IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 10, October 2017

A Survey on Developing Web Services with SOAP and REST

Madhuri Surendrakumar Warvante[1]

ME Student, Department of Computer Engineering, Pune Institute of Computer Technology, Pune, Maharashtra, India[1]

Abstract: A web service is a software system designed to support interoperable machine-to-machine interaction over a network. Web service is a service offered by one electronic device to another electronic device by communicating with each other over World Wide Web. Web services can be classify in two classes REST-compliant web services and arbitrary web services. SOAP (Simple Object Access Protocol) is intermediate language so that the applications written in different language can talk with each other. That means SOAP is language independent. SOAP is also platform independent. SOAP is based on XML and XML is light-weight so SOAP is also light-weight. REST stands for Representational State Transfer. It relies on a stateless, client-server, cacheable communications protocol and in virtually all cases, the HTTP protocol is used. REST is an architecture style for designing networked applications. The idea is that, rather than using complex mechanisms such as CORBA, RPC or SOAP to connect between machines, simple HTTP is used to make calls between machines. If we used SOAP or CORBA then there is requirement for client side to use the same because there is tight coupling. In many ways, the World Wide Web itself, based on HTTP, can be viewed as a REST-based architecture. RESTful applications use HTTP requests to post data (create and/or update), read data (e.g., make queries), and delete data. Thus, REST uses HTTP for all four CRUD (Create/Read/Update/Delete) operations. In REST two software can be integrated. In REST each data element is a resource, addressed by a URI (Uniform Resource Identifier).

Keywords: web service, SOAP, WSDL, UDDI, RPC, REST.

I. INTRODUCTION

A web service is a software system designed to support interoperable machine-to-machine interaction over a network [1]. Web service is a service offered by one electronic device to another electronic device by communicating with each other over World Wide Web. Web services are self-contained, modular, distributed, dynamic applications that can be described, published, located, or invoked over the network to create products, processes, and supply chains. These applications can be local, distributed, or Web-based Web services that are built on top of open standards such as TCP/IP, HTTP, Java, and XML [2]. Web service is a collection of standards or protocols for exchanging information between two devices or application. In a web service web technologies such as HTTP can be used to transfer data. Web services uses XML data formats for messaging. Web services can be classify in two classes

- REST-compliant web services
- Arbitrary web services[3]

There are three major components of web service

- Simple Object Access Protocol (SOAP)
- Web Services Description Language (WSDL)
- Universal Description, Discovery and Integration (UDDI)

A. SOAP

SOAP stands for Simple Object Access Protocol. SOAP is based on XML. SOAP is intermediate language so that the applications written in different language can talk with each other. That means SOAP is language independent. SOAP is also platform independent. SOAP messages can be carried by a variety of network protocols; such as HTTP, SMTP, FTP, RMI/IIOP, or a proprietary messaging protocol. But the simplest way is to use HTTP. SOAP is a light-weight protocol that is used for data interchange between applications. SOAP is based on XML and XML is light-weight so SOAP is also light-weight.

B. WSDL

WSDL stands for Web Services Description Language. WSDL is written in XML. WSDL is a xml document which contains information about web services. WSDL describes Web services starting with the messages that are exchanged between the requester and provider agents. The messages themselves are described abstractly and then bound to a

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 10, October 2017

concrete network protocol and message format [1]. The three major elements of WSDL that can be defined separately are:

- Types
- Operations
- Binding

C. UDDI

UDDI stands for Universal Description, Discovery and Integration. UDDI is a XML based framework for describing, discovering and integrating web services. UDDI is a directory of web service interfaces described by WSDL, containing information about web services.

II. SOAP WEB SERVICES

A. SOAP Architecture

The SOAP specification describes a standard, XML-based way to encode requests and responses, including:

- Requests to invoke a method on a service, including in parameters
- Responses from a service method, including return value and out parameters
- Errors from a service

SOAP architecture is shown in figure 1.

