Web Services:
SOAP, REST and OAS

Aasminpreet Singh Kainth
Department of Computer Science
Middlesex College, University of Western Ontario
London, Ontario, Canada.
akainth4@uwo.ca

***Abstract* -** Through the use of web services has led to uncertainty in selecting the correct type of web services during the design stage of the project. The most favoured implementing types are SOAP (Simple Object Access Protocol) and REST (Representative State Transfer Protocol). The maintainability problems of REST and SOAP services also emerged with the emergence of online services technologies. Choosing the correct solution is not a simple choice, since it is affected by the requirements of growth and maintenance. In the current topic I tried to use software evaluation metrics to compare SOAP and REST based Web services. To achieve this goal, a comprehensive analysis of the literature and essential terminologies to compare REST and SOAP Web services in terms of software evaluation metrics are made. This paper also describes the interface definitions for REST APIs during the research and found a way to deploy the API definition on the AWS API Gateway, straight from SwaggerHub i.e., Serverless deployment.

I. Introduction

A. Problem Formulation

Exchanging the data among information systems is an easy task with the use of web services, where web services are modular, dynamic and self-contained applications by their nature [1]. The most favoured types of implementations are SOAP (Simple Object Access Protocol) and REST (Representational State Transfer Protocol). Both of these solutions have their own benefits and demerits, so choosing the right form of web service is essential, otherwise it may lead to some issue in data sharing or it may enforce other restrictions. This paper will help you select the right type of Web service in the design process of the project. I also tried to explore interface definitions for REST APIs and a way to make use of serverless deployment.

There are a number of related studies on these protocols, primary benefits and final value [2], [3], [4]. Other articles review real implementation examples [5], [6]. This research paper summarises these studies and gives a comparison, which helps identify the key differences and benefits of SOAP and REST protocols.

B. Motivation

The aim of the paper is to review the main SOAP and REST protocol advantages and disadvantages.

To accomplish this goal, the following tasks are set:
(i) identifying the criteria on which SOAP and REST protocols should be jointly compared
(ii) comparing SOAP and REST on identified criteria
(iii) determining the type of protocol is best based on software evaluation metrics.

The rest of the paper is organized as follows. Section 2 provides the background information of the papers read. Section 3 outlines the analysis for my work. The discussion to research is presented in Section 4, while Section 5 aims to conclude. Section 6 gives references to research papers discussed.

II. Background

A. THEORETICAL BASIS

This section addresses the key terminologies that will be used in the paper.

A. Web Services

A way in which two computers can connect with each other over a network where a server operating on another computer listens to a request from a device that is considered a web site. On request the requested resource is exchanged by the user over the network.

This tool may be JSON, XML, Text, Images, Audio Files, and so on. For example, a user opens the website (client) in a stock price prediction programme and enters data (e.g., ticker symbol), the website server processes the data and delivers the answer, in this case a stock price. For lay man the term web service and website may have the same meaning but in real there's an important difference between the two I.e. Web services do not have the GUI, but the websites do. They are also completely different entities as website is a set of web pages where a web page can contain multiple web services as appropriate.

The sending and receiving of the messages can be done by using XML messages over HTTP protocol. Webservices can run on any browser and operating system i.e. they are browser and operating system independent service. They take web applications to a whole new level. [1] Let's say, you are a .net developer and you can publish your resource on web or LAN through .net web service so any other developer (let's say java developer) can access your resource.

B. SOAP

SOAP is a lightweight protocol designed for the exchange of information in a decentralised and distributed environment. It is reliant on XML which allows messages to be exchanged between varied platforms independent of the programming language or platform. SOAP messages can be exchanged over different underlying protocols such as HTTP, SMTP and FTP. The faults are gracefully handled as SOAP has an error structure as well. [13]

SOAP spans four primary sectors: [10]

1. A one-way communication message format that explains how to bundle a message into an XML document

2. A summary of how a SOAP message should be transmitted through the Web (using HTTP) or e-mail (using SMTP) (the XML document that makes up).

