API Features Individualizing of Web Services: REST and SOAP

Anshu Soni, Virender Ranga

Abstract: ***Web Services are combination of open protocols and standards to allow communication between client and server. It provides an interoperability between contrasting applications. Representational state Transfer (REST) and Simple Object Access Protocol (SOAP) are the two main popular used web services now-a-days. REST is an architectural style based, whereas SOAP is a underlying protocol. Both services are used to handle the communication on the world wide web (www). Both services have some advantages and drawbacks and it is the decision of web developer to decide which service is best to use according to its requirements. The aim of this research work is to design a REST API and SOAP API by JAX-RS and JAX-WS, respectively and gives a comparative analysis of Application Programming Interface (API) features (in terms of response time, memory usage, execution speed and so on) of these services by using API testing tool like Postman. This gives insight view of which service is better to use as per requirements. The result of experiments shows that the response time of SOAP is approximate takes 4ms to 7ms more than REST. It has been observed that as number of API increases, SOAP takes approximate 1MB to 2MB more memory usage than REST.***

Index Terms**:** ***Architectural style, JAX-RS, JAX-WS, Jersey, Postman, Protocol, REST, SOAP, Tomcat, Web service.***

I. INTRODUCTION

Web service is a client-server application that provides medium to communicate between different applications over the network [1]. It is a kind of a server or a software that uses the XML messaging in standard format through the Internet as XML can be used to cipher all transmission. It can be built over TCP/IP, HTTP, Java, HTML, and XML. It provides inter-portability between contrasting applications and language independence also. It is a software object set up on network providing services over the Internet. Fig. 1 shows the virtual representation of web service. Web service accepts request from a client and deliver the response to the client applications. Web services basically have three components: SOAP, WSDL and UDDI. SOAP is a protocol that relies on XML for permeate web services. For communication, SOAP recommends world wide web consortium. Web Services Description Language (WSDL) acts as an interface between web applications and contains data about web services like method name, parameter and how to access it. It is a part of UDDI (Universal Description, Discovery and Integration), a framework for defining, declaring, combining the web service. It can make communication through CORBA, SOAP, Java RMI protocol. Web service permits to uncover the functionality of code over the network. Mainly two types of web services are there, first is REST and second one is SOAP. REST, an architectural style for developing web services. In REST, the client sends request and server returns response to the client by the use of resources. The language of REST is based on nouns and verbs. REST web services is not dependent on any protocol, but almost every REST service makes the use of HTTP verbs are used to regulate the operations on resources. Resources can be videos, web pages, pictures, anything that can be allowed in computer-based system. REST does not have any restriction over the representation format. SOAP is a protocol that defines how web services can interact with each other or interacting with client applications. The communication in the SOAP service is XML based. SOAP follows strongly typed strict standards whereas REST does not follow many standards, but follows constraints defined by Roy Fielding in their dissertation [2]. SOAP is language and platform independent also. It defines its own security i.e. WS-Security.

Both services are well suited for communications, but there is a slight difference between these two. SOAP follows a protocol where as REST is an architectural style for developing web services. JAX-RS and JAX-WS are the Java APIs for REST and SOAP, respectively. Both services have advantages and disadvantages in some area of concern. Hence, it is better to figure out in which situation which service can be used to get the better results. This research work aims to provide such a decision between these two web services based on our strict analysis. The rest of the paper is organized as follows: section II considers the related work in this research area. In section III, background study is shown in brief. Section IV depicts our main proposed work. In section V, comparison is discussed between REST and SOAP based on state-of-the art parameters. Section VI concludes our research findings.

Revised Manuscript Received on June 15, 2019

Anshu Soni, anshusoni1995@gmail.com, Department of Computer Engineering, National Institute of Technology, Kurukshetra, Haryana.
Virender Ranga, virender.ranga@nitkkr.ac.in, Department of Computer Engineering, National Institute of Technology, Kurukshetra, Haryana.

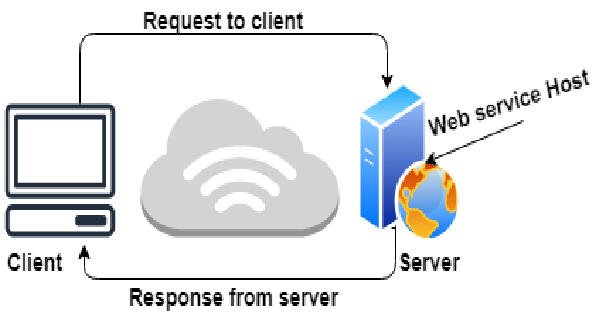


Fig.1 Virtual representation of web service

II. RELATED WORK

Chuangwei Zhang et.al[1] generate a design that makes web services management more convenient, reduce system implementation and also improves efficiency.