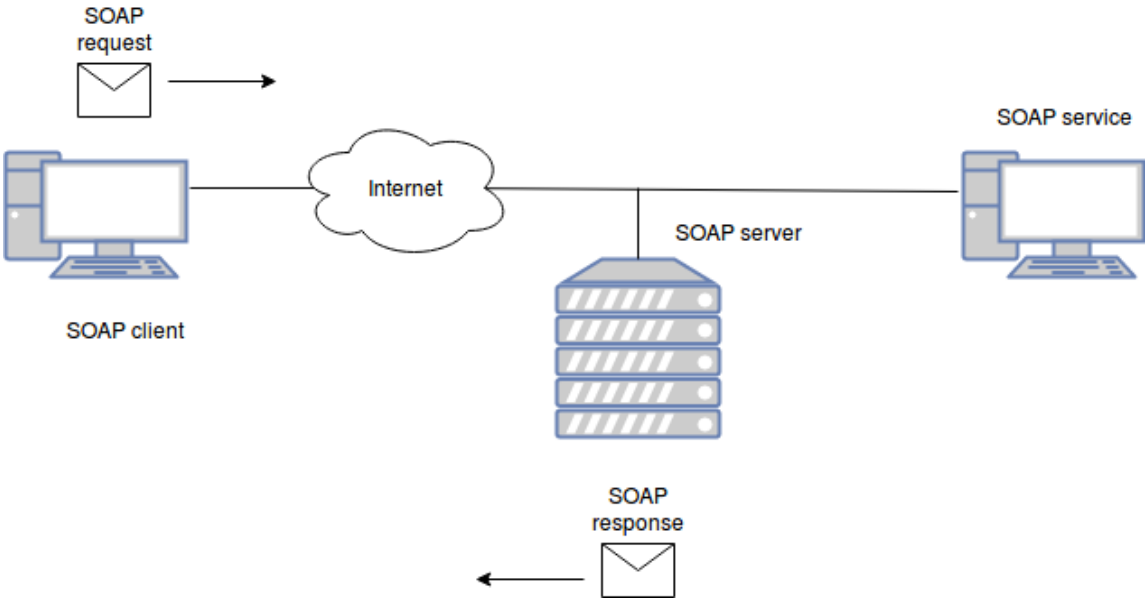

Fig. 1 SOAP Architecture

B. SOAP Building Blocks

The diagram below (Fig. 2) shows the various building blocks of a SOAP Message.

- An Envelope, which defines what to be included in the message and how this massage should be processed, a set of encoding rules for expressing instances of application defined data types, and a convention for representing procedure calls and responses [6].
- A Header element that contains header information.
- A Body element that contains call and response information.
- A Fault element containing errors and status information.

SOAP is a W3C recommendation, which means that it is a technical report that is the end result of an extensive consensus building inside and outside of the W3C about a particular technology or policy.

There are two types of SOAP requests. The first is the Remote Procedure Call (RPC) style request similar to other distributed architectures. This is usually synchronous; the client sends a message and waits to get a response or fault message back from the server. The second type of SOAP request is the document request. In this case, a full XML document is passed to/from the client and server, inside a SOAP message [5]. Moreover; SOAP has three major characteristics: Extensibility (security and WS-routing are among the extensions under development), Neutrality

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 10, October 2017

(SOAP can be used over any transport protocol such as HTTP, SMTP or even TCP), and Independence (SOAP allows for any programming model) [3].

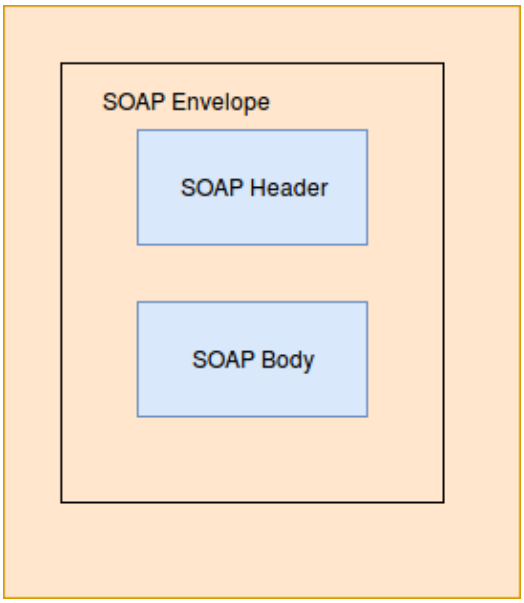

Fig. 2 SOAP Building Blocks

C. Remote Procedure Call (RPC)

RPC is client-server model where client needs services provided by server and server provides services to client. RPC is extension of local procedure call where calling and called processes may be on same systems or nay be on different systems with a network connecting them [7].

When making a remote procedure call:

- The calling environment is suspended, procedure parameters are transferred across the network to the environment where the procedure is to execute, and the procedure is executed there.
- When the procedure finishes and produces its results, its results are transferred back to the calling environment, where execution resumes as if returning from a regular procedure call.