3. A collection of rules to follow when processing a SOAP message and a basic list of the individuals involved in processing the message. It also determines who will read parts of the messages and how to respond if the content is not understood.

4. A collection of conventions on how to transform and return an RPC call into a SOAP message and how to enforce the RPC interaction style.

C. REST

REST is an architecture used to build Web services. It uses the same architecture that uses HTTP or related protocols, and also uses the same operations i.e. GET, PUT, POST, & HTTP DELETE. The emphasis here is on communicating with state-of-the-art resources, instead of messages or operations.

The key concept behind REST was to use well-developed HTTP to transfer data between computers, instead of using a protocol that works for transmitting messages on top of the HTTP layer.

REST treats everything as a resource and, in Dr. Roy Fielding's words, a resource is the same as a file that is stored on a computer or as an entity in a database.

Statelessness is another REST concept; statelessness means that all HTTP requests are independent. [7]

D. Software Evaluation Metrices

There are two types of evaluation metrics used in the production of software [2]: direct, measurable, and indirect or non-measurable metrics. Table below summarises the most common metrics

Direct	Indirect
Cost	Functionality
Effort	Quality
Lines of code	Complexity
Execution speed	Efficiency
Memory	Reliability
Errors	Maintainability

E. OpenAPI

With the help of RESTful web services, Web APIs have gained a lot more popularity with enhancements in new features. Swagger, which is now known as Open API Specification was made available in 2010 and it has played a huge role in the API documentation field. After forming the In 2015 Swagger concentrated on standardising its offerings and improving its presence in industry under the Linux Foundation Open API Initiative.

OpenAPI Specification (formerly known as the Swagger Specification) is a format for REST API specifications. This can be used for creating new APIs and recording current APIs. OpenAPI allows you to define the entire API, including the endpoints, processes, request and response formats available, authentication methods supported, contact support and other information available.

OpenAPI definitions can be written in YAML or JSON. SwaggerHub Editor uses YAML, but you can import and download both YAML and JSON.

Swagger is, in short, a collection of open source tools developed around the OpenAPI standard. That can be used for creating, documenting and consuming REST APIs.

Swagger Tools consists of:

Swagger Editor

Swagger UI

Swagger Codegen

SwaggerHub combines these tools for a seamless experience, and adds hosted storage, access control and collaboration features on top of it.

F. Serverless Architecture

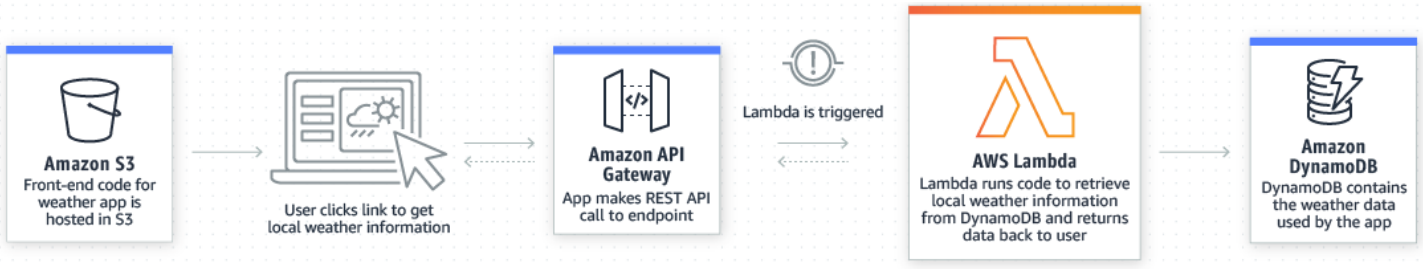
A serverless architecture is a means of building and running software and utilities without the need for network management. Your code is already operating on servers so AWS does all the system maintenance.

To run your applications, databases, and storage systems, you no longer need to have, scale, and manage servers

Use of Serverless Architecture

The developers will concentrate on their main project by using a serverless architecture, rather than thinking about maintaining and running servers or runtime, either in the cloud or on-site. This reduced overhead helps developers to recover time and resources that can be expended on creating better, scale-up and stable products.