Paul Adamczyk [3] distinguishes two architectural styles REST and SOAP and explain briefly about the RESTful web services and demonstrate four principles of REST coined by Roy Fielding based on a survey of existing web services.

Mark Massé [4] discusses simple rules to design web services. The authors explain why REST API should be designed and configured and also discussed the tips to address the client's needs.

Chia Hung Kao et.al [5] introduce the testing framework for REST based web applications and examine their performance. The software tester provides an unified process for test case composition and test scripts generation for the tests' execution. The proposed framework decreases the effort to understand the design and implementation of application.

Florian Haupt et.al [6] provide a framework for structural analysis of REST APIs which consists of metrics and graphical representation of APIs. They also convert the description of REST API, present in various languages into a approved model that illustrates the design of REST API explicitly.

Munonye K* et.al[7] analyze the performance of REST web services based on Java and .Net implementations an

d conclude their performance. On various test scenarios, they found that Java-based web services perform better for GET method around 80.36\% and .Net performs better for PUT method and performance is 11.6\% lower. They came across the conclusion that java based web services should be used on business-to-business purposes while .Net based web services preferable on forms updating applications having user involvement regular update and deletion takes place.

Ritesh Sinha et. al[8] scrutinize a new approach for application integration on cloud with REST web services.

Pascal Giessler et. al[9] identify best practices for developing REST services and clarify the usage of web services on different applications. This paper illustrates eight different categories of web services and 23 best practices.\par

Dong Qiu et.al[10] simplify the usage of API in Java programming language. They discuss how API engage in software projects. They have also done the empirical analysis on 5000 open source Java projects, figure out the core API and mediator API in Java, knowing the concept of API libraries. They conclude that core API is not widely used, most of the APIs that are deprecated are still used.

Festim Halili et.al [11] discuss the information about the web services and describes their advantages and disadvantages with the tabular format.

Snehal Mumbaikar et.al [12] show the comparison between REST and SOAP web services for the mobile environment on the basis of different metric.

Antonio Navarro et.al [13] give an approach that conducts the SOAP and REST conversion. SOA Intermediate model act as a bridge between SOAP and REST.

Madhusudhan Govindaraju et.al [14] compare the performance of SOAP toolkit and give a conclusion on the basis of performance characteristics. They have conducted the experimental study on following toolkit : gSOAP 2.4, XSOAP4/XSUL, AxisJava 1.2 (with and without streaming enabled), AxisC++ 1.1.1 and .NET 1.1.4322.

Sehrish Malik et. al [15] describe the web services comparison in actuator network and design the structure of indoor actuator with SOAP and REST services and draw conclusion that REST gives result better as compared to SOAP.

Andy Neumann et.al [16] compare almost 4000 sites through Alexa.com to provide 500 sites that provide REST service, analyze JSON support and software documentation.

Juris Tihomirovs et.al [17] use software evaluation metrics to compare and analyze the SOAP and REST web service. For research purpose, they include data resources from various sites. They conclude for maintainability use rest service. The lines of code is lesser in SOAP compared with REST and also finds that response time of REST is lower than SOAP.

Jiali Bian et.al [18] come up with the notion of web services service layer as middle ware depends on OMS platform. They also introduce the design of web services packet layer and compare with response time, response usage and so on.

Christopher Kiama et.al[19] show a comparative study of REST and SOAP on the basis of performance, scalability measured via throughput values and so on take data from political parties registrar of Kenya.

Anil Dudhe et.al[20] concentrate on creating web service SOAP, REST with jersey, conduct experiments and perform tests on Apache Tomcat and draw the conclusion that REST is performed better than SOAP.

Vibha Kumari[21] list out the pros and cons of both REST and SOAP service and make conclusion of what to use and when to use.

Kishor Wagh et.al[22] present the comparison between two frameworks that are based on REST and SOAP and analyze their problems also.

To find out the most advisable framework between two that are good enough for wireless communication. REST is considered to be suitable but SOAP finds out to be more secure.

G.Sambasivam et.al[23] focus mainly on quality of service of web services by Multifaceted Matchmaking framework. They finds 21 QoS parameters that are relevant for service discovery. QoS-UDDI model has been developed to expand web service and presented a structure that establish the ranking of web service.