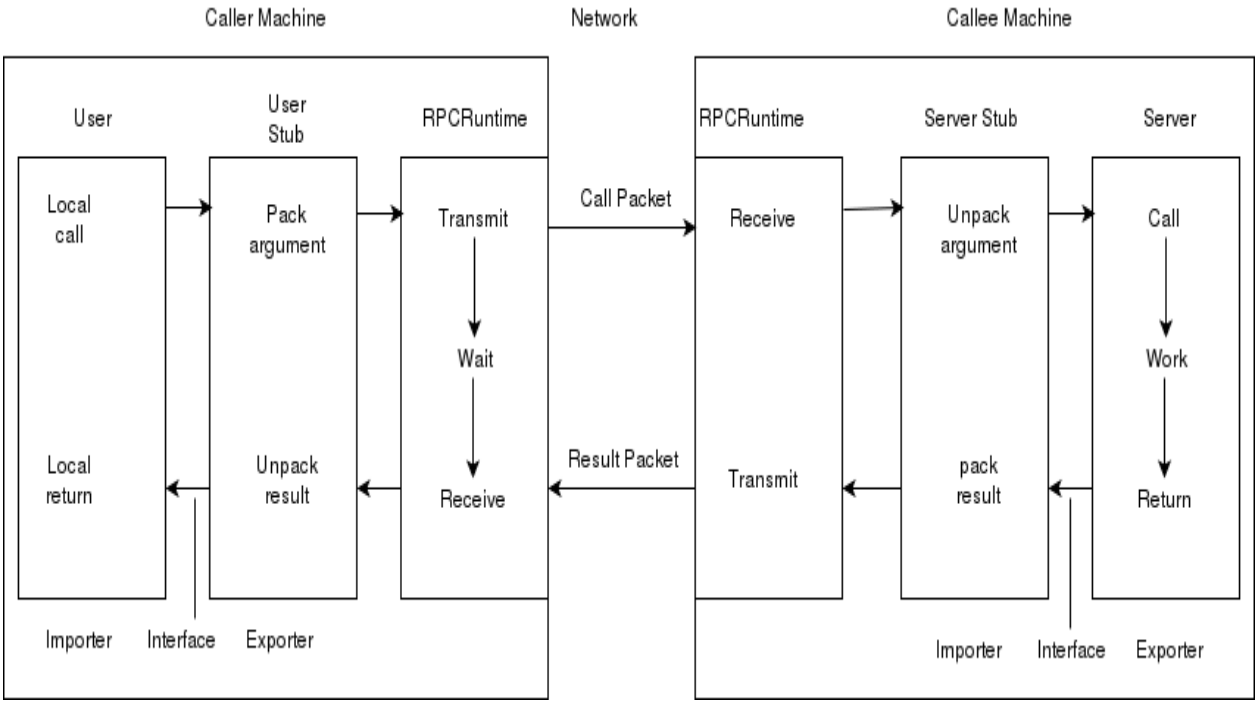

Fig. 3 The components of the system, and their interactions for RPC (image Source : [10]).

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 10, October 2017

D. Advantages of SOAP

- SOAP is platform independent.
- SOAP is portable.
- Firewall-friendliness: SOAP is capable of getting past firewalls which are totally blocking for other protocols. This is possible due to use of the HTTP protocol [6].
- Use of open standard: SOAP is based on the open Standard XML. As a consequence, SOAP becomes easily extendable and well supported [6].
- Interoperability: SOAP relies on open instead of vendor-specific technologies and thus enables distributed interoperability and loosely coupled applications [6].
- Resilience to changes: It is unlikely that future modifications of SOA infrastructure will have any impact on applications using the method, as long as no significant serialization changes are made to SOAP specification [6].

E. Disadvantages of SOAP

- Operation interface: Useful information such as operation details and data are encapsulated within the services, just exposing only one endpoint of API and all operations use the POST method [10][11].
- Complexity: It is time-consuming to serialize and deserialize native languages into SOAP messages. Furthermore, the WSDL protocol stack is also complex so that only programmers can understand how to deploy a service [6][10].
- Interoperability: Since a specific service interface is defined for each service, a client must be bound to a specific WSDL. Once the WSDL has changed, the client has to follow these changes [10].
- Performance: Much information in the SOAP and WSDL is redundant and meaningless. It increases the network communication volume and server side payload and it is difficult to support the proxy and cache servers because clients cannot identify the useful information straight forwardly from the URI and HTTP [10][11].