Example: Weather application (*Fig. below)*



B. LITERATURE REVIEW

Cesare Pautasso et al.'s study included a theoretical distinction between SOAP and REST in terms of architectural principles, conceptual as well as various practical decisions[4]. Michael Zur Muehlen et al. worked on a contrast that portrays the compositional requirements of the two interaction styles[5] taking basic procedure and specification into consideration. But, it did not mention any hands-on details about the discrepancies in application creation between the two interfaces. IMS is the service delivery framework of the modern age telecommunications networks that used the SOAP to create a web interface where a session user can serve as an endpoint and receive information parse and send SOAP messages. David Lozano et al. developed multimedia conferencing applications using the Web service RESTful[7]. In the IP Multimedia Subsystem[6], Gmez and Miguel discussed the general problem of SOAP-based Web service multimedia conferencing.

The authors in [8] have shown methods for improving the efficiency of SOAP web services where the crux of their research work is to find the common parts of the SOAP message so that these common parts are only processed once that helps to minimise the overhead. SOAP processing involves sorting of messages, serialisation, deserialization, compression, multicasting, and security assessment. To improve the SOAP efficiency, each step is discussed in detail.

In the year 2012 P. K. Potti et al. have been working on a comparative performance analysis between the SOAP web services and the REST web services in terms of response time and throughput. They have developed web services to conduct simple CRUD operations on a database and locations to be retrieved. The efficiency of these Web services was compared with metrics such as response time and throughput[11].

Snehal and P, in [9]. Padiya[9] contrasted the efficiency of SOAP and RESTful web services and a multimedia conference based on specific metrics for the mobile world. SOAP and REST for CRUD operation are planned in their analysis, and they compare results. However, the most outcome was insufficient because they were working on a limited collection of data for their experiment. To explore the variations between REST and SOAP interaction types, payloads such as customer data and photos used do not seem appropriate for CRUD operations.

III. Analysis

For comparison between the REST and SOAP, some metrics are used to illustrate main standards to be taken into account when selecting the perfect approach for a problem [2]. These metrics are divided into two categories direct and indirect metrics.

A. Cost

REST compared to SOAP services is cost-effective, because they can be produced with limited use of the production tool. [4] In addition, REST does not involve the client at the beginning of the implementation or testing stages, because REST resources can be accessed directly from the web browser by searching for a specific resource address. This also effectively reduces the cost of achieving the same result with the SOAP method.

B. Effort

While developing and publishing the webservice on internet, it is suggested to use REST approach in order to reduce the maintenance costs. Whereas, SOAP WSDL approach is suggested when one tries to use any other webservice for the reduction in maintenance costs.[2]

C. Lines of Code

The study by Fortes and V. Brusamolin, compared SOAP and REST in terms of lines of code in order to achieve the same functionality. [2] 4016 lines of code were written in order to develop a REST service whereas to achieve the same functionality 3844 lines of code were written in SOAP service.

D. Execution Speed

To calculate the execution speed of a web service, the response time of the resource returned by the web service to the client determines the execution speed. The University of North Florida[7][11] made this response time comparison: The experiment revealed that REST has a shorter response time than SOAP. It is noted that in cases of slow data transfer

speed or a huge load in the network, SOAP is not the proper solution. In [12], simulation was used to build a basic client-server scenario for a response time measurement experiment.

It is observed that the processing time of the SOAP-XML service is 4–5 times higher than the REST approach as shown in (Fig. 1).

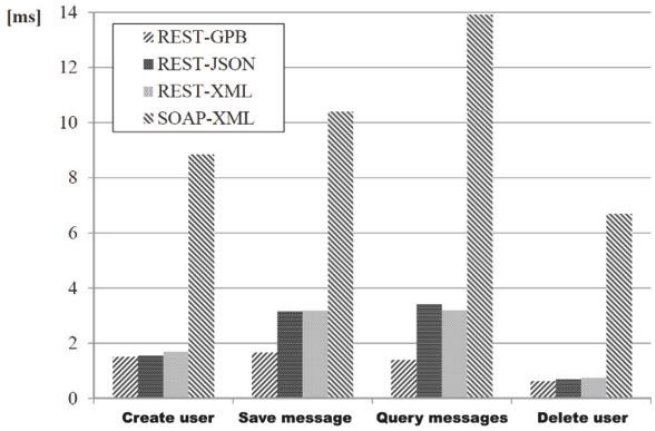


Figure 1 Measuring REST and SOAP response time [12]