R.Padmanaban et.al[24] discuss on RESTful web services resources using primitive and also conduct a case study on REST services which results in high portability, scalability and reliability, which concludes high performance. electronically for review.

III BACKGROUND STUDY

A. REST API

Representational State Transfer (REST) is first given by Roy Fielding in 2000 as an architectural style for the advancement of web services. It designates the set of constraints to promote web services. Roy Fielding [2] defines seven constraints described below in their dissertation report:

- **Starting with NULL style**: It is a vacant set of constraints. It is the starting point to describe REST.
- **Client - Server**: Separation of client and server architectural style to evolve independently of each other. Thus, enhance the flexibility under user interface and scalability by simplify the components of server.
- **Stateless**: Transmission between client and server must be stateless. Any request of the client will not be saved. No session nor history be made of a client request. Management of resources is not required. This constraint bring about the properties of scalability, visibility, and reliability.
- **Cache**: To improve network efficiency, cache restraint is combined with client and server communication. This constraint defines that reply is stated as cacheable (or non-cacheable) explicitly (or implicitly). Cacheable response requires the client to reuse the response. Caching improves the performance over client-side and scalability be progressed over the server side.
- **Uniform Interface**: REST style is distinct from other network style through uniform Interface constraint. It decouples and simplifies the architecture enables to evolve independently. Four Interface constraints have been defined i.e. resources identification, resources Manipulation by representations, self-descriptive messages and hypermedia as the engine of application state (HATEOAS).
- **Layered System**: To increase Internet-Scale requirements, one more constraint has been added. In this architecture be composed of hierarchical layers so that each component was not able to view beyond the immediate layer towards they were connected. A layered constraint is applied to enclose legacy services. The disadvantage over this system is that it increases the overhead and delay to the data processing.
- **Code-on-demand** : Functionality be added under REST to downloading and executing code. It allows the client to reduce features that would be needed to be pre-implemented. This constraint improves system extensibility.

API acts as an interface that allows communication between two software programs. API uses HTTP protocol for cooperation between different programs are web services (REST or SOAP). Basically, the interaction of APIs is the request and response type between client and server. An API follows REST guidelines or standards to build web services are called REST API. REST is not defined as a protocol that develops over HTTP, but it follows constraints while developing APIs. For request and response, REST uses HTTP to define the desired action. To allow communications, there are resource methods or request types available:

- GET : Retrieve resource information.
- POST : New sub-ordinate Resource has been developed.
- PUT : Existing Resource will be updated.
- DELETE : Delete the existing resource identified by Request URI.

Fig. 2 represents REST API work flow following HTTP methods and return response to clients in JSON/XML format.

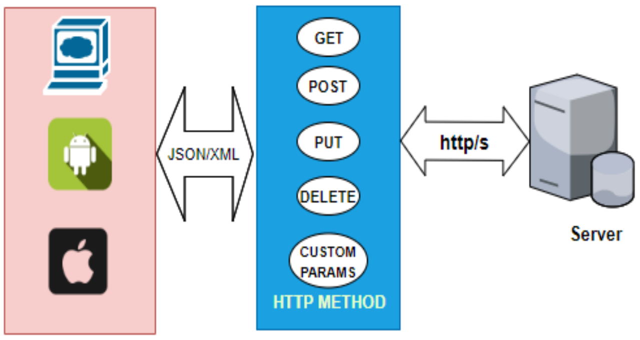


Fig.2 REST Flowchart

HTTP methods described above can be safe and idempotent as well. Table I exhibits the characteristics of HTTP methods which are idempotent and safe. Idempotent methods are those HTTP methods that can be called as many times as without any effect on stored data, whereas safe methods are those HTTP methods which do not update the resources and these methods are read-only.

Table I Characteristics of HTTP Methods

HTTP Method	Idempotent	Safe
GET	Yes	Yes
POST	No	No
PUT	Yes	No
DELETE	Yes	No

The representation of state can be in JSON or XML format. Usually, today most preferable response format will be in JSON format. There is some structure that is to be followed while creating a REST API.