F. Document

With a document style message the client and/or server passes an XML document as the body of the SOAP message instead of parameters [5]. Document style messaging has other advantages over a remote procedure call, which is meant to be relatively static, and any changes to the RPC interface would break the contract between the service and the application. Changing the RPC description would cause all of the applications that rely on a specific method signature of SOAP-RPC to break [5].

III. REPRESENTATIONAL STATE TRANSFER

REST stands for Representational State Transfer [12]. It relies on a stateless, client-server, cacheable communications protocol and in virtually all cases, the HTTP protocol is used [12]. REST is an architecture style for designing networked applications. The idea is that, rather than using complex mechanisms such as CORBA, RPC or SOAP to connect between machines, simple HTTP is used to make calls between machines. If we used SOAP or CORBA then there is requirement for client side to use the same because there is tight coupling.

In many ways, the World Wide Web itself, based on HTTP, can be viewed as a REST-based architecture [13]. RESTful applications use HTTP requests to post data (create and/or update), read data (e.g., make queries), and delete data [14]. Thus, REST uses HTTP for all four CRUD (Create/Read/Update/Delete) operations. In REST two software can be integrated. In REST each data element is a resource, addressed by a URI(Uniform Resource Identifier).

REST was defined by Roy Thomas Fielding in his 2000 PhD dissertation “Architectural Styles and the Design of Network-based Software Architectures [15]". Management of the API occurs throughout the entire lifecycle of the API, from planning and design, to development, operations and maintenance of it [13].

A. The Goals and Design Principle of REST

- Operation interface: Useful information such as operation details and data are encapsulated within the services, just exposing only one endpoint of API and all operations use the POST method [18].
- Scalability of component interactions
- Generality of interfaces
- Independent deployment of components
- Intermediary components of reducing interaction latency, enforcing security, and encapsulating legacy system

The components in the REST system must comply with the following constraint [18]:

- Identification of resources
- Manipulation of resources through representations
- Self-descriptive messages
- Hypermedia as the engine of application state

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 10, October 2017

B. RESTful API

Representational State Transfer is an architectural style used for web development. we can achieve fast performance, reliability and the ability to scale by using this style. Developers work with reusable components that can be managed and updated without affecting the system as a whole while it is running [16].

C. REST API Lifecycle

RESTful APIs that enable you to integrate their tools into your own development lifecycle. We can use RESTful APIs to write scripts or code that programmatically deploy REST API web services, or that migrate REST API services from one environment to another, as part of a larger automated process that also deploys or migrates other applications [17].
As shown in Fig. 3 RESTful API Lifecycle will have different states.

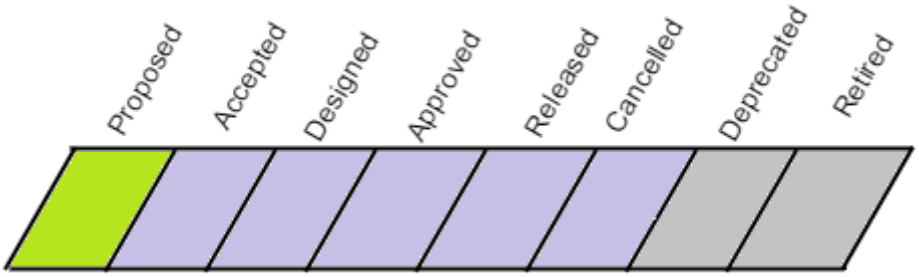

FIG.3 RESTFUL API LIFECYCLE (IMAGE SOURCE: [17])

- Proposed: This state shows new API or version of new API being proposed.A new proposed API is presented to the API Governance Board, and they will evaluate it from a product and technical perspective. The board will decide whether to reject or approve the proposed API.
- Accepted: This state shows API is reviewed by the API Governance Board and approved to move further.
- Designed: The API goes to design and its specifications are saved into a specifications document. After design is completed, the API will then be presented to the architecture review for final approval.
- Approved: Once it passes the architecture design review, the API service is approved, and authorized for production release.
- Released: The API Service is live in Production.
- Cancelled: API are cancelled either by the API Governance Board or by the author(s) depending on the life cycle stage when the decision is made.
- Deprecated: When new version of API has been deployed or when the product is no longer part of company portfolio then An API is deprecated.
- Retired: A retired API is no longer deployed in the production environment and is no longer in use.