E. Memory

In [5][8], authors shown how a creation of SOAP request message from phones can take 9 times more memory resources. Mobile devices are resource limited. Additionally, it is said that REST can perform Additionally it is claimed that when bandwidth and resources are reduced, REST works better. [9]

F. Errors

SOAP has built-in error handling mechanism which is advantage over REST [9] also it is possible to use WSDL file for the automatic generation of client-side code. This helps in reducing the number of errors that could be done mistakenly if a programmer code manually.

G. Functionality

Nowadays, the commonly adopted solutions to system integration are point-to-point and distributed computing, where SOAP is ideal for use in distributed computer systems as the message can be transmitted by one or more intermediaries [9],[14].

H. Quality

Since many of the qualities such as service time, availability, security, etc., [10] have been defined in this study as a comparison metrics, this point is not viewed in detail.

I. Complexity

REST protocol uses the architectural style of the Network that is closely related to the HTTP. REST solution is easier as it follows commonly recognised W3C / IETF (HTTP, XML, Url, and MIME) formats. The creation of Web Services using REST is therefore more "user-friendly"[12].

J. Efficiency

REST web services can used by a large number of users at the similar time. REST requests can be served via the JSON, or even plain text (which are light weight data format) to send messages that in total reduces the server load.

K. Reliability

WS-Security is used as the standards for signing and encrypting messages in SOAP protocol making the data transfer process a lot safer.[3] Whereas the REST approach doesn’t have any own security model.

L. Maintainability

RESTful webservices on the server side are easier to maintain while the SOAP webservices on the client side are easier to maintain. This is because the SOAP Web services use WSDL file to define a service GUI. Based on the available WSDL file, a client-side code can be automatically created in today's integrated development tools, such as Eclipse.

M. Comparison Summary

For summarising the results which are presented in Table 1, the best protocol which can be used to integrate internal information systems and applications with other systems can not be clearly defined.

Metric	SOAP	REST
Cost (lower)		✓
Effort (lower)		
Server side		✓
Client side	✓	
Lines of code (fewer)	✓	
Execution speed (faster)		✓
Memory (low consumption)		✓
Errors (less)	✓	
Functionality		
Protocol independent	✓	
Complexity (lower)		✓
Efficiency		
Performance (better)		✓
Reliability	✓	
Maintainability		
Server side		✓
Client side	✓	

Table 1 COMPARISON SUMMARY

This implies that the functional and non-functional specifications have to be assessed modularly for each project. Taking those criteria into account, it is possible to determine which of the approaches to use.

IV. Discussion

Upon reviewing the findings of the study, it was acknowledged that it is not feasible to specifically choose the right approach for a project, because it is recommended that each method will be analysed independently. As SOAP and others have pros and cons of their own.

A. REST

Choosing REST is best if the project needs greater scalability, compatibility and efficiency. Complexity of REST implementation, speed of execution, consumed memory resources and performance compared with SOAP protocol was higher. So if the project needs easy point-to-point integration or large-scale mobile app availability, then REST is the right option. Based on these factors, the major Web service providers are using precisely REST : Twitter, Yahoo, etc. [9].

B. SOAP

SOAP is a safer option as the project needs protection and reliability, faster client-side support, as well as less potential mistakes. Furthermore, many business system integration projects require intermittent requests for data processing which is the SOAP benefit. Consequently, the conclusion is that SOAP is better suited to large integration of information systems, such as integration projects in banking information systems.

SOAP is a better choice when security and reliability are needed for the project; simpler client-side maintenance, as well as less potential errors. Furthermore, several enterprise system development ventures involve intermittent demands for data processing which is the SOAP benefit. Consequently, the inference is that SOAP is best adapted to broad integration of information systems, such as transformation ventures of banking information systems.