- Base URI: To identify resources Uniform Resource Identifier(URI) be used.
  Example: http://api.example.com.
  URI format:
  scheme"://"authority"/"path["?"query]["\#"fragment]
- Resource Modelling: Representation of data is called resource. It can be a singleton or can be a collection and it may contain sub-resources also. Resource modelling can be separated by a forward slash in the URI path.Example:

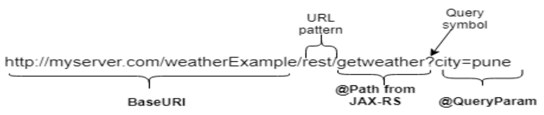

URL pattern
Query symbol
http://myserver.com/weatherExample/rest/getweather?city=pune
BaseURI
@Path from JAX-RS
@QueryParam

Fig.3. REST API structure with example

http://api.example.com/customers/\{customerid\}
- Query Design(Parameter): Query string is separated by "?" symbol. It contributes to the identification of resources uniquely. It contains parameters that identify hierarchically along with path segment.

Example : http://api.example.com/customer?item=pen

Fig.3 shows rest API structure consists of BaseURI, resources, Parameters, Query String.

B. SOAP API

SOAP composition of Simple Object Access Protocol is a messaging protocol that uses XML messaging format for providing communication over network. SOAP is an essential component of Service-Oriented Architecture (SOA) and SOA-related web services requirements. It was developed first time by the Mentor, User Land Company and Microsoft in 1998. In 1999, its first version was released. With version 1.2, SOAP service was used with high keenness. The main objective of SOAP is to forward data over the network. It used HTTP to transfer any information over the internet. To start the operations, SOAP messages are used. SOAP service is platform and language independent also.
Fig.4 represents the format of a SOAP message that is directed between the server and the client.

- Envelope - It is a decisive aspect in the SOAP message format that determines that given document as SOAP, characterize the begin and end of the message format. It is necessary for the SOAP message format.
- Header - This element contains optional data such as information of verification like authorization.
- Body - It is a necessary element that contains XML data. It contains request and response information having the data that is sent to the application.
- Fault - This element conveys information about errors occurs while sending the message such as Version Mismatch, client fault when message was incorrectly formed, Server fault when there was a problem with server. It contains errors and status information as well.

SOAP Binding is a method that gives how messages are exchanged over the Transport layer.
RPC(Remote Procedural Call) and Document Style are two different Binding style messaging requests under SOAP.

- RPC - It is basically an interaction of a client and a server with request and response. Format of request and response is XML, Envelope the root element determine the overall architecture of the message. RPC used the depiction and design of the message that is used for request and response. There is Synchronous communication in RPC that receives a response until a message request is sent. It is a simpler style and less capable as it is used to send smaller messages.
- Document Style - It has rich content and complex also. It is also stated as message-oriented style. XML data is passed as a body rather than as a parameter.

SOAP request defines two Header: Content-Type and Content-Length. Content-Type exemplify the MIME type for message request and response and character encoding also available for XML body. On the other hand, Content-Length header defines the amount of bytes required for message body in request and response.

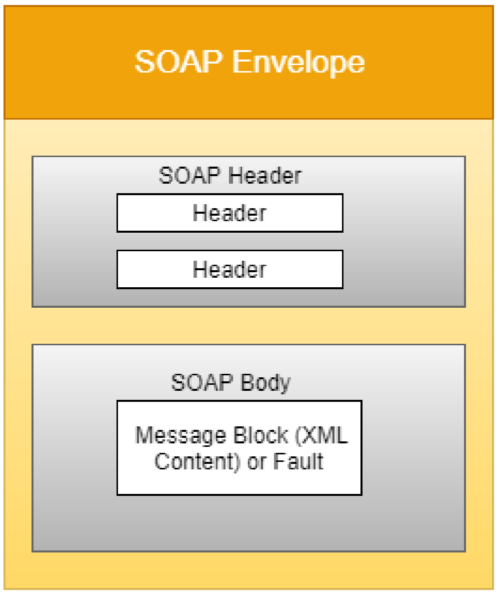


Fig.4 SOAP Messaging format

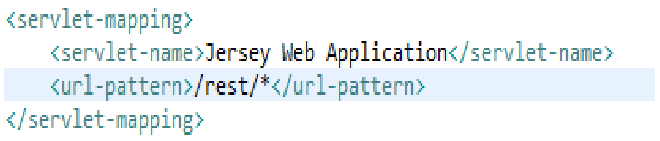
IV. PROPOSED APPROACH

This research work aims to compare REST and SOAP web service on the basis of parameters like response time, architecture, security, reliability, efficiency and compares their development and architectural style. Also makes a conclusion that which service is best to choose on the basis of requirements. It is also observed that which service is more preferable to apply for communication between a client and a server. In the netshell, this research work creates a REST API with JAX-RS and SOAP API with JAX-WS and make the comparison by showing bar-graphs, figures, etc. and test those API in Postman 7.1.1 for web service performance and their response time evaluation.

