Difference between SOAP and RESTful webservices

(OR)

Difference between SOAP and REST

S.No	SOAP	REST
1	**Developer View:** Object oriented	**Developer View:** Resource Oriented
2	**Standards Based:** Yes . SOAP web services are based on SOAP and WS-* specifications For acquiring security tokens,it uses WS-Trust. For conveying security tokens, it uses WS-Security For defining policy, it uses WS-Policy For suppoting distributed ACID transactions, it uses WS-AtomicTransaction and WS-Coordination For acquiring interface definitions, it uses WS-MetadataExchange For providing end-to-end reliability, it uses WS-ReliableMessaging For establishing security context, it uses WS-SecureConversation	**Standards Based:** No
3	**Security:** SSL, WS-Security . WS-Security provides end-to-end security covering message integrity and authentication	**Security:** SSL
4	**Transactions :**	**Transactions :**

	WS-AtomicTransaction	No
5	**Reliability :** WS-ReliableMessaging	**Reliability :** Application specific
6	**Performance:** Good	**Performance:** Better Caching and lower message payload makes RESTful web services performance efficient and scalable
7	**Caching :** No	**Caching :** GET operations can be cached
8	**Message Size :** Heavy, has SOAP and WS-* specific markup	**Message Size :** Lightweight, no extra xml markup
9	**Message Communication protocol :** XML	**Message Communication protocol :** XML, JSON, other valid MIME type . This flexibility of REST makes its extremely useful in providing consumer need specific message payloads
10	**Message Encoding :** Yes SOAP Web Services support text and binary encoding	**Message Encoding :** No RESTful encoding is limited to text
11	**Service Description :** WSDL	**Service Description :** No formal contract definition In REST, no formal way to describe a service interface means more dependence on written documentation
12	**Human intelligible Payload :** No	**Human intelligible Payload :** Yes
13	**Developer Tooling :**	**Developer Tooling :**

	Yes Complexity of SOAP Web Services dictates the need for using frameworks to facilitate rapid application development.	Minimal or none REST on the other hand due to its simplicity can be developed without any framework
14	**Orientation :** Wraps business logic	**Orientation :** Accesses resources/data
15	**Abbreviation:** SOAP stands for Simple Object Access Protocol	**Abbreviation:** REST stands for Representational State Transfer
16	**Who is using SOAP?** Google seams to be consistent in implementing their web services to use SOAP, with the exception of Blogger, which uses XML-RPC. You will find SOAP web services in lots of enterprise software as well.	**Who is using REST?** All of Yahoo's web services use REST, including Flickr, del.icio.us API uses it, pubsub, bloglines, technorati, and both eBay, and Amazon have web services for both REST and SOAP.
17	**Simplicity:** No	**Simplicity:** Yes
18	**Transport protocol support:** HTTP, SMTP, JMS Multiple transport protocol support makes SOAP Web Services flexible	**Transport protocol support:** HTTP

Areas where SOAP based WebServices is a great solution:

Asynchronous processing and invocation: If application needs a guaranteed level of reliability and security then SOAP 1.2 offers additional standards to ensure this type of operation. Things like WSRM – WS-Reliable Messaging etc.

Formal contracts: If both sides (provider and consumer) have to agree on the exchange format then SOAP 1.2 gives the rigid specifications for this type of interaction.

Stateful operations: If the application needs contextual information and conversational state management then SOAP 1.2 has the additional specification in the WS* structure to support those things (Security, Transactions, Coordination, etc). Comparatively, the REST approach would make the developers build this custom plumbing.

Areas where RESTful WebServices are a great choice:

Limited bandwidth and resources: Remember the return structure is really in any format (developer defined). Plus, any browser can be used because the REST approach uses the standard GET, PUT, POST, and DELETE verbs. Again, remember that REST can also use the XMLHttpRequest object that most modern browsers support today, which adds an extra bonus of AJAX.

Totally stateless operations: If an operation needs to be continued, then REST is not the best approach and SOAP may fit it better. However, if you need stateless CRUD (Create, Read, Update, and Delete) operations, then REST is suitable.

Caching situations: If the information can be cached because of the totally stateless operation of the REST approach, this is perfect.

And, further updates on difference between questions and answers, please visit my blog @ http://onlydifferencefaqs.blogspot.in/