D. The RESTful Web Service Design Principles [19]

- Prerequisite gathering-similar to traditional software requirement gathering practices.
- Resource identification-similar to OOD.
- Resource representation definition.
- URI definition-define the API, which consists of URIs for clients and servers to exchange resources' representations.

E. RESTful Style Webservices

REST is a distributed software architecture that takes full use of Web features. It can reduce the complexity of system development, provide scalability for the system. REST Web services are the resource-oriented services, you can use normal simple operation to generate an action for the clear resources.

The followings are several key principles for RESTful Web services [19]:

- Defining the ID for all resources;
- Linking all the resources together;
- Using standard methods;
- Resources to multiple statements;
- Stateless communication;

If the developers can stick with the five principles in the design of the system applications, they will get a Web service architecture using high-quality system.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 10, October 2017

The first key principle of REST illustrates that, the ID of the resources in RESTful Web services (whether a single resource, or a collection of resources; whether the virtual resources, or physical resources) can use a unified, unique concept URI to identity.

The second key principle of REST illustrates that, all the resources of the RESTful Web services, are available through hyperlinks in the form of URI dynamically reference in any case, in order to achieve the interconnection of network resources.

The third key principle of REST illustrates that, all of the resources in RESTful Web services should be properly achieved through HTTP, using a common, uniform and nonspecific HTTP interface. HTTP not only unique positioning of resources, but also indicates how to operate the resource. Using the standard HTTP GET, PUT, POST, and DELETE operations and other methods of resources can make the client program and the Web services to the resources of mutual collaboration.

The fourth key principle of REST illustrates that, RESTful Web services can use HTTP to allow the data processing and call the relationship between the separation of operating characteristics, by providing resources to the multiple statements made between the client and server. The HTTP data can be applied to resources interaction.

The final key principle illustrates that, in order to improve system scalability, RESTful Web services should be put into the resources required state, or saved on the client. This makes the change of Web server is not visible for the client.

F. Advantages of REST API

- Addressability: Resources are marked with global URIs. They are accessible once they are exposed on the Web rather than require a separate resource discovery and location mechanism [6].
- Links and Connectedness: Resources are linked with each other by using hyperlinks which point to valid future states in representations. What most important is that, state transfer can be implemented by following the links [11].
- Statelessness: Each request includes all the necessary information for the servers to understand, so each transaction is independent and unrelated to previous ones. Servers do not need to keep states between requests [20].
- Uniform Interface: Resources are manipulated using a fixed set of HTTP methods (GET, PUT, DELETE and POST) without a special code to deal with each one. These methods (except POST) are Idempotent [11].

G. Disadvantages of REST API

- Encoding a large amount of input data in the resource URI is impossible because the server either refuses such requests or crashes [6].
- It may also be challenging to encode complex data structures into URI as there is no commonly accepted marshalling mechanism. Inherently, the POST method does not suffer from such limitations [6][11].

IV. DIFFERENCE BETWEEN REST AND SOAP

There are currently two schools of thought in developing Web Services – one being the standards-based traditional approach [SOAP] and the other, simpler school of thought [REST]. Because of some disadvantages of SOAP over REST it is convenient to use REST over SOAP. The detailed difference is listed in following table. Some of major disadvantages of SOAP over REST are SOAP is slow, in SOAP there is tight coupling. tight coupling means if server is using some specific protocols then client should use the same protocol.

TABLE I DIFFERENCE BETWEEN REST AND SOAP

No.	REST	SOAP
1	Assumes a point-to-point communication model–not usable for distribute computing environment where message may go through one or more intermediaries.	Designed to handle distributed computing environments.
2	Minimal tooling/middleware is necessary. Only HTTP support is required.	Requires significant tooling/middleware support.
3	URL typically references the resource being accessed/deleted/updated	The content of the message typically decides the operation e.g. doc-literal services
4	Not reliable – HTTP DELETE can return OK status even if a resource is not deleted.	Reliable.
5	Formal description standards not in widespread use. WSDL 1.2, WADL are candidates.	Well defined mechanism for describing the interface e.g. WSDL+XSD, WS-Policy.
6	Better suited for point-to-point or where the intermediary does not play a significant role.	Well suited for intermediated services.
7	No constraints on the payload.	Payload must comply with the SOAP schema.

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 10, October 2017

8	Only the most well established standards apply e.g. HTTP, SSL. No established standards for other aspects. DELETE and PUT methods often disabled by firewalls, leads to security complexity.	A large number of supporting standards for security, reliability, transactions.
9	Built-in error handling (faults)	No error handling
10	Tied to the HTTP transport model.	Both SMTP and HTTP are valid application layer protocols used as Transport for SOAP.
11	Less verbose.	More verbose.