Technologies used to create REST API with JAX-RS and SOAP with JAX-WS:

- Jersey version - 2.25
- JAX-WS - 2.1
- Apache Maven - 3.8.0
- Tomcat 7.0
- Eclipse Java IDE Mars 2.0
- Postman 7.7.1

Jersey[25] is an open source framework for developing REST services in java that supports JAX-RS APIs. JAX-RS has been designed to make it easier for development of RESTful web services in java.
Under the web.xml, these changes to be reflected while creating REST service with URL Pattern "rest" as shown in Fig.5. Table II describes some of the annotations by JAX-RS. JAX-RS is a Java API for Restful web services grant backing for developing web service. It provides some annotations that export java package javax.ws.rs to make development of web services easier.



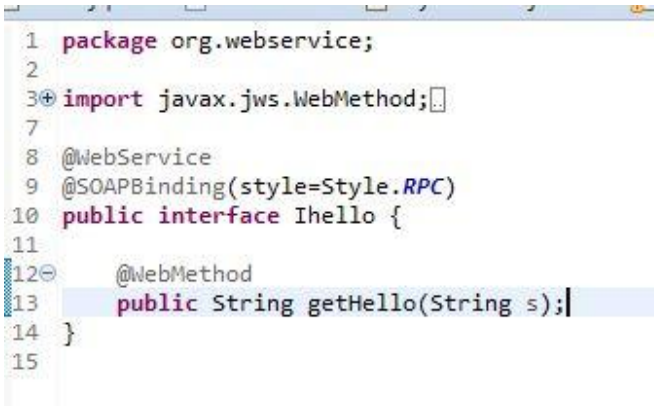
```
<servlet-mapping>
    <servlet-name>Jersey Web Application</servlet-name>
    <url-pattern>/rest/*</url-pattern>
</servlet-mapping>
```

Fig.5 Representation of web.xml

SOAP with JAX-WS[26] created with two styles one with RPC style and another one with Document Style(default) annotated with @SOAPBinding. For Interface Implementation @WebService is used as shown in Fig.6

Table II JAX-RS annotations

Annotations	Description
@Path	Specify the URI path
@GET	Annotate HTTP GET request method
@POST	Annotate HTTP POST request method
@DELETE	Annotate HTTP DELETE request method
@PathParam	Extract URI Path Parameters
@QueryParam	Extract URI Path Query Parameters
@Produces	Specify Media Type to be produced
@Consumes	Specify Media Type to be produced by REST



```
package org.webservice;

import javax.jws.WebMethod;

@WebService
@SOAPBinding(style=Style.RPC)
public interface Ihello {

    @WebMethod
    public String getHello(String s);
}
```

Fig.6 SOAP with JAX-WS

Table III. describes some of the JAX-WS annotations to make web service development simple. JAX-WS is a Java API for XML-based web service mainly SOAP. It exports the package javax.jws for specific Java to WSDL annotations. For core JAX-WS package is javax.xml.ws.

Table III JAX-WS annotations

Annotations	Description
@WebService	Used for implementing web service over class or over an Interface.
@SOAPBinding	Specify SOAP Messaging Style.
@WebMethod	Only apply over a method specify web service operation.
@WebResult	Specify how WSDL file looks.

These web services are tested on API testing tool Postman[27]. It is a API development tool that is used for testing, build and modification of APIs. HTTP requests(GET, POST, PUT, Patch) can be made through Postman. Integration tests can be made by Postman so that APIs work as expected and it returns response time with response size when API request is made.

V. API FEATURES COMPARISON

Fig.7 shows the flow chart comparison of REST and SOAP, identifies the differences in their development style. Here, we can make a conclusion that REST is easy to develop than SOAP as REST uses architectural style having HTTP methods which are easy to understand than SOAP WSDL files. While sending data with SOAP service, it gets attached with SOAP standard follows an envelope style that makes its payload heavier. On the contrary, REST transfers data as it is without any payload on the basis of URI Interface. Hence, REST is lightweight and requires less bandwidth compared with SOAP, which uses more bandwidth. In our research paper, response time comparison has been done during testing of REST and SOAP on Postman while creating, deleting and made query over messages shown in Fig.8.

From these results, we conclude that REST has less response time than SOAP while performing different operations with web services. With response time analysis, we can conclude that REST allows lower bandwidth usage and SOAP allows high bandwidth usage as it has envelope-style payload and consumes more resources. On the basis of performance analysis, it clearly clarifies that execution speed of REST is much faster than SOAP.