Based on the project's priorities, the option of SOAP and REST solution is outlined in Table V. It should be remembered that this is not a rigid guideline as to whether projects can take the correct approach to data sharing.

Table 2 SELECTION OF SOAP AND REST PROTOCOLS

	Main criteria	Project example
REST	Greater scalability; Compatibility; Performance; Simplicity; Point-to-point communication model; Limited bandwidth.	Mobile app integration with the information systems; Simple client-server data exchange solutions.
SOAP	Higher security and reliability; Lower number of errors; Asynchronous requests; Distributed computing; Support from other standards (WSDL, WS).	Business information systems (B2B); Banking information systems; Payment systems.

V. Conclusion

The present paper focuses mainly on the forms of services SOAP and REST. In the future, study will rely on other forms of Web service as comprehensive information about other software assessment metrics could not be provided, and that is left for further analysis.

This paper provides a comparative study of the types SOAP and REST. It is important to select a suitable technology to build web services for the deployment of web services, because it influences the actual design and functionality of the Enterprise software.

VI. References

[1] S. Kumari and S. K. Rath, "Performance comparison of SOAP and REST based Web Services for Enterprise Application Integration," in International Conference on Advances in ICACCI, Kochi, 2015, pp. 1656–1660.

[2] Fortes and V. Brusamolin, "Comparative Evaluation of the Maintainability of RESTful and SOAP-WSDL Web Services," in 2013 IEEE 7th International Symposium on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems, Eindhoven, 2013, pp. 40–49.

[3] N. Serrano, J. Hernantes and G. Gallardo, "Service-Oriented Architecture and Legacy Systems," IEEE Software, 2014.

[4] C. Pautasso, O. Zimmermann and F. Leymann, "RESTful Web Services vs. "Big" Web Services: Making the Right Architectural Decision," in Proceedings of the 17th international conference on World Wide Web, Beijing, China, 2008, pp. 805–814.

[5] F. Belqasmi, J. Singh, S. Y. B. Melhem and R. H. Glitho, "SOAP-Based Web Services vs. RESTful Web Services for Multimedia Conferencing Applications: A Case Study," IEEE Internet Computing, vol. 16, issue 4, pp. 54–63, July–Aug. 2012.

[6] F. AlShahwan and K. Moessner, "Providing SOAP Web Services and RESTful Web Services from Mobile Hosts," in 2010 5th International Conference on Internet and Web Applications and Services (ICIW), Barcelona, 2010, pp. 174–179.

[7] Pavan Kumar Potti, "On the Design of Web Services: SOAP vs. REST" in University of North Florida, 2011.

[8] Juris Tihomirovs, Jānis Grabis, "Comparison of SOAP and REST Based Web Services Using Software Evaluation Metrics", in Riga Technical University, 2016.

[9] Swagger specifications tools, SmartBear Software, 2019. https://swagger.io/docs/specification/about/

[10] 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.

[11] P. K. Potti, S. Ahuja, K. Umapathy, Z. Prodanoff, "Comparing Performance of Web Service Interaction Styles: SOAP vs. REST," in *Proceedings of the Conference on Information Systems Applied Research*, New Orleans Louisiana.

[12] G. Serme, A. S. de Oliveira, J. Massiera and Y. Roudier, "Enabling Message Security for RESTful Services," in *2012 IEEE 19th International Conference on Web Services*, Honolulu, HI, 2012.

[13] T. Aihkisalo and T. Paaso, "Latencies of Service Invocation and Processing of the REST and SOAP Web Service Interfaces," in *2012 IEEE 8th World Congress on Services*, Honolulu, HI, 2012.

[14] J. Kangasharju, S. Tarkoma and K. Raatikainen, "Comparing SOAP performance for various encodings, protocols, and connections," in *Personal Wireless Communications* (Lecture Notes in Computer Science (LNCS)

[15] “What is Amazon API Gateway?” © 2020, Amazon Web Services, Inc. https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html