V. CONCLUSION

This paper focused on developing web services with SOAP and REST. Introduction and architecture of SOAP, building blocks of SOAP, advantages and disadvantages of SOAP is explained. Two types of SOAP requests Remote Procedure Call (RPC) and Documents are also explained. Section 3 introduces the goals and design principles of REST, RESTful API, RESTful API lifecycle, RESTful Web service design principles and RESTful style Web service. Section 4 contain difference between REST and SOAP.

REFERENCES

[1] David Booth, W3C Fellow / Hewlett-Packard Hugo Haas, "Web Services Glossary". W3C. 2004-02-11. Retrieved 2011-04-22.
[2] IBM, "WebSphere Business Integration Adapters", Accessed May 2011 form http://publib.boulder.ibm.com/infocenter/wbihelp/v6rxmx/index.jsp?topic=/com.ibm.wbia_adapters.doc/doc/webservices/webservices17.htm
[3] David Booth, W3C Fellow / Hewlett-Packard, Hugo Haas, W3C, Francis McCabe, Fujitsu Labs of America, Eric Newcomer (until October 2003), Iona, Michael Champion (until March 2003), Software AG, Chris Ferris (until March 2003), IBM, David Orchard (until March 2003), BEA Systems, "Relationship to the World Wide Web and REST Architectures", Web Services Architecture. W3C. Retrieved 2011-04-22.
[4] Oracle9i Application Server Oracle9iAS SOAP Developer's Guide, https://docs.oracle.com/cd/A97335_02/integrate.102/a90297/overview.htm, Release 1 (v1.0.2.2), Part Number A90297-01.
[5] Brian Suda, "SOAP Web Services", School of Informatics, University of Edinburgh, 2003.
[6] Albreshne, A., Fuhrer and Pasquier, "Web Services Technologies: State of the Art", 2009.
[7] Dave.Marshall, "Remote Procedure calls(RPC)", https://users.cs.cf.ac.uk/Dave.Marshall/C/node33.html, 1 May 1999.
[8] Yash Singla, "Remote Procedure call (RPC)", http://www.geeksforgeeks.org/operating-system-remote-procedure-call-rpc/.
[9] ANDREW D. BIRRELL and BRUCE JAY NELSON "Implementing Remote Procedure Calls", Xerox Palo Alto Research Center, ACM Transactions on Computer Systems, Vol. 2, No. 1, February 1984, Pages 39-59.
[10] Meng, J. , Mei,S. and Yan , Z., "RESTful Web Services: A Solution for Distributed Data Integration", Computational Intelligence and Software Engineering, 2009. CiSE 2009.
[11] Cox, J., Harvey, D. and Ramsbrock, D., "SOAP vs. REST For Mobile Services", http://blogs.captechconsulting.com/blog/jack-cox/soap-vs-rest-mobile-services, September 22, 2015.
[12] Michael Jak l, "Representational State Transfer", University of Technology Vienna, 2006.
[13] Roberto Medrano, "The Reality of API Lifecycle Management", June 9, 2013.
[14] N.L.Drotz, F.G.P.Zetterberg, "Wi-Fi Fingerprint Indoor Positioning System using Probability Distribution Comparison", IEEE International conference on Acoustics, Speech and Signal Processing, Kyoto, Japan, March 2012.
[15] Roy T. Fielding and Richard N. Taylor, "Principled design of the modern web architecture", ACM Trans. Inter. Tech., 2(2):115–150, 2002.
[16] Megan Lunde, Richard May, Daivid B. Peterson, "API Governance", https://projects. tmforum.org/wiki/display/TP/API+Governance, 08 Jul 2016.
[17] Guy Levin, "REST API, API Driven Development, Development Lifecycle", http://blog.restcase.com/development-lifecycle-of-an-api-service, 20 December 2015.
[18] R. T. Fielding, "Architectural Styles and the Design of Network based Software Architectures," California, Irvine: University of California, 2000.
[19] Hongjun Li, "RESTful Web Service Frameworks in Java", IEEE International Conference, Signal Processing, Communications and Computing (ICSPCC), 2011.
[20] Hamad, H., Saad, M. and Abed, R., "Performance Evaluation of RESTful Web Services for Mobile Devices", International Arab Journal of e-Technology, Vol. 1, No. 3, January 2010.