Fig.9 analyze the memory usage for REST and SOAP web services and finds that memory requirement of SOAP is higher as compared with REST as REST uses JSON and it is easy to parse than XML, but SOAP uses XML and it is hard to parse and also SOAP has heavy payload than REST. Therefore, SOAP takes more memory and CPU time.

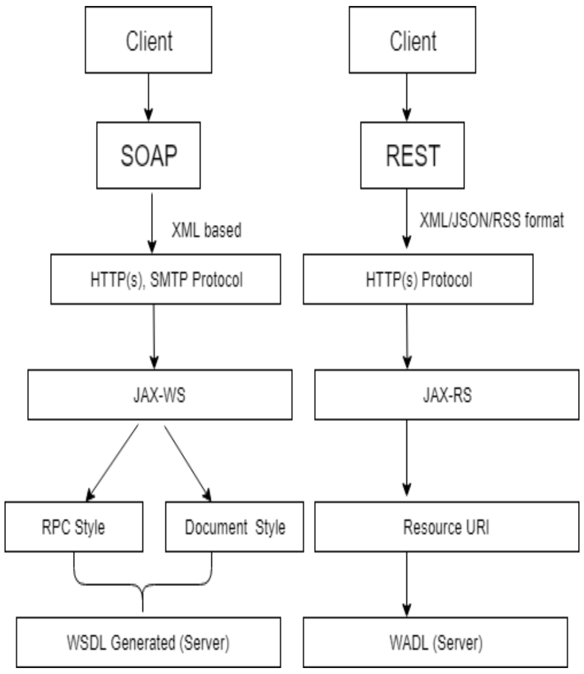


Fig.7 Flow Chart of REST and SOAP

The benefit of using SOAP is that many middleware applications can send the same SOAP signal in the same format. SOAP supports asynchronous requests, therefore, can be used in distributed computing whereas REST supports point-to-point service, hence not allow distributed computing. While considering security, SOAP protocol has WS-Security for encryption makes data safer. On the other side, of course, REST has security based on HTTP (or HTTPS) and extra security techniques are also permitted according to the project requirements. Hence, SOAP is more reliable as compared to REST. So, it is difficult to define which is more secure.

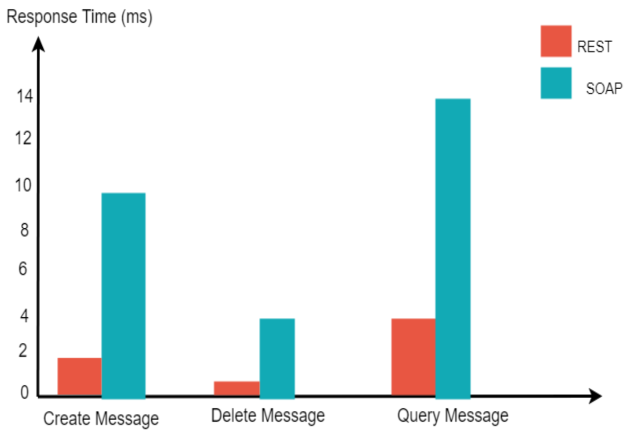


Fig.8 Response time comparison

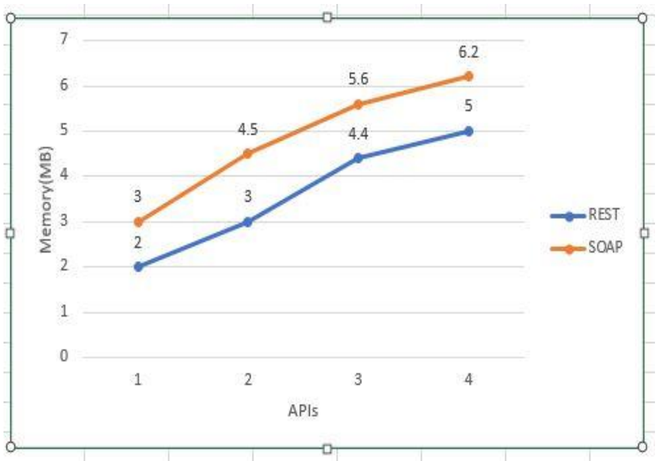


Fig.9 Memory Required Comparison

From the maintainability perspective, REST web service is effortless to maintain on the server side while SOAP is complicated to maintain on client-side. In the case of communication failure, SOAP provides with built-in retry logic compared with REST that has no built-in messaging systems and can deal with failure. REST has only retrying at the client-side. Table IV illustrates the overall comparison of REST and SOAP web services.

