOAP-based APIs.

SOAP relies exclusively on XML to provide messaging services

REST is based on HTTP – which itself is a very unsecure protocol. It supports basic authentication and communication encryption through TLS. Any further security should be additionally implemented at the server.

**SOAP** achieves **interoperability** by providing an XML format that can be transferred as simple text over HTTP or any other protocol capable of transferring XML text.

protocols such as Simple Mail Transfer Protocol (SMTP), File Transfer Protocol (FTP), or any other protocol that can transfer text. These other protocols could be used to make asynchronous SOAP calls, meaning the client application could invoke the SOAP service but would not have to wait for a response from the server.

**REST** is an architecture that allows you to transfer data between computers over the net. It uses HTTP methods and protocol. The Service exposes its resources by providing an end point (maybe an api doc to explain how to use its endpoints) that the client will use to interact with the service. The data is transferred between the two in a representational format (json, xml, text, html, image) so, it does not matter what each program is written in, as long as both applications are using rest. Rest is stateless so, each request/response happen in isolation and only the client keeps track of that or any future state details (only client remembers that or any past conversations).

REST -REpresentational State Transfer – is an architectural style that makes use of HTTP + Can build SOA Applications

REST can structure data into XML, YAML, or any other machine-readable format, but usually, JSON is preferred.

REST is almost synonymous to HTTP, through REST specification does not mandate it

REST can use data in Comma Separated Value (CSV), JavaScript Object Notation (JSON), and Really Simple Syndication (RSS)

SOAP security is well standardized through [WS-SECURITY](https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss), which is very much feature-rich and easy to implement into application code

REST allows a greater variety of data formats, whereas SOAP only allows XML.

Coupled with JSON REST is generally considered easier to work with and thanks to JSON, REST offers better support for browser clients.

REST provides superior performance, particularly through caching for information that’s not altered and not dynamic.

REST is the protocol used most often for major services such as Yahoo, Ebay, Amazon, and even Google.

REST is generally faster and uses less bandwidth. It’s also easier to integrate with existing websites with no need to refactor site infrastructure.

Questions:

**What does SOAP do?** allows programs that run on totally different operating systems Windows/Ubuntu/Unix to communicate using Extensible Markup Language (XML).

**What are SOAP Headers/Attributes?** Custom information such as session IDs, authentication token, etc.

**WSDL?** A contract that says how to work with a service. What operations it has, how you must call them, what parameters it expects

**What is a SOAP endpoint?** It is a URL.  It identifies the location on the built-in HTTP service where the web services listener listens for incoming requests. Calling applications must specify this **endpoint** when they send web services messages

**What is the HEADER used for?** Basically, its metadata about the Req -or-Resp as key value pairs. Use the headers to provide information about the entity or other resource that you are requesting.

**What is an HTTP request header?** It has the client info. What page are they requesting, what browser are they using/capability/version and their OS type etc…

**What is an HTTP response header?** It has the Server details and info about the response and the data being sent. Response code, last time date the resource was modified. Content-length, cache control

**What is the format for making URI/URL in a RESTful webservice?** http://[www.xyz.com/apiService?anyparams](http://www.xyz.com/apiService?anyparams)

Protocol + hostname + main-resource ? filter / methods & any parameters

**What does HTTP status code 406 mean?** Not Acceptable. Sent when the Server doesn't have formats that the Client can accept.

**What is statelessness?** Server does not keep track of state, it is the responsibility of the Client . Rest is stateless so, each request/response happen in isolation