Table IV Comparison of Web Services

Feature	SOAP	REST
Definition	Messaging Protocol	Architectural Style

Interaction	Tightly Coupled	Loosely Coupled
Message Format	XML	JSON/XML/Plain different format
Performance	Slower	Faster
Complexity	High	Low
Bandwidth Utilization	Less	More
Payload	Heavy	Low(Lightweight)
Cost	High	Low
Execution Speed	Slower	Faster
Memory Consumption	High	Low
Reliability	Yes	No

VI. CONCLUSION

The decision between web services totally depends on the project requirements. Both web services have their own usage, strengths and weaknesses. Our research work lists out some concert important points to be considered while choosing the best web service.
From the above analysis, the following conclusions have been made while considered relevant parameters.

- Experimental results conclude that response time of REST is less than SOAP API which is approximate 4ms to 7ms.
- With the enhancement of number of API, SOAP takes approximate 1MB to 2MB more memory usage than REST as SOAP has heavy payload than REST and also takes more CPU time.
- REST is less coupled to a server which is an advantage over SOAP. With less interaction, it is easier to make updates without getting the whole knowledge of API documentation which is easier for the client.
- SOAP is more reliable because it has built in successful/retry logic when a failure occurs whereas REST has no such kind of logic.
- SOAP has WS-Security feature while REST is limited to HTTPs but we can add additional methods to enhance security of REST.
- REST is easier to understand and develop, lightweight, works with any format while SOAP is restricted to XML only.

SOAP is best suited for long-haul working projects having more focus on security and reliability such as banking, financial, telecommunication services. While REST is best suited for projects having more focus on simplicity and performance such as web chats, mobile services. Applications that focus more on back-and-forth messaging always prefer to use REST.

REFERENCES

1. C.Zhang and X.Yin,"Design and implementation of single-service multifunction webservice",2011 International Conference on Computer Science and Service System (CSSS), Nanjing,2011, pp. 3912-3915.
2. R.T.Fielding, "Architectural Styles and the Design of Network-based Software Architectures DISSERTATION", 2000.
3. P.Adamczyk, P.H. Smith, R.E.Johnson, and Munawar Hafiz,"REST and Web Services: In Theory and in Practice", DOI 10.1007/978-1-4419-8303-9_2, Springer Science+Business Media, 2011.
4. Mark Massé,"Designing Consistent RESTful Web Services Interface",Published by O'Reilly Media,ISBN: 978-1-449-31050-9.
5. C.H.Kao, C.C.Lin and J.Chen, "Performance Testing Framework for REST-basedWeb Applications", 13th International Conference on Quality Software,2013 IEEE, DOI 10.1109/QSIC.2013.32, 2013.
6. F.Haupt, F.Leymann, A.Scherer and K.Vukojevic-Haupt,"A Framework for the Structural Analysis of REST APIs", IEEE International Conference on Software Architecture,DOI 10.1109/ICSA.2017.40, 2017.
7. Munonye K* and Martinek P**, "Performance Analysis of the Microsoft .Net- and Java-Based Implementation of REST Web Services", IEEE 16th International Symposium on Intelligent Systems and Informatics, Subotica, Serbia, September 13-15, 2018.
8. R.Sinha, M.Khatkar and S.Chand Gupta, "Design & Development of a REST based Web Service Platform for Applications Integration on Cloud", IJISET - International Journal of Innovative Science, Engineering & Technology, Vol. 1 Issue 7, September 2014.
9. P.Giessler, M.Gebhart, D.Sarancin, R.Steinegger and S.Abeck "Best Practices for the Design of RESTFul Web Services," Tenth International Conference on Software Engineering Advances, Barcelona, 2015.
10. D.Qiu, B.Li and H.Leung,"Understanding the API usage in Java",Information and Software Technology, Elsevier, 2016, pp 81-100.
11. F.Halili and E.Ramadani, "Web Services: A Comparison of Soap and Rest Services", Canadian Center of Science and Education, Modern Applied Science; Vol. 12, No. 3; 2018.
12. S.Mumbaikar and Puja Padiya,"Web Services Based On SOAP and REST Principles", International Journal of Scientific and Research Publications, Volume 3, Issue 5, May 2013.
13. A.Navarro, A.d.Silva, "A metamodel-based definition of a conversion mechanism between SOAP and RESTful web services", Computer Standards & Interfaces, pp.49-70,2016.
14. M.Govindaraju, A.Slominski, K.Chiu, P.Liu, R.v.Engelen and M.J.Lewis,"Toward Characterizing the Performance of SOAP Toolkits", Fifth IEEE/ACM International Workshop on Grid Computing, Pittsburgh, PA, 2004, pp. 365-372.
15. S.Malik and S.Malik,"A Comparison of RESTful vs. SOAP Web Services in Actuator Networks", 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN), Milan, 2017, pp. 753-755.
16. A.Neumann, N.Laranjeiro and J.Bernardino, "An Analysis of Public REST Web Service APIs", IEEE Transactions on Services Computing,2018.
17. J.Tihomirovs and J.Grabis,"Comparison of SOAP and REST Based Web Services Using Software Evaluation Metrics",Information Technology and Management Science,2016,pp. 92-97.
18. J.Bian, Y.Cheng and J.Kuang, "RESEARCH ON WEBSERVICE TECHNOLOGY BASED ON OMS PLATFORM",2010 International Conference on Advanced Intelligence and Awarenss Internet (AIAI 2010), Beijing, China, 2010, pp. 162-165.
19. C.Kiama and L.Muchemi,"Comparative Study of REST and SOAP: Case of Registrar of Political Parties' Kenya", Trends in Distributed Computing, 2014, pp.105-116.
20. A.Dudhe and S.S. Sherekar, "Performance Analysis of SOAP and RESTful Mobile Web Services in Cloud Environment", International Journal of Computer Applications (0975 – 8887) Second National Conference on Recent Trends in Information Security, GHRCE, Nagpur, India, Jan-2014.

21. V.Kumari, "Web Services Protocol: SOAP vs REST",International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 4 Issue 5, May 2015, pp.2467-2469.
22. K.Wagh and R.Thool, "A Comparative Study of SOAP Vs REST Web Services Provisioning Techniques for Mobile Host", Journal of Information Engineering and Applications,Vol 2, No.5, 2012,pp 12-16.
23. G. Sambasivam , J. Amudhavel, T. Vengattaraman and P. Dhavachelvan,"An QoS based multifaceted matchmaking framework for web services discovery",Future Computing and Informatics Journal Volume 3, Issue 2, pp.371-383, December 2018.
24. R. Padmanaban, M. Thirumaran, P. Anitha and A. Moshika," Computability evaluation of RESTful API using Primitive Recursive Function", Journal of King Saud University–Computer and Information Sciences, accepted (article in press), 2018.
25. M.Z.Gashti, "INVESTIGATING SOAP AND XML TECHNOLOGIES IN WEB SERVICE", International Journal on Soft Computing (IJSC) Vol.3, No.4, November 2012.
26. jersey: https://jersey.github.io/, online Accessed on 5-oct-2018.
27. JAX-WS: https://javaee.github.io/metro-jax-ws/, online Accessed on 15-oct-2018.
28. Postman: https://www.getpostman.com/, online accessed on 20-nov-2018.

AUTHORS PROFILE

Anshu Soni completed her M.tech degree in 2019 from Computer Engineering Department of National Institute of Technology, Kurukshetra, Haryana, India. She completed her B.Tech from University of Delhi in 2017. Her one of the paper has been published in Springer Lecture Notes in Networks and Systems, (Scopus Indexed). She has also written technical content articles on geeksforgeeks.

Dr. Virender Ranga received his Ph.D. degree in 2016 from Computer Engineering Department of National Institute of Technology, Kurukshetra, Haryana, India followed by M.Tech. and B.Tech. He has published more than 50 research papers in various International SCI Journals and reputed International Conferences in the area of Computer Communications. Presently, he is Assistant Professor in the Computer Engineering Department since 2008. He has been conferred by Young Faculty Award in 2016 for his excellent contributions in the field of Computer Communications. He has been acted as member of TPC in various International conferences of repute. He is a member of editorial board various reputed journals like Journal of Applied Computer Science & Artificial Intelligence, International Journal of Advances in Computer Science and Information Technology(IJACSIT), Circulation in Computer Science (CCS), International Journal of Bio Based and Modern Engineering (IJBBME) and International Journal of Wireless Networks and Broadband Technologies. Currently, he has been selected Guest Editor for a special issue to be published in International Journal of Sensors, Wireless Communications and Control (Bentham Science Publication) He is an active reviewer of many reputed journals of IEEE, Springer, Elsevier, Talyor & Francis, Wiley and InderScience. His research area includes Wireless Sensor & Adhoc Networks, SDN, IoT security, FANET